

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
28 December 2006 (28.12.2006)

PCT

(10) International Publication Number
WO 2006/138469 A2

(51) International Patent Classification:
H05B 6/04 (2006.01)

(21) International Application Number:
PCT/US2006/023319

(22) International Filing Date: 16 June 2006 (16.06.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/691,370 17 June 2005 (17.06.2005) US

(71) Applicant (for all designated States except US): **VER-
ANO, INC.** [US/US]; 575 West Street, Suite 120, Mans-
field, MA 02048 (US).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **HUTCHINSON,
Thomas, W.** [CA/CA]; 10567 Shillington Crescent S.w.,
Calgary, Alberta T2W 0N8 (CA). **GINTER, Andrew,
Francis** [CA/CA]; 59 Rosery Drive Nw, Calgary, Al-
berta T2K 1L4 (CA). **HARMS, Daryl, Dean** [CA/CA];
1307-310 Grier Avenue Ne, Calgary, Alberta T2K 5X5
(CA). **JESEN, John, Bretton** [CA/CA]; 2110d 35 Avenue
Sw, Calgary, Alberta T2T 2E4 (CA).

(74) Agent: **SATURNELLI, Anne, E.**; MUIRHEAD & SAT-
URNELLI, LLC, 200 FRIBERG PARKWAY, Suite 1001,
Westborough, MA 01581 (US).

(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN,
CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI,
GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE,
KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU,
LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG,
NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD,
SE, SG, SK, SL, SM, SY, 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, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM,
ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM),
European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI,
FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT,
RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA,
GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— without international search report and to be republished
upon receipt of that report

For two-letter codes and other abbreviations, refer to the "Guid-
ance Notes on Codes and Abbreviations" appearing at the begin-
ning of each regular issue of the PCT Gazette.

(54) Title: DURATION OF ALERTS AND SCANNING OF LARGE DATA STORES

(57) Abstract: Described are techniques used in monitoring the performance, security and health of a system used in an industrial application. Agents included in the industrial network report data to an appliance or server. The appliance stores the data and determines when an alarm condition has occurred. Notifications are sent upon detecting an alarm condition. The alarm thresholds may be user defined. A threat thermostat controller determines a threat level used to control the connectivity of a network used in the industrial application.



WO 2006/138469 A2

DURATION OF ALERTS AND SCANNING OF LARGE DATA STORES

CROSS REFERENCE TO RELATED APPLICATIONS

This application claims priority to U.S. Provisional Patent Application No. 60/691,370, filed on June 17, 2005, Attorney Docket No. VRS-002PR, and is a continuation in part of U.S. Patent No. 10/815,222, filed on March 31, 2004, Attorney Docket No. VRS-00101, which claims priority to U.S. Provisional Patent Application No. 60/477,088, filed on June 9, 2003, Attorney Docket No. VRS-00160, all of which are incorporated by reference herein.

BACKGROUND

Technical Field

This application generally relates to a network, and more particularly to event monitoring and management therein.

Description of Related Art

Computer systems may be used in performing a variety of different tasks. For example, an industrial network of computer systems and components may be used in controlling and/or monitoring industrial systems. Such industrial systems can be used in connection with manufacturing, power generation, energy distribution, waste handling, transportation, telecommunications, water treatment, and the like. The industrial network may be connected and accessible via other networks, both directly and indirectly, including a corporate network and the Internet. The industrial network may thus be susceptible to both internal and external cyber-attacks. As a preventive measure from external cyber-attacks, firewalls or other security measures may be taken to separate the industrial network from other networks. However, the industrial network is still vulnerable since such security measures are not foolproof in the prevention of external attacks by viruses, worms, Trojans and other forms of malicious code as well as computer hacking, intrusions, insider attacks, errors, and omissions that may occur. Additionally, an infected laptop, for example, can bypass the firewall by connecting to the industrial network using a modem, direct connection, or by a virtual private network (VPN). The laptop may then introduce worms or other forms of malicious code into the industrial network. It should be noted that an industrial network

may be susceptible to other types of security threats besides those related to the computer systems and network.

Thus, it may be desirable to monitor events of the industrial network and accordingly raise alerts. It may be desirable that such monitoring and reporting be performed efficiently minimizing the resources of the industrial network consumed. It may further be desirable to have the industrial network perform a threat assessment and respond in accordance with the threat assessment. In performing the assessment, it may also be desirable to take into account a wide variety of conditions relating to performance, health and security information about the industrial network, such as may be obtained using the monitoring data, as well as other factors reflecting conditions external to the industrial network.

SUMMARY OF THE INVENTION:

15 In accordance with one aspect of the invention is a method for monitoring data elements comprising: receiving one or more sets of data elements, each set representing a grouping of one or more data elements; receiving an associated resource allocation of a first resource for each of said one or more sets; and performing processing for each of said one or more sets of data elements, wherein said processing includes scanning and monitoring data
20 elements for differences, and wherein at least one of said scanning and said monitoring is performed in accordance with the associated resource allocation for each set. The resource allocation may indicate an amount of said first resource to be consumed during a defined time period. A sum of all the associated resource allocations for all the sets of data elements may indicate a total amount of said first resource allocated for consumption during a defined time
25 period. The first resource may be a processor and the resource allocation may indicate an amount of processor execution time to be consumed during a defined time period. An associated resource allocation for one of said sets may be specified as a percentage of said total amount. An associated resource allocation for one of said sets may be specified as a numeric value. Each of the sets of data elements may have an associated level of priority indicating a level of priority of data elements included therein. The method may also include, for each set of data elements: providing a set of one or more rules; and determining data elements belonging to said each set using said one or more rules as a filter. The set of rules may include at least one inclusionary rule defining one or more criteria, wherein a data element satisfying said one or more criteria is included in said each set of data elements. The set

of rules may include at least one exclusionary rule defining one or more criteria, wherein a data element having said one or more criteria is not included in said each set of data elements.

One or more data stores may be specified for each set from which data elements are selected for inclusion into said each set. The one or more data stores may include at least one data

store selected from a database and a file system. At least one of said data stores specified may have a hierarchical structure and rules may be specified using said hierarchical structure to determine which data elements belong to said each set of data elements. The method may also include: determining one or more attributes of each data element, each attribute characterizing data, metadata, or schema information for said each data element, wherein said

determining one or more attributes is performed as part of said scanning;

determining differences between a current value for each of said one or more attributes and a predetermined value; and generating an alert in accordance with any of said differences. The step of determining differences may be performed by comparing hash values for a current value and a predetermined value for at least one attribute including sensitive data. A second

resource may be specified for at least one of said sets, said second resource having an associated resource allocation for each of said data sets, and wherein at least one of said scanning and said monitoring may be performed in accordance with the associated resource allocations for said first and said second resources for each set, said at least one of said scanning and said monitoring consuming allocated resources for a data set if any one the

associated resource allocations of said first or said second resources has been consumed. The resource allocation may indicate a number of disk inputs performed during at least one of said scanning and said monitoring of a data set for a defined time period, each of said disk inputs representing a portion of data read from a device during said at least one of said scanning and said monitoring. The method may also include: determining an elapsed time

for said at least one of said scanning and said monitoring to consume said total amount of said first resource allocated during a defined time period; if the elapsed time is less than said defined time period, scheduling said processing to resume at a beginning of a next defined time period; and if the elapsed time is not less than said defined time period, scheduling said processing to resume execution immediately. A first process may perform scanning and

monitoring data elements for differences, and after said first process processes one or more data elements, another process waiting for execution may be executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor. A first process may be performing scanning and monitoring data elements for differences, and after said first

process processes a predetermined amount of data with respect to at least one of said scanning and said monitoring, another process waiting for execution may be executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor. A first
5 process may be performing scanning and monitoring data elements for differences, and after said first process consumes a fixed amount of a resource with respect to at least one of said scanning and said monitoring, another process waiting for execution may be executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor. The
10 predetermined value may be an expected value obtained in accordance with one or more previous scans of a data element at a prior point in time. The predetermined value may be determined using at least one of: simulation and theoretically expected results.

In accordance with another aspect of the invention is a method for scanning data
15 elements comprising: receiving one or more sets of data elements, each set representing a grouping of one or more data elements; receiving an associated resource allocation of a first resource for each of said one or more sets; and performing scanning for each of said one or more sets of data elements in accordance with the associated resource allocation for each set.

In accordance with another aspect of the invention is a computer readable medium
20 which comprises code stored thereon for monitoring data elements that: receives one or more sets of data elements, each set representing a grouping of one or more data elements; receives an associated resource allocation of a first resource for each of said one or more sets; and performs processing for each of said two or more sets of data elements, wherein said
25 processing includes scanning and monitoring data elements for differences, and wherein at least one of said scanning and said monitoring is performed in accordance with the associated resource allocation for each set.

In accordance with yet another aspect of the invention is a method for displaying
30 alerts on an alert display area comprising: receiving an alert message in connection with an alert condition in a system; and displaying said alert message for an alert condition, wherein, said alert message includes a duration time representing an amount of time said alert condition has existed in the system. If the alert condition has not subsided at a time when said alert message is displayed, said duration time may include as said duration time an indicator

that said alert condition is ongoing. The alert message that is displayed may represent an alert condition having a non-zero duration time. The duration time may be determined by using a model describing severity transitions of transitioning from a first severity level to a second severity level. Each severity level included in said model may be associated with

5 only a single state and said model may include transitions from a first state having said first severity level to a second state having said second severity level. At least one severity level included in said model may be associated with more than one state and said model includes transitions from a first state having said first severity level to a second state having said

10 second severity level. The method may also include: displaying an acknowledgement indicator indicating that said alert message requires manual acknowledgement, wherein said alert message is displayed until said alert message has been acknowledged using said

acknowledgement indicator and said alert condition has cleared. In response to performing an acknowledgement by selecting said acknowledgement indicator, said acknowledgement indicator may be displayed in a visually differentiable state than when said alert message has

15 not been acknowledged. The alert condition may be associated with one of a security alert and a performance alert. The duration time may have a starting time and an ending time, said starting time being determined when said alert condition is first detected and said ending time indicating when said alert condition has subsided. At a point in time, a display including alert messages may include at most a single alert message associated with said alert condition

20 which includes a duration time indicating that said alert condition is ongoing. At a point time, a display including alert messages may include a plurality of alert messages associated with said alert condition, each of said plurality of alert messages including a duration time indicating that said alert condition is ongoing. Each of said plurality of alert messages may be associated with a unique severity level. A state of said alert condition may be determined

25 using a model, and wherein said starting time and said ending time of said duration time may be determined in accordance with one or more states of said model. The alert message may be a first alert message for a non-instantaneous alert, and the method may further include: displaying on a display screen said first alert message for a non-instantaneous alert and a

second alert message for an instantaneous alert. The model may include a set of states and a

30 set of severities, each of said severities being associated with one or more states, said model may utilize one of an overlapping technique or a non-overlapping technique in connection with determining said duration time, wherein said overlapping technique has at most one ongoing alert, and, when transitioning from a first severity to a second severity, said first severity is determined to end and said second severity is determined to start, wherein said

non-overlapping technique has one or more ongoing alerts so that a time duration for one alert may overlap with a duration time of a second alert, and, when transitioning from a lower severity to a higher severity, said higher severity is determined to start, and when transitioning from a higher severity to a lower severity, said higher severity is determined to end. The model may indicate whether to output alert messages for intervening severities when transitioning from a first severity to a second severity. The model may include a rule indicating at least one severity for which alert messages are not to be generated when transitioning into the at least one severity.

In accordance with yet another aspect of the invention is a computer readable medium comprising code stored thereon for displaying alerts on an alert display area that: receives an alert message in connection with an alert condition in a system; and displays said alert message for an alert condition, wherein said alert message includes a duration time representing an amount of time said alert condition has existed in the system. The alert condition may represent a non-instantaneous alert having a duration time determined using a model for modeling said alert condition, said model may include a set of one or more severities, a set of one or more states, a set of one or more transitions defining transitioning between states, a first parameter indicating whether said model utilizes an overlapping technique or a non-overlapping technique in connection with determining said duration time, wherein when transitioning from a first severity to a second severity, a duration time for said first severity is determined to end and a duration time for said second severity is determined to start with said overlapping technique, and wherein, with said non-overlapping technique, when transitioning from a lower severity to a higher severity, a duration time for said higher severity is determined to start, and when transitioning from a higher severity to a lower severity, a duration time for said higher severity is determined to end. The model may indicate wherein said model indicates whether to output alert messages for intervening severities when transitioning from a first severity to a second severity. The model may include a rule indicating at least one severity for which alert messages are not to be generated when transitioning into the at least one severity.

BRIEF DESCRIPTION OF THE DRAWINGS:

Features and advantages of the present invention will become more apparent from the following detailed description of exemplary embodiments thereof taken in conjunction with the accompanying drawings in which:

Figure 1 is an example of an embodiment of a system described herein;

Figure 2 is an example of an embodiment of components that may be included in a
5 corporate network of the system of Figure 1;

Figure 3 is a more detailed example of an embodiment of components that may be included in an industrial network of the system of Figure 1;

10 Figure 4 is a more detailed example of an embodiment of components that may be included in the watch server of Figure 3;

Figure 4A is a more detailed example of an embodiment of the threat thermostat controller;
15

Figure 5 is an example of the different types of agents that may be included in an embodiment on systems from Figure 3;

Figure 6 is an example of an embodiment of an architecture of each of the agents from
20 Figure 5;

Figure 7 is a flowchart of steps of one embodiment for control flow within an agent;

Figure 8 is an example of an embodiment of the real time database and alarm engine
25 (RTAP) of Figure 4;

Figure 9 is an example of a representation of a database schema used by an embodiment of RTAP;

30 Figure 9A is an example of representing an alarm function within an attribute with the database schema of Figure 9;

Figures 10-11 are examples of embodiments of an alarm state table that may be used by RTAP;

Figure 12 is an example of a state transition diagram representing the states and transitions in the alarm state table of Figure 11;

5 Figure 13-14 are examples of user interface displays that may be used in an embodiment of the system of Figure 1.

Figure 15 is an example representation of a hierarchical data store that may be used in connection with the techniques herein;

10

Figure 16 is an example illustrating the information that may be defined for one or more buckets of data elements;

Figures 17-22 are flowcharts of processing steps that may be performed in an embodiment in connection with scanning techniques described herein;

15

Figure 23 is an example illustrating a prior art alert display;

Figures 24-35 illustrate examples that may be performed in connection with the various techniques described herein related to alert messages and durations; and

20

Figures 36—38 are flowcharts of processing steps that may be performed in an embodiment in connection with alert messages and durations.

DETAILED DESCRIPTION OF EMBODIMENT(S):

Referring now to Figure 1, shown is an example of an embodiment 10 of the system that may be used in connection with techniques described herein. The system 10 may be part of an infrastructure used in connection with, for example, manufacturing, power generation, energy distribution, waste handling, transportation, telecommunications, water treatment, and the like. Included in the system 10 is a corporate network 12 connected through a hub, switch, router and/or firewall 16 to an industrial network 14. The corporate network 12 may be connected to one or more external networks such as the Internet 20 through a firewall 18 and/or other devices. Also connected to the corporate network 12, either directly or via the firewall 18, may be a mail server 30, a web server 32 and/or any one or more other hardware and/or software components.

It should be noted that although the system 10 of Figure 1 includes a firewall 18 and may also include one or more other firewalls or security measures, the corporate network as well as the industrial network may be susceptible to cyber attacks and other types of security threats, both malicious and accidental. As will be described in following paragraphs, different computer systems that may be included within an embodiment of the industrial network 14 must operate in accordance with an extremely high rate of failsafe performance due to the critical applications and tasks that the industrial network may be used in connection with. In other words, there is a very low tolerance for failure of components included in the industrial network 14. Loss of control and failure within the industrial network 14 may result in much more catastrophic conditions than a failure that may occur within the corporate network 12. For example, a catastrophic failure within the corporate network 12 may force a back-up retrieval of information. However, in the event that the industrial network 14 is being used in connection with supplying power or controlling a system such as train switching, failure may result in a catastrophic loss in terms of both human and economic dimensions.

In connection with the system 10, it should also be noted that external threats such as may be encountered from an external hacker coming through the Internet 20 to access the industrial network 14 may only account for part of the security threats. A large number of cyber attacks and other threats may come from within the system 10 itself such as, for example, within the corporate network 12 or from within the industrial network 14. For

example, a disgruntled employee may attempt to perform a malicious attack from within the industrial network 14 as well as within the corporate network 12 in an attempt to cause operation failure of one or more components of the industrial network 14. As another example, someone may connect to the industrial network or the corporate network 12 using a laptop that might be infected, for example, with a form of malicious codes such as a Trojan, a virus, a worm, and the like. This malicious code may be introduced within the system 10 on the corporate network 12 or within the industrial network 14 independent of the firewall 18 and/or firewall 16 functioning. Such types of internal threats may not be caught or prevented by the firewall or other security measures developed for preventing primarily external threats. Thus, an embodiment of the system 10 may ideally include and utilize other techniques in connection with controlling, supervising, and securing operation of the components within the system 10 in a failsafe manner.

The corporate network 12 may include components generally used in office and corporate activities such as, for example, systems used by individuals in performing accounting functions, and other administrative tasks. The web server 32 may be used, for example, in servicing requests made to a website associated with the corporate network 12. Incoming e-mail from the internet 20 to the corporate network 12 may be handled by the e-mail server 30. It should be noted that an embodiment of the system 10 may include other components than as described herein in accordance with a particular functionality of each embodiment.

The corporate network 12 may be connected to the industrial network 14 through the hub, switch, router, or firewall 16. It should be noted that the corporate network 12 may be connected to the industrial network 14 by one or more of the foregoing mentioned in connection with element 16. In other words, the element 16 in Figure 1 may represent a layering or hierarchical arrangement of hardware and/or software used in connecting the corporate network 12 to the industrial network 14. The different arrangements of 16 included in an embodiment may vary in accordance with a desired degree of security in accordance with the particular use of the components within the industrial network 14.

Included in the industrial network 14 in this embodiment is a Watch server 50. The Watch server 50 may be characterized as performing a variety of different monitoring,

detection, and notification tasks in connection with the industrial network 14 and connection to the corporate network. The Watch server 50 is described in more detail elsewhere herein.

Components included in an embodiment of the system 10 may be connected to each other and to external systems and components using any one or more different types of communication medium(s). The communication mediums may be any one of a variety of networks or other type of communication connections as known to those skilled in the art. The communication medium may be a network-connection, bus, and/or other type of data link, such as a hardwire or other connections known in the art. For example, the communication medium may be the Internet, an intranet, network or other non-network connection(s) which facilitate access of data and communication between the different components.

It should be noted that an embodiment may also include as element 16 other types of connectivity-based hardware and/or software as known to those of ordinary skill in the art to connect the two networks, the corporate network 12 and the industrial network 14. For example, the element 16 may also be a dial-up modem connection, a connection for wireless access points, and the like.

The different components included in the system 10 of Figure 1 may all be located at the same physical site, may be located at different physical locations, or some combination thereof. The physical location of one or more of the components may dictate the type of communication medium that may be used in providing connections between the different components. For example, some or all of the connections by which the different components may be connected through a communication medium may pass through other communication devices and/or switching equipment that may exist, such as a phone line, a repeater, a multiplexer, or even a satellite.

Referring now to Figure 2, shown is an example of an embodiment of components that may be included within a corporate network 12. Included in this embodiment 12 of Figure 2 are user systems 40a-40b, and a hub, switch, firewall, or WAN router 42. The component 42 may be used in connecting this particular corporate network to one or more other corporate networks, to the firewall 18, and also to any other components included in 16 previously described in connection with Figure 1.

Each of the user systems 40a-40b may include any one of a variety of different types of computer systems and components. Generally, in connection with computer systems included within the corporate network 12 as well as in connection with other components
5 described herein, the processors may be any one of a variety of commercially available single or multi-processor systems such as, for example, an Intel-based processor, an IBM mainframe, or other type of processor able to support the incoming traffic and tasks in accordance with each particular embodiment and application. Each of the different components, such as the hub, switch, firewall, and/or router 42, may be any one of a variety
10 of different components which are commercially available and may also be of a proprietary design.

Each of the user systems 40a-40b may include one or more data storage devices varying in number and type in accordance with each particular system. For example, a data
15 storage device may include a single device, such as a disk drive, as well as a plurality of devices in a more complex configuration, such as with a storage area network (SAN), and the like. Data may be stored, for example, on magnetic, optical, silicon-based, or non-silicon-based media. The particular arrangement and configuration may vary in accordance with the parameters and requirements associated with each embodiment and system.

20 Each of the user systems 40a-40b, as well as other computer systems described in following paragraphs, may also include one or more I/O devices such as, for example, a keyboard, a mouse, a display device such as a monitor, and the like. Each of these components within a computer system may communicate via any one or more of a variety of
25 different communication connections in accordance with the particular components included therein.

It should be noted that a corporate network may include other components besides user systems such as, for example, a network printer available for use by each user system.
30

Referring now to Figure 3, shown is a more detailed example of an embodiment 100 of components previously described in connection with the system 10 of Figure 1. Included in the industrial network 14 in one embodiment may be a process LAN 102, a control network 104, an I/O network 106, one or more other I/O networks 124a and 124b, and a

Watch server 50. In this example, the industrial network 14 may be connected to the corporate network 12 by the hub, switch, router, or firewall 16. It should be noted that the industrial network 14 may include other components than as described herein as well as multiple instances of components described herein. In one embodiment, component 16 may
5 be an integrated security appliance such as, for example, the Fortinet Fortigate appliance.

The process LAN 102 may be characterized as performing tasks in connection with data management, integration, display, and the like. The control network 104 may be used in connection with controlling the one or more devices within the I/O network 106 as well as
10 one or more other I/O networks 124a and 124b. The Watch server 50 may be characterized as performing a variety of different monitoring, detection, and notification tasks in connection with the industrial network 14 and connection to the corporate network. The Watch server 50 and other components included within an embodiment of 14 described in more detail in the following paragraphs may be used in connection with the operation of the
15 industrial network 14 in order to provide for proper operation of the industrial network 14 and component 16 and security threat management.

The process LAN 102 of Figure 3 includes a switch or hub 110a connected to component 16 and one or more other components within the process LAN 102. Components
20 included in this example of the process LAN 102 are a historian 114 and an application server 116. The historian 114 may be used, for example, in storing a history of the different monitoring data that may be gathered by other components included within the network 14. The historian 114, for example, may serve as a data archive for the different types of data gathered over time within the network 14. The application server 116 may be used to execute
25 an application that performs, for example, process optimization using sensor and other data. The application server 116 may communicate results to the SCADA server for use in controlling the operations of the network 14.

The SCADA (Supervisory Control and Data Acquisition) server 118 may be used in
30 remotely monitoring and controlling different components within the control network 104 and the I/O network 106. Note also that the SCADA server included in Figure 2 generally refers to a control system, such as a distributed control system (DCS). Additionally, the SCADA server 118 may also be responsible for controlling and monitoring components included in other I/O networks 124a and 124a. For example, the SCADA server 118 may

issue one or more commands to the controller 122 in connection with controlling the devices 130a-130n within the I/O network 106. The SCADA server 118 may similarly be used in connection with controlling and monitoring other components within the I/O networks 124a and 124b. As known to those of ordinary skill in the art, a SCADA server may be used as
5 part of a large system for remotely monitoring and controlling, for example, different types of energy production, distribution and transmission facilities, transportation systems, and the like. Generally, the SCADA server 118 may be used in connection with controlling and remotely or locally monitoring what may be characterized as components over possibly a large geographically distributed area. The SCADA server may rely on, for example,
10 communication links such as radio, satellite, and telephone lines in connection with communicating with I/O networks 124a and 124b as well as I/O network 106. The particular configuration may vary in accordance with each particular application and embodiment.

The workstation 120 may include a human machine interface (HMI), such as a
15 graphical user interface (GUI). The workstation 120 may be used, for example, in connection with obtaining different sensor readings, such as temperature, pressure, and the like, from the devices 130a-130n in the I/O network 106, and displaying these readings on the GUI of the workstation 120. The workstation 120 may also be used in connection with accepting one or more user inputs in response, for example, to viewing particular values for different sensor
20 readings. For example, the workstation 120 may be used in connection with an application monitoring a transportation system. An operator may use the GUI of workstation 120 to view certain selected statistics or information about the system. The selections may be made using the GUI of the workstation 120. Other inputs from the workstation 120 may serve as instructions for controlling and monitoring the operation of different devices and components
25 within the industrial network 14 and one or more I/O networks. For example, the transportation system may be used in dispatching and monitoring one or more trains.

The SCADA server 118 may also be used in connection with performing data acquisition of different values obtained by the device sensors 130a-130n in performing its
30 monitoring and/or controlling operations. The data may be communicated to the SCADA server 118 as well as the workstation 120. The SCADA server 118, for example, may monitor flow rates and other values obtained from one or more of the different sensors and may produce an alert to an operator in connection with detection of a dangerous condition. The dangerous condition or detection may result in an alarm being generated on the

workstation 120, for example, such as may be displayed to a user via the GUI. The SCADA server 118 monitors the physical processing within the industrial network and I/O network(s). The server 118 may, for example, raise alerts to an operator at the workstation 120 when there is a problem detected with the physical plant that may require attention.

5

The controller 122 may be used in connection with issuing commands to control the different devices, such as 130a-130n, as well converting sensor signal data, for example, into a digital signal from analog data that may be gathered from a particular device. An embodiment may also have a controller 122 perform other functionality than as described herein.

10

The Watch server 50 may be used in connection with monitoring, detecting, and when appropriate, notifying a user in accordance with particular conditions detected. The Watch server 50 may include a Watch module which is included in an appliance. The Watch server 50 may also be installed as a software module on a conventional computer system with a commercially available operating system, such as Windows or LINUX, or a hardened operating system, such as SE LINUX. In one embodiment, the Watch server 50 may be, for example, a rack mount server-class computer having hardware component redundancy and swappable components. The appliance or conventional computer system may be executing, for example, SE LINUX on an IBM X-series server, or RedHat Enterprise Linux on an Intel rack-mount, that monitors the logs and performance of the industrial network 14. The foregoing may be used in connection with monitoring, detecting and notifying a human and/or controlling computer system or other components where appropriate.

15

20

25

It should be noted that the Watch server 50 may be used in raising alerts detected in connection with the SCADA system, associated networks, and computer processors. In other words, the tasks related to monitoring the computers and networks of Figure 3 are performed by the Watch server 50. In contrast, as known to those of ordinary skill in the art, the tasks related to the physical plant processing, sensor data gathering, and the like for controlling and monitoring the operation of the particular industrial application(s) are performed by the SCADA server 118.

30

Included in an embodiment of the network 14 are one or more agents 132a-132d that may be used in collecting data which is reported to the Watch server 50. It should be noted

that each of the agents 132a-132d may refer to one or more different agents executing on a computer system to perform data gathering about that computer system. The agents 132a-132d report information about the system upon which they are executing to another system, such as the Watch server 50. The different types of agents that may be included in an embodiment, as well as a particular architecture of each of the agents, are described in more detail elsewhere herein. In addition to each of the agents reporting information to the Watch server 50, other data gathering components may include an SNMP component, such as 112a-112c, which also interact and report data to the Watch server 50. Each of the SNMP components may be used in gathering data about the different network devices upon which the SNMP component resides. As known to those of ordinary skill in the art, these SNMP components 112a-122c may vary in accordance with each particular type of device and may also be supplied by the particular device vendor. In one embodiment, the Watch server 50 may periodically poll each of the SNMP components 112a-112c for data.

In one embodiment of the industrial network 14 as described above, the Watch server 50 may be executing the SE LINUX (Security Enhanced LINUX) operating system. Although other operating systems may be used in connection with the techniques described herein, the SE LINUX operating system may be preferred in an embodiment of the Watch server 50 for at least some of the reasons that will now be described. As known to those of ordinary skill in the art, some operating systems may be characterized as based on a concept of discretionary access control (DAC) which provides two categories of a user. A first category of user may be an administrator for example that has full access to all system resources and a second category of user may be an ordinary user who has full access to the applications and files needed for a job. Examples of operating systems based on the DAC model include for example, a Windows-based operating system. DAC operating systems do not enforce a system-wide security policy. Protective measures are under the control of each of the individual users. A program run by a user, for example, may inherit all the permissions of that user and is free to modify any file that that user has access to. A more highly secure computer system may include an operating system based on mandatory access control (MAC). MAC provides a means for a central administrator to apply system wide access policies that are enforced by the operating system. It provides individual security domains that are isolated from each other unless explicit access privileges are specified. The MAC concept provides for a more finely-grained access control to programs, system resources, and files in comparison to the two level DAC system. MAC supports a wide variety of categories

of users and confines damage, for example, that flawed or malicious applications may cause to an individual domain. With this difference in security philosophy, MAC may be characterized as representing a best available alternative in order to protect critical systems from both internal and external cyber attacks. One such operating system that is based on the

5 MAC concept or model is the SE LINUX operating system. The SE LINUX operating system is available, for example, at <http://www.nsa.gov/selinux>.

The components included in the industrial network 14 of Figure 3, such as the agents 132a-132d, SNMP components 112a-112c, and the Watch server 50, may be used in

10 connection with providing a real time security event monitoring system. The different agents 132a-132d included in the industrial network 14 may be installed on the different computer systems included in the industrial network 14 and may report, for example, on machine health, changes in security log files, application status, and the like. This information gathered by each of the agents 132a-132d and SNMP components 112a-112c may be

15 communicated to the Watch server 50. This information may be stored in a real-time database also included on the Watch server 50. From the Watch server 50, alarm limits may be set, alerts may be generated, incident reports may be created, and trends may also be displayed. Watch server 50 may also run a network intrusion detection system (NIDS), and has the ability to monitor network equipment, such as switches, routers, and firewalls, via the

20 SNMP components included in various network devices shown in the illustration 100.

The agents 132a-132d described herein may be characterized as intelligent agents designed to run and report data while minimizing the use of system and network resources. A control network 104 may have a relatively low amount of network bandwidth.

25 Accordingly, when deploying a monitoring device, such as an agent 132a-132d within such a control system, there may be inadequate bandwidth available for reporting the data from each agent. The lower bandwidth of a control network is typical, for example, of older legacy systems upon which the various agents may be deployed. Agents within an embodiment of the network 14 may be designed to minimize the amount of CPU usage, memory usage, and

30 bandwidth consumed in connection with performing the various data gathering and reporting tasks regarding the industrial network 14. In one embodiment, agents may be written in any one or more of a variety of different programming languages such as PERL, C, Java, and the like. Generally, agents may gather different types of information by executing system commands, reading a data file, and the like on the particular system upon which the agents

are executing. It should be noted that the agents also consume resources in a bounded manner minimizing the variance of consumption over time. This is described elsewhere herein in more detail.

5 Agents 132a-132d may format the information into any one of a variety of different message formats, such as XML, and then report this data to the Watch server 50. In one embodiment, the agents 132a-132d may communicate the data to the Watch server 50 over TCP/IP by opening a socket communication channel just long enough to send the relevant data at particularly selected points in time. In one embodiment, the agents 132a-132d operate
10 at the application level in communicating information to the Watch server 50. The Watch server 50 does not send an application level acknowledgement to such received data. Additionally, the agents never read from the communication channel but rather only send data out on this communication channel as a security measure. It should be noted that although the embodiment described herein uses TCP, the techniques described herein may be
15 used in an embodiment with UDP or another type of connectionless communication.

As described in the following paragraphs, the agents that may be included in a system 10 of Figure 1 may be generally characterized as 2 different classes of monitoring agents. A first class of agent may be used in monitoring control systems upon which the agent actually
20 executes. Agents in this first class are those included in the industrial network 14 of Figure 3, such as agents 132a-132d. A second class of agent that may be included in an embodiment is described elsewhere herein in connection with, for example, the Watch server 50. Agents of this second class may be used in monitoring input from third party equipment or applications or other activity about a system other than the system upon which the agent is executing. As
25 described in more detail elsewhere herein, different types of agents of either class may be used in an embodiment to gather the different types of data.

It should be noted that the various activities performed by the agents described herein may be used in connection with monitoring and reporting, for example, on cyber-security
30 incidents as well as the performance of different system devices such as, for example, the different computer systems for CPU load, space and memory usage, power supply voltage, and the like. Although one particular type of security threat is a cyber-security threat as described herein, it should be noted that the techniques and systems described herein may be used in connection with monitoring security threats that are of different types. For example,

control system damage may be caused, for example, by a faulty power supply, faulty performance of other components, and the like. The various agents deployed within an embodiment of the system 10 of Figure 1 may be used in detecting both malicious as well as accidental threats to the industrial network 14. It may also be useful to correlate performance information with security information when assessing the likelihood of system compromise as a result of an attack on one or more systems.

Although the agents 132a-132d are illustrated in connection with particular components included in Figure 3, agents 132a-132d may also be used in monitoring a controller 122, devices 130a-130n, and the like as needed and possible in accordance with each embodiment. Agents may be used on components having a computer processor capable of performing the tasks described herein and meeting predefined resource limitations that may vary with each embodiment. For example, agents may be executed on "smart" controllers, such as 122, if the controller has a processor able to execute code that performs the agent functionality. There may also be a connection from the controller 122 to the Watch server 50 to communicate the agent gathered data to the Watch server 50.

In Figure 3, it should be noted that each of 113a, 113b, and 113c may refer to one or more connections and may vary in accordance with the particular component connected to the Watch server 50 by each connection. For example, component 16 may be a hub, switch, router or firewall. If component 16 is a switch, element 113a may refer to two communication connections between 16 and the Watch server 50 where one of the connections is connected to the spanning port of the switch and is used for the monitoring operations for network intrusion detection by the Watch server 50. In an embodiment in which the component is a hub, a single connection may be used. The foregoing also applies to connections 113b and 113c and, respectively, components 110a and 110b.

Referring now to Figure 4, shown is an example of an embodiment of components that may be included in a Watch server 50. Watch server 50 in one embodiment may include a threat agent 200, an SNMP Watch agent 202, an SNMP Guard Agent 203, a NIDS agent 204, an ARPWatch agent 206, and a Guard log agent 209. Data 201 produced by the agents executing in the industrial network may be received by the Watch server. The Watch server 50 itself may be monitored using one or more agents 208 of the first class. Data from these agents is referenced as 208 in Figure 4. Each of 200, 201, 202, 203, 204, 206, 208 and

209 communicates with the receiver 210 to store data in RTAP (real-time database and alarm engine) 212. The Watch server 50 may also include a web server 214, a notification server 216, a threat thermostat controller 218 and one or more firewall settings 220.

5 The agents included in the Watch server 50, with the exception of the agents reporting data 208, are of the second class of agent described elsewhere herein in which the agents included in the Watch server gather or report information about one or more systems other than that system upon which the agent is executing. It should be noted that this class of agent is in contrast, for example, to the agents 132a-132d previously described in connection with
10 Figure 3 which are executed on a computer system and report information to the Watch server 50 about the particular computer system upon which the agent is executing. The agents included within the Watch server 50 may be used in gathering data from one or more sources. As described in connection with the agents of the first class of agent, the second class of agents may be written in any one or more programming languages.

15

 The threat agent 200 receives threat assessments from one or more sources which are external to the industrial network 14. The inputs to the component 200 may include, for example, a threat assessment level or alert produced by the corporate network, a security or threat level produced by the US government, such as the Homeland Security Threat level, an
20 input published on a private site on the Internet, and the like.

 Data from agents of both classes executing within the industrial network 14 of Figure 3 communicate data to the receiver 210 as input source 201. It should be noted that any one or more of the metrics described herein may be reported by any of the agents in connection
25 with a periodic reporting interval, as well as in accordance with the occurrence of certain thresholds or events being detected.

 The SNMP Watch agent 202 periodically polls the different devices including hubs, switches, and routers having a vendor supplied SNMP component, such as 112a and 112b.
30 The Watch server performs this periodic polling and communicates with each of the SNMP components 112a-112b to obtain information regarding the particular component or network device. SNMP Components, such as 112a, report data to agent 202 such as, for example, metrics related to the activity level on switches and the like being monitored. Similarly, the SNMP Guard agent 203 periodically polls the firewall(s) for data. In one embodiment, the

component 203 works with the Fortinet Fortigate series of firewalls. The particular firewall(s) supported and utilized may vary in accordance with each embodiment. The components 202 and 203 use the SNMP protocol to request the information set forth below from network switches, firewalls, and other network equipment.

5

In one embodiment, the following metrics may be reported by 203 at the end of each periodic reporting interval. It should be noted that the units in an embodiment may vary from what is specified herein. Also, not all metrics may be available and tracked by all devices:

10

Uptime - How long the device has been running continuously since it was last rebooted or reset.

Configuration - Information from the device that describes the device. This may include, for example, the name of the vendor, the model number, the firmware version, and any other software or ruleset versions that the device supports and may be relevant to an understanding of the behavior of the device.

15

Communications Status - An indication as to whether the device's SNMP component responded to the previous and/or current SNMP requests for information.

20

Total incoming and outgoing network traffic, in kilobytes, per reporting interval.

Per-interface incoming and outgoing network traffic, in kilobytes, for a reporting interval.

25

%CPU Load - The percentage of CPU Utilization in the reporting interval.

%Disk space - For devices with disks, like some firewalls, report the percentage used for every filesystem or partition.

30

%Memory Used - Report the fraction of physical memory in use.

Open Session Count - a count of open communications sessions.

VPN Tunnel count - a count of the number of machines and/or users connected through a firewall device using an IPSEC, SSL or other Virtual Private Network (VPN) technology. For each such connection, also report the source IP address, and if available:

- 5 host name, user name, certificate name and/or any other information that might serve to identify who or what is connected to the control network via the VPN connection.

- Administrative User Count - A count of how many "root" or other administrative users are logged into the device and so are capable of changing the configuration of the device. For each such user, when the information is available via SNMP, report the user name, source IP address and any other information that might help to identify who is logged in.
- 10

- With reference to the foregoing metrics, in one embodiment, the agent 202 may report at periodic intervals uptime, total incoming and outgoing network traffic, and information regarding the particular operating system, version number and the like.
- 15

- The NIDS agent 204 monitors the process and control LAN communications by scanning messages or copies of messages passing through LANS, hubs, and switches. It should be noted that referring back to Figure 3, a dedicated connection between a Watch server and these components may be used in connection with performing NIDS monitoring. The NIDS Agent 204 receives data and determines metrics, for example, such as a message count, network intrusion alerts raised, and the like. The NIDS agent 204 may be used in connection with monitoring both the process LAN and the control LAN communications. As an input to the NIDS agent 204, data may come from the NIDS component 204a. Data from LANs, hubs and switches may be input to the NIDS component 204a and used in connection with detection of network intrusions. Included within the NIDS component 204a is a library of signatures that may be used in connection with detecting different types of network intrusions. It should be noted that the NIDS agent 204 may be used in performing real time traffic analysis and packet logging of control networks. The NIDS component 204a may also be used in connection with performing protocol analysis, content searching and matching, and may be used to detect a variety of attacks and different types of probes including, for example, buffer overflows, stealth port scans, CGI attacks, SMB probes, fingerprinting
- 20
- 25
- 30

attempts and the like. The NIDS agent 204 may also be used in connection with monitoring an existing third party network intrusion detection system installed on a control network.

In one embodiment, the NIDS component 204a is implemented using SNORT technology. As known to those of ordinary skill in the art, SNORT is described, for example, at www.snort.org, and may be used in connection with monitoring network traffic, for example, by monitoring and analyzing all messages on the network, such as through one of the connections connected into a spanning port of a switch used in an embodiment of Figure 3. SNORT reports data on any messages exchanged between any two ports. SNORT uses a pattern matching engine which searches one or more messages for known patterns. The known patterns are associated with known viruses, worms, and the like. For example, a message packet may include a particular bit pattern indicative of a known worm. The NIDS agent may be implemented using both customized and/or conventional NIDS engines. In one embodiment using the SNORT technology, the SNORT open source NIDS engine may be used with the entire SNORT NIDS rules set. An embodiment may also use other NIDS technologies such as, for example, the Fortinet Fortigate NIDS system with the Fortinet Fortigate rules set. An embodiment may also use more than one NIDS system. Specific rules may be disabled or made more specific at particular sites if normal background traffic at the site is found to generate an unacceptable number of false positive alerts as a result of enacting particular rules. This may vary in accordance with each embodiment. An embodiment may connect one or more NIDS engines to the hubs and other components of the industrial network. In the event that the industrial network includes switches, the NIDS engines may be connected to the spanning ports on the switches as described elsewhere herein. If spanning ports are not available, it may be preferable to connect the NIDS engines to monitor the traffic between the industrial network 14 and the corporate network 12, using an Ethernet tap, hub, or other technique as known to those of ordinary skill in the art.

In one embodiment of the NIDS component 204a, customized signatures may be used in addition to those that may be supplied with the NIDS technology such as publicly available at the Snort.org website. These additional signatures may identify network traffic that would be "normal" on a business network, but is not associated with normal activity within an industrial network. These may include, for example, signatures identifying one or more of the following: telnet login traffic; ftp file transfer traffic; web browsing through normal and encrypted connections; reading email through POP3, IMAP and Exchange

protocols; sending email through SMTP and Exchange protocols; and using any of the instant messaging products, such as those from Microsoft, Yahoo, and America Online. The foregoing may be considered normal traffic on a business network, such as the corporate network 12, but not within a dedicated-purpose network like the industrial network 14. It should be noted that plant operators within the network 14 may have access to email and other such facilities, but such access may be gained using workstation computers that, while they may sit physically beside critical equipment, may actually be connected to the corporate network and not to the industrial network.

In one embodiment, the following additional NIDS signatures may be used to report all attempts to communicate with well-known UDP and TCP ports having support which is turned off on correctly-hardened Solaris machines running the Foxboro IA industrial control system software:

	<u>Port number</u>	<u>application or service</u>
	7	(ECHO)
	9	(DISCARD)
	13	(DAYTIME)
	19	(CHARGEN)
	21	(FTP)
	23	(TELNET)
	25	(SMTP)
	79	(FINGER)
	512	(REXEC)
	513	(RLOGIN)
	514	(RSH)
	540	(UUCP)
	1497	(RFX-LM)

The foregoing are pairings of a port number and an application or service that may typically access this port for communications. In connection with an industrial network, there should typically be no communications for these ports using the TCP or UDP communication protocols by the above service or application. If any such communications are observed, they

are flagged and reported. The foregoing pairings are well-known and described, for example, in RFC 1700 Assigned Number, section entitled "Well-known Port Numbers", at <http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1700.html>.

- 5 An embodiment may also include additional signatures representative of traffic that should not be present on an industrial network. The signatures below may be used, for example, to monitor and report on ports used for TCP and/or UDP to identify uses that may be characteristic and typical within the corporate or other business network, but are not desirable for use within an industrial network. This may include, for example, using
- 10 particular applications and services such as Microsoft's NetMeeting, AOL and other Instant Messaging services, and the like.

For both TCP and UDP, report communications for the following:

15	<u>Port number</u>	<u>application or service</u>
	80	(HTTP)
	443	(HTTPS)
	143	(IMAP)
	993	(IMAPS)
20	110	(POP3)
	995	(POP3S)
	989	(FTPS-DATA)
	990	(FTPS)
	992	(TELNETS)
25	389	(LDAP-NETMEETING)
	552	(ULS-NETMEETING)
	1503	(T.120-NETMEETING)
	1720	(H.323-NETMEETING)
	1731	(MSICCP-NETMEETING)
30	1590	(AOL-AOL_IMESSENGER)
	194	(IRC)

For embodiments using Yahoo Instant Messenger, report activity on the following:

TCP ports 5000, 5001, 5050, 5100

UDP ports 5000-5010

5

For embodiments using Microsoft Instant Messenger, report activity on the following:

TCP ports 1863, 6901 for application/service: (VOICE)

TCP ports 6891-6900 for application/service: (FILE-XFER)

10

UDP port 6901 for application/service: (VOICE)

Note that the particular signatures used may vary with that activity which is typical and/or allowed within each embodiment.

15 The foregoing are examples of different activities that may not be characterized as normal within operation of an industrial network 14. Any additional signatures used may vary in accordance with each embodiment.

20 In one embodiment of the NIDS component 204a, customized signatures may be used in an inclusionary manner. Exclusionary NIDS signatures, such as those illustrated above, typically identify and report upon abnormal or undesirable messages detected on the network being monitored. As additional undesirable, abnormal or atypical messages are identified, new and corresponding NIDS signatures are developed for use in the embodiment. An embodiment may use inclusionary NIDS signatures to identify messages that are part of the network's normal, or desirable operating state and report upon any message not identified as such. New inclusionary NIDS signatures are developed in such an embodiment whenever a type of message is determined to be within the normal or acceptable behavior of a given network. In an embodiment using the inclusionary signatures, no additional signatures are needed to identify new undesirable or abnormal messages. Inclusionary signatures therefore

25 incur lower signature update costs than do exclusionary signatures on networks whose normal or desirable network or message traffic patterns change infrequently. On such networks, it can be argued that inclusionary signatures may provide greater security, because they are immediately able to identify new abnormal or undesirable messages, without waiting for exclusionary signatures to be identified, developed or installed.

30

In one embodiment, the NIDS agent 204 may have a reporting interval every 10 seconds. The password age agent 310, described elsewhere herein, may have a reporting interval of every hour. The other agents may have a reporting interval of once every minute which may vary in accordance with the resources within each embodiment and computer systems therein.

The ARPWatch agent 206 may detect one or more events of interest in connection with the ARP (address resolution protocol) messages monitored. ARP may be used with the internet protocol (IP) and other protocols as described in RFC 826 at <http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc0826.html>. In one embodiment, the ARPWatch 206a is based on open source software, as described in <ftp://ftp.ee.lbl.gov/arpwatch.tar.gz>, with additional functionality described herein to minimize false positives. As known to those of ordinary skill in the art, ARP may be used in performing IP address resolution and binding to a device address. The ARPWatch 206a may, for example, look for any new devices, such as computers, detected on a network. The ARPWatch 206a may monitor the industrial network 14 for the introduction of new types of devices and log information regarding these new devices. The ARPWatch agent 206, for example, may use information on new devices provided by 206a and report information to RTAP in a fashion similar to that used with the NIDS agent using SNORT based technology. For example, a laptop computer may be turned on and connected for a short time period to a network by a wireless connection. When the laptop is introduced into the network, a network address is associated with the laptop while in the network. This may be done by binding an IP address to the device address unique to the particular laptop. When first introduced into the network, there is no known IP address for the laptop. In connection with assigning an IP address to this new laptop, one or more messages may be monitored in the network to detect this event of introducing a new device in the network causing a new binding of an IP address to the unique address of the laptop. The IP addresses associated with the device may be assigned on a visitor or temporary basis, such as with the laptop. Thus, IP addresses may be reused such that a same IP address may be bound to different devices at different points in time.

In one embodiment, a customized version of ARPWatch may raise an alert if the device address of the computer was not previously known or active. An alert may also be raised if the device address was assigned to some other IP address or the IP address had some other device address assigned to it within a prior time period, for example, such as 15-20 minutes. Within an embodiment, the prior time period may vary. It should be noted that the prior time period may be long enough to be confident that the device to which the other IP address was assigned is no longer connected to the network.

Such activity may be a sign of, for example, ARP spoofing or ARP poisoning. As known in the art, ARP spoofing occurs when a forged ARP reply message is sent to an original ARP request, and/or forged ARP requests are sent. In the forged replies and/or forged requests, the forger associates a known IP address with an incorrect device address, often the forger's own device address. Thus, receiving the forged ARP messages causes a reassignment of the known IP address to the address of the forger's device, in one or more other devices on the network. This can also cause the network's own table of address bindings to be updated or poisoned with the forged binding. The time period described above may be used to minimize the false positives as may be generated when using the standard open source ARPWatch in IP-address-poor DHCP environments.

The agent 206 in one embodiment uses a version of ARPWatch that issues fewer messages than the conventional open source version for ARP frames containing invalid IP addresses that do not fall within the local network mask. The conventional open source ARPwatch logs a message for every ARP frame containing an invalid IP address resulting in up to 100 messages per minute for a given invalid IP address detection.

The ARPWatch 206a in one embodiment keeps track of the invalid addresses detected and reports invalid IP/device address binding for as long as the ARPWatch 206a is running. The RTAP 212 may also track changed IP addresses reported by the ARPWatch 206a via the ARPWatch agent 206, and the web server 214 may then present a list of these to the network administrator for approval such that these addresses are no longer considered as "new." An embodiment may also provide functionality for approving new and/or changed IP/device address bindings, and/or provide functionality for individually approving new IP addresses and/or new and/or changed IP/device address bindings as well as approving the entire list presented at once. Once the foregoing new device address/IP bindings are approved, the

number of "new" network devices drops to zero and any RTAP alert that may be outstanding against that count is reset to a normal condition or state.

In one embodiment, agents of the first class described elsewhere herein may be executing on the Watch server monitoring the health, performance, and security of the Watch server. The data from these agents 208 is sent to the receiver 210 to report data about the Watch server 50. The particular types of this first class of agent are described elsewhere herein, for example, in connection with Figures 5 and 6. An embodiment may include any number of these types of agents of the first class to report data about the Watch server 50.

The Guard log agent 209 monitors log files of the Watch server 50 and also the firewall log files of one or more firewalls being monitored, such as an embodiment including a firewall as component 16 of Figure 3. In one embodiment, an option may be set on the Fortinet Fortigate firewall to automatically transmit all firewall log information to the Watch server 50. On the Watch server 50, the received firewall log information may be included as part of the system logging for the Watch server 50. In one embodiment, the system log files of the Watch server may be segregated into different physical files of the file system. The firewall log files may be included in a separate physical file in the Watch Server's log file area. The Guard log agent 209 may also obtain additional information from the firewall using SSH (Secure SHell). The agent 209, using SSH (Secure SHell), remotely logs into a machine via a shell. As known to those of ordinary skill in the art, SSH may be characterized as similar in functionality to telnet, however unlike telnet, all data exchanged is encrypted. In one embodiment, the agent 209 may download a copy of the firewall rules and other configuration information currently in use in a firewall using SSH. Other embodiments may use other secure command and communications technologies such as, for example, IPSEC, sftp, HTTPS, SSL, and the like. The agent 209 may have a master set of firewall configuration information which it expects to be currently in use. The downloaded copy of firewall configuration information currently in use may be compared to the master set to determine any differences. In the event that any differences are detected, an alert may be raised and signaled by RTAP 212.

In one embodiment, the following metrics may be reported by the Guard log agent 209 in connection with the firewall log files every reporting interval:

firewall violations - Report the total number of firewall violations messages detected in the reporting interval. On platforms that distinguish different kinds of violation messages, report those counts separately. For example, one embodiment uses ipchains which, as known to those of ordinary skill in the art, is available on some versions of Linux to set up, maintain and inspect the IP firewall rules in the Linux kernel. Ipchains provides for distinguishing "dropped" packets from "rejected" packets, and these counts may be reported out separately.

Report the first three firewall violation messages detected for each type of violation message in each reporting interval.

10

Report a summary information for each types of firewall violation message. The summary information may include, for example, one or more IP addresses identified as the source for most of the messages, what percentage of the messages are associated with each IP address, one or more IP addresses that are the target in a majority of the messages, and what percentage of messages are associated with each target address.

15

Network IDS (intrusion detection system) and IPS (intrusion prevention system) reports - Report the total number of intrusion detection and prevention reports logged in the reporting period. For systems that report different kinds or priorities or severities of intrusion attempts, report the total number of each class of attempts as separate metrics.

20

For each separately-reported class of intrusion attempts, report the first three attempts logged in each reporting interval as well as the total number of attempts.

25

Summary reporting information for each class of intrusion attempt--Report the most common source IP address, destination IP address and attack type and the percentage of total attempts in that class of attempt that had that most common source IP, destination IP or attack type.

30

Firewall configuration change -- This is described above in which the agent 209 may report a boolean value indicating whether any aspect of the industrial network firewall configuration has changed at the end of the reporting interval. As described above, the agent 209 agent uses firewall-specified technologies (eg: ssh or tftp) to download the currently

active firewall configuration to the server 50. If the configuration downloaded at the end of one reporting interval differs from the configuration downloaded at the end of the previous reporting interval, the agent 209 reports a one, otherwise if the downloaded configuration differs from the saved firewall settings for the current threat level, the agent 209 reports a one, otherwise if any of the saved firewall settings for any threat level have changed in the reporting interval, the agent reports a one, otherwise it reports a zero. The agent 209 also reports a one-line summary of what area of the firewall configuration has changed, such as, for example, the saved settings, the downloaded configuration, and the like, with further detail as to what part of the configuration changed, such as, for example, the firewall rules, the number of active ports, address translation rule, and the like. If the configuration has changed, the alert may remain in the elevated alarm state until an authorized administrator, for example, updates the saved configuration data (firewall configuration set) on the Watch server 50 as the new master set of configuration data.

15

It should be noted that in industrial networks, for example, paths through the firewall may be opened temporarily by a short term firewall configuration change such as while short-term projects or contractors are on-site. The firewall configuration change metric allows for automatic tracking and determination of when there has been such a change. In response to a one for this metric, for example, RTAP 212 may generate an alert condition. This condition may continue to be tracked as long as the configuration is non-standard until the configuration is restored to a firewall configuration known to be safe.

20

Threat thermostat configuration change - Reports the number of saved firewall configurations corresponding to threat thermostat threat levels that have changed in the reporting interval.

25

Note that the agent 209 keeps a copy of the saved firewall configurations corresponding to the different threat levels. At the end of the reporting interval, the agent 209 compares the copies to the current firewall configurations corresponding to the different threat levels and reports if a particular pairing of firewall rule sets with an associated threat level has changed, or if there has been a change to any one of the rules sets. If there have been any changes, then after reporting, the agent 209 replaces its set of saved firewall configurations with the modified firewall configurations. For every configuration that

30

changed in a reporting period, the agent 209 also reports a one-line summary of what has changed in the configuration.

Other activity and innocuous activity filters – In log files of firewalls and other
5 systems, various metrics may be derived from a master system log file. The "other activity" metric is a count of "unusual" messages detected in the master log file during the reporting interval. A set of messages may be defined as unusual using a default and/or user specified set. For example, unusual messages to be included in this metric may be filtered using regular expressions. Any line of the log file that is not counted as some other metric is a candidate for
10 "other activity." These "other activity" candidate lines are compared to each of the saved regular expressions and discarded if there is a match with any expression. Otherwise, the candidate log entry is counted as an incident of "unusual activity". When all the log entries generated during the reporting interval are processed, the "unusual" count is reported as the value of the "other activity" metric.

15

The receiver 210 is used to interface with RTAP 212. In one embodiment, use of facilities in RTAP 212 is in accordance with a predefined interface or API (application programming interface). One of the functions of the receiver 210 is to convert the agent protocol data received into a format in accordance with a known RTAP API in order to
20 populate the database of RTAP 212. Additionally, the receiver 210 may perform agent authentication of messages received. For example, in one embodiment, a private unique key may be used by each device or processor sending a message to the Watch server 50. The receiver 210 knows these private keys and uses these to authenticate the received messages as being from one of the expected devices. The receiver 210 records the IP address reporting
25 every metric and rejects new values reported for a metric from any IP address but the last address to report legitimately to the metric. Other embodiments may use other encryption techniques and accordingly may use different techniques in authenticating received messages.

The notification server 216 and/or the web server 214 may be used in connection with
30 providing incident notification. When configuring the security or system performance metrics used by the Watch server 50, a user may specify an e-mail address or other destination for receiving different types of alert notifications that may be produced. The notification in the event of an alert may be sent, for example, to a PDA, pager, cell phone, and the like upon the occurrence of an event determined by the Watch server. Such an event

may include, for example, reported data reaching predetermined alarm or other thresholds, detection of a cyber-attack, detection of a component failure within the industrial network requiring immediate repair, and the like. Once a user has been notified upon such a device, the user may then use a web browser to gain secure access to the Watch server allowing one
5 to examine the problem and acknowledge any one or more alarms. The notification server
216 may also send a message to a direct phone connection, such as to a phone number, rather than an e-mail address.

The web server 214 may be used in connection with displaying information and/or
10 accepting input from a user in connection with any one of a variety of different tasks in this embodiment. Other embodiments may use conventional command line, Windows, client/server or other user interfaces known to those of ordinary skill in the art. In this embodiment, for example, the web server 214, through a web browser, may be used in displaying a security metric set-up page allowing for customization of security conditions,
15 and definitions used for recording and alarming. For example, a user may specify limit values or thresholds associated with different warnings or alarms. When such thresholds have been reached, a notification message may be sent to one or more specified devices or addresses. Additionally, the web server 214 in connection with a browser may be used, for example, in connection with displaying different types of information regarding a security
20 status, details of selected metrics, and the like. In connection with the use of the web server 214 and a browser, the different agents may be configured and monitored.

The web server 214 may be any one of a variety of different types of web servers known to those of ordinary skill in the art such as, for example, a TOMCAT web server. The
25 web server 214 may be used in connection with obtaining input and/or output using a GUI with a web browser for display, browsing, and the like. The web server 214 and a browser may be used for local access to appliance data as well as remote access to appliance data, such as the RTAP 212 data. The web server 214 may be used in connection with displaying pages to a console in response to a user selection, in response to a detected alert or alarm,
30 obtaining settings for different threshold and alarm levels such as may be used in connection with notifications, and the like. The web server 214 may also be used in connection with communicating information to a device such as a pager in the event of a notification when a particular designated threshold for example of an alarm level has been reached.

RTAP 212 may provide for collection, management, visualization and integration of a variety of different automated operations. Data may be collected and reported to the Watch server and stored in RTAP 212, the Watch server's database. As described elsewhere herein, RTAP 212 may be used in connection with performing security monitoring and providing for appropriate notification in accordance with different events that may be monitored. RTAP may raise alerts, for example, in the event that predetermined threshold conditions or events occur in accordance with the data store maintained by RTAP 212. One embodiment of RTAP is described in following paragraphs in more detail.

RTAP 212 may be implemented using a commercially available real time control system, such as Verano's RTAP product, or the Foxboro Intelligent Automation (IA) product or other SCADA or DCS system. In other words, a conventional control system may be used not to control a physical process, but to monitor the security activity of an industrial network and connections. RTAP 212 may also be implemented using customized software that uses a relational database. As will be appreciated by one of ordinary skill in the art, other embodiments may use other components.

In operation, each of the different agents may report data to RTAP 212 through use of the receiver 210. RTAP 212 may then store the data, process the data, and perform event detection and notification in accordance with predefined alarm levels and thresholds such as may be obtained from user selection or other defined levels. For example, as described above, a user may make selections in accordance with various alarm or alert levels using a browser with a GUI. These particular values specify a threshold that may be stored and used by RTAP 212. As RTAP 212 receives data reported from the different agents, RTAP 212 may process the data in accordance with the threshold(s) previously specified. In the event that an alarm level has been exceeded or reached, the RTAP 212 may signal an alert or alarm, and provide for a notification message to be sent on one or more devices using the web server 214 and/or notification server 206. It should be noted that the various designated location or device to which notification messages are to be sent may also be specified through the same GUI by which the threshold levels are specified.

The threat thermostat controller 218 may be used in generating a response signal in accordance with one or more types of security threat inputs. In one embodiment, the threat thermostat controller 218 may use as inputs any one or more raw or derived parameters from

the RTAP 212, other inputs that may be external to the Watch server, and the like. In one embodiment, in accordance with these various input(s), the threat thermostat controller 218 selects one or more of the firewall settings from 220 which controls access between the corporate network 12 and the industrial network 14 as well as access to the industrial network 14 from other possible connections.

In one embodiment, the threat thermostat controller 218 may use one of three different firewall settings from 220 in accordance with one or more inputs. Each of the firewall settings included in 220 may correspond to one of three different threat levels. In the event that a low threat level is detected for example the firewall rule settings corresponding to this condition may allow all traffic between the corporate network 12 and the industrial network 14 as well as other connections into the industrial network 14 to occur. In the event that a medium threat level is determined, a second different set of firewall settings may be selected from 220. These firewall settings may allow, for example, access to the industrial network 14 from one or more particular designated users or systems only within the corporate network 12. If a high threat level is determined by the threat thermostat controller 218, all traffic between the corporate network 12 and industrial network 14 may be denied as well as any other type of connection external into the industrial network 14. In effect, with a high threat level a determination, for example, an embodiment may completely isolate the industrial network 14 from any type of outside computer connection.

Actions taken in response to a threat level indicator produced by the threat thermostat controller 218 may include physically disconnecting the industrial network 14 from all other external connections, for example, in the event of a highest threat level. This may be performed by using a set of corresponding firewall rules disallowing such connections. Additionally, a physical response may be taken to ensure isolation of one or more critical networks such as, for example, disconnecting a switch or other network device from its power supply. This may be done in a manual or automated fashion such as using a control system to implement RTAP 212. Similarly, a mechanism may be used to reconnect the critical network as appropriate.

In connection with low threat level determinations, the corresponding firewall settings from 220 may allow data to be exchanged between the industrial network and less trusted networks in predefined ways and also allow authorized users on less trusted networks to

remotely log into computers on a critical network, such as the industrial network. When the threat level as generated or determined by the threat thermostat controller 218 increases, the second set of firewall rule settings from 220 may be used which provide for a more restrictive flow of communication with a critical network such as the industrial network 14. For
5 example, corporate may notify the industrial network that a particular virus is circulating on the corporate network 12, that a Homeland Security alert status has increased, and the like. Using these different inputs, the second set of rules may be selected and allow critical data only to be exchanged with less trusted networks and also disable remote log in capabilities. In the event that the highest or third level of threat is determined by the threat thermostat
10 controller 218, what may be characterized as an air gap response may be triggered leaving all less trusted networks physically disconnected until the threat(s) have been addressed, such as, for example, by installing any proper operating system and application patches.

In connection with the threat thermostat 218 in one embodiment, five threat levels
15 may be utilized. Associated with each threat level may be a text file with a series of commands that define a particular firewall configuration including firewall rule sets, what network ports are enabled and disabled, address translation rules, and the like. All of this information may be included in each of the text files associated with each of the different threat levels.

20

One of the inputs to the threat thermostat controller 218 may include, for example, a security level as published by the Homeland Security, an assessment or threat level as produced by a corporate department, and/or another source of a threat level that may be gathered from information such as available on the Internet through a government agency or
25 other type of private organization and reported by the threat agent 200. These assessments may be weighted and combined by the threat thermostat controller 218 to automatically determine a threat level causing a particular set of firewall settings to be utilized. A particular weighting factor may be associated with each of multiple inputs to 218 making the determination of a specific indicator or threat level.

30

It should be noted that the particular firewall settings included in each of the sets of 220 may include a particular set of firewall rules, address translations, addresses to and from which particular communications may or may not be allowed, intrusion detection and prevention signatures, antivirus signatures, and the like. Inputs to the threat thermostat

controller may also include, for example, one or more raw metrics as provided from RTAP, and/or one or more derived parameters based on data from RTAP and/or from other sources. It should be noted that the threat thermostat controller may generate a signal causing data to be displayed on a monitor connected to the Watch server 50 such as through a console as well
5 as to send one or more notification messages to previously designated destinations. In one embodiment, the threat thermostat control level may be displayed on a GUI. In one embodiment, an alert may be generated when there is any type of a change in a firewall rule set or threat level either in an upward or a downward threat level direction.

10 An embodiment may provide for a manual setting of a threat thermostat level used in the selection of the firewall settings, and the like. This manual setting may be in addition to, or as an alternative to, automated processing that may be performed by the threat thermostat controller 218 in determining a threat level. Additionally, an embodiment may include one or more user inputs in the automatic determination of a threat level by the threat thermostat
15 controller 218. It should be noted that in one embodiment, once the threat level has risen out of the lowest level, only human intervention may lower the thermostat or threat level.

It should also be noted that although various levels of access with respect to a critical network, such as the industrial network, have been suggested in examples herein in
20 connection with different threat levels, an embodiment may vary the particular access associated with each of the different threat levels. Although three or five threat levels and associated rule sets are described herein, an embodiment may include any number, more or less, of threat levels for use in accordance with a particular application and embodiment.

25 Additionally, in connection with the data that has been gathered by RTAP 212 such as raw data, alerts may be generated using one or more derived or calculated values in accordance with the raw data gathered by the agents.

An embodiment may implement the database portion of RTAP 212 as an object
30 oriented database. RTAP 212 may include a calculation engine and an alarm engine in one embodiment. The calculation engine may be used to perform revised data calculations using a spreadsheet-like data flow process. The alarm engine may determine an alarm function or level using a state table. Details of RTAP 212 are described elsewhere herein in more detail.

It should be noted that any one or more hardware configurations may be used in connection with the components of Figures 3 and 4. The particular hardware configuration may vary with each embodiment. For example, it may be preferred to have all the components of Figures 3 and 4 executing on a single computer system in a rack-mount arrangement to minimize the impact on the physical layout of a plant or other location being monitored. There may be instances where physical location and layout of a system being monitored require use of extra hardware in a particular configuration. For example, NIDS and ARPWatch may be monitoring the activity of 3 different switches in an industrial network using the spanning ports of each switch. Each of the 3 switches may be located in physical locations not in close proximity to one another or another computer system hosting the components of the Watch server 50. Two switches may be located in different control rooms and one switch may be located in a server room. One hardware configuration is to have the computer system upon which the Watch server components execute monitor the one switch in the server room. Two additional processors may be used in which each processor hosts agents monitoring execution of one of the remaining two switches. The two additional processors are each located in physical proximity near a switch being monitored in the control rooms. The two additional processors are capable of supporting execution of the agents (such as the NIDS agent 204 and ARPWatch Agent 206) and any software (such as NIDS 204a, ARPwatch 206a) used by the agents. These processors are connected to, and communicate with, the computer system upon which the Watch server components execute. As will be appreciated by those of ordinary skill in the art, the hardware and/or software configurations used may vary in accordance with each embodiment and particular criteria thereof.

In one embodiment, it should be noted that the receiver 210 of the Watch server 50 may track the last time a report was received by each agent (class 1 and class 2). In the event that the component 210 determines that an agent has not reported to the receiver 210 within some predetermined time period, such as within 150% of its expected periodic reporting interval, an alert is raised by sending a notification to one of the notification devices. Such an alert may indicate failure of an agent and/or machine and/or tampering with the watch system and/or with agents. Alerts may also be raised if agents report too frequently, indicating that someone may be trying to mask an attack or otherwise interfere with agent operation. Alerts may also be raised if agent reports are incorrectly authenticated, for example, if they are incorrectly encrypted, have an incorrect checksum, contain an incorrect timestamp or

sequence number, are from an incorrect IP address, are of incorrect size, or are flagged as being destined for an IP address other than the address on which the receiver 210 is listening.

It should be noted that components 202, 203 and 209 may preferably send encrypted communications where possible to other components besides the receiver 210. Whether encryption is used may vary with the functionality of the components communicating. An embodiment may use, for example, V3.0 or greater of the SNMP protocol with the components 202 and 203 in order to obtain support for encryption. Component 209 may also use encryption when communicating with the firewall.

10

Referring now to Figure 4A, shown is an example 400 of an embodiment of a threat thermostat controller 218 in more detail. In particular, the example 400 illustrates in further detail the one or more inputs that may be used in connection with a threat thermostat controller 218 as described previously in connection with the Watch server 50 of Figure 4.

15 An embodiment of the threat thermostat controller 218 may automatically determine a firewall rule set and threat indicator 410 in accordance with one or more inputs 402, 406 and/or 408 and 220. Inputs 402, 404, 406 and 408 may be characterized as selection input which provides for selection of one of the firewall settings from 220. As an output, the threat thermostat controller 218 may automatically send the selected firewall settings from 220 and
20 a threat indicator level as a signal or signals 410. Inputs 402 may come from external data sources with respect to the industrial network 14. The external data may include, for example, an indicator from a corporate network, one or more inputs from an internet site such as in connection with a Homeland Security alert, a threat indicator generated by another commercial or private vendor, and the like. This external data may come from network
25 connections, or other type of remote log in connections with respect to the industrial network 14. Other types of input may include one or more RTAP inputs 404. The RTAP inputs 404 may be raw data inputs as gathered by agents and stored within the RTAP 212 database, particular threshold levels, and the like. RTAP inputs 404 may also include a resultant value or indicator that is generated by processing performed by RTAP in accordance with one or
30 more of RTAP data values. An RTAP indicator included as an RTAP input 404 to the threat thermostat controller 218 may be, for example, an indicator as to whether a particular threshold level for one or more metrics is exceeded. The input to the threat thermostat controller 218 may also include one or more derived parameters 406. The derived parameters 406 may be based on one or more raw data values as gathered by the agents and stored in

RTAP. These derived values may be stored within RTAP or determined by another source or module. Another input to threat thermostat controller 218 may be one or more manual inputs 408. The manual input or inputs 408 may include, for example, one or more values that have been selectively input by an operator such as through GUI or configuration file.

- 5 These values may include a metric that may be manually input rather than being received from an external source in an automated fashion.

Although the various inputs described and shown in 400 have been illustrated for use with a threat thermostat controller 218 in one embodiment, it should be noted that any one or
10 more of these as well as different inputs may be used in connection with the threat thermostat controller to produce an output threat indicator. The outputs of the threat thermostat controller 218 include a firewall rule set and threat indicator 410. The firewall rule set and other settings may be communicated, for example, to a firewall as a new set of rules to be used for subsequent communications and controlling access to one or more critical networks.
15 In one embodiment, a new set of firewall rules may be remotely loaded from the Watch server location 220 to the firewall using SSH (described elsewhere herein) and/or any of a variety of secure communications mechanisms known to those of ordinary skill in the art such as, for example, IPSEC, HTTPS, SSL, and the like.

20 The threat indicator that may be produced by a threat thermostat controller 218 may also serve as an input to RTAP 212 and may be used, for example, in connection with generating one or more notifications through use of the web server and/or notification server as described elsewhere herein when a particular threat indicator level has increased or decreased, a firewall rule setting selection has been modified and the like. Additionally, data
25 recording for the threat level, date, time, and the like may be recorded in RTAP 212. The threat thermostat controller 218 may also produce an output signal 411 used in connection with automatically controlling the operation of a connecting/disconnecting the industrial network from the corporate network in accordance with the threat indicator. For example, the signal 411 may be input to RTAP, a control system, switch or other hardware and/or software
30 used to control the power supply enabling connection between the industrial network and corporate network as described elsewhere herein.

It should be noted that in one embodiment, only manual inputs may be used. A single manual input may be used in one embodiment, for example, in selection of a threat indicator

causing the threat thermostat controller 218 to make a selection of a particular firewall setting. Another embodiment may provide for use of a combination of automated and/or manual techniques where the automated technique may be used to produce a threat indicator unless a manual input is specified. In other words, rather than weight one or more manual
5 inputs in connection with one or more other inputs in an automated fashion, the manual input or inputs may serve as an override of all of the other inputs in connection with selecting a particular firewall rule set from 220 and generating a threat indicator. Such a manual override may be provided as an option in connection with a mode setting of a threat thermostat controller 218. If the override setting which may be a boolean value is set to on or
10 true, the manual input will act as an override for all other inputs and an automated technique for producing a threat indicator. In the event that override is set to off, the manual input may not be considered at all, or may also be considered along with other inputs in connection with an automated technique used by the threat thermostat controller.

15 Referring now to Figure 5, shown is an example 300 of the different types of agents of the first class of agent that may be utilized in an embodiment of the industrial network 14. It should be noted that the agents 300 may be included and executed on each of the computer systems in the industrial network 14 as indicated by the agents 132a-132d. In other words, the different agent types included in 300 are those types of agents that may execute on a
20 system and report information about that system to the Watch server 50. It should be noted that although an embodiment may include the particular agent types of 300, an embodiment may include different types of agents and a different number of agents than as described herein in accordance with the particular application and embodiment and may vary for each computer system included in the industrial network 14.

25 Included in 300 is a master agent 302, a critical file monitoring agent 304, a log agent 306, a hardware and operating system agent 308, a password age agent 310, and an application specific agent 312. In one embodiment, the master agent 302 is responsible for control of the other agents included in the computer system. For example, the master agent
30 302 is responsible for starting and monitoring each of the other agents and to ensure that the other agents are executing. In the event that the master agent 302 detects that one of the other agents is not executing, the master agent 302 is responsible for restarting that particular agent. The master agent 302 may also perform other tasks, such as, for example scheduling different agents to run at different periods of time, and the like.

The critical file monitoring agent 304 may be used in connection with monitoring specified data files. Such data files that may be monitored by agent 304 may include, for example, operating system files, executable files, database files, or other particular data file
5 that may be of importance in connection with a particular application being performed within the industrial network 14. For example, the agent 304 may monitor one or more specified data and/or executable files. The agent 304 may detect particular file operations such as file deletion, creation, modification, and changes to permission, check sum errors, and the like. Agent 304, and others, gather information and may report this information at various time
10 intervals or in accordance with particular events to the Watch server 50.

The log agent 306 may be used in monitoring a system log file for a particular computer system. The log monitoring agent 306 may look for particular strings in connection with system activity such as, for example, "BOOT", or other strings in connection with
15 events that might occur within the computer system. The log agent 306 searches the log file for predetermined strings of interest, and may store in memory the string found as well as one or more corresponding metrics such as, for example, the number of occurrences of a string. For example, the log agent 306 may count occurrences of a BOOT string and report the count in a single message which may be sent to the Watch server or appliance. The sending of a
20 single communication to the Watch server may be performed as an alternative, for example, to sending a message reporting the occurrence of each string or event. Techniques such as these provide for efficient and bounded use of resources within the industrial network 14 resulting in reduced bandwidth and CPU and memory usage consumed by the agents.

25 In one embodiment, the agent 306 may report the following metrics at periodic intervals:

Login failures - Report the number of "failed login" messages in the system log in the reporting interval. The format of these messages may vary in accordance with software
30 platform, such as operating system and version and login server, such as for example, ssh, telnet, rlogin, and the like. Reported with this metric may be the names of the top three accounts reporting login failures in a reporting interval, and what percentage of the total number of failure reports is associated with each of these three accounts.

Password change failures - Report the number of "failed password change attempt" messages in the system log in the reporting interval. Some of these failures may be the result of an authorized user trying to change his/her own password. This metric may indicate false positives such as these in addition to indicating a brute force password attack by an unauthorized user. Reported with this metric may be the top three accounts reporting failed password attempts in a reporting interval and a corresponding percentage of failed attempts associated with each account.

Network ARPwatch – Using the modified version of ARPwatch described elsewhere herein, this metric reports the number of unapproved IP/device address bindings currently on the network. The ARPwatch metric also reports the first three ARPwatch log messages detected in each reporting interval, and if the metric is non-zero in an interval, reports the top three IP addresses and device addresses responsible for those messages.

Host IDS audit violations - Report the total number of IDS and failure audit messages detected in the reporting interval. When the IDS classifies the messages, report a count for each classification -- eg: critical, warning. When multiple audit systems are running on a machine, report each system's output independently. For example, on SELinux systems, the SELinux system reports authorization failures for all failed accesses to protected resources. Such authorization failures are reported as a separate SELinux authorization failure metric. Additionally, report the first three log messages detected in each classification in each reporting interval and a count of the messages not reported. This metric may be extended to report a summary of all the messages detected for each kind of message in the reporting interval as well, including, where process information is available, the top three processes responsible for the messages and the percentage of total messages associated with each process and/or, where file information is available, the top three files that are reported as the targets of audited file manipulations, and what percentage of all the IDS messages each file was mentioned in.

Host IDS antivirus alerts - for host IDS systems that monitor and report on viruses detected on control system hardware. Note that while some computers in the industrial network may not execute antivirus software for performance, compatibility, or other reasons, other computers within the industrial network may utilize antivirus software. An

embodiment of this agent may also report the first three such messages detected in a reporting interval.

The agent 306 may also include metrics related to the following:

5 Web page authentication failures, web page permission violations, total web page failures, firewall violations (described herein in more detail), and tape backup failures. This last metric may be useful in connection with notifying administrators, for example, in the event that security history or other information is no longer being backed up on a tape or other backup device.

10

Windows event logs - Report the number of Windows "Error," "Warning," "Informational," "Success Audit," and "Failure Audit," log entries detected in the reporting interval. The agent also reports the first 256 characters of text for the first three such log entries discovered for every type of log in every reporting interval.

15

The hardware and operating system agent 308 may be used in connection with gathering and reporting information in connection with the operating system and hardware. For example, through execution of a status commands or others that may be available in an
20 embodiment, information may be produced using one or more operating system utilities or calls. As an output of the command, data may be produced which is parsed by the hardware operating system agent 308 for the particular statistics or metrics of interest. In one embodiment the hardware operating system agent 308 may use one or more status commands, for example, to obtain information about CPU load, disk space, memory usage,
25 uptime, when the last reboot has occurred, hardware and software information such as related to version numbers, and the like. Similar to the behavior of other agents, the hardware operating system agent 308 may parse the output of different status commands and send a single report to the Watch server at different points in time rather than report multiple messages to the Watch server. For example, the agent 308 may combine information from
30 multiple status commands and send a single communication to the Watch server or appliance 50 at particular time periods or in accordance with particular events.

In one embodiment, the following metrics may be reported by agent 308 at periodic intervals:

Counts of interprocess communications resources used - System V message counts and segment counts for any message queues used by the control system. If the system is becoming congested such as, for example, messages requests are backing up because the system does not have the resources to process them fast enough, or because of a failure of some component of the system that is supposed to be processing them, an alarm may be raised when any message queue's message or segment count exceeds a preset limit.

Operating System type - This is a constant value reported to assist in auto-configuration of the Watch server. For example, when an administrator is searching for a particular machine when using the GUI on the Watch server, that person may become confused by the many machines whose information is available via the GUI. Operating system type, version number, and the like, may be used in identifying a particular machine through the GUI.

"Uptime" - How long it has been since the machine running the agent 308 has been rebooted. This is sometimes useful in post-mortem analysis. For example, a number of anomalous alerts are generated by the Watch server in connection with resource and system performance metrics. It may also be observed that a particular computer in the industrial network was also being rebooted during this time period. It may be determined that the abnormal resource usage, for example, was due to the machine reboot or restart activity which may differ from the usage when the computer is in steady state. Thus, it may be useful to determine why the machine rebooted and restarted rather than investigating the individually raised resource-generated alerts.

User count and root user count -- How many login sessions are active on the machine, and how many of those belong to "root" or other account with an elevated level of permissions. Each metric reports not just the count of logins, but for the first N logins, such as 20 for example, report where the user is logged in from, such as the machine's console, or the IP address, or host name of some other machine the user is logged in from. In one embodiment, the foregoing metrics may be determined on a Unix-based system. Other systems may provide similar ways to obtain the same or equivalent information. Note that

below as known to those of ordinary skill in the art, "tty" is a UNIX-specific reference to a UNIX device that manages RS232 terminal connections:

- 1) Determine who is logged into the system. In one embodiment, this may be determined by examining the output of the "who" command. (i.e., On HP-UX, "who -uR".
5 On Tru64 use "who -M").
- 2) From the output of 1), extract the user name, the source machine if any, and the "tty" or other login user identifier.
- 3) Determine which user(s) are currently using system resources. This may be determined, for example, using the Unix-based "ps" command.
- 10 4) Since "Who" command output may be sometimes stale, the number of current users may be determined in a Unix-based system by remove from the "who" list (from 1) any user whose identifier is not associated with any active process identified in 3).
- 5) When reporting root user counts, an embodiment may also search the /etc/passwd file for each user "who" reports. Any user with a numeric user ID of 0 is in fact running as
15 root and is reported as such. Since a single user logged in on the console may have many terminal windows open and "who" reports each as a separate login, it may be desirable to report the foregoing as a single user.
- 6) All of the login sessions whose source "who" reported as ":0" as associated with the console display device, may be combined into a single entry.
- 20 7) Count and report all root users whose "source" is reported by the "who" command as a machine other than the local machine.
- 8) Examine the "tty" of all potential root users. Increment the root count for, and record the source for, every unique standard input device:
 - "ptyN" - a pseudo tty used by X11 windows
 - 25 - "ttyN" - a Linux pseudo console created using the keystroke combination:
<Ctrl><Alt><Fn>
 - serial port logins -- "ttySN" on Linux or "ttyN" on HP-UX,
Solaris and Tru64,
 - "console" for console device logins and ":0" as well for
30 console logins on Tru64.
- 9) Determine the owner of any of the standard window manager processes that might be running on the machine. If a root console login has not already been identified, and any of these processes is running as root, that counts as a root console login.

(10) Determine and count the number of rsh (remote shell) users. Use the Unix "ps" command to identify remote shell daemon processes (eg: remshd) and their child processes. Count and report as users the user identifiers associated with all such child processes.

5 CPU Load – On a Windows-based system, report the %CPU used in the reporting interval. On UNIX systems, report the CPU load average for the interval. The load average is the average number of processes competing for the CPU. Note that on a Windows system, if some process is spinning in a tight loop, the machine reports 100% CPU usage. This metric may be characterized as a rather blunt reporting instrument since it gives no idea of how
10 much "other" work the machine is accomplishing over and above the process that is currently looping. On a UNIX system in a similar state, a report may indicate, for example, a load average of 1.5. The process in a loop accounts for 1 load unit (it was always wanting the CPU). The additional .5 indicates that over and above the executing process, 1/2 of the time some other process wanted the CPU for execution purposes. Additionally, the top three
15 process names consuming CPU in a reporting interval may be reported along with what portion (such as a fraction or percentage) of the CPU used in the interval was used by each of these top three processes.

%Disk space - for every disk partition or volume, report the % used for each. It
20 should be noted that RTAP may be configured to alert when the percentage of this metric is not in accordance with an absolute threshold.

%Swap space - Report the %used for swap space. An alert may be generated when that metric increases to an absolute threshold. Additionally, in each reporting interval, the
25 top three processes using memory and/or swap space may be reported and what % of the available resource each of these processes consumes.

%Memory Used - Report the fraction of physical memory used. It should be noted that some operating systems report swap and memory as one metric and so the %swap space
30 metric may be the metric reported. In that case %swap space combines memory used and swap used into one total and reports the fraction of that total in use.

Hardware (such as LM) Sensors and Disk Status (such as SMART) – Report metrics from sensors on different computer components, such as the CPU. The values

reported on different platforms may differ in accordance with the different hardware and/or software and/or monitoring circuits. Examples of the metrics that may be reported by one or more sensors may include CPU temperature, case temperature, fan speed for any of several fans, power supply working/failed status in machines with multiple power supplies, soft hard disk failures, etc. These sensors can provide early warning of impending hardware failure of critical industrial control computer components. Note that "SMART" refers to the Self Monitoring Analysis and Reporting Technology as described, for example, at <http://smartmontools.sourceforge.net>. "LM" refers to, for example, "LM-78" and "LM-75" hardware monitoring functionality that is standard in some vendor chipsets and is described, for example, at <http://secure.netroedge.com/~lm78>.

Network Traffic - Report total incoming and total outgoing traffic for every network interface on the computer. An abnormal increase in such traffic can mean either that the machine is under attack, that the machine has been compromised and is being used to attack another machine, that the machine has been compromised and is being used for some purpose other than it was intended for, or that there has been some sort of malfunction of the control system on the machine.

Open listen sockets - Reports a count of open listen sockets on the computer. Listen sockets are almost always associated with long-running server processes, and the count of such processes and sockets almost always changes very predictably on control system computers. For example, the count of such sockets may fall within a very small range having little variance. When the count moves out of this range, an alert may be generated. When the listen socket count falls out of the bottom of the range, it may be an indication that some server component of the operating system or of the control system itself has failed, causing the associated listen socket to close. When the listen socket count rises out of the top of the normal range, it may indicate that some server component has been added to the system. This may be, for example, a new control system component, a debugging or other tool that it may or may not be wise to deploy on a production control system, or a component installed by an intruder or authorized user seeking to gain unauthorized access to the system in the future.

The password age agent 310 may be used in monitoring the status of different passwords and accounts. Such activity may include password aging. The particular metrics that may be gathered by the agent 310 may relate to the security of the computer system being monitored as a security measure to detect hackers trying to log in to the system.

5

In connection with one embodiment, the agent 310 may report the following at periodic intervals:

Max password age - Once an hour, a maximum password age metric may be reported that measures the age of every password on every account on the system. Included in this reported metric may be the age of the oldest password on the system, and, for each of the first 100 users on the system, the user name and age of that user's password. This metric may be useful when raising alerts when passwords become too old. Some sites have a policy of changing passwords regularly, eg: at minimum every 180 days. Because user names and password ages can serve to help identify vulnerable accounts on a computer system, these information may be separately encrypted when the primary communication channel for the agent report is not already encrypted.

The application specific agent 312 may be customized, for example, to monitor specific application parameters that may vary with a particular embodiment and/or application executing in the industrial network 14. Generally the application specific agent 312 is an agent that may be built and specified by a particular operator of the industrial network.

In one embodiment, the agent 312 may report information about the particular application at periodic intervals including any of the following metrics:

Abnormal process terminations- A count of control system processes that have terminated unexpectedly or with an improper exit/termination status. The names of a number of such processes which failed in the reporting period may also be reported. These names occupy a fixed maximum size /communications budget, and mean that the next level of detail is available in the Web GUI. It should be noted that this metric may include reporting information regarding the first processes in time to terminate unexpectedly rather than the last processes in time to terminate unexpectedly since the last processes may have terminated as a result of the earliest failed processes. For example, a later terminated process

(unexpectedly terminated) may have terminated execution as a result of being unable to communicate with an earlier terminated process (unexpectedly terminated). Also included with this metric when non-zero are the names or other identifiers of the processes that failed most frequently, and what percentage of the total number of failures is associated with each.

5

Installed software –Reports a count of software packages installed, uninstalled and/or updated in the last reporting period. This information comes from whatever sources are available on the computer such as, for example, one or more log files that are appended to when software is installed, uninstalled or updated, and one or more system databases of installed software that are updated when software is installed, uninstalled or updated.

10

Another type of application specific agent 312 may be a control system software agent with knowledge of the expected behavior of a specific control system such as, for example, a Foxboro IA system, a Wonderware InSQL server, or a Verano RTAP server (such as the RTAP component 312), and the like. Such agents may report some metrics already described herein such as:

15

Installed software - When the control system software itself has new components installed, or installed components updated or removed.

20

Process terminations - either terminations reported as abnormal by the control system software application itself, or processes no longer active that the agent "knows" should be active because a correctly functioning Foxboro or RTAP or other system should have such processes active to run correctly.

25

Open listen sockets – The number of open listen sockets. An embodiment may report and monitor the number of open listen sockets that are managed by the control system and are expected to be open for the correct operation of the control system. Note that the number of open listen sockets refers to an embodiment that may use, for example, UDP or TCP. This metric may be more generally characterized as the number of communication endpoints or communication channels open on a server machine upon which the server is listening for client requests.

30

Control system shutdown - Reports all controlled and uncontrolled shutdowns of the control systems application. In the case of an unexpected shutdown of the entire computer

running the control system, where there may not have been an opportunity to report the shutdown before the computer itself shuts down, the shutdown may be reported when the computer and the agent restart.

5

An embodiment may also include other types of agents not illustrated in 300. Some of these may be optionally included in accordance with the amount of resources available as well as the particular task being performed by a system being monitored. For example, an embodiment may also include a type of agent of the first class reporting on file system integrity characteristics, such as changes in file checksum values, permissions, types, and the like. Execution of such an agent may be too CPU and/or disk I/O intensive to scan entire filesystems, or to determine checksums for large number of files in a system, so this agent may be selectively included in particular embodiments. The file system integrity agent may report the following metric at periodic intervals:

15

Integrity failures – For IDS that monitor and report on file integrity, report the total number of integrity failure messages detected in the reporting interval. For systems that report different kinds of, or priorities of, integrity failures, an embodiment may report the total integrity failures in each classification. For each classification of failure, also report the first three integrity failure log messages or events detected in the reporting interval and a count of the remaining messages not reported. Integrity failures may be discovered by the agent itself, or the agent may monitor the results of conventional integrity checking tools such as Tripwire or may invoke installation integrity checking tools such as fverify. For more information on Tripwire, see <http://www.tripwire.org>. For more information on fverify, see:

25

http://h30097.www3.hp.com/docs/base_doc/DOCUMENTATION/V51_HTML/MAN/INDEXES/INDEX_F.HTM

In addition, to provide a degree of filesystem integrity checking on a resource-constrained computer system, an embodiment may pace filesystem integrity checking such that only a small number of files are checked in a given reporting interval to reduce and/or limit the CPU and disk I/O impact of such checking. Such an embodiment may, for example, also classify files into two or more levels of importance, and may scan some number of files from each level of importance in each reporting interval. This way, when the lowest level of

30

importance has many more files in it than the highest level of importance, more important files will tend to be checked more frequently than less important files, while still controlling the resource impact of the scanning.

5

It should be noted that the particular metrics reported by each agent as well as other particular agent characteristics, such as periodic reporting intervals, may vary with each embodiment. Although the description of the various metrics herein may be made with reference to elements particular to this embodiment, such as a the UNIX-based operating systems, it will be appreciated by one of ordinary skill in the art that equivalents may be used in connection with other elements, such as other operating systems that may be used in an embodiment.

It should be noted that an embodiment may issue an alert using RTAP when any one of more of the metrics, reported as a count or value rather than a boolean, described herein is not in accordance with an absolute threshold.

Data flows from each of the different types of agents of the first class 132a-132d on computer systems in the industrial network to the Watch server 50. In this embodiment, the Watch server may be characterized as an appliance that is a passive listening device where data flow is into the appliance. Within the Watch server, a process may be executed which expects a periodic report from the agents 132a-132d as well as a report from the agents 132a-132d when a particular event occurs. If there is no report from a particular agent 132a-132d within a predefined time period, the Watch appliance may detect this and consider the agent on a particular system as down or unavailable. When a particular system, or the particular agent(s) thereon, in the industrial network 14 are detected or determined to be unavailable or offline, as may be determined by the RTAP 212 of Figure 4, an alarm or alert may be raised. Raising an alarm or alert may cause output to be displayed, for example, on a console of a notification device.

30

Collectively the different types of agents provide for gathering data that relates to the health, performance, and security of an industrial network. This information is reported to the Watch appliance or server 50 that uses the health, performance and security data in connection with security threat monitoring, detection, and determination.

Each of the agents may open up its own communication connection, such as a socket, to send data to the Watch server. An embodiment may alternatively use a different design and interaction of the different types of agents than as illustrated in 300. In one embodiment
5 using the example 300, each agent may be implemented as a separate process. In an alternative embodiment, a single process may be used performing the processing of all the different types of agents illustrated in Figure 5 and all data may be communicated to the Watch server over a single communication connection maintained by this single process. An embodiment may use another configuration to perform the necessary tasks for data gathering
10 described herein.

It should be noted that an embodiment may include the master agent with any one or more of the different types of agents for use with a system being monitored. Using the model of Figure 5, the master agent is necessary to control the operation of one or more of the other
15 types of the first class.

Referring now to Figure 6, shown is an example 350 of the architecture of each of the agents of the first and second classes described herein. It should be noted that the architecture 350 may vary in a particular embodiment or with a particular class of agent. The
20 particular illustration of Figure 6 is only an example and should not be construed as a limitation of the techniques described herein.

An agent data source 352 is input to an input data parser 354. The particular data source 352 may vary in accordance with the particular type of agent. For example, in the
25 event that the agent is a log file agent, the agent data source may be a system log file. In the event that the agent is the hardware operating system agent, the agent data source may be the output of one or more status commands. The one or more data sources 352 are input to the data parser 354 for parsing. The particular tokens which are parsed by 354 may be passed to the pattern matching module 356 or the metric aggregator and analyzer 358. It should be
30 noted that there are times when the parsed data may be included in a message and does not require use of pattern matching. The pattern matching module 356 searches the data stream produced by 354 for those one or more strings or items of interest. The pattern matching module 356 may report any matches to the metric aggregator and analyzer 358. The component 358 keeps track of summary of the different strings as well as counts of each

particular string that have occurred over a time period as well as performs processing in connection with immediate notification. As described elsewhere herein, an agent may report data to the Watch server 50 at periodic reporting intervals. Additionally, the agent may also report certain events upon immediate detection by the agent. This is described elsewhere
5 herein in more detail. The metric aggregator and analyzer 358 also controls the flow of data between the different components and is also responsible for compressing the messages to minimize the bandwidth function.

Once the metric aggregator and analyzer 358 has determined that a message is to be
10 reported to the Watch server 50, such as for immediate reporting or periodic reporting of aggregated data over a predetermined time period, the metric aggregator and analyzer 358 may send data to the XML data rendering module 362 to form the message. The XML data rendering module 362 puts the information to be sent to the Watch server 50 in the form of an XML message in this particular embodiment. Subsequently, component 362 communicates
15 this XML message to the message authentication and encryption module 360 for encryption prior to sending the XML message to the Watch server or appliance.

In connection with the message authentication and encryption module 360 of Figure 6, it should be noted that any one of a variety of different types of encryption techniques may
20 be used. In one embodiment, a timestamp and agent host name or identifier may be included in a message body or text. The authentication processing on the Watch server 50, such as may be performed by the receiver 210, may require that the timestamp values always increase and otherwise reject duplicate or out of date messages. Additionally, an encryption technique may be used which utilizes a key, such as a shared secret key, and the entire message may be
25 encrypted with this key. The shared secret key provides the message authentication information. An embodiment may also use other well-known techniques such as, for example, the MD5 cryptographic checksum and encrypt the checksum of the entire message. The authentication processing performed within the Watch server 50 may vary in accordance with the techniques used by the agents. In one embodiment, an agent may encrypt the
30 checksum of the message and not the message itself. Alternatively, in an embodiment in which a checksum determination of a message is not available, the agent may encrypt the message.

The different types of data reported by the types of first class of agents illustrated in Figure 5 relate to the health, performance, and security of a critical network, such as the industrial network 14. This data as reported to the Watch server 50 enables the Watch server 50 to generate signals or alerts in accordance with the health, performance, and security of the critical network. In particular, the RTAP 212 of the Watch server may be characterized as a global aggregator and monitor of the different types of data reported to a central point, the Watch server 50.

The agents 132a-132d (of the first class described elsewhere herein) as well as the second class of agents that communicate data to the Watch server 50 may be characterized as distributed monitoring agents. In one embodiment, these agents may raise alerts or send reports to the Watch server in summary format in accordance with predefined time periods, or in accordance with the detection of certain events in order to conserve the bandwidth within the industrial network 14. In existing systems, agents may report every occurrence of a particular event, such as a suspicious activity, and may result in the consumption of excessive bandwidth when a system is under attack. An agent, such as one of the first class executing in the industrial network 14, may report attack summaries at fixed intervals to conserve network resources. For example, an agent 132a-132d may report the occurrence of a first suspicious event and then report a summary at the end of a reporting period. In other words, reports may be sent from an agent at predetermined time intervals. Additionally, the agents may send messages upon the detection or occurrence of certain conditions or events.

The agents (first class and second class when communicating with the receiver 210) included in an embodiment may be designed in accordance with particular criteria. As described in connection with the above embodiment, the agents are "one-way" communication agents at the application level for increased security so that operation of an agent, such as on a component in the industrial network 14, minimizes added vulnerability to a network attack. The agents communicate with the Watch server by opening a TCP connection, sending an XML document over the connection, and closing the connection after the XML communication is sent. The agents do not read commands or requests for information from this connection from the Watch server.

It should be noted that a computer hosting an agent does receive and process messages from the Watch server. However, the processing performed by such a host to an

agent are limited to processing steps at lower network levels. For example, in an embodiment using the XML messages described herein, this processing may include the TCP-level connection setup, teardown and data acknowledgement messages performed at levels lower than the application level. Any vulnerabilities existing at these lower levels exist independent of whether the agents described herein are utilized. In other words, use of the agents described herein does not introduce any additional vulnerabilities into monitored and networked control system equipment.

The agents, in particular the first class of agents described herein, may be characterized as bandwidth limited agents designed to consume a fixed portion of available network resources. Conventional security agents tend to report every anomalous event upon the occurrence of the event consuming potentially unbounded communication resources under denial-of-service attacks. Conventional agents may regard every security event as important and make a best-effort attempt to communicate every such event to their management console. Agents that consume an excessive amount of a limited network communications resource risk causing the entire system to malfunction, triggering safety relays and other mechanisms to initiate an emergency shutdown of the industrial process.

In contrast, the agents described herein are designed to transmit small fixed-size messages at fixed intervals, thus consuming a bounded portion of available communications resources, even under denial-of-service attack conditions. The first class of agents herein gather information, produce condition reports and event summaries, and report those conditions and summaries at fixed intervals. The reports may include: statistics in accordance with the particular first class agent type, such as, for example, resource usage statistics like % CPU used, CPU load factors, % memory used, % file system used, I/O bus utilization, network bandwidth utilization, number of logged in users, and the like. The report may also identify the top N, such as, for example, two or three, consumers of one or more resources. The consumers may be identified by, for example, process names, directories, source IP addresses, and the like, and may identify, when appropriate, what portion of a resource each is consuming. The report may also include other information that may vary with agent type and class such as, for example, counts of log messages and other events, like login failures, network intrusion attempts, firewall violations, and the like detected in the reporting interval that match some criterion or search expression; representative samples of the complete event description or log message for the events counted in the reporting interval, and a short

statistical summary of the events, such as what host or IP address hosted the most attacks and what percentage of attacks overall were hosted by a particular computer, which host was most attacked and what percentage of attacks were targeted at a particular host, what user account was most used in connection with launching an attack and what portion of attacks are targeted at a particular user account. In one embodiment, a reporting threshold for an agent may be specified indicating a maximum amount of data the agent is allowed to transmit during one of the reporting intervals. The reporting threshold may specify, for example, a number of bytes that is equal to or greater than a size of a summary report sent at the reporting interval. For a given reporting interval or period, an agent's reporting budget may be the reporting threshold. The agent may also report one or more other messages as needed besides the summary in accordance with the specified reporting threshold. Prior to sending a report, the agent makes a determination as to whether it is allowed to send a next report by determining if the total amount of data reported by an agent would exceed the reporting threshold by sending the next report. If the threshold is exceeded, the agent does not send the report.

The agents described herein, in particular the first class of agents, are also designed to limit the amount of processing time and storage (disk and memory) consumed. Conventional intrusion detection and performance monitoring agents are known for the negative performance and storage impact on the system being monitored. SNMP components, for example, have been known to consume all of the memory on a host of the SNMP component. AntiVirus scanners may impair the performance of the machines they are monitoring by up to 30-40% depending on the particular processor. The foregoing may not be acceptable in connection with legacy systems, such as may be encountered in industrial networks. Industrial control applications respond to inputs from the industrial process within very short time windows due their real-time processing nature. Furthermore, such systems render information about the process to operators in a timely manner. Anti-virus solutions, for example, may not generally be deployed on control system hardware, such as in the industrial network 14 described herein, because the anti-virus processing may impair the operation of a system sometimes causing system failure. The agents described herein are designed to minimize the resource impact on the system being monitored. Expensive metrics, like filesystem checksums, are gathered over a very long period of time, or for only the most security-critical components so that the impact of the data gathering on the system being

monitored is within a small fixed budget. For example, in one embodiment, 1-3% of all of a machine's resources can be allotted to the monitoring agents executing thereon.

An embodiment of RTAP 212 may use an event reporting technique referred to as the exponentially decreasing attack reporting. In some embodiments, when a metric goes into an alert state and a user has requested notification of the alert, an e-mail or other notification message is sent indicating that a particular metric has gone into alert state. If the "current value" of the metric, for example, returns to the "normal" range, a notification message may also be sent regarding this transition. The foregoing may cause a large burst of notification messages to be sent to an administrator and important information may be overlooked due to the volume of notification messages received in a short time interval. For example, in the event that the alert or alarm condition exists for some time period, an initial set of notification messages may be sent when an attacker succeeds in compromising one machine. Reported by agents on that machine in the industrial network may be high memory and network usage as a consequence of being compromised and an illicit web server started. When usage levels return to normal, another set of notification messages may be sent. However, suppose that the memory and network alert conditions do not return to normal. The foregoing conditions may be overlooked in the burst of notification messages. An administrator with access to a web browser could log into the Watch web user interface and see that metrics on a particular host were still in an alert state, but email notification may also be used by administrators who do not have such access. Accordingly, an embodiment may use the "exponentially decreasing notifications" technique which reports the initial alert. Instead of staying silent until the next alert state change, additional alert notices are sent while the metric stays in an alert state. The frequency with which these additional alert notices are sent may vary in accordance with the length of time an alarm condition or state persists. In an embodiment, this frequency may decrease exponentially, or approximately exponentially. In one embodiment, the following alert or alarm notification messages may be sent upon the first detection of an alarm or alert condition, and at the end of a first defined reporting interval. At this point, there may be additional summary information that may be optionally sent to the user with the notification message. This is described in more detail herein using the enhanced email notification described elsewhere herein. Subsequently, a notification message is sent at increasing intervals while the condition persists. These time intervals may be user specified as well as defined using one or more default values that may vary with an embodiment. For example, in one embodiment, an initial reporting interval of

an alarm condition may be every minute. After the end of the first reporting interval or minute, notification messages may be sent at time intervals so that a current reporting interval is approximately 10 times longer than the previous reporting time interval. In this example, if the first notification message is sent after 1 minute, the second notification message may be sent after 10 minutes and include any additional information such as may be available using the enhanced e-mail reporting described elsewhere herein. The third notification message may be sent at about 1 ½ hours later, and so on. The reporting interval may reach a maximum of, for example, 12 hours so that if an alarm or alert state persists, notification messages with enhanced reporting (such as enhanced e-mail) may be sent every 12 hours until the alert condition clears, or the user otherwise removes themselves from the notification list.

Using the foregoing notification reporting technique, persistent alert conditions that may otherwise be lost in a burst of notification messages may remind the administrator that there is a persistent problem condition, and provide the administrator with current summary information so that the administrator can see if the nature of the attack or compromise is changing over time. Using this type of exponentially decreasing attack reporting techniques, the bandwidth of the network may be more wisely utilized for the duration of the attack as well. The foregoing exponentially decreasing notification reporting may be performed by the notification server 216 of Figure 4. The alarm or alert conditions may be produced using the calculation as described elsewhere herein causing the notification server to be notified. However, the foregoing may be performed by the notification server to reduce the number of times that a notification message is sent.

Additionally, an embodiment may use different techniques in connection with when agents report to the Watch server 50. One design concern, as described elsewhere herein, is minimizing the amount of network bandwidth used for reporting due to the possible bandwidth limitation of the industrial network. In one embodiment, for one or more designated metrics in a defined reporting interval, the log agent 306 may report the first detection of a log message that causes the metric to increment as soon as the log message is detected. Subsequently, the agent does not report any additional information to the Watch server about the metric until the end of the reporting interval, when the agent 306 then reports the total count for the metric in the reporting interval. Using the foregoing, immediate notification may be achieved upon the occurrence of the metric increase and then an update

received at the next reporting interval. The foregoing immediate notification may be used with metrics determined using log files. An embodiment may also use other agent types to report other metrics that may be readily determined on an event basis such as, for example, a line being added to a log file, a file descriptor becoming readable, or a command executing.

5

An embodiment may use a combination of the foregoing immediate notification and periodic interval reporting. In an embodiment using just the periodic interval reporting by the agents to the Watch server 50, there may be an unacceptable delay in reporting alarm conditions indicating an attack or attempted attack. For example, consider an agent's reporting interval of 60 seconds. One second into that interval, the agent viewed a failed login attempt indicated by a metric and then another 10,000 such attempts in the minute. A single report is sent at the end of the minute reporting interval to the Watch server with a report metric indicating the 10,001 attempts. However, there is a delay and an administrator may expect to receive a notification of the event prior to 10,000 being detected. Using the immediate notification, the agent also reports the first occurrence of the failed login attempt when it is detected. Accordingly, the Watch server 50 may respond with an immediate notification message with the first occurrence or detection of the metric increase.

10

15

20

It should be noted that the foregoing immediate notification may be performed in accordance with user selected and/or default specified conditions. This may vary with each embodiment. In one embodiment, the metric aggregator and analyzer 358 may perform processing steps in connection with the immediate reporting and also periodic interval reporting to the Watch server 50.

25

30

Referring now to Figure 7, shown is a flowchart 450 of processing steps describing the control flow previously described in connection with 350 of Figure 6. Generally, the processing steps of 450 may be performed in an embodiment by each of the agents of the first and second classes when processing the one or more input data sources. At step 452, a determination is made as to whether input data has been received by the agent. If not, control continues to wait at step 452 until input data has been received. It should be noted that the input data may be received in accordance with the agent performing a particular task such as executing a command producing input, waiting for input on a communications channel, reading a data file, and the like, in accordance with one or more predefined criteria. The one or more predefined criteria may include performing a particular task at predefined intervals,

when a particular data file reaches a certain level of capacity in accordance with a number of operations, and the like. The particular criteria which causes the input data to be received by the agent may vary in accordance with each embodiment. At step 452 once data has been received, control proceeds to step 454 where the input data is read and then parsed at step 5 454. Once the input data stream has been parsed, control proceeds to step 455 where a determination is made as to whether pattern matching is needed. If not, control proceeds to step 460. It should be noted that pattern matching may not be needed, for example, if no selective filtering of the parsed input source is needed when all metrics from a source are reported. Otherwise, control proceeds to step 456 where pattern matching is performed. At 10 step 458, a determination is made as to whether the input data has any one or more matches in accordance with predefined string values indicating events of interest. If not, no strings of interest are located and control returns to step 452. Otherwise, control proceeds to step 460 where data may be recorded for the one or more metrics derived from the parsed input source. For example, a particular metric and its value may be stored and recorded, for 15 example, in the memory of a computer system upon which the agent is executing. At step 462, a determination is made as to whether any messages reporting data to the Watch server are to be sent. As described herein, an agent may report data at periodic intervals when summary information is reported. An embodiment may also provide for immediate reporting the first time a designated metric increases in value such as may be the case, for example, at 20 the beginning of an attack or an attempted attack. This processing may be performed, for example, by the metric aggregator and analyzer 358. If no message is to be sent to the Watch server 50, control proceeds to step 452 to obtain additional data. Otherwise, control proceeds to step 464 where the message to be sent to the Watch server is prepared in accordance with a message protocol and/or encryption technique that may be used in an 25 embodiment. As described herein, for example, a message being sent to the Watch server is sent in an XML or other format and an encryption technique described elsewhere herein may also be used. Control then proceeds to step 466 where the message is sent to the Watch server. Control then returns to step 452 to wait for additional input data to be received by the agent.

30

Referring now to Figure 8, shown is an example of an embodiment 212 of components that may be included in RTAP. It should be noted that a version of RTAP is commercially available from Verano Corporation as described, for example, at the website www.verano.com. Included in Figure 5 is RTAP scheduler 502, an alarm server 504, a Java

server 506, and a database server 508. The database server 508 in this embodiment includes a calculation engine 510. The database server 508 may output data, such as the metrics gathered by the agents described herein, to one or more devices 514 which may be stored, for example, on data storage devices such as disks. Included in this embodiment is a memory resident portion of the database 512 used to store designated portions of the data in memory in order to increase efficiency by reducing the amount of time it takes to retrieve data. An embodiment may therefore designate one or more portions of the database to be stored in a memory resident portion 512.

The RTAP scheduler 502 schedules and coordinates the different processes within the RTAP component 212. The RTAP scheduler may perform various process management tasks such as, for example, ensuring that other processes in 212 are executing, scheduling different processing for execution, and the like. The alarm server 504 may be used in connection with interfacing to one or more other components described elsewhere herein for notification purposes. For example, the alarm server 504 may interface with the notification server 216 and the threat thermostat controller of the Watch server 50. The alarm server 504 may be signaled in the event of a detection of a particular alert or alarm condition by the database server 508 and may accordingly interact with components external to RTAP 212. The Java server 506 may be characterized as a bi-directional server communicating with the web server 32 of Figure 4. The Java server 506 may interface with the web server 32 as needed for notification, message sending, and other communications with RTAP 212. The Java server 506 may also output one or more inputs to the threat thermostat controller 218, and also receive input from the receiver 210 to store data gathered by agents. The database server 508 may be used in connection with storing data either on a data storage device, such as a disk 514, as well as the one or more memory resident portions of the database, as may be included within memory 512. In one embodiment, the memory resident portion 512 may be implemented, for example, as a shared memory segment. The data stored in 512 and/or 514 may be an object-oriented database. Prior to use, objects of the database may be designated for inclusion in the memory resident portion 512.

In one embodiment, write operations of the database are made to the database server using the calculation engine 510. Read operations may be performed by having another RTAP component perform the processing rather than reading the data through the use of the database server 508. The RTAP component, such as the Java server, processing a read

request for data first consults the memory resident portion 512 and may obtain the one or more other portions of data from disk storage 514 as needed. All write operations in this embodiment are processed through the database server 508 and the calculation engine 510 is used to determine revised data values having dependencies on a modified data value being written. The database server 508 uses an alarm state table 520 in this embodiment to determine alarm conditions in accordance with modified data values stored in the database. The component 520 may be included in the disk or memory resident portion of the database in an embodiment depending on how the database is configured. The shared memory segments of portion 512 may be stored at various time intervals to disk or other non-volatile storage as a back up. Such a time interval may be, for example, every 30 seconds or another time interval selected in accordance with the particular tolerance for data loss in the event that data included in the memory resident portion of the database 512 is lost, for example, in connection with a power or memory failure. It should be noted that in this embodiment, a synchronization technique between readers and writers to the database may be omitted. Data attributes and/or objects which are being written may be synchronized to prevent reading incomplete values. However, the data that is read may also not include all of the recently data reported. Write operations may be synchronized by the database server 508. Thus, the database within RTAP may operate without the additional overhead of using some synchronization techniques.

20

The database objects may be represented using a tree-like structure. Referring now to Figure 9, shown is an example of an embodiment 600 of one representation of a database tree or schema that may include the objects of the object oriented database of RTAP. In the representation 600, at level 0 is a root of the tree 600. A security object node and an object template node are children of the root located at level 1. The security object is referred to as the parent node with respect to all of the related metrics and other data stored within RTAP. The object templates may include one or more nodes that correspond to the different templates for each of the different object types. For example, in this embodiment there is a metric name object type, a category name object type, and a host name object type. There are child nodes of the object templates node at level 1 for each of these different types. When a new host is added to the system, an object of the type "host name" is created for that particular new host in accordance with the appropriate object template. Each host name object corresponds to one of the hosts or computer systems. Data associated with each particular host or computer system is organized by category name. A category name may

30

refer to a particular category of metrics. For example, one category may be login information having associated metrics such as number of failed password attempts, and the like. Each of the different metrics associated with a particular category is a child node of a category node corresponding to that particular category. Referring back to the category of login data, for example, the failed password attempts may be one metric stored in a metric object which is a child node with respect to the category name object for login information. This 3-level tree of objects is only one possible embodiment of a database of metrics. Other embodiments that may also be implemented by one of ordinary skill in the art may include, for example: conventional relational database representations of metrics, as well as other tree structures for object-tree representations of metrics, such as metric objects as children of host objects without intervening category objects, multiple levels of category objects providing additional metric grouping information, and multiple levels of objects above the level of host objects, providing groupings for hosts, such as functional or geographic host groupings.

In this embodiment, there may be one or more object templates. Similarly, there may be one or more different host objects. Associated with a single host object may be one or more category objects. Associated with a single category object may be one or more metric objects. Each of the objects shown in 600 may also have an associated one or more attributes. For sake of simplicity of illustration, all the attribute nodes of each object are not included in the tree 600. As an example, object 602 is shown in more detail in Figure 9.

In connection with the object names included in the database schema or tree of objects, a particular metric may be referred to by including the name of all of the intervening objects in the path between the root and the particular node of interest.

Included in Figure 9 is a support data object having one or more child objects which may be used to store information used primarily to configure standard components of the RTAP system. What will now be described are how the alarm state tables used by the calculation engine, as described elsewhere herein, may be stored within the representation 600. In this embodiment, one child node of the support data object is an alarm class object. The children of the alarm class object correspond to the different types of alarm classes. In this example, there are three alarm classes: Boolean or 2-state, 3-state, and 5-state. For each class, such as the Boolean class 606, a child node, such as 608, includes the alarm state table

for that class. In this example, the number of states in an alarm class specifies the number of bins or alarm levels. An embodiment may include other alarm classes than as shown herein.

5 A host object may be created, for example, the first time a new agent reports to the Watch server. A new host may be determined by the receiver 210 of Figure 4. On first notification of a new agent, an alert or alarm condition is raised. A notification message may be sent. A user or administrator may be presented with a list of one or more new agents and a selection may be made to authorize or reject each new agent. When a new agent is authorized/approved, data from the agent may be processed by the receiver 210. Otherwise, 10 data from an unauthorized/unapproved agent is rejected. Note that the data from the agent is not stored or queued for later processing after approval since this may cause an overflow condition. An agent reports one or more metrics or other data to the receiver 210 which, upon successful authentication of the message, may perform a write operation to the database of RTAP. The RTAP database server 508 then writes the values to the objects and executes or 15 runs the calculation engine in order to propagate all other values dependent on this new value written to the database. For example, a particular metric, such as the number of failed password attempts, may be referenced as a metric attribute. The first metric attribute may be the number of failed password attempts as a raw value. A second metric attribute may be the raw value used in a mathematical representation to calculate a percentage. Accordingly, 20 when a new metric raw value of failed password attempts is written causing an update for the value of the first attribute, the calculation engine is then executed and updates any other values in the database dependent on this new raw value. In this instance, a revised percentage is determined as a revised second attribute value.

25 In this embodiment, the calculation engine 510 has a built-in alarm function which uses values from the alarm state table 520 to determine if a revised data value triggers an alarm condition. After writing a new data value to the database, the calculation engine determines revised data values as described above in accordance with data dependencies. Once the revised data values have been determined, alarm state table 520 may be consulted 30 to determine if any of the revised values now trigger a new or revised alarm condition. In the event that the calculation engine determines that an alarm condition has been detected, a message is sent from the database server 508 to the alarm server 504 which subsequently sends a message to the one or more notification servers.

In one embodiment, an alarm state vector or alarm instance may be defined for an attribute of a metric object. In determining a value for this attribute, the alarm function described above may be invoked.

5 Referring now to Figure 9A, shown is an example 620 of more detail of a node in the tree 600 for a metric using the alarm function. In this example, the attribute 1 622 has an associated alarm instance 624 and an alarm function whose result is assigned as the value of the attribute 1 622. The alarm instance 624 includes one or more subvalues 628a-628c that may be used by the alarm function. In this example, the subvalues include a current alarm
10 state 628a, a current acknowledged state (Ack state) 628b, and a sequence number 628c. It should be noted that other information may be included as one or more subvalues than as described herein in this example. Use of these subvalues 628a-628c is described in more detail in following paragraphs. In one embodiment, the subvalues may be included in a vector or other data structure.

15

In one embodiment, the alarm function may have a defined API of the following format:

Alarm (alarmclass, value, limits, other data ...)

20

The limits in the above refer to the alarm limits vector, as described elsewhere herein. The alarm limits vector may include one or more levels associated with the different thresholds. Each level in the alarm limits vector above refers to an alarm level or threshold that may be associated with varying degrees of alarm conditions or severity levels such as, for
25 example, warning, high, and the like. Each of these levels may be stored within another attribute, such as attribute 2 of Figure 9A and may have a default value as specified in the original template. These values may be changed in an embodiment, for example, through a user specifying or selecting a new threshold level. The alarmclass may be used to specify a particular class of alarm (such as 2-, 3-, or 5-state) in order to determine the proper alarm
30 class from the tree 600 to obtain the corresponding alarm state table for a particular alarm class.

It should be noted that state tables, as may be used in connection with alarm states, are known to those of ordinary skill in the art and may include one or more rows of input. Each

row may specify a next state and action(s) based on a current state and given input(s). Using the alarm state table as input, the built in alarm function of the calculation engine may determine, in accordance with the revised data values, whether a predefined level associated with an alarm condition has been exceeded. The predefined levels may be default or user-
5 specified alarm thresholds.

The foregoing is an example of how data gathered by an agent may be processed and stored within an embodiment of the Watch server including RTAP. Other embodiments may use other representations in accordance with the particular metrics and communication
10 protocols used in an embodiment.

The RTAP component 212 may be characterized as described herein in one embodiment as an environment which is a set of cooperating processes that share a common communications infrastructure. In one embodiment, these processes may communicate using
15 SysV UNIX messaging techniques, semaphores, shared messages, and the like. As known to those of ordinary skill in the art, an embodiment using SysV messaging techniques may experience problems due to insufficient memory allocated for message use, such as with RTAP communications. Messages that may be communicated between processes within RTAP, as well as between RTAP and other components, may use a prioritization scheme in
20 which low priority activities involving message sending are suspended when the amount of memory in a message pool falls below a designated threshold. This particular designated threshold may vary in accordance with each particular embodiment. In connection with the techniques described herein, a portion of the memory for message use, such as 80%, may be designated as a maximum threshold for use in connection with requests. In the event that
25 more than 80% of the message memory or pool has been consumed and used in connection with message requests, any new requests are blocked until conditions allow this threshold not to be exceeded. However, message responses are processed. The foregoing may be used to avoid a deadlock condition by blocking a request in the event that the threshold portion of the message pool is consumed for use in connection with requests. In one embodiment, the
30 foregoing special message management functionality may be included in one or more routines or functions of an API layer used by the different RTAP components when performing any messaging operation or function. These routines in the API layer may then invoke the underlying messaging routines that may be included in the operating system.

An embodiment may utilize what is referred to herein as "latching alerts" where a particular alarm level does not decrease until an acknowledgment of the current alarm state has been received. An acknowledgment may be made, for example, by an operator through a GUI. An embodiment may define an alarm state table 520 such that an alarm or an alert state
5 may be raised or remain the same until an acknowledgement of the alarm or alert state has been received. Until an acknowledgment is received, the alarm state table does not provide for reducing the alarm or alert state. It should be noted that the foregoing latching alerts may be performed in connection with one or more of those metrics associated with an alert or an alarm state. The latching alerts may be used in an embodiment in connection with particular
10 indicators or classes. Other classes of metrics, such as those associated with performance indicators, may not be subject to the latching condition. This may vary in accordance with each embodiment.

Referring now to Figure 10, shown is an example of an embodiment of the alarm state
15 table 520 that may be used in connection with implementing latching alerts. Included in the state table 520 is a line of information corresponding to a current level or state. Each line of information includes a current level or state, an acknowledgment, and input value or range, a next level or state, and an associated action. In this example 520, a normal level is associated with a level one indicator for a range of input values between 0 and 100, inclusively. An
20 alarm condition is associated with a second level for a range of input values between 101 and 200, inclusively. Finally, a third alarm level is associated with an input range of values from 201 to 300, inclusively. Line 652 indicates that the current level or state of normal level one is maintained as long as the input is between the range of 0 and 100. Line 654 indicates that
25 when the current level is normal (level 1) and a current input value is between the range of 101 to 200, level 2 is the next designated level or alarm condition. The action designates that an alarm condition is generated. In connection with line 652 and 654, it is not relevant whether an acknowledgment has been received because an acknowledgment does not apply
in this example in connection with a normal alarm or level condition. Referring to line 656,
when the system is in the second level of alarm and the input value drops down to the normal
30 range between 0 and 100, but an acknowledgement of the alarm condition has not yet been received with respect to the level 2 alarm condition, the system remains in the level 2 state of alarm. With reference to line 658, in the event that the system is in the second level of alarm and acknowledgement of this alarm has been received, when the input value or range drops to the normal range between 0 and 100, the level of the current state decreases to level 1 and the

alarm condition is cleared. Referring to 660, if the system is in the second level of alarm state and the input value increases to the third level between 201 and 300, an alarm condition is generated to signify this increase in alarm level to level 3. This is performed independent of whether an acknowledgement to the previous alarm state of level 2 has been
 5 acknowledged.

It should be noted that the different ranges or values specified in connection with the third column of 520 may be associated with threshold values or ranges. The thresholds may be specified using default values as well as in accordance with one or more user selected
 10 values or ranges. It should also be noted that although the table 520 shows specific numeric values for the ranges in the input column, these alarm range thresholds may be parameterized to use the input values (i.e., alarm LEVELs) of the alarm function described elsewhere herein.

Referring now to Figure 11, shown is an example of another embodiment of an alarm state table 700 and an alarm limits vector 200. The elements of vector 200 identified in the lower left corner may be passed as input parameters when invoking the alarm function described herein specifying the LEVELs or thresholds for the different alarm states as described above. In this example, the table 700 represents a 3-state alarm (normal, warning,
 20 and alert) with 2 thresholds (100 and 200) forming three partitions or ranges (0-99, 100-199, 200 and greater). Each row of the table 700 corresponds to a state and includes:

- a state identifier for the row in the table 704;
- a named identifier for the state 706;
- a color 708 as may be displayed, for example, by an illuminated light on a display
 25 panel or other indicator; and
- an indication as to whether an acknowledgement (ACK) is required for this state 710.

The portion 702 of this example includes the transition functions (beginning with & in this example) used in determining state transitions from one row of the table to another.
 30 Other embodiments of 700 may include information other than as described herein. State transitions occur as a result of evaluating transition functions. It should be noted that if a column name contains a space character (such as between RANGE and LEVEL in 712), the transition function name ends at the space character such that the remaining text (LEVEL)

following the transition function (RANGE) is a modifier to the function, telling the function how to interpret the values in the column.

5 The alarm system determines the new state for an alarm by repeatedly evaluating transition functions in the context of the alarm state table for the alarm. The transition functions are evaluated in the order in which they appear from left to right as columns in the state table. Each transition function may take optional input parameters such as, for example, the current alarm state, values from the alarm state table and other optional values as arguments. As an output, the transition function returns a new state in accordance with the input parameters. Each function is evaluated repeatedly, until the new state returned is the same as indicated by the state value in 704 for a particular row, or until a transition that would loop is detected. Evaluation then proceeds to the transition function indicated by the next column moving from left to right.

15 In this example 700, two functions are illustrated, &ACK and &RANGE as described in more detail in following paragraphs. The &RANGE function takes an alarm limit vector like the one illustrated in Figure 11, lower left corner 720, as an example, as well as the following and other possible suffixes in column names:

- 20 level - The level number corresponding to the current alarm state;
- < - What new state to move to when the current value for the metric is in a range of values lower than the range for the current alarm state;
- == - What new state to move to when the current value for the metric is in the range of values associated with the current alarm state; and
- 25 > - What new state to move to when the current value for the metric is in a range of values higher than the range associated with the current alarm state.

The alarm limit vector 720 in this example contains an integer (2 in this example) as the first vector element indicating what level number to assign to the highest range of values the metric can assume. The highest range of values includes all values greater than or equal to the limit specified as the second vector element, which is 200 in this example. The third vector element specifies another next-lower limit, which is 100 in this example. The fourth vector element specifies the final next-lower limit, which is 0 in this example. The three ranges or partitions of values are specified using the vector elements 2-4 for the following:

Range 0: values less than 100

Range 1: values from 100 to 199

Range 2: value 200 or more.

- 5 In connection with the foregoing, it should be noted that a fourth range is implied for values less than zero (0). In this example, values in this fourth implied range correspond to an error state and are not further described in connection with this example.

10 In this example, the &ACK function is a transition function that returns the current state (as indicated by column 704) when the alarm has not yet been acknowledged, otherwise returns the new state as indicated in column 714 when the alarm has been acknowledged.

Referring back to Figure 9A, the current alarm state 628a and the ack state 628b may be used in determining the current state of the alarm and its corresponding row in the alarm state table. The sequence number 628c may be used in connection with race conditions that may be associated with alarm conditions and acknowledgments thereof. A unique sequence number is associated with each alarm state that caused a notification message to be sent to a user. Each such message contains a copy of that sequence number for the state causing the message to be sent. In one embodiment, a unique sequence number may be generated for each alarm condition associated with a single invocation of the alarm function. The single invocation of the alarm function may result in transitioning from a current input state to an output state, and may also transition through one or more other intermediate states to arrive at the output state. In this embodiment, a unique sequence number is not associated with the one or more intermediate states. Rather, a first unique sequence number is associated with the current input state and a second unique sequence number is associated with the output state. For example, a first alarm condition notification message having a first sequence number may be sent to a user and indicated on a user interface display. A second alarm condition, indicating a greater severity than the first alarm condition and having a second sequence number, may be determined during the same time period in which a user's acknowledgement of only the first alarm condition is received and processed. The user's acknowledgement is processed in accordance with the particular sequence number associated with the alarm condition being acknowledged. In this example, the acknowledgement indicates an acknowledgement of the first alarm condition, but not the second. Without the use of the sequence number, or some other equivalent known to those of ordinary skill in the

art, the acknowledgment may also result in acknowledgement of the second alarm condition depending on whether the acknowledgement is processed before or after the second alarm condition is determined.

5 Referring now to Figure 12, shown is a state transition diagram 800 illustrating the state transitions associated with the functions 702 of alarm state table 700. In the example 800, each state is represented as a node. The arrows between the nodes represent the transitions possible between the different states in accordance with the information in 702 of table 700. Note that the diagram does not indicate transitions causing a same state or a
10 transition to a state A from a same state A.

What will now be described is an example of acknowledging an alarm illustrating the use of the table 700 in determining a new state from a current state. In an example, the alarm function is invoked including parameters indicating that the current state or initial state
15 is Alert Unacked (6) with a metric value of 250. A human user or other agent acknowledges the alarm, causing the alarm state to be re-evaluated. Examining the row of table 700 for state 6, the &ACK function is evaluated with a state (6) as the current state. Since the alarm is now acknowledged, &ACK returns (5) as indicated in the &ACK column of row 6 of the table as the new state. As a result, the new state of the alarm becomes Alert Acked (5).
20 &ACK is re-evaluated for state 5 using row 5 of the table 700. Since the alarm has been acknowledged, the &ACK function returns a 5 as the new state. Since the new state matches the current state, the evaluation of the &ACK function is complete and evaluation proceeds with the next transition function in state 5. The next transition function, is &RANGE. Recall that the metric value for which evaluation is being performed is 250. &RANGE uses the
25 limits vector as described above, and determines that the current metric value of 250 is greater than the first limit of 200 classifying the current metric value as being within the highest range of 2 (greater than 200). The "&RANGE level" column indicates that range (2) corresponds to the current state (5) and so the &RANGE function returns the (5) which is contents of row 5 in the "&RANGE =" column as the new state. Since the new state (5) is
30 identical to the current state (5), evaluation of the &RANGE function is complete. Since the &RANGE function is the last transition function in the state table in this example, the state change is complete and the alarm system proceeds to carry out whatever actions are configured for the new state and any other notifications or other controls that may be associated with the alarm. For example, the alarm color in the example changes from "Red

Flashing" to "Red" as indicated by column 708. Column 708 may be characterized as an associated action to be taken when a transition completes after evaluation of all transitions functions. Other embodiments may include other actions.

5 Similar examples can be constructed to demonstrate that the state table illustrated in Figure 11 embodies the following behavior for a latched alert or alarm:

 -When a metric value enters a value range associated with a "higher" or "more serious" alert state than the current state, the current state transitions to that higher alert state,
10 unacknowledged.

 - If the metric enters a value range associated with a "lower" or "less serious" alert state, and the alarm has not been acknowledged, no state change takes place -- the alarm is said to "latch" at the highest alert state represented by the metric while the alarm is
15 unacknowledged.

 - If the alarm is acknowledged, the state changes to whatever state currently reflects the value of the alarm metric.

20 The state table in Figure 11 illustrates an alarm that, if a metric associated with the alarm assumes a value corresponding to a lower alarm state while the alarm is latched at a higher state, and the alarm is acknowledged, the alarm transitions into the lower alarm state with an unacknowledged status. If the alarm is in a high-severity acknowledged state, and the underlying metric changes to reflect a lower-severity state, the alarm also changes to a
25 lower-severity unacknowledged state. Comparable alarm tables can be constructed that preserve the latter behavior of the alarm while altering the former behavior to transition into an acknowledged state, rather than an unacknowledged state. Such transition tables however, may be characterized as more complex than the example described herein and may include additional states than as illustrated in Figure 11. The table 700 of Figure 11 models an
30 "analog" or "floating point" metric. Comparable state tables can be constructed for digital metrics, boolean and other kinds of metrics.

 It should be noted that an embodiment of the alarm state table may utilize values to implement hysteresis for one or more of the ranges. Hysteresis may be characterized as a

behavior in which a metric value may transition from a first state into a second state in accordance with a first threshold value, and transition from the second state to the first state in accordance with a second threshold value that differs from the first threshold value. Such threshold values may be used in connection with a metric that may assume values at different points in time which hover near a boundary or range threshold. The use of the two different thresholds may be used in order to reduce constantly changing states for metric values hovering near a boundary condition. For example, a range threshold may be 100. It may be that the associated metric assumes a series of values: 98, 99, 101, 99, 101, 99, 102, etc. at consecutive points in time. The range threshold may be used to cause a transition from a first state to a second state (from 99-101). However, it may be undesirable to have an alarm state change associated with changes from 101 to 99 especially if the metric value may hover around the boundary of 100. An embodiment may determine that once the first threshold is reached causing a transition from a first range under 100 to a second range of 100 or more, a value of 95 or less is needed to cause a transition from the second back to the first range using 95 as the second threshold value. As will be appreciated by those of ordinary skill in the art, state tables as described herein may be modified to include the foregoing use of multiple thresholds to account for hysteresis conditions.

The foregoing is one example of an embodiment of how data may be managed within the system and how alarm conditions may be determined and generated. Other embodiments may store the data in one or more other types of data storage in accordance with one or more organizations and determine alarm conditions using techniques other than as described herein.

The XML messages or documents sent by the agents to the receiver may include data corresponding to objects and the tree-like structure as illustrated in Figure 9. The XML document may include data represented as:

host name	- name of host sending the report
- category name	- name of a group of metrics
- metric name	- name of a metric
- value	- metric value
- attr1	- other attribute/value
- ...	

It should be noted that an embodiment may use other formats, such as an alternative to XML, and protocols than as described herein for communications between agents and other components.

5 Attributes that may be associated with a metric include "value" and "units." Other kinds of metrics have other attributes. For example, the "Operating System Type" metric may have corresponding attributes unique to every platform. The attributes may include, for example, a version number, machine ID or other information that may be useful on that platform, but not exist on any other platform.

10 Other groups of metrics may share some additional "standard" or common attributes, for example:

15 - log - used by the log agent 306 for all metrics derived from log files. The "log" attribute is a table of strings containing the complete text of the first three log messages of the type counted by the metric in the reporting interval. These log messages may provide additional detail, for example, to a network administrator who is trying to understand why an alert was raised on a particular metric; and

20 - summary - used by all agents that support the enhanced e-mail or enhanced reporting feature as described elsewhere herein. The "summary" attribute contains a human-readable summary of the "next level of detail" for the metric.

25 As described herein, data from the RTAP database may be combined using the calculation engine and elements from the tree-structure object oriented database to produce one or more inputs to the threat thermostat controller 218 previously described herein. The calculation engine as described above may process a data-flow language with a syntax and operation similar to a spreadsheet. A calculation engine expression may be defined as expressions attached to data attributes of objects in the database. When the calculation
30 engine processes an object, all of the expressions in the object are evaluated and become the new values of the attributes to which they are attached. This evaluation process is repeated to re-evaluate all objects dependent on changed values.

Expressions in an embodiment may reference other attributes using a relative pathname syntax based on the Apple Macintosh filesystem naming conventions. For example,

- 5 - '^' means the parent object
- ':' separates object names
- '.' separates an object path from an attribute name

10 In one example, there may be 5 external threat thermostat values communicated by the threat agent 200 and stored in the RTAP database called E1, E2, E3, E4, and E5. An input to 218 may be determined as a weighted average of the foregoing five values. The threat agent 200 may monitor or poll the external data sources and write the threat thermostat level indicators to the five "E" external points. In the RTAP database, these five (5) points
15 may be defined as child objects of a parent object representing the combined weighted average in an expression. The value of this expression may be assigned to the parent object having the following expression:

$$\begin{aligned} & ([E1.indicator] + 3*[E2.indicator] + [E3.indicator] + [E4.indicator] + \\ 20 \quad & [E5.indicator]) / 7 \end{aligned}$$

In the foregoing, the ".indicator" operator obtains the value of the identified attribute referenced. In the foregoing, external indicator 2 is determined to be three times as valuable or relevant as the other indicators. Whenever an indicator is updated, the calculation engine
25 calculates the tree of other points having expressions referencing the contents of any of the attributes of a changed point. The engine is executed to evaluate the expressions on all of those objects, similar to that of a spreadsheet. An embodiment may use any one or more known techniques in evaluating the expressions in an optimal order.

30 In one embodiment, an approach may be taken with respect to combining inputs with respect to the different metrics as may be reported by the different agents. A resulting combination may be expressed as a derived parameter used, for example, in connection with generating an alarm or alert condition, as an input to the threat thermostat 218, and the like. A derived value or signal indicating an attack may be produced by examining one of more of

the following: one or more metrics from the NIDS agent, the ARPWatch agent, IPS, the number of bad logins and root user count exceeding some predetermined normal threshold tuned for that particular system. Once an initial set of one or more of the foregoing indicate an alert condition indicative of an attack or attempted attack, secondary condition(s) may be used as confirmation of the attack indication. The secondary information may include a resource usage alert on some machine that occurs simultaneous with, or very shortly after, a reliable indication of an attack on that machine, or on the network at large using the initial set of conditions. An embodiment may, for example, generate an alarm condition or produce an input to the threat thermostat 218 based on the foregoing. An alarm condition may be generated in connection with a yes or true attack indicator value based on the first set of one or more conditions. Once this has occurred, another subsequent alert or alarm condition may also be generated based on the occurrence of one or more of the second set of conditions occurring with the first set of conditions, or within a predetermined time interval thereof, for the network, or one or more of the same computers.

In connection with the foregoing, resource usage metrics may not be used as first level attack indicators or used without also examining other indicators. Resource usage may be characterized as a symptom of potential machine compromise rather than an attack on the machine. Usage metrics may be "noisy" causing false positive indicators of an attack if examined in isolation. However, if such an alert occurs in connection with resource usage simultaneously with, or very shortly after, another reliable "attack" indicator (as in the initial or first set of indicators above), the resource usage metric's credibility increases. Accordingly, the resource usage metrics may be consulted in combination with, or after the occurrence of, other attack indicators to determine, for example, if any one or more particular computers have been compromised. Based on the foregoing, an embodiment may define a derived parameter using an equation or formula that takes into account security metrics and combines them with one or more resource metrics. Such a derived parameter may be used, for example, in connection with producing an input to the threat thermostat controller. Such a derived parameter may be produced using the weighting technique described above.

An embodiment may include a form of enhanced reporting or notification as made by the Watch server to a user upon the detection of an alarm condition. As described herein, metrics and associated information may be reported by an agent. The values of the metrics

may be one or more factors used in determining an alarm condition. The values of the metrics used to detect and report the occurrence of an alarm condition may be characterized as a first level of alarm or event notification information. Once an alarm condition has been detected, additional information that may have been gathered by the agent may also be useful in proceeding to take a corrective action or further diagnosing a problem associated with the alarm condition. This additional information that may be used in further diagnosing the problem or taking corrective action may be characterized as a second level of information.

An embodiment may provide an option for enabling/disabling notifications of alarm conditions to include this additional information. The additional information may be reported by the agents to the Watch server for optional incorporation into notification messages. An enable/disable option may also be associated with agents gathering the data. Whether an embodiment includes this feature or uses it for selective metrics may vary with each embodiment and its resource limits such as, for example, of the industrial network. Thus, the cost and feasibility of obtaining the second level of information may be balanced with the benefits to be gained in taking corrective actions using the second level of information from an alert message rather than obtaining the information some other way. For example, if a second level of information is not included in an alert notification, an administrator may use a user interface connected to the Watch server 50 to gain the additional information.

It should be noted that the particular additional information included in the second level of enhanced notification may vary with each metric. It may include additional information to assist in determining a problem source such as a user account, IP or other address, particular component(s) or process(es) consuming a resource and the associated percentage(s), and the like. Some of this second level of information is described above with associated metrics. In one embodiment, the enhanced notification feature may be optionally enabled for use with one or more metrics of the SNMP Guard agent 203 and the Guard log agent 209 described herein.

For example, the agent 203 may report the following additional information with each of the metrics for enabled enhanced reporting:

Communications status - In the event there is a problem indicated by this metric, corrective action includes determining the reason for the communication failure. A message such as the component is not responding, its IP address and the time of the failure may help.

Login failures – Identify the most frequent type of connection, such as a VPN, remote dial-in, or other connection, a percentage of the failures on this connection, one or more of the user IDs associated with the top number of failures and the percentage and
5 originating IP address or other communication channel associated with each user ID.

Administrative user count, dialup user count VPN user count – identify the IP or other addresses from which the most recent administrative users have logged on.

10 Memory usage, CPU usage, disk space, other resource usage – The top consumers of the resource are identified along with an associated percentage along with which process or user or other information to identify the consumer as appropriate for the resource.

Open session count – identifies the number of open communication sessions
15 between any two points. Additional information may include two or more IP addresses, host names, and the like, identified as being included as a connection endpoint.

Agent 209 may include the following additional enhanced reporting or notification information for the following metrics:

20

Configuration – In the event that there has been a change to the configuration as monitored by 209, it may be helpful to know what changed such as whether there is a change to the threat thermostat rule sets, such as included in 220, and/or the current firewall configuration, and what portion of the rules changed. Additionally, for any such change,
25 was the state change (previous state to what current state), from what user account, address, process and the like, made this change.

Threat thermostat change – An embodiment may indicate an alarm condition when a change occurs to the threat thermostat setting. The change may be the result of a manual
30 change, an automated change in accordance with the functionality included in an embodiment. Additional detail for enhanced reporting may include what user made the change, what was the status changed to/from, the frequency that such changes have been made within a reporting period, identify the uses that most frequently changed the setting and what percentage of the time each user changed the setting.

NIDS and IPS reports – An address or other identifying source of the most frequent alerted NIDS/IPS conditions, an associated percentage of these conditions attributed to a particular source, information about the type of attack, and the target of the attack (what machine by host name, IP address and the like).

Antivirus events –The metric may identify a total number of antivirus events. Additional information may include a break down by type of event within a reporting period to identify what viruses (virus signatures) have been removed from a communication streams with an associated frequency or percentage, what source and/or destinations (such as source and destination networks) appeared most frequently for each type, and a frequency or percentage associated with each of the source and destinations.

Other activity – This metric identifies other activity that does not belong in any other category. Additional information may include the text of the first one or more messages of this type detected.

Referring now to Figure 13, shown is an example 900 of a graphical user interface display. The example 900 may be displayed, for example, using a web browser to view alarm incident reports resulting from notification messages sent in accordance with alarm conditions determined. The example 900 may be used to view and acknowledge one or more of the alarm conditions in an embodiment. In one embodiment, this display of 900 may be viewed when tab 902 for incidents is selected from one of multiple user interface tabs. Each tab may be selected in connection with different viewing and/or processing. In one embodiment, the tab 902 may flash if a new alarm condition is detected from that which is displayed in 900 at a point in time to a user. In other words, this embodiment may not automatically update the display 900 with additional information for alert conditions detected since the user selected tab 902. Rather, this embodiment flashes coloring on tab 902 to indicate such additional alert conditions detected while the user is in the process of using an instance of display 900. The inventors believe that the flashing tab is less disruptive of user concentration during alarm burst conditions than other notification techniques such as, for example, redrawing the display 900 with updated alarm information as available.

The display 900 may indicate in column 906 (labeled "A") whether a particular condition indicated by a line of displayed data has been acknowledged. An incident or alarm condition associated with a line of displayed data in 900 may be acknowledged, as by selecting the exclamation point icon to the left of a particular line of data, selecting the option 908 to acknowledge all displayed incidents, or some other option that may be provided in an embodiment. The status in 906 for each incident may be updated in accordance with user acknowledgement. For example, 904 indicates that the associated incident has not been acknowledged (e.g., exclamation point notation in column 906). The two incidents as indicated by 910 have been acknowledged (e.g., no exclamation point notation in column 906).

Referring now to Figure 14, shown is an example of a user interface display 1000. The example 1000 may be displayed when the monitor tab 1020 is selected to view a metric tree. With reference to Figure 9, the information displayed in 1000 is that information included in a portion of 600 – the subtree formed with the security object as its root including all child nodes. The display 1000 shows an aggregate view of the different metrics and associated alarm conditions. The display 1000 reflects the hierarchical representation in the subtree by showing a nesting of hosts (Guard and Watch), categories for each host (such as Intrusion attempts, Resource Usage, and the like), and metrics (such as CPU Usage, Memory Usage and Sessions) associated within each category (such as Resource Usage). In the subtree, these metrics may be defined as leaf nodes having a parent node (category name) defined as Resource Usage.

Associated with each of the metrics is a level indicator. The level indicator may indicate a color or other designation associated uniquely with each alarm state within an embodiment. For example, in one embodiment, the indicator may be green when the metric level is in the normal range, yellow when the metric level is in the warning range, and red when in the highest severity range.

The elements in 1000 representing the parent node of one or more other nodes may have a color or other designation corresponding to the aggregate condition of all the child nodes. For example, the indicator for Resource Usage may represent an aggregate of each of the indicators associated with metrics for CPU Usage, Memory Usage, and Sessions. In one embodiment, the aggregate indicator of a parent node may be determined to be the maximum

indicator value of its one or more child nodes. For example, a parent node indicator, such as 1006, is yellow if any one or more of its child node indicators, such as 1008, are yellow but none are red.

5 In 1000, the user may select to view a graph of a particular metric in the right portion of 1000 by selecting an option in the left portion of 1000 as indicated by selection 1010. In one embodiment, it should be noted that the graph portion is not immediately updated in response to a user selection. Rather, the graph may be updated when the web page is refreshed in accordance with the one or more selections made at that point in time. Note
10 that in 1000 the icon or indicator displayed for Watch 1002 has a different shape than other machines, such as the Guard machine or host. The different shape makes it easier for users to find the Watch server in the list of hosts since many of the important network monitoring metrics are found in the Watch server branch of the metric tree.

15 The foregoing 900 and 1000 are examples of user interface displays that may be included in an embodiment and displayed, such as using a web browser on the web server 214 of Figure 4. Other embodiments may use other interface displays than as described herein.

20 Described elsewhere herein are techniques that may be used in connection with performing periodic integrity checks such as, for example, upon files of a filesystem. These techniques may also be used in connection with other types of data, associated data elements and data stores, such as, for example, the data elements of a Windows registry as may be included in a Microsoft Windows® operating system, or a database, such as a relational
25 database or hierarchical database. As described elsewhere herein in connection with filesystems, different data elements, such as files, may be categorized or placed in various "buckets". Similarly, defining various "buckets" or groupings of data elements in which each bucket has an associated category, priority, and the like, may also be applied for use with the Windows registry, database, such as a relational database or other data store.

30 When performing integrity checking, scanning of the data elements may be performed in accordance with one or more metrics. The particular metrics associated with each category or "bucket" may vary in accordance with the priority and other features of a category. In one

embodiment, the scanning may result in an alert generation in connection with unexpected changes to either the structure or contents of the particular data store.

As described elsewhere herein, the scanning may be performed in a process control
5 system by one or more agents executing on a component in the process control system. In an embodiment, the agents may perform the actual scanning or may otherwise communicate with another component which performs the scanning. In either instance, an agent is able to monitor and detect any changes or differences to data elements of one or more data stores. For example, the critical file monitoring agent of Figure 5 may perform the processing to
10 monitor files. Additionally, similar to the critical file monitoring agent 304 of Figure 5, an agent may be used to monitor and detect changes to data elements of other types of data stores.

Different attributes, features, and the like, may be used to characterize and identify a
15 particular data store and its associated data elements. When there is an unexpected change to one or more of these, an alert may be generated. For example, schema information for a particular data element and associated data store may be available and may vary with type of data and data store. As known in the art, schema information may relate to the structure, definition, or model of a data store and its data elements. Schema information may include,
20 for example, filesystem paths, registry paths, or a database table, field, stored procedures, and the like. A set of initial schema associated with a data store and data elements associated therewith may be recorded at a first point in time. At a later point in time, a second set of schema may be obtained and compared to the initial schema raising alerts when changes to the current schema are observed while scanning the data store and associated data elements.

25 An embodiment may also use and compare metadata information. Metadata may be defined as information about the data or otherwise characterizing the data. For example, metadata about data in a file may include the file size, file type, (e.g., based on file extension, file attribute indicating whether the data is in binary form). Examples of metadata may
30 include data ownership, permissions, timestamps of data accesses or modifications, or metadata accesses or modifications, data type, and the like. As set forth above regarding the use of schema information, an initial or approved set of metadata may be determined. This initial or approved set of metadata may then be compared to other metadata periodically for a

current set of data elements raising alerts when changes to those metadata are observed while scanning the data store.

It should be noted that in some instances, schema information may be characterized as a subset of metadata, for example, in connection with a database.

An embodiment may also compare aspects of the actual data of the various data elements alone, or in combination with, any one or more of schema and metadata information as described above in connection with periodic integrity checks and scans of data elements.

An original or approved set of data may be recorded and compared periodically to a current set of data, or portions thereof, raising alerts when changes are observed while scanning the data store. The data set with respect to which changes are detected may be, for example, an approved or expected benchmark value determined using observed or recorded values from one or more previous scans of the data element, determined based on simulated, theoretical, or other techniques besides actual observed data. Such data sets against which comparisons are made may be set as a configurable option or parameter which is manually specified or specified using an automated technique, for example, such as using software that determines a running average over a period of time for a numeric metric.

When any one or more of the schema, metadata and/or data are characterized as "large" such as, for example, in comparison to the size of a corresponding checksum, hash, or cryptographic hash, or "secure" in which case a reference copy of the item may not be appropriate, an embodiment may choose to utilize and compare hashes of elements of the schema, metadata and/or of the data in order to reduce storage requirements and costs for scanning.

The scanning and integrity checking performed in an embodiment may preferably choose not to scan the entire, potentially very large, data store. Rather, as described elsewhere herein, integrity checks may be performed at periodic time intervals in accordance with a number of defined scanning buckets, each describing a part of the data store. Each bucket may be associated with a fixed or varying resource budget. For example, each defined bucket or category may be associated with a fixed or variable amount of time indicating how much time the scanner may spend scanning data elements and/or determining differences with respect to data elements, in that particular bucket at one time before suspending a

current bucket's scanning and/or difference detection for the current bucket, and possibly proceeding to the next bucket. If and when the resource budget permits processing with respect to a bucket to resume, the scanner may resume scanning and/or difference detection with respect to the parts of the data store identified by that bucket. The "budget" associated with one or more buckets may also be defined using other units that may vary with resource. For example, a "budget" may be defined in terms of percentage utilization of a resource during a defined time period such as, for example, percentage of CPU utilization, memory, and the like. Rather than specify a percentage, a threshold amount may also be specified. As used in connection with the techniques herein, it should be noted the resource budget defined for a bucket may be applied to the scanning and/or difference detection processing that may be performed in an embodiment.

It should be noted that it may be costly or difficult to accurately determine the amount of utilization of certain resources, such as CPU utilization or cache misses. In such cases, an embodiment may estimate such utilization by various means. For example, a file scanner may estimate cache misses by measuring how much of a large file has been scanned, or a scanner may estimate CPU utilization by measuring the passage of real time, when the passage of real time is cheaper or easier to measure than is CPU utilization.

For each bucket, or for each data element or set of elements in a bucket, an embodiment may define which aspects of the data are to be monitored such as, for example, one or more aspects of schema information, metadata information, whether to perform a comparison of checksums for a data element or to compare the actual data, and the like. The particular aspects defined for a particular bucket may also vary with each bucket as well as for each data element.

It should be noted that a scanner may perform processing which includes scanning and/or detecting differences with respect to a set of data elements. As described herein, the scanning may include performing processing to scan a current data element, and the difference detection may include comparing information regarding a currently scanned data element to other information from a previous scan for the data element.

Existing scanners, such as virus and spyware scanners, scan large data stores, such as the Windows registry or computer filesystems as well. These scanners can be supplied with a

specification of what parts of the data store to scan. The scanners can also work in a "real-time" mode where they scan new data entered into or retrieved from the data store for changes or suspicious patterns, rather than scan the entire store.

5 However, some systems and applications, such as Process Control Systems as described herein, are particularly sensitive to unexpected additional resource usage such as might be incurred by on-going real-time scanning or by a full scan of a large data store.

Existing systems could be configured with many small scans, each scanning a part of the large data store, each on a different schedule. However, partitioning the data store into
10 pieces small enough so that any individual scan would result in an acceptable resource usage is impractical. Each sub-scan still runs at full speed, consuming as much of the system's resources as necessary to complete the scan. As a result, one may need an enormous number of small "sub-scans" configured, each small enough to impact the overall system minimally. Managing all of those sub-scans may be error prone, costly and impractical.

15 The techniques described herein allow a data store to be partitioned into a comparatively small number of partitions or "buckets." Each bucket describes a part of the data store, be that part small or large. Furthermore, a resource budget can be assigned to each bucket and the scanner carries out its scanning and/or difference detection processing so to
20 fully comply or best-effort comply with the requested budget. This minimizes management costs and complexity for the scanner configuration. Additionally, an embodiment using the techniques described herein may assign priorities to different aspects of the data store that are processed. For example, a small set of data in a first bucket that is of vital or critical importance may be assigned a comparatively large budget in comparison to other defined
25 buckets so that data elements in the first bucket are processed more frequently than less important data included in other buckets.

An embodiment may provide any one or more techniques in connection with defining the particular data elements for each bucket or category. For example, data elements
30 included in a particular bucket may be specified using a list of file names, directories, objects, trees, data element relationships, query string and resulting set of data elements, data element attribute(s), and the like. When specifying trees of data elements an embodiment may specify a depth limit in the tree or use other mechanisms to specify which portions of the tree are defined as elements of the bucket.

As mentioned above, one type of data store that may be used in connection with the techniques described herein for scanning and detecting any differences or changes to specified data elements included in the data store is a database. The database may be a relational database, may be a hierarchical database with a defined hierarchical structure between data elements, such as data records, included therein.

In one aspect, the Windows registry may be characterized as a hierarchical database of information, such as settings and options for configuration of an operating system and computer system upon which the operating system resides. The information may be configuration information for one or more users, software, such as installed software applications, hardware devices, and configuration preferences for the computer. For example, whenever a change is made to a "Control Panel" setting, file association, system policies, installed software, and the like, the changes are reflected and stored in the registry. In connection with the techniques described herein, particular portions of the registry including security related information may be of interest. For example, portions of the registry with information about what software is installed, whether new software has been installed, particular settings for installed software, and the like, are one type of registry information that may be used in connection with the techniques described herein. Other non-security related information, such as user preferences related to the visual appearance or location of displayed elements on a screen, may not be useful in connection with the techniques described herein. Furthermore, different portions of the security related information may have varying levels of importance. For example, a first set of attributes or settings associated with each application may include more pertinent or critical security-related information than a second set of attributes or settings for installed software applications.

A registry of such information as described above for a system may be represented in a variety of different ways reflecting the registry's hierarchical structure.

Referring now to Figure 15, shown is an example illustrating one representation of information that may be included in a system registry. It should be noted that the example 1100 may not reflect the exact structure of any particular embodiment of a registry. Rather, it is meant to generally illustrate that a system registry, such as the Windows registry for a Windows operating system, may include different types of system wide data organized in a

hierarchical structure. The example 1100 includes a tree structure to represent the hierarchical structure of the registry information. In this example, the information in the registry may be partitioned into one of four general groupings as being related to hardware, software, users, or system settings and information. The foregoing general groupings are represented by the first level of the tree or the children of the root node. It should be noted that each of the first level nodes may represent nodes of a subtree including one or more other sub-levels of nodes not included in the example 1100 for the sake of simplicity of illustration.

In connection with the techniques described herein, only portions of the registry information may be of interest for use in connection with the techniques described herein. As mentioned above, security-related information included in the registry may be of interest in connection with the techniques described herein and may be included in one or more buckets. In other words, a first subset of the information in the example 1100 may be included in a first bucket. A second subset of the information in the example 1100 may be included in a second bucket. As an example, information related to installed software may be of interest for use in connection with the techniques described herein. A first bucket may be designated to include the subtree with the root node 1110. In connection with the first bucket, an embodiment may perform scanning and determine changes with respect to one or more settings for each application (e.g., Setting 1 for App1) as well as whether there are any newly installed applications (e.g., any new child nodes of node 1110). A second different bucket may include the subtree with the root node 1112 related to system security information.

As illustrated and will be appreciated by those skilled in the art, a data store having a hierarchical structure provides a structure that can be used in connection with easily forming subsets as may be specified in connection with buckets. In one embodiment, buckets may be specified by defining one or more inclusion rules and, optionally, one or more exclusion rules. The rules can take a variety of different forms. The buckets may be defined to include elements based on, for example, named paths in the hierarchy, as included in a particular subtree or branch, or subset of a subtree or branch, a particular naming pattern of data elements (e.g., all data elements having the string "SYS"), a list of one or more fully or partially named data elements, and the like. Additionally, a bucket may be defined to exclude one or more elements also based on the foregoing. For example, data elements may be excluded which match the naming pattern "*.LOG", "*.TMP", and the like. The exclusionary rules may be characterized as selectively pruning data elements from the data

store that are included in a bucket. One use of exclusionary rules may be to exclude data elements which may change at a higher frequency rate than desired since changes in such data elements may raise unnecessary alerts or false positive alerts. The use of the inclusionary and exclusionary rules function as a filter of what data elements are included in a bucket.

5

In connection with data elements in a data store having a hierarchical structure between the data elements, there is an underlying assumption that similar kinds of data elements are grouped together in the structure.

10 It should be noted that the techniques described herein may also be used in connection with data that is not initially organized or partitioned in accordance with a defined hierarchy. In such an embodiment, rules may be defined which group together like data elements. For example, a file system may be flat file system such as one not having a directory structure. Files included therein may be named using a naming convention such as with the occurrence
15 of certain strings in the file names. Rules may be specified which group files together to be included/excluded from buckets based on the naming conventions.

20

The rules defined for each bucket determine which data elements are included in each bucket. As such, the rules determine which data elements are scanned and monitored for changes.

25

As described above, the types of information that can be monitored for changes may include one or more of data, metadata, and schema information for each data element. As also mentioned above, the hash of the foregoing information may be utilized in connection
25 with assessing any changes to the foregoing information. Reasons for using a hash value rather than the actual information (e.g., data, metadata, or schema information) about the data element may relate to performance criteria, the amount of storage consumed/available, and/or the sensitivity of the information being monitored.

30

When utilizing hash values for information, less storage and computation time may be consumed than when examining the actual information. Additionally, storing a copy of a hash value of information may be desirable for security related reasons due to the sensitivity of the information. For example, information being monitored for changes may include a password, private key value, or other authentication information used, for example, to log

into a website, or for other verification purposes. As such, even though the information may be small in size, it may be desirable to utilize a secure or cryptographic hash, such as MD5 or SHA1. For example, the registry as described elsewhere herein may include authentication information. A set of access permissions may be used in connection with controlling access to the authentication information in the registry. If a copy of the authentication information, rather than the hash of the authentication information, is stored elsewhere for use in connection with the techniques described herein, then a security risk or vulnerability may be created since the same permission control should be duplicated in connection with the copy of the authentication information.

Referring now to Figure 16, shown is an example representation of information that may be used in connection with the techniques described herein. The example 1200 includes information that may be defined and associated with one or more buckets of data elements in an embodiment. It should be noted that although the information included in the example 1200 is illustrated in tabular form, an embodiment may store the information in any one or more different data structures for use in connection with the techniques described herein. Each row in the table includes information about a particular bucket. In this example, each bucket is characterized by seven pieces of information as included in the seven columns of information.

The first column, as denoted by element 1202, corresponds to a defined bucket. In this example, each bucket may be uniquely denoted by a bucket number. However, it should be noted that a bucket may be uniquely identified using any one of variety of different unique identifiers. The second column, as denoted by element 1204, indicates the one or more data stores from which data elements are selected and included in the associated bucket as identified in column 1202. The third column, as denoted by element 1206, indicates the one or more inclusionary rules. The fourth column, as denoted by element 1208, indicates the one or more exclusionary rules. Exclusionary rules may be defined to apply to the entire bucket, or may be defined to apply only to files included in one or more inclusionary rules. For example, a first inclusionary rule may specify to include all files in a first directory. A second inclusionary rule may specify to include all files in a second directory. An exclusionary rule may be defined to exclude all files named by the first inclusionary rule matching the pattern *.LOG. As described elsewhere herein, the combination of the inclusionary rules and any exclusionary rules that may be defined for a particular bucket

determine the one or more data elements included in that particular bucket. The fifth column, as denoted by element 1210, indicates the information to be scanned for each data element in the bucket. The sixth column, as denoted by element 1220, indicates a first resource and an associated allocation or budget of the resource for that particular bucket. The seventh column, as denoted by element 1222, indicates an nth resource and an associated allocation or budget of the resource for that particular bucket. It should be noted that one or more resources may be associated with a particular bucket.

The first row 1230 includes information for a first bucket of data elements. In this example, the first bucket may include data elements from the filesystem and/or the registry as specified in column 1204. The inclusion rules in column 1206 specify that any data element having a name with the pattern "config" is included in the bucket. The exclusion rules in column 1208 specify that any data element having a name with the pattern "*TMP" or "*.LOG" is to be excluded from the bucket. The information to be scanned as set forth in column 1210 includes access permissions. In this example, access permissions may specify the level of access permissions needed in order to access the particular data element. Columns 1220 and 1222 define two resources and an associated budget for each resource. The first resource indicated in column 1220 is CPU time. The budget or allocation of this first resource for bucket 1 is designated as a percentage although an embodiment may also designate an absolute value or threshold such as, for example, a number of CPU seconds. In this example, the allocated percentage is 10% indicating that 10% of the total resource budget available to all the buckets is allocated to bucket 1. When percentages are used to indicate allocations or budgets for the various buckets, it should be noted that the total percentages for any single resource should total 100%. For example, all of the percentages for resource one as indicated in column 1220 should total 100%. Similarly, all of the percentages set forth in column 1222 should total 100%.

In this embodiment, the percentage specified is with respect to a total amount of that particular resource allocated for use in connection with scanning and/or detection processing for a defined resource consumption period. For example, a total of six (6) CPU seconds per minute may be allocated as the overall resource budget for all of the buckets included in the example 1200. The first bucket is allocated 10% of the total 6 CPU seconds, or .6 CPU seconds. The defined resource consumption period in this example may be, for example, one minute of actual or elapsed time. It should be noted that other embodiments may define a

resource consumption period other than as described herein. The particular resource consumption period may vary with the particular resource.

5 Column 1222 indicates a second resource as disk inputs. In this example, disk inputs may represent a metric indicating the number of pages brought in from disk during the scanning and/or detection processing operations for this particular bucket. The number of disk inputs may also be characterized as representing a number of cache misses incurred during the scanning and/or difference detection processing for this particular bucket. In connection with the techniques described herein, the foregoing metric may be measured with
10 respect to a resource consumption period.

It should be noted that in an embodiment in which more than one resource is budgeted for a particular bucket, the resource budget for the bucket may be deemed to be consumed when any single allocated resource has been consumed. When the resource budget for any
15 single allocated resource has been consumed, any unused budget for other resources may be added to the corresponding resource budget for some other bucket.

The second row 1232 includes information for a second bucket of data elements. In this example, the second bucket may include data elements from the filesystem as specified
20 in column 1204. The inclusion rules in column 1206 specify that any data element having a name with the pattern "*.SYS" in the pathname as indicated by "pathname1" is included in bucket 2. There are no exclusion rules defined for bucket 2 in column 1208. The information to be scanned as set forth in column 1210 includes a hash of the data and the file size. In connection with bucket 2, the information to be scanned includes the data itself (e.g.,
25 hash of the data) as well as a portion of its metadata (e.g., file size). For the second bucket, only a single resource is budgeted as indicated in column 1220.

It should be noted that a priority may be associated with each of the buckets. The particular priority may indicate a level of importance of the particular data elements included
30 in that bucket. The priority level may be used in determining the particular resource allocation budget for each of the buckets.

In connection with performing the scanning and/or difference detection in accordance with a specified resource budget allocated amongst the one or more buckets, an embodiment

may utilize any one or more scheduling techniques. For example, an embodiment may scan and perform difference detection with respect to each of the buckets serially or sequentially in order of priority, starting with the first bucket and continuing scanning with the second bucket only after the allocated resource(s) for bucket one have been consumed. The scanning and difference detection processing for each of the buckets may be performed in a cyclic fashion such that when processing has been completed for the last or lowest priority bucket, processing then continues with the first or highest priority bucket. Particular examples of scheduling techniques that may be included in an embodiment are described in more detail in following paragraphs. The amount of one or more resources that are consumed may be measured with respect to processing associated with scanning and/or difference detection.

Referring now to Figure 17, shown is a flowchart of processing steps that may be performed in an embodiment in connection with the techniques described herein. The steps of flowchart 1250 summarize processing as just described. At step 1252, one or more buckets and related information for each of the buckets may be defined. Step 1252 may include, for example, specifying the information included in the table 1200 of the Figure 16. At step 1254, scanning and difference detection processing may be performed. As part of step 1254 processing, changes or differences of the scanned information may be determined. At step 1256, associated processing may be performed in accordance with the detected differences or changes. In one embodiment, alerts may be generated as part of step 1256 processing in connection with selected detected changes. From step 1256, control proceeds to step 1254. As described herein in one embodiment, the processing of steps 1254 and 1256 can be continually repeated as the entity performing the processing continually executes. The processing of steps 1254 and 1256 may utilize the information set forth in step 1252 for each bucket and in accordance with a defined schedule. Additionally, resource budget or allocation as may be defined in step 1252 may be performed with respect to resource consumption attributable to scanning and/or difference detection processing. In following paragraphs, described are examples and embodiments in which the resource budget and resource consumption are with respect to both the scanning and difference detection processing. This should not be construed as a limitation of the techniques described herein. An embodiment may utilize the resource budget and resource consumption with respect to processing for scanning and/or difference detection. As described herein, for example, the process performing the scanning and difference detection may be a virus scanner. As such, an embodiment may measure resource consumption and the resource budget or allocation

specified for a bucket with respect to both the scanning and difference detection processing performed by the process. Alternatively, an embodiment may have a first process perform the scanning and a second different process performing the detection processing. The latter embodiment may elect to measure the resource consumption and resource budget or allocation with respect to only one of the first or second process.

Referring now to Figure 18, shown is a flowchart of processing steps that may be performed in an embodiment in connection with defining buckets and associated information for each bucket. The steps of a flowchart of 1300 set forth more detailed processing of step 1252 of the Figure 17. At step 1302, the number of buckets to be included in an embodiment is determined. At step 1304, a determination is made as to whether the information has been specified for each of the buckets. If so, processing stops. Otherwise, control proceeds to step 1306 to determine the one or more data stores from which the data elements for the current bucket are to be selected. As described and illustrated elsewhere herein, a single data store may be specified for a bucket. Alternatively, multiple data stores may be specified for a bucket. The multiple data stores may be of the same or of different types. As one example illustrated by row 1230 of Figure 16, data elements of a bucket may be selected from the filesystem and the system registry. For example, a bucket may be defined to include data elements that may be characterized as critical security data elements for a system. The bucket may include a number of registry items and a set of files. At step 1308, rules may be specified to define data elements to be included in the bucket. Inclusionary and/or exclusionary rules may be specified. At step 1310, the information to be scanned and monitored for changes with regard to each data element is specified. As described herein, the information specified at step 1310 may include data, metadata, and/or schema information about each data element. At step 1312, one or more resources and an amount of each resource allocated for use with the current bucket are specified. The amount may be determined using one or more specified metrics. The amount of each resource totaled over all the buckets may be characterized as the total budget for that resource for scanning and/or difference detection purposes with respect to a resource consumption period. The quantity of the total budgeted amount of the resource specified in step 1312 indicates a portion of that total for the current bucket. Control proceeds to steps 1314 and 1304 to process any remaining buckets.

What will now be described are scheduling techniques that may be included in an embodiment. The technique will be illustrated with respect to a single metric, an amount of CPU time. Although specific examples are set forth in following paragraphs, it will be appreciated that an embodiment may utilize any one of a variety of different scheduling algorithms and variations of what is set forth herein.

Additionally, as pointed out elsewhere herein, it should be noted that the resource allocations or budgets and resource consumption for a bucket may be used in connection with performing scanning and/or difference detection.

The total budget for the resource, the CPU, may be measured by an amount of CPU time. In one embodiment, the total budgeted amount of CPU time may be 1.8 seconds or 3% of each minute. If there are 2 buckets, the 1.8 seconds of each minute is apportioned between the two buckets in accordance with the priority of each bucket. For example, if bucket 1 includes data elements of a higher priority than bucket 2, bucket 1 may be allocated 1.2 seconds and bucket 2 may be allocated the remaining .6 seconds. The scanning process which performs the scanning and difference detection of the data elements is scheduled in accordance with the total amount of budgeted CPU time for a defined resource consumption period. Additionally, the scanning process apportions its total amount of budgeted CPU time in accordance with the per bucket budgeted allocations. In connection with the foregoing example, it should be noted that there may be a similar number of data elements in each bucket. An embodiment may consider the per-data element budget allocation which may vary with the number of data elements in each bucket and the total resource budget of each bucket and accordingly select values for the foregoing so that the per-data element budget of a data element in the higher priority bucket is greater than the per-data element budget of a data element in the lower priority bucket.

As known in the art, a process in a time-share computing system may be in one of several defined states. In one embodiment, an embodiment may be in a READY state if the process is READY for execution. There may be a queue of one or more processes in the READY state to be executed by the CPU. A process may also be in a WAIT or NOT READY TO EXECUTE state, for example, if the process is waiting for its scheduled execution time. When the process is in the WAIT state, the process may be characterized as being "asleep". In one embodiment, a process may be placed in the WAIT state for a

This may be done, for example, to ensure that the scanning and difference detection is performed while also further minimizing the perceived impact of scanning and difference detection on other processes having the same execution priority level as the scanning process.

In one embodiment, the scanning process may execute at a normal priority level as may be the default priority level for other user tasks and processes. Steps may be taken in an embodiment to interleave the execution of other processes in between an amount of scanning performed. The amount of scanning and difference detection performed may be specified using one or more predetermined scanning criteria. For example, the scanning process may perform a sleep(0) after each scan of a data element causing the scanning process to go to the end of the READY queue after each data element is processed. As such, any other processes which are in the READY queue are then executed before processing the next data element. If there are no other processes in the READY queue, the scanning process resumes execution with the next data element. The processing just described in which the scanning process performs an additional sleep(0) between data elements scanned may be performed as an alternative to the processing steps of flowchart 1450. An embodiment may also specify other scanning criteria besides performing a sleep(0) between successively processed data elements, or hashes thereof. For example, an embodiment may specify that a sleep(0) is performed between a predetermined number of scans in which the number is greater than one. Since the amount of processing performed when scanning and/or performing difference detection associated with each data element may vary, an embodiment may specify that a sleep(0) is performed between successive processing of a predetermined amount of data. To generalize the foregoing, an embodiment may also perform the sleep(0) after consuming a fixed amount of any one or more resources. This may include, for example, performing a sleep(0) after utilizing a portion of the allocated CPU time for one resource consumption period. To further illustrate, if a first bucket has a resource allocation for a resource consumption period of 6 seconds per minute, the scanning process may perform a sleep(0) each time .5 CPU seconds are consumed. It should be noted that the use of the sleep(0) as just described effectively performs similar processing as may be associated with time slicing when the scanning process is swapped out because its specified time slice has elapsed.

Referring now to Figure 21, shown is a flowchart of processing steps that may be performed in an embodiment in connection with scheduling the scanning process. The steps of flowchart 1500 summarize processing just described and may be performed as an alternative to processing steps of flowchart 1450. Note that steps 1502 and 1508 are

specified time period using a system defined function call such as the sleep function which takes as a parameter an amount of time a process is to remain asleep, or in the WAIT state. When the scheduled execution time arrives (e.g., the amount of time specified as an argument to the sleep function has elapsed), the scanning process is awakened by the operating system and placed on the READY queue. The scanning process is executed in accordance with its position in the READY queue since there may be more than a single process which is in the READY queue waiting for execution by the CPU. It should be noted that any priority assigned to a process may affect its position in the READY queue as well as other variations from what is described herein.

In the example described in following paragraphs, the scanner or scanning process may perform processing for scanning and difference detection. As described herein, the difference detection may include comparing results from a current scan to other results for a data element to determine any differences therebetween. The scanning process begins executing at a first time, X. The absolute value of the start time X may be any time. For purposes of illustration, let X represent a time which is 15 seconds after the start of the previous minute of wall clock or actual time. The scanner process scans and performs difference detection processing for its total budgeted time of 1.8 CPU seconds allocated its 1.8 seconds in accordance with the budgeted amounts for bucket 1 and bucket 2. When the total CPU time of 1.8 seconds has been consumed, the actual end time, Y, is denoted. The amount of elapsed time, Y-X, is determined. It should be noted that the difference or the elapsed time may be longer than the 1.8 seconds of budgeted CPU time. For example, the CPU may have performed other tasks causing execution of other processes during the elapsed time. The scanning process may have been swapped out to allow a higher priority task. The scanning process may have been delayed waiting for information to be brought into memory from an external storage device. The foregoing are just several reasons why the elapsed time may be greater than the amount of budgeted CPU time. It is determined if the elapsed time is greater than one resource consumption period, such as one minute of actual time. In other words, the budgeted amount in this example is that 1.8 seconds out of every minute of CPU time is allocated to scanning. If more than one minute has elapsed, then the scanning process has not consumed its budgeted time of 1.8 seconds for the last one CPU minute. As such, if more than one minute has elapsed, then the scanner is scheduled to execute at the next available time. The scanning process may perform a sleep(0) placing it at the end of the READY queue.

If the elapsed time is equal to or less than one minute, then the scanning process goes to sleep/waits for an amount of time, $60-(Y-X)$, and is then scheduled for execution. In one embodiment, the scanning process may be placed in the WAIT state for the time $60-(Y-X)$, after which the scanning process is awakened and placed at the end of the READY queue. For example, if the elapsed time is 40 seconds, the scanning process may perform a sleep (20 seconds).

It should be noted that when the scanner process does a sleep(0) or is otherwise placed at the end of the READY queue, if there are no other processes waiting for execution, then the scanner process may be immediately executed. It should be noted that the foregoing example, unless otherwise mentioned, is set forth with respect to all processes executing at a same priority level as the scanning process. For example, an embodiment may execute higher priority processes prior to those of lower priorities affecting the position of the scanning process in the READY queue as a result of a sleep(0). Operating systems may also implement other techniques which affect the execution order and position in the READY queue. Such techniques may vary with each embodiment.

In the foregoing, steps are taken to schedule the scanning process for execution for its budgeted time every resource consumption period which, in this example, is a minute. It should also be noted that the actual elapsed time, $Y-X$, as well as subsequent start times (e.g., X_s) may change between scheduled execution times of the scanning process.

Referring now to Figure 19, shown is a flowchart of processing steps that may be performed in an embodiment in connection with scheduling the scanning process. It should be noted that the processing steps of flowchart 1400 summarize scheduling as just described. The steps of flowchart 1400 present more detailed processing as may be performed in connection with step 1254 of Figure 17. At step 1402, the scanning process is ready for execution. This may be indicated by the scanning process being placed in the READY queue. At step 1404, the scanning process begins execution at a time X . At step 1406, the scanning process performs scanning for its total resource allocation budget as specified for a resource consumption period. As described above, the scanning process allocates its total budget in accordance with the budgeted amount for each bucket. At step 1408, the scanning process ends at a time Y . At step 1410, the actual elapsed time, $Y-X$, is determined. At step

1412, a determination is made as to whether the actual elapsed time is more than one resource consumption such as, for example, one minute. If step 1412 evaluates to no, control proceeds to step 1414 where the scanning process is placed at the end of the READY queue to perform scanning at the next available time. As described above, this may be implemented in an embodiment using the sleep function and by issuing a sleep (0). It should be noted that as described herein, step 1414 processing may be optionally performed in an embodiment. In other words, an embodiment utilizing the techniques herein may perform processing steps of flowchart 1400 omitting step 1414. From step 1414, control proceeds to step 1402 since, in this example, the scanning process may execute continuously.

If step 1412 evaluates to yes, control proceeds to step 1416 where the scanning process goes to sleep until the start of the next resource consumption as determined at step 1410. At step 1418, the scanning process is awakened at the start of the next resource consumption period. In the example described herein, the scanning process is placed at the end of the READY queue when awakened so that the process can begin execution at the next available time.

Referring now to Figure 20, shown is an example of a flow chart of processing steps that may be performed in an embodiment by the scanning process. The steps of flowchart 1450 set forth more detailed processing that may be performed by the scanning process when performing scanning and difference detection processing in accordance with the budgeted resource amounts for each bucket. The flowchart 1450 sets forth more detailed processing as may be performed in an embodiment in connection with step 1406 of Figure 19. At step 1452, a determination is made as to whether the current resource is consumed for the current bucket. If so, control proceeds to step 1454 to continue processing (e.g., scanning and difference detection) with the data element from the next bucket. From step 1454, control proceeds to step 1458. If step 1452 evaluates to no, control proceeds to step 1456 where processing continues with a next data element from the current bucket. At step 1458, a determination is made as to whether all budgeted resources, or the total budgeted amount for all buckets (e.g., 1.8 seconds), have been consumed. If so, processing stops. Otherwise, control proceeds to step 1452.

An embodiment may also perform processing to further ensure that other processes executing at a same priority level as the scanning process are executed in a timely fashion.

analogous, respectively, to step 1452 and 1458 of flowchart 1450. At step 1502, a determination is made as to whether the scanning process has consumed the resources specified for the current bucket for a resource consumption period. If step 1502 evaluates to yes, control proceeds to step 1504 to perform processing (e.g., scanning and difference
5 detection processing) using a data element from the next bucket in accordance with the predetermined scanning criteria. Control then proceeds to step 1508.

If step 1502 evaluates to no, control proceeds to step 1506 to perform processing using a data element from the current bucket in accordance with the predetermined scanning
10 criteria. At step 1508, a determination is made as to whether all budgeted resources (e.g., the total amount budgeted across all buckets) have been consumed. If so, processing stops. Otherwise, control proceeds to step 1510 to place the scanning process at the end of the READY queue such as by performing a sleep(0). As described above, the predetermined criteria may include one or more conditions. In one embodiment, the criterion may include a
15 single condition causing a sleep(0) to be performed after performing scanning and difference detection with respect to each data element, or after processing a predetermined amount.

An embodiment using the techniques of flowchart 1500 performs processing in which the scanning process is interrupted between defined occurrences, such as between processing
20 (e.g., scanning and difference detection) for one or more data elements, processing a predetermined amount, and the like. Use of such techniques may further minimize any perceived impact of scanning and difference detection processing on the performance of other processes executing at the same priority level as the scanning process.

Other techniques may be utilized in an embodiment in connection with implementing scheduling. For example, rather than perform processing of the flowchart 1400 of Figure 19, an embodiment performing scheduling utilizing priority changes of the scanning process. For example, if the scanning process has a total resource allocation budget of 1 out of every 10 seconds, the scanner may run continuously and measure the values of X and Y as illustrated
30 in flowchart 1400. However, the scanning process may not be scheduled to execute, or put to sleep, for a defined time period. Rather, the scanner's execution priority may be adjusted up or down so that on average, the scanner executes 1 out of every 10 elapsed seconds. In such an embodiment, a running average of the amount of time the scanner has executed during a

resource consumption period may be maintained for multiple resource consumption periods (e.g., multiple 10 second intervals).

Referring now to Figure 22, shown is a flowchart of processing steps that may be performed in an embodiment in connection with scheduling. The steps of 1600 summarize processing just described and may be performed as an alternative to processing of flowchart 1400. It should be noted that a portion of the steps of flowchart 1600 are similar to those of flowchart 1400 as so numbered. As a result of steps 1402, 1404, 1406 and 1408, the scanning process is executed and values for X and Y are determined. At step 1602, the actual elapsed time is determined (e.g., $Y - X$) and the running average amount of execution time is updated. The foregoing average may be represented as the average of the elapsed time values over multiple resource consumption periods. At step 1604, a determination is made as to whether the average is more than the budgeted average. If step 1604 evaluates to yes, it means that, on average, the scanner is executing for more than its allocated or budgeted total time. Control proceeds to step 1608 to adjust the priority downward and then place the scanning process in the READY queue to be executed in accordance with its adjusted priority. If step 1604 evaluates to no, control proceeds to step 1414 where a determination is made as to whether the measured average is the same as the budgeted amount. It should be noted that the determination at step 1414 may be performed with respect to a window or threshold so that if the measured average is within the range of budgeted amount $\pm$ window, step 1414 evaluates to yes. If step 1414 evaluates to yes, control proceeds to step 1418. If step 1414 evaluates to no, meaning that the scanning process is not consuming its budgeted resource(s), control proceeds to step 1606 to adjust the priority upward. From step 1606, control proceeds to step 1418. At step 1418, the scanner process is placed in the READY queue for execution in accordance with its adjusted priority. Control then proceeds to step 1402.

In connection with the techniques described herein, data elements may be classified and prioritized and accordingly assigned a resource budget of one or more resources. The resource budget may be defined so that the scanning process consumes an amount of a defined resource consumption period partitioned among one or more buckets.

An embodiment may also utilize techniques described herein in which a scanning process voluntarily relinquishes control of the CPU and lets another currently waiting process

execute independent of any operating system implemented technique for sharing the CPU, such as time slicing. For example, as described above, the scanning process may perform a sleep(0) in between successive data element scans or after scanning a predetermined amount of data. By performing the sleep(0), the currently executing scanning process voluntarily
5 relinquishes control of the CPU so that any waiting process on the WAIT queue is allowed to execute prior to resuming scanning. The foregoing may be performed within a defined resource consumption period prior to the scanning process consuming its total allocated resources for the defined resource consumption period.

10 It should be noted that in connection with existing scanners, such as virus scanner, the scanning process may execute as a background task but may still consume varying portions of cache thereby adversely impacting a subsequently executed process or task. For example, when a scanner executes as a background task, the scanner may displace a certain number of pages from cache swapping out pages of an application. As the scanner scans files, the files
15 are brought into cache (e.g., measured as disk input pages as described elsewhere herein) and may displace cache portions containing pages for an application. When the application again resumes execution, the application experiences cache misses since its pages have been displaced by the scanner previously executing as a background task. Accordingly, the application's pages are again brought in from disk and into cache causing a performance
20 penalty for the application. In other words, when a foreground application resumes execution, the application finds that its pages have been displaced from cache. These pages may be reloaded from disk causing additional disk inputs to be performed for the application. Thus, even though the scanner has executed in the background and is no longer executing, it may adversely impact a subsequently executed foreground application process since the
25 application's pages have been displaced from cache. System resources are utilized to resume execution of the application once the scanner has been swapped out (e.g., is no longer is executing) by fetching any displaced pages. In connection with the techniques described herein, such adverse impacts may be minimized. Using the techniques and scheduling described herein, the scanner is not typically able to displace large amounts of other pages
30 utilized by other running foreground applications during a single resource consumption period.

It should be noted that in connection with the forgoing techniques, it is possible that scanning of all data elements within a bucket may complete within the resource allocation

budget for that bucket. In other words, the scanning process may have processed each data element in the bucket and not consumed the entire resource allocation budget. In such instance, an embodiment may continue by rescanning data elements in the same bucket until the resource budget is consumed. An embodiment may alternatively disregard any
5 unconsumed portion of the budgeted resource for the current bucket and continue with scanning of the next bucket in accordance with the resource budget of the next bucket. As yet another alternative, an embodiment may continually add any unconsumed portion of the budgeted resource for the current bucket to the budget associated with a next bucket and continue with scanning of the next bucket. After all buckets have been processed in a first
10 cycle, any remaining or unconsumed budget may be discarded.

What will now be described in the following paragraphs relates to techniques that may be used in connection with alerts, such as performance alerts and/or security alerts representing information about performance and/or security conditions, respectively.

15 An embodiment may model alarms or alerts using a state machine as described herein. As known in the art, the state machine may be specified by defining a set of states and transitions between the states. As described herein, a severity may be associated with one or more states. The model may utilize the overlapping or non-overlapping technique and may
20 perform processing for the skipping or no skipping variation of either of the foregoing. In connection with modeling alerts, the model may also include one or more additional rules or model parameters. The rules may specify whether alerts are to be generated when transitioning between severities or between states. Furthermore, the rules may also indicate whether an alert is generated when transitioning into any state or severity, or only when
25 transitioning into one or more selected states. For example, an embodiment may define a first model in which alerts are output in connection with transitioning between severities. The first model may also include a rule which states that alerts are output when transitioning into any severity level except the normal severity level. Another embodiment may define a second model in which alerts are output in connection with transitioning between states rather
30 than severities. The second model may include a rule which states that alerts are output only when transitioning into a specified list of one or more states. One metric which may be measured, as will be described in following paragraphs, is a "duration" metric representing the amount of time or duration recorded for a particular alert as may be associated with one

or more states. In other words, the duration metric may be used to track state changes as may be defined in accordance with an underlying state machine model.

An embodiment can measure the “duration” metric representing the amount of time a system may be characterized as being in a particular condition in different ways that vary in accordance with an embodiment. Different ways in which the duration metric may be measured are described in following paragraphs. Additionally, an embodiment may include a measured “duration” in some unit of time in a display in connection with both security alerts and performance alerts.

Alert messages may be displayed, for example, on a display device of a computer system. In existing systems, both security and performance alerts may be included in the same display. Alerts may be characterized as being an instantaneous alert or a non-instantaneous or condition-based alert. An instantaneous alert is a single reported alert occurrence having no duration, or a duration time of 0. For example, security alerts tend to generally be instantaneous alerts reporting on a single occurrence, such as a failed password attempt. In contrast, non-instantaneous or condition-based alerts are not based on a single occurrence. Rather, a condition associated with the alert may persist for a time period having a non-zero duration such as, for example, a high CPU usage condition. A condition which persists for some time may cause more than one corresponding reported alert message to appear on an alert display. For example, performance alerts tend to generally be non-instantaneous alerts associated with a condition having a duration. Non-instantaneous alerts may be modeled in a variety of different ways, for example, using a state-based model in which one or more states may be associated with each alert severity level. Systems may issue various alerts as described by such models.

Referring now to Figure 23, shown are examples illustrating how performance and/or security alerts may be included in a display for reporting purposes. In the example 1700, the element 1710 represents a generic list of alert messages as may be viewed on such a device. The list may include both performance and security alerts intermixed. Additionally, the list may include alerts in connection with both instantaneous and non-instantaneous occurrences or conditions. As a further example, element 1720 illustrates a list of alerts including more detail regarding the alerts generally depicted in 1710. To an operator viewing such a list as in 1720, the list can be confusing due to the intermixing of both security and performance alerts

including both the instantaneous and non-instantaneous alert messages. More than one alert message may be associated with a non-instantaneous condition. For example, the portion 1722 includes multiple alerts messages that appear to be associated with one or more related performance conditions. However, it may not be readily apparent to one viewing the
5 information which alert messages are related such as, for example, when many alerts from many sources need attention from a human operations. It may also not be easy to derive information from the display regarding the particular alerts.

The techniques described herein utilizing a duration metric may be used in connection
10 with non-instantaneous conditions. Use of the duration metric may provide for a clearer presentation of the alert messages and associated conditions. For example, non-instantaneous conditions may have multiple alert messages associated with the same condition or a progression of one or more severity levels. Use of the duration metric may result in combining information from one or more alert messages as may be displayed in an existing
15 system thereby reducing the amount of information on the alert display and placing it into a more readable and understandable form.

It will be appreciated by those skilled in the art that the non-instantaneous alerts may be modeled in an embodiment, such as in process control systems and network monitoring
20 systems, using any one of a variety of different implementations that may be used in connection with the duration metric and techniques described herein. For example, the non-instantaneous conditions may be modeled using boolean state machines, with only two states reflecting a measured value, threshold-value state machines, where every state has a corresponding range of values for one or more measurement points and the current state for
25 the state machine is the one that corresponds to the value range into which the current measured value input falls, as well as more complex state machines, such as those where state changes may be triggered not only by new measured values, but by delays and timeouts, such as state machines which reduce state-change "jitter" by delaying state changes until the inputs which would otherwise have caused a state change result in a new state persisting for
30 some period of time. Additionally, in the more complex state machines, state changes may be triggered by more complex conditions, for example, operator commands which are awaiting approval or acknowledgement, or by inputs from other state machines, such as those in alarm aggregator systems where a higher level condition may be triggered by one or more other inputs or lower level conditions.

In connection with using such state-based models, a severity level may be associated with one or more states as will be illustrated in connection with the techniques described herein.

5 The duration metric may be applied to any state machine. The duration of a condition in some part of a physical or logical system may be defined as the length of time in which a state machine modeling that part of the system spends in one or more predefined states, which can be called the states corresponding to the condition. While the duration metric can
10 be applied to any state machine, reference will often be made, by way of example, to a simple threshold-value state machine in order to illustrate the new duration metric.

As used in connection with descriptions herein, for any time T after an alert condition begins, an alert corresponding to the condition may be characterized as being open at T if the
15 duration metric for the alert is increasing at time T. An alert may also be characterized as being closed at such time T if the duration metric for the alert is no longer increasing.

It should be noted that in connection with the descriptions and examples herein, reference is made to transitioning between severities or transitioning from one severity to
20 level to another in connection with a related condition. The foregoing transitioning with respect to severities is in connection with the transitioning between the underlying states of a state machine and in which the states are associated with the severities.

25 Referring now to Figure 24, shown is an example of thresholds that may be used in connection with a 4 state threshold value based state machine used to model non-instantaneous alerts. In this example 1800, each state is associated with a single unique severity level although an embodiment may associate more than one state with each unique severity level. In this example, when a measured value exceeds threshold V1, a low severity alert condition exists. When a measured value exceeds threshold V2, a medium severity alert
30 condition exists. When a measured value exceeds threshold V3, a high severity alert condition exists. For example, the foregoing threshold values may be associated with performance conditions in an embodiment regarding memory usage on a computer system. Each threshold level V1-V3 may be associated with a percentage of in-use system memory.

Referring now to Figure 25 shown is an example illustrating a series of measurements taken at various times. Each such measurement is represented by one of the data points (e.g., x, y, z, xx, yy, zz, and q) on the graph of 1900. The horizontal lines at each of V1, V2 and V3 graphically represent the threshold values from Figure 24. Each of the denoted elements a, b, c, d, and e along the horizontal time axis corresponds to a time period or time window. Any measurement taken during time periods a and e correspond to a low severity alert. Any measurement taken during time periods b and d correspond to a medium severity alert. Any measurement taken during time period c corresponds to a high severity alert.

What will now be described and illustrated are various ways in which the duration metric may be determined in an embodiment. An embodiment may render alerts in accordance with a non-overlapping duration determination technique or an overlapping duration determination technique.

In connection with the non-overlapping duration, at most a single alert may be open at a point in time for an associated condition. As a condition transitions from a first severity to a second severity the first severity alert is characterized as closed or ended, and the second severity alert is opened or started. When computing the duration time, an end time associated with an alert is determined when closing the alert and a start time associated with an alert is determined when opening the alert.

In connection with the overlapping duration, more than one alert may be open at a point in time for a single alert condition so that the time period of duration of one alert may overlap with respect to the time period of duration of a second different alert. As a condition transitions from a lower severity to a higher severity, an alert is opened for the higher severity, and the alert for the lower severity remains open. As a condition transitions from a higher severity to a lower severity, the higher severity alert is closed.

Additionally, in connection with both the overlapping and non-overlapping duration techniques, an embodiment may apply either a state skipping variation or a non-skipping variation. When transitioning from a first severity to a second severity, the underlying model of the associated condition may include one or more other severities which intervene between the first and second severities. For example, a first measurement may be associated with a

low severity. The second measurement, being the next consecutive measurement taken in time, may be associated with a high severity. The underlying model of the associated condition may also have a medium severity intervening between the low and high severities. When using the skipping variation, no alerts are opened for the intervening severities. For example, in connection with the foregoing example transitioning from low to high severity, no alert is opened for the medium severity when transitioning into the high severity. When using the non-skipping variation, alerts may be opened for the intervening severities. For example, in connection with the foregoing example transitioning from low to high severity, an alert is opened for the medium severity when transitioning into the high severity. When transitioning from a higher severity to a lower severity using the skipping variation, an alert is opened for the lower severity if not yet opened.

The foregoing is illustrated in more detail in connection with following examples and explanations set forth herein.

Referring now to Figure 26, shown is an example illustration in an embodiment utilizing the overlapping technique in connection with the graph of Figure 25. It should be noted that for purposes of illustration of the overlapping and non-overlapping technique, an example is used in which no severities are skipped when transitioning between states using the values as indicated in Figure 25. Each of the boxes included in 1922-1932 in Figure 26 represent a subset of how relevant alerts may be rendered on a computer display in an embodiment. Element 1922 includes information relevant to the value y obtained at a time T1 indicating a transition from a normal severity to a low severity. A low severity alert may be opened and the information as included in the bottom portion of 1922 may represent the display of an alert console. Note that in an embodiment, the duration time may be represented as ONGOING indicating that the associated alert is open and ongoing and has not subsided. Other embodiments may utilize other visual indicators to represent the fact that the displayed alert is open and ongoing. Additionally, as will be appreciated by those skilled in the art, other embodiments may display the alert messages as indicated in connection with examples herein in different orders than as shown and in connection with other information. T1 is the starting time of the duration associated with the low alert just opened.

Element 1924 includes information relevant to the value z obtained at a time T2 indicating a transition from a low severity to a medium severity. A medium severity alert

may be opened and the information as included in the bottom portion of 1924 may represent the display of an alert console. Note that there are two alerts opened as indicated by the ONGOING duration. T2 is the starting time associated with the medium alert just opened.

5 Element 1926 includes information relevant to the value xx obtained at a time T3 indicating a transition from a medium severity to high severity. A high severity alert may be opened and the information as included in the bottom portion of 1926 may represent the display of an alert console. Note that there are 3 alerts opened as indicated by the ONGOING duration. T3 is the starting time associated with the high alert just opened.

10

 Element 1928 includes information relevant to the value yy obtained at a time T4 indicating a downward severity transition from a high to medium. When performing a downward transition to a target severity, any open alerts associated with a severity level higher than the target severity are closed. The information as included in the bottom portion
15 of 1928 may represent the display of an alert console. In connection with 1928, the high severity alert is closed and the ending time for the duration is determined as T4-T3.

 Element 1930 includes information relevant to the value zz obtained at a time T5 indicating a transition from a medium severity to a low severity. The information as included
20 in the bottom portion of 1922 may represent the display of an alert console. In connection with 1930, the medium severity alert is closed since a transition is made to the low severity. The ending time of the duration for the medium severity alert is T5 and the duration may be determined as represented by the time interval T5-T2.

25 Element 1932 includes information relevant to the value q obtained at a time T6 indicating a transition from a low severity to a normal severity. The bottom portion of 1932 may represent the display of an alert console. In connection with 1932, the low severity alert is closed since a transition is made to the normal severity. The ending time of the duration for the low severity alert is T6 and the duration may be determined as represented by the time
30 interval T6-T1.

Referring now to Figure 27, shown is an example illustration in an embodiment utilizing the non-overlapping technique in connection with the graph of Figure 25.

As in connection with Figure 26, reference will be made to elements including information as may be displayed on an alert console at each point in time when a next value is obtained at a corresponding point in time. Element 1964 represents the alerts open at each point in time.

5 Element 1952 includes information relevant to the value y obtained at a time T1 indicating a transition from a normal severity to a low severity. A low severity alert may be opened and the duration is indicated as ONGOING. Time T1 is the starting time of the duration associated with the low alert just opened.

10 Element 1954 includes information relevant to the value z obtained at a time T2 indicating a transition from a low severity to a medium severity. A medium severity alert may be opened as indicated by the ONGOING duration and the low severity alert is closed. T2 is the starting time associated with the medium alert just opened. In connection with the non-overlapping approach, at most one alert may be opened as illustrated by 1964. The
15 duration for the low severity alert may be determined as T2-T1. As such, with each transition at a time Tn from a source severity to a target severity, an alert associated with the source severity is closed, an ending time Tn of the duration is determined for the source severity alert, a new alert associated with the target severity is opened, and a starting time Tn of the duration is determined for the newly opened alert.

20 Element 1956 includes information relevant to the value xx obtained at a time T3 indicating a transition from a medium severity to a high severity. A high severity alert may be opened as indicated by the ONGOING duration and the medium severity alert is closed. T3 is the starting time associated with the high alert just opened. The duration for the
25 medium severity alert may be determined as T3-T2.

30 Element 1958 includes information relevant to the value yy obtained at a time T4 indicating a downward severity transition from high to medium. A medium severity alert may be opened as indicated by the ONGOING duration and the high severity alert is closed. T4 is the starting time associated with the medium alert just opened. The duration for the high severity alert may be determined as T4-T3.

 Element 1960 includes information relevant to the value zz obtained at a time T5 indicating a transition from a medium severity to a low severity. A low severity alert may be

opened as indicated by the ONGOING duration and the medium severity alert is closed. T5 is the starting time associated with the low alert just opened. The duration for the medium severity alert may be determined as T5-T4.

- 5 Element 1962 includes information relevant to the value q obtained at a time T6 indicating a transition from a low severity to a normal severity. The low severity alert is closed. The duration for the low severity alert may be determined as T6-T5.

- 10 In connection with defining the model associated with the foregoing example of Figures 24-27, an embodiment may define a rule specifying that one or more of the severities or states are not alert-generating so that transitioning into such severities or states does cause an alert message to be generated. In connection with the foregoing example, an embodiment may define the normal severity to be such a non-alert generating severity.

- 15 What will now be presented are examples to further illustrate the techniques described herein in connection with the duration techniques. In the following example, a severity level may be associated with more than one state. Additionally, example values observed will be used to illustrate the skipping and non-skipping variations of both the non-overlapping and overlapping techniques.

- 20 Referring now to Figure 28, shown is a graphical representation of threshold values used in connection with a 5 state model having 3 severity levels. It should be noted that a low alert state and a high alert state are both associated with a same severity level S2, and a low warning state and a high warning state are both associated with severity level S1.

- 25 Referring now to Figure 29, shown is an example illustrating a series of measurements taken at various times. Each such measurement is represented by one of the data points (e.g., x, y, z, and zz) on the graph of 2000. The horizontal lines at each of V1, V2, V3, and V4 graphically represent the threshold values from Figure 28. Each of the denoted
30 elements a-k along the horizontal time axis corresponds to a time period or time window.

Referring now to Figure 30, shown is an example illustration in an embodiment utilizing both the overlapping and the non-overlapping techniques with skipping in connection with the graph of Figure 29. As in connection with other examples, reference will

be made to elements including information as may be displayed on an alert console at each point in time when a next value is obtained at a corresponding point in time.

The use of the overlapping technique with skipping is represented using elements 5 2120 and 2124. The use of the non-overlapping technique with skipping is represented using elements 2122 and 2126. Elements 2124 and 2126 represent the alerts open at each point in time in connection with each respective technique.

In connection with the overlapping technique with skipping, element 2102 represents 10 time T1 when the transition is made from severity S0 to S2. As a result, an S2 alert is opened. T1 is the starting time of the duration for the S2 alert. Element 2104 represents time T2. As illustrated, a transition is made from state High Alert to state Low Alert. However, in this example, each state is associated with the same alert severity and thus both states are associated with the same open alert. At time T2, the S2 alert is noted as ONGOING. At time 15 T3, a transition is made from the High Alert state to the Normal state. The severity transitions from S2 to S0 resulting in closing the alert for S2. The ending time of the duration for the S2 alert is T3 and the duration time may be measured as T3-T1.

In connection with the non-overlapping technique with skipping, element 2110 20 represents time T1 when the transition is made from severity S0 to S2. As a result, an S2 alert is opened. T1 is the starting time of the duration for the S2 alert. Element 2104 represents time T2. As illustrated, a transition is made from state High Alert to state Low Alert. However, in this example, each state is associated with the same alert severity and thus both states are associated with the same open alert. At time T2, the S2 alert is noted as 25 ONGOING. At time T3, a transition is made from the High Alert state to the Normal state. The severity transitions from S2 to S0 resulting in closing the alert for S2. The ending time of the duration for the S2 alert is T3 and the duration time may be measured as T3-T1.

Referring now to Figure 31, shown is an illustration of the foregoing example using 30 the overlapping technique without skipping. Element 2202 represents time T1 when the transition is made from severity S0 to S2. As a result, an S2 alert is opened. T1 is the starting time of the duration for the S2 alert. Additionally, the severity S1 is skipped since there is a transition directly from S0 to S2 at time T1. In an embodiment utilizing the non-skipping variation, an alert is also opened for severity S1 having a starting time of T1. Both

alerts are open as indicated by the ONGOING duration. Element 2104 represents time T2. As illustrated, a transition is made from state High Alert to Low Alert. However, in this example, each state is associated with the same alert severity and thus both states are associated with the same open alert severity S2. Neither alerts for S1 or S2 are closed since
5 the transition to the target severity S2 does not represent a downward or reduced severity transition with respect to either open alert for S1 or S2. At time T2, the S1 and the S2 alerts are noted as ONGOING. At time T3, a transition is made from the High Alert state to the Normal state. The severity transitions from S2 to S0 resulting in closing the alert for S1 and S2. The ending time of the duration for the S1 and S2 alerts is T3 and the duration time for
10 both may be measured as T3-T1. Element 2212 indicates the open alerts at each time.

Referring now to Figure 32, shown is an illustration of the foregoing example using the non-overlapping technique without skipping. Element 2302 represents time T1 when the transition is made from severity S0 to S2. As a result, an S2 alert is opened. T1 is the
15 starting time of the duration for the S2 alert. Additionally, in connection with the non-skipping variation, an alert for S1 is opened and also closed with the duration time of zero. Element 2304 represents time T2. As illustrated, a transition is made from state High Alert to state Low Alert. However, in this example, each state is associated with the same alert severity and thus both states are associated with the same open alert. At time T2, the S2 alert
20 is noted as ONGOING. At time T3, a transition is made from the High Alert state to the Normal state. The severity transitions from S2 to S0 resulting in closing the alert for S2. The ending time of the duration for the S2 alert is T3 and the duration time may be measured as T3-T1. Additionally, in connection with the non-skipping variation, an alert for S1 is opened and also closed with the duration time T3-T2. In connection with the non-skipping variation
25 when transitioning to a lower severity, alerts are opened and closed for any intervening severities. Element 2320 indicates the open alerts at each time.

As a further illustration and example in connection with Figure 29, measurements taken
30 in time periods (a) and (c) have severity S1, and a measurement in period (b) has severity S2. Using the non-overlapping duration technique, three alerts may be rendered: one for each time period (a), (b), and (c), each with a duration corresponding to how long measured values were seen to lie between the corresponding thresholds. It should be noted that in this example, there are no skipped severities when transitioning between states and associated

severities so use of skipping and no-skipping may render the same alerts. Using the overlapping duration technique, two alerts may be rendered: an S2 alert which has a starting time when the measurement is taken during time period (b) and an ending time when the measurement is taken during time period (c), and an S1 alert which has a starting time when the measurement is taken during time period (a) and which is ongoing when the measurement is taken during time period (c).

Referring back to Figures 28-32, illustrated is an example utilizing a model in which alerts are output when transitioning between severities. The different variations of utilizing the overlapping or non-overlapping technique and the skipping/no skipping technique are illustrated. The model also specifies that an alert is not output when transitioning to the S0 severity associated with the normal state (e.g., see, for example element 2106 in which no new alert is opened and no alert message is output in connection with the observed value zz in the normal range). In connection with any one of the different variations (e.g., utilizing the overlapping or non-overlapping technique and the skipping/no skipping technique), an embodiment may alternatively specify that alerts are to be output in connection with transitioning between states and that no alert is to be output when transitioning into the normal state.

Referring now to Figure 33 shown is another example illustrating a series of measurements taken at various times in accordance with the threshold values of Figure 24 designated 4 levels and severities. Each such measurement is represented by one of the data points (e.g., x, y, z, xx) on the graph of 2400. The horizontal lines at each of V1, V2 and V3 graphically represent the threshold values from Figure 24. Each of the denoted elements a, b, c, and d along the horizontal time axis corresponds to a time period or time window.

With reference to the example of Figure 33, only the overlapping technique with skipping and then without skipping will be further illustrated in connection with Figures 34 and 35.

Referring now to Figure 34, shown is an example illustrating the overlapping technique without skipping using the measurements of Figure 33. Element 2520 represents the alerts open at each point in time. Element 2502 represents information relevant to time T1

where there is a transition from a normal severity to a high severity alert. An alert is opened for each of the low, medium and high severities having a start time of T1. At time T1, there is a transition from normal to high severity and using the overlapping non-skipping technique, alerts are opened for the target severity and all severities intervening (e.g., low and medium) between the start severity (e.g., normal) and the target severity (e.g., high).

5 Element 2504 represents information relevant to time T2 where there is a transition from a high severity to a low severity alert. In connection with this downward transition, alerts for the medium and high severities are closed since the target severity (e.g., low) is a lesser severity than either of the foregoing. Both the medium and high severity alerts have an
10 ending time T2 and their duration times may each be represented as T2-T1. The low alert is still open as represented by the displayed ONGOING duration. Element 2506 represents information relevant to time T3 where there is a transition from a low severity to a normal severity alert. The low severity alert is closed thereby specifying an ending time T3 for the low severity alert. At time T3, a transition is made to the target severity, the normal severity
15 level, which is of a lesser severity than the severity associated with the low alert. Thus, the low alert is closed at time T3 having a duration time represented by T3-T1.

Referring now to Figure 35, shown is an example illustrating the overlapping
technique with skipping using the measurements of Figure 33. Element 2620 represents the
20 alerts open at each point in time. Element 2602 represents information relevant to time T1 where there is a transition from a normal severity to a high severity alert. An alert is opened for the high severity having a start time of T1. Using the overlapping skipping technique, alerts are only opened for the target severity and not for any of the severities intervening (e.g., low and medium) between the start severity (e.g., normal) and the target severity (e.g.,
25 high). Element 2604 represents information relevant to time T2 where there is a transition from a high severity to a low severity alert. In connection with this downward transition, the high severity alert is closed since the target severity (e.g., low) is of a lesser severity. The duration time for the high severity alert is represented as T2-T1. Additionally, an alert is opened for the low severity having a starting time T2 of the duration. The low alert is open
30 as represented by the displayed ONGOING duration. Element 2606 represents information relevant to time T3 where there is a transition from a low severity to a normal severity alert. The low severity alert is closed thereby specifying an ending time of T3 for the low severity alert. At time T3, a transition is made to the target severity, the normal severity level, which

is of a lesser severity than the severity associated with the low alert. Thus, the low alert is closed at time T3 having a duration time represented by T3-T1.

Referring now to Figure 36, shown is a flowchart of processing steps that may be used in connection with the techniques described herein when examining a new value in connection with generating alerts and determining associated durations when alert generation is based on changes in state severity. The flowchart 2700 generally summarizes the processing just described. At step 2702, the start and target severities are determined. The start severity is that severity associated with a measurement taken at the immediately previous point in time. For example, in a state machine having states which change in response to measured values, if a new measurement value is obtained at time T2, the previous measurement value is that obtained at time T1. In this instance, the severity of the value at time T1 is the starting severity and the severity of the value at time T2 is the target severity. At step 2704, a determination is made as to whether the severities of the source and target are the same. If so, no updates are made to the starting and ending time values for the durations for open alerts. If step 2704 evaluates to no, control proceeds to step 2708 where a determination is made as to whether the start transition has a higher severity than the target. If so, control proceeds to step 2710 to apply the appropriate downward transition rule in accordance with whether an embodiment is utilizing the overlapping or non-overlapping technique and whether the embodiment is using the skipping or non-skipping variation thereof. If step 2708 evaluates to no, control proceeds to step 2712 to apply the upward transition rule in accordance with whether an embodiment is utilizing the overlapping or non-overlapping technique and whether the embodiment is using the skipping or non-skipping variation thereof.

The upward and downward transition rules as illustrated and described herein will now be specified. It should be noted that in connection with the generalized rules below, the starting and/or target severities may be characterized as a non-alert generating severity, such as the normal severity level illustrated in connection with the techniques herein. As such, the generalized rules are written to perform processing associated with opening and closing alerts as appropriate depending on whether the starting and target severities cause alerts to be generated.

In connection with the overlapping technique using no skipping, following is the upward transition rule processing steps for transitioning from severity A at time T1 to B at time T2:

1. open an alert for B if B is an alert-generating severity.
- 5 2. open alerts for any intervening severities between A and B which are alert generating severities.
3. record starting times for the durations for all alerts just opened in 1. and 2. as T2.

Following is the corresponding downward transition rule for transitioning from
10 severity B at time T1 to severity A at time T2:

1. close any open alerts with severity > severity of A.
2. record ending time of the duration for all alerts just closed in 1. as T2.

15 In connection with the overlapping technique using skipping, following is the upward transition rule processing steps for transitioning from severity A at time T1 to B at time T2:

1. open an alert for B if B is an alert-generating severity.
2. record starting times for the durations for all alerts just opened in 1. as T2.

20 Following is the corresponding downward transition rule for transitioning from severity B at T1 to A at time T2:

1. close any open alerts with severity > severity of A.
2. record ending time for the duration as T2 for all alerts just closed in 1.
- 25 3. open an alert for A if A is an alert-generating severity and an alert is not already opened for A. Record T2 as the start time for the alert A if such an alert A is opened.

30 In connection with the non-overlapping technique using no skipping, following is the upward transition rule processing steps for transitioning from severity A at time T1 to B at time T2:

1. open an alert for B if B is an alert-generating severity and record the starting time for the duration as T2.
2. open and close alerts for any intervening alert-generating severities between A and B and record the duration time as zero.

3. close A if there is an alert open associated with severity A (e.g., A is an alert generating severity) and record the ending time for the duration as T2.

Following is the corresponding downward transition rule for transitioning from severity B at T1 to A at time T2:

1. close the alert for B if there is an alert open for B (e.g., B is an alert-generating severity) and record and ending time for the duration as T2.

2. open and close alerts for any intervening alert-generating severities between A and B and record the duration time as $T2 - T1$.

3. open an alert for A if A is an alert-generating severity and record a starting time for the duration as T2.

In connection with the non-overlapping technique using skipping, following is the upward transition rule processing steps for transitioning from severity A at time T1 to B at time T2:

1. close A if there is such an open alert for A (e.g., A is an alert generating severity) and record ending time for the duration of T2.

2. open an alert for B (e.g., B is an alert generating severity) and record starting time for the duration as T2.

Following is the corresponding downward transition rule for transitioning from severity B at T1 to A at time T2:

1. close B if there is an alert open for B (e.g., B is an alert generating severity) and record ending time of T2 for the duration.

2. open A if there is an alert open for A (e.g., A is an alert generating severity) and record starting time for the duration as T2.

It should be noted that in connection with displaying alerts, the ordering on the display may be performed in accordance with one or more sorting criteria. For example, in one embodiment, a priority level may be associated with each severity level and thus each alert message. One or more severity levels may be associated with a priority level. A priority leveling may be applied to further group or categorize the different severities associated with performance and security alerts. The priority level may be the primary key used to sort alerts on the alert display. Additional secondary sorting criteria may be specified

such as by time which includes the starting time of those alerts having a non-zero duration and the time of each instantaneous alert. It should be noted that the instantaneous alerts may be further characterized as having a same starting time and ending time for the duration time=0. Other embodiments may use other criteria than as described herein.

5

An embodiment of the alert display may include an acknowledgement button or selection option in connection with those alerts that require operator acknowledgement.

Referring now to Figure 37, shown is an example illustrating the use of
10 acknowledgements in connection with the techniques described herein. The example 2800 includes a first display 2802 that may be the contents of an alert display at an operator's screen at a first point in time. The three displayed alerts may require operator acknowledgement such as by selecting the ACK button on a displayed GUI next to the corresponding alert. Acknowledgement of alerts may be indicated on those alerts requiring
15 operator attention. In connection with recognizing this fact that a human has viewed this alert and is aware of such alerts, a manual acknowledgement may be required. In one embodiment, this may be performed by manually selecting the ACK button for a corresponding alert to acknowledge the alert. In connection with the techniques described herein, if an operator acknowledges an alert and the alert is ongoing or not closed, the alert
20 may continue to be displayed on the computer display or other display area of alerts until the associated condition clears and the associated alert has closed. For example, an operation may acknowledge all the alerts in 2802 causing an update of the display to be as illustrated in 2810. In 2810, those alerts which are closed, as indicated by a displayed duration time, may be cleared from the display. However, the low alert is still ongoing so the acknowledgement
25 button ACK may be set to some other visually differentiable state (e.g., change color, flash, appear bolded, and the like) than as displayed in 2802 to indicate that the associated alert has been acknowledged. However, the associated alert may continue to be displayed until the condition associated with the alert has cleared or subsided. In such an embodiment, for example, an unacknowledged alert may flash until acknowledged, and also remain visible on
30 the alert display screen until both the alert has been acknowledged by the operator, and the associated value has returned to a non-alert generating or a normal state. In other words, the alert message remains until both the alert has been acknowledged and the associated alert condition has subsided, been cleared from the system or otherwise no longer persists.

Referring now to Figure 38, shown is a flowchart of processing steps that may be used in an embodiment as an alternative to the steps illustrated in Figure 36 for generating alerts and determining alert durations. The steps of Figure 38 may be used, for example, in state machines where paths through the state machine from a source state to a target states exist and the paths are associated with severities that are neither monotonically increasing nor monotonically decreasing.

At step 2902, the source and target states are determined, and the path through the state machine leading from source state to target state is determined. At step 2904 a determination is made as to whether the severities of the source, the target, and all the states in the path from the source to the target states are the same. If so, control transfers to 2906 where no updates are made to the starting and ending time values of the durations for open alerts. If step 2904 evaluates to no, control proceeds to step 2908 wherein a segment of the state path is identified, said segment starting at the current start state in the state path and ending in the furthest state along the path wherein the severities associated with states in the path segment are all monotonically increasing or decreasing. The furthest such state found is said to be the intermediate target state. The corresponding aforementioned upward transition rules described for Figure 36 are then applied if the path segment has monotonically increasing severities, and the corresponding downward transition rules are applied if the path segment has monotonically decreasing severities. Control then passes to step 2910 where a determination is made as to whether the current intermediate target is in fact the ultimate target. If the current target is the end of the path through the state machine, the update of open alert durations is complete and control is transferred to 2912. If the current intermediate target is not the ultimate target, then the current intermediate target becomes the current source and control is transferred back to step 2908 to determine and process the next path segment.

While the invention has been disclosed in connection with preferred embodiments shown and described in detail, their modifications and improvements thereon will become readily apparent to those skilled in the art. Accordingly, the spirit and scope of the present invention should be limited only by the following claims.

What is Claimed is:

1. A method for monitoring data elements comprising:
receiving one or more sets of data elements, each set representing a grouping of one
5 or more data elements;
receiving an associated resource allocation of a first resource for each of said one or
more sets; and
performing processing for each of said one or more sets of data elements, wherein
said processing includes scanning and monitoring data elements for differences, and wherein
10 at least one of said scanning and said monitoring is performed in accordance with the
associated resource allocation for each set.

2. The method of Claim 1, wherein said resource allocation indicates an amount of
said first resource to be consumed during a defined time period.

3. The method of Claim 2, wherein a sum of all the associated resource allocations
for all the sets of data elements indicates a total amount of said first resource allocated for
consumption during a defined time period.

4. The method of Claim 3, wherein said first resource is a processor and said resource
allocation indicates an amount of processor execution time to be consumed during a defined
time period.

5. The method of Claim 3, wherein an associated resource allocation for one of said
25 sets is specified as a percentage of said total amount.

6. The method of Claim 3, wherein an associated resource allocation for one of said
sets is specified as a numeric value.

7. The method of Claim 1, wherein each of said sets of data elements has an
associated level of priority indicating a level of priority of data elements included therein.

8. The method of Claim 1, further comprising, for each set of data elements:
providing a set of one or more rules; and

determining data elements belonging to said each set using said one or more rules as a filter.

5 9. The method of Claim 8, wherein said set of rules includes at least one inclusionary rule defining one or more criteria, wherein a data element having said one or more criteria is included in said each set of data elements.

10 10. The method of Claim 8, wherein said set of rules includes at least one exclusionary rule defining one or more criteria, wherein a data element having said one or more criteria is not included in said each set of data elements.

11. The method of Claim 1, wherein one or more data stores are specified for each set from which data elements are selected for inclusion into said each set.

15 12. The method of Claim 11, wherein said one or more data stores includes at least one data store selected from a database and a file system.

20 13. The method of Claim 12, wherein at least one of said data stores specified has a hierarchical structure and rules are specified using said hierarchical structure to determine which data elements belong to said each set of data elements.

14. The method of Claim 1, further comprising:
determining one or more attributes of each data element, each attribute characterizing data, metadata, or schema information for said each data element, wherein said determining
25 one or more attributes is performed as part of said scanning;
determining differences between a current value for each of said one or more attributes and a predetermined value; and
generating an alert in accordance with any of said differences.

30 15. The method of Claim 14, wherein said determining differences is performed by comparing hash values for a current value and a predetermined value for at least one attribute including sensitive data.

16. The method of Claim 1, wherein a second resource is specified for at least one of said sets, said second resource having an associated resource allocation for each of said data sets, and wherein at least one of said scanning and said monitoring are performed in accordance with the associated resource allocations for said first and said second resources for each set, said at least one of said scanning and said monitoring consuming allocated resources for a data set if any one the associated resource allocations of said first or said second resources has been consumed.

17. The method of Claim 1, wherein said resource allocation indicates a number of disk inputs performed during at least one of said scanning and said monitoring of a data set for a defined time period, each of said disk inputs representing a portion of data read from a device during said at least one of said scanning and said monitoring.

18. The method of Claim 3, further comprising:
determining an elapsed time for said at least one of said scanning and said monitoring to consume said total amount of said first resource allocated during a defined time period;
if the elapsed time is less than said defined time period, scheduling said processing to resume at a beginning of a next defined time period;
if the elapsed time is not less than said defined time period, scheduling said processing to resume execution immediately.

19. The method of Claim 1, wherein, a first process is performing scanning and monitoring data elements for differences, and after said first process processes one or more data elements, another process waiting for execution is executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor.

20. The method of Claim 1, wherein, a first process is performing scanning and monitoring data elements for differences, and after said first process processes a predetermined amount of data with respect to at least one of said scanning and said monitoring, another process waiting for execution is executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor.

21. The method of Claim 1, wherein, a first process is performing scanning and monitoring data elements for differences, and after said first process consumes a fixed amount of a resource with respect to at least one of said scanning and said monitoring,
5 another process waiting for execution is executed prior to resuming execution of said first process by said first process relinquishing control of a processor independent of an operating system technique to share said processor.

22. The method of Claim 14, wherein said predetermined value is an expected value
10 obtained in accordance with one or more previous scans of a data element at a prior point in time.

23. The method of Claim 14, wherein said predetermined value is determined using at least one of: simulation and theoretically expected results.

24. A method for scanning data elements comprising:
receiving one or more sets of data elements, each set representing a grouping of one or more data elements;
receiving an associated resource allocation of a first resource for each of said one or
20 more sets; and
performing scanning for each of said one or more sets of data elements in accordance with the associated resource allocation for each set.

25. A computer readable medium which comprises code stored thereon for
25 monitoring data elements that:
receives one or more sets of data elements, each set representing a grouping of one or more data elements;
receives an associated resource allocation of a first resource for each of said one or more sets; and
30 performs processing for each of said two or more sets of data elements, wherein said processing includes scanning and monitoring data elements for differences, and wherein at least one of said scanning and said monitoring is performed in accordance with the associated resource allocation for each set.

26. A method for displaying alerts on an alert display area comprising:
receiving an alert message in connection with an alert condition in a system; and
displaying said alert message for an alert condition, wherein, said alert message
includes a duration time representing an amount of time said alert condition has existed in the
5 system.

27. The method of Claim 26, wherein if said alert condition has not subsided at a
time when said alert message is displayed, said duration time includes as said duration time
10 an indicator that said alert condition is ongoing.

28. The method of Claim 26, wherein said alert message that is displayed represents
an alert condition having a non-zero duration time.

15 29. The method of Claim 26, wherein said duration time is determined by using a
model describing severity transitions of transitioning from a first severity level to a second
severity level.

20 30. The method of Claim 29, wherein each severity level included in said model is
associated with only a single state and said model includes transitions from a first state
having said first severity level to a second state having said second severity level.

25 31. The method of Claim 29, wherein at least one severity level included in said
model is associated with more than one state and said model includes transitions from a first
state having said first severity level to a second state having said second severity level.

32. The method of Claim 26, further comprising:
displaying an acknowledgement indicator indicating that said alert message requires
manual acknowledgement, wherein said alert message is displayed until said alert message
30 has been acknowledged using said acknowledgement indicator and said alert condition has
cleared.

33. The method of Claim 32, wherein, in response to performing an
acknowledgement by selecting said acknowledgement indicator, said acknowledgement

indicator is displayed in a visually differentiable state than when said alert message has not been acknowledged.

34. The method of Claim 26, wherein said alert condition is associated with one of a security alert and a performance alert.

35. The method of Claim 26, wherein said duration time has a starting time and an ending time, said starting time being determined when said alert condition is first detected and said ending time indicating when said alert condition has subsided.

10

36. The method of Claim 27, wherein, at a point in time, a display including alert messages includes at most a single alert message associated with said alert condition which includes a duration time indicating that said alert condition is ongoing.

15

37. The method of Claim 27, wherein, at a point time, a display including alert messages includes a plurality of alert messages associated with said alert condition, each of said plurality of alert messages including a duration time indicating that said alert condition is ongoing.

20

38. The method of Claim 37, wherein each of said plurality of alert messages is associated with a unique severity level.

39. The method of Claim 35, wherein a state of said alert condition is determined using a model, and wherein said starting time and said ending time of said duration time are determined in accordance with one or more states of said model.

25

40. The method of Claim 26, wherein said alert message is a first alert message for a non-instantaneous alert, and the method further comprising:

displaying on a display screen said first alert message for a non-instantaneous alert and a second alert message for an instantaneous alert.

30

41. The method of Claim 29, wherein said model includes a set of states and a set of severities, each of said severities being associated with one or more states, said model utilizing one of an overlapping technique or a non-overlapping technique in connection with

determining said duration time, wherein said overlapping technique has at most one ongoing alert, and, when transitioning from a first severity to a second severity, said first severity is determined to end and said second severity is determined to start, wherein said non-overlapping technique has one or more ongoing alerts so that a time duration for one alert may overlap with a duration time of a second alert, and, when transitioning from a lower severity to a higher severity, said higher severity is determined to start, and when transitioning from a higher severity to a lower severity, said higher severity is determined to end.

42. The method of Claim 41, wherein said model indicates whether to output alert messages for intervening severities when transitioning from a first severity to a second severity.

43. The method of Claim 41, wherein said model includes a rule indicating at least one severity for which alert messages are not to be generated when transitioning into the at least one severity.

44. A computer readable medium comprising code stored thereon for displaying alerts on an alert display area that:

receives an alert message in connection with an alert condition in a system; and displays said alert message for an alert condition, wherein said alert message includes a duration time representing an amount of time said alert condition has existed in the system.

45. The computer readable medium of Claim 44, wherein said alert condition represents a non-instantaneous alert having a duration time determined using a model for modeling said alert condition, said model including a set of one or more severities, a set of one or more states, a set of one or more transitions defining transitioning between states, a first parameter indicating whether said model utilizes an overlapping technique or a non-overlapping technique in connection with determining said duration time, wherein when transitioning from a first severity to a second severity, a duration time for said first severity is determined to end and a duration time for said second severity is determined to start with said overlapping technique, and wherein, with said non-overlapping technique, when transitioning from a lower severity to a higher severity, a duration time for said higher severity is

determined to start, and when transitioning from a higher severity to a lower severity, a duration time for said higher severity is determined to end.

46. The computer readable medium of Claim 45, wherein said model indicates
5 whether to output alert messages for intervening severities when transitioning from a first severity to a second severity.

47. The computer readable medium of Claim 46, wherein said model includes a rule
indicating at least one severity for which alert messages are not to be generated when
10 transitioning into the at least one severity.

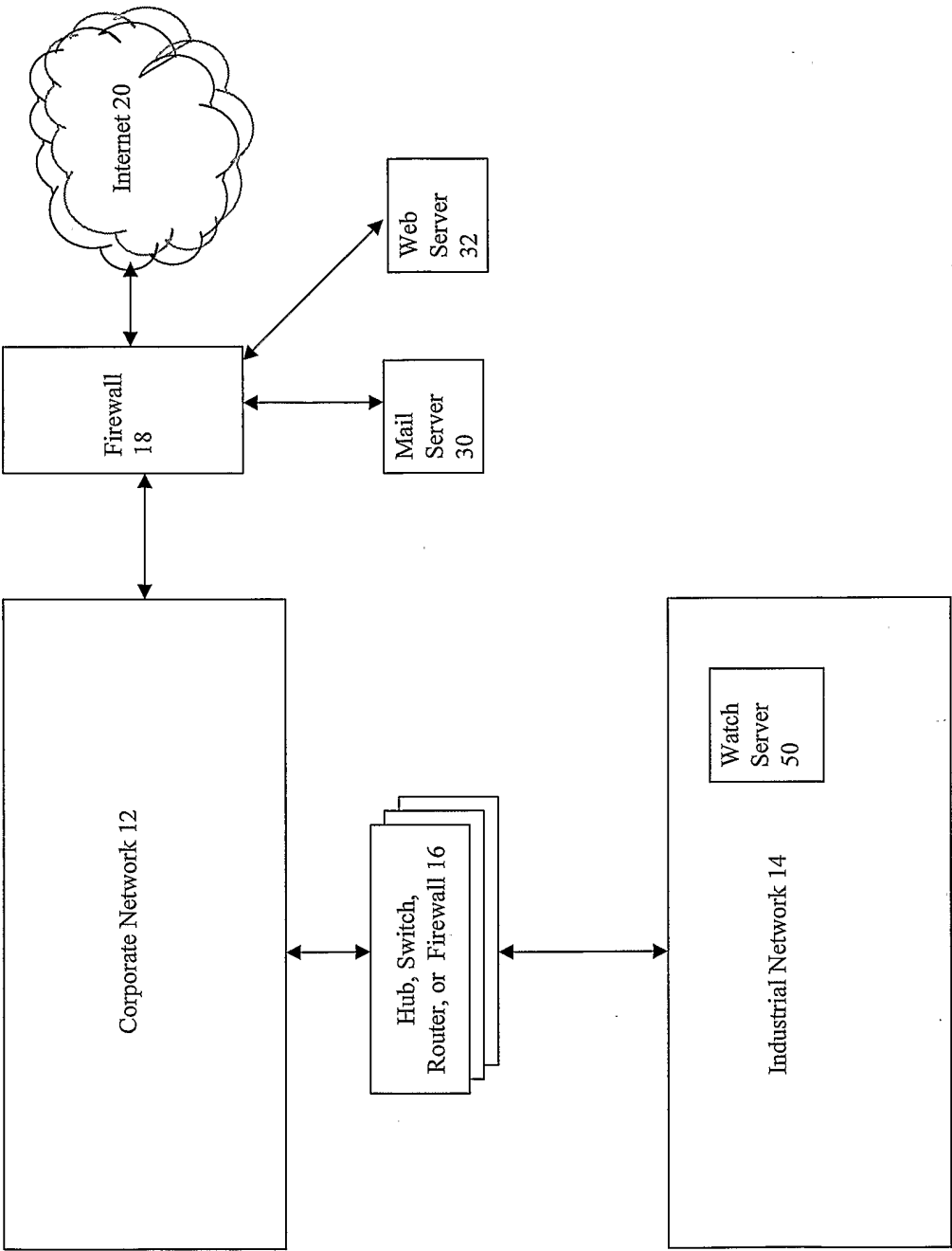


FIGURE 1

12

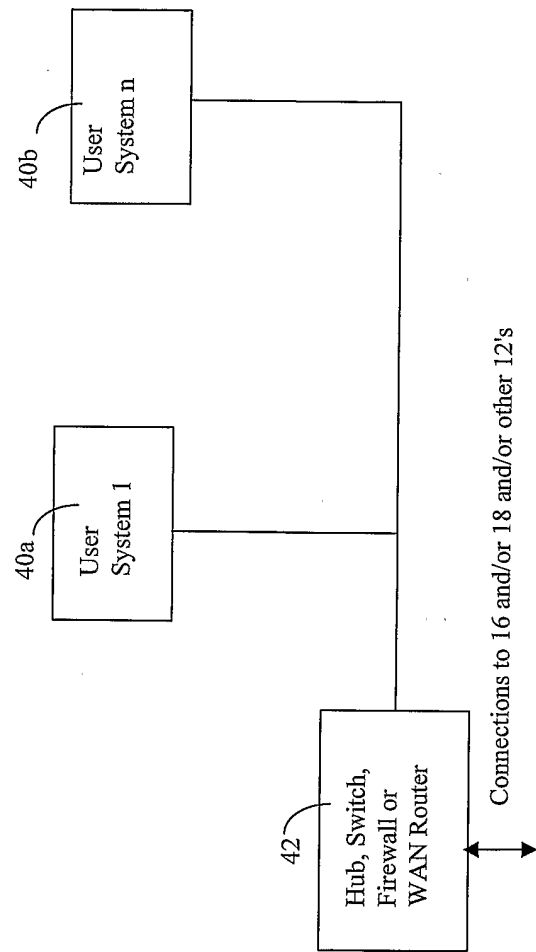


FIGURE 2

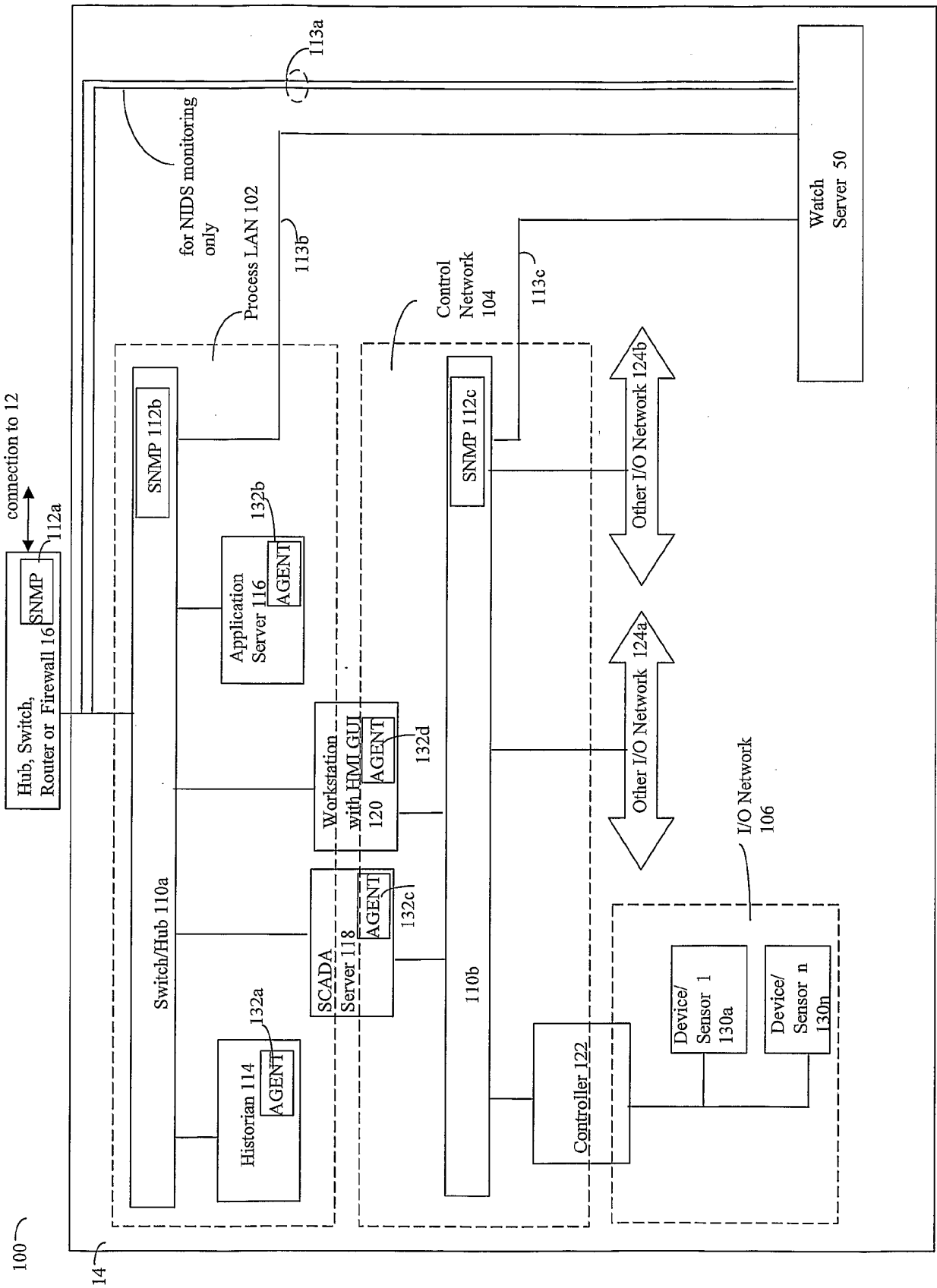


FIGURE 3

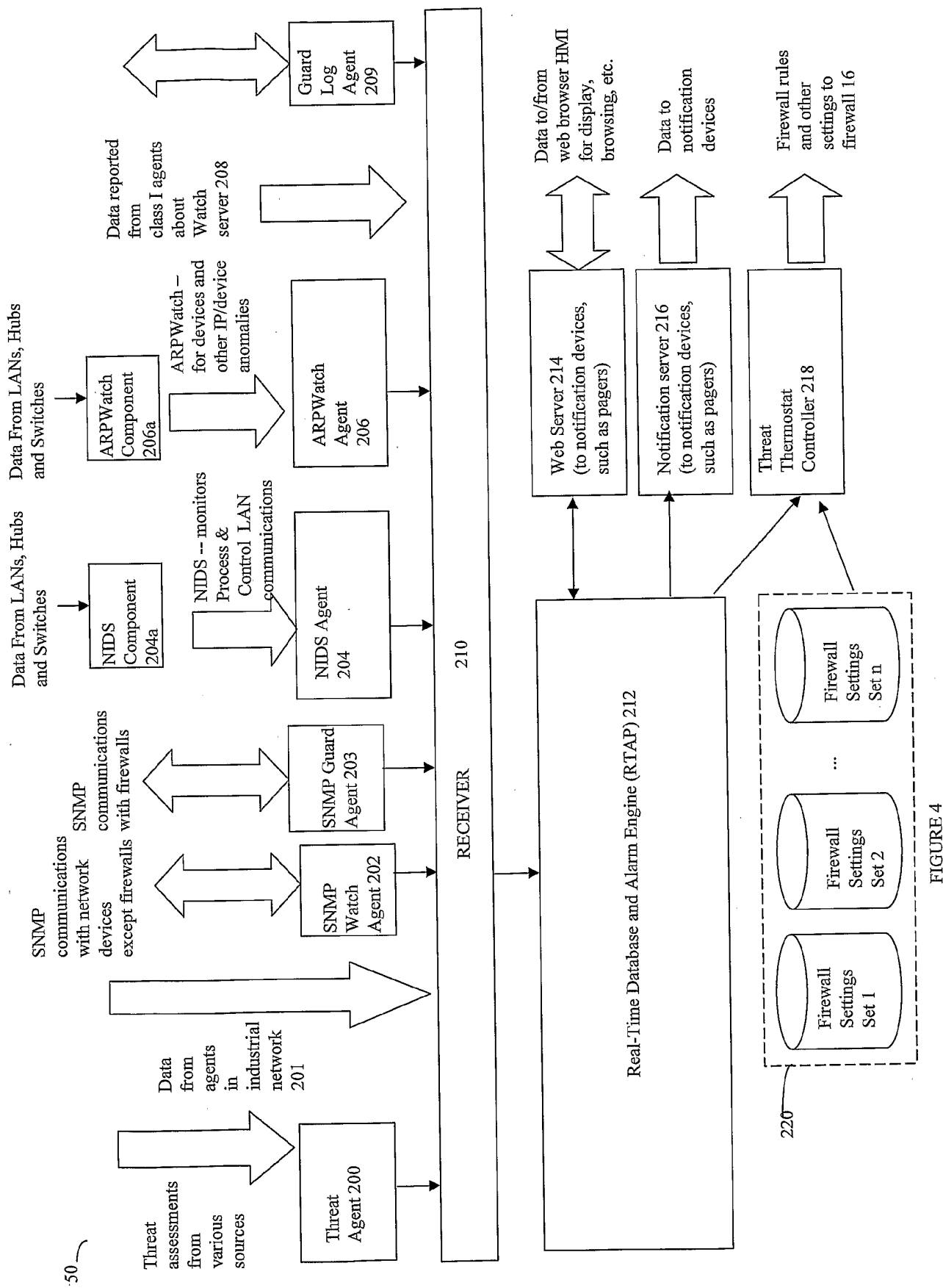


FIGURE 4

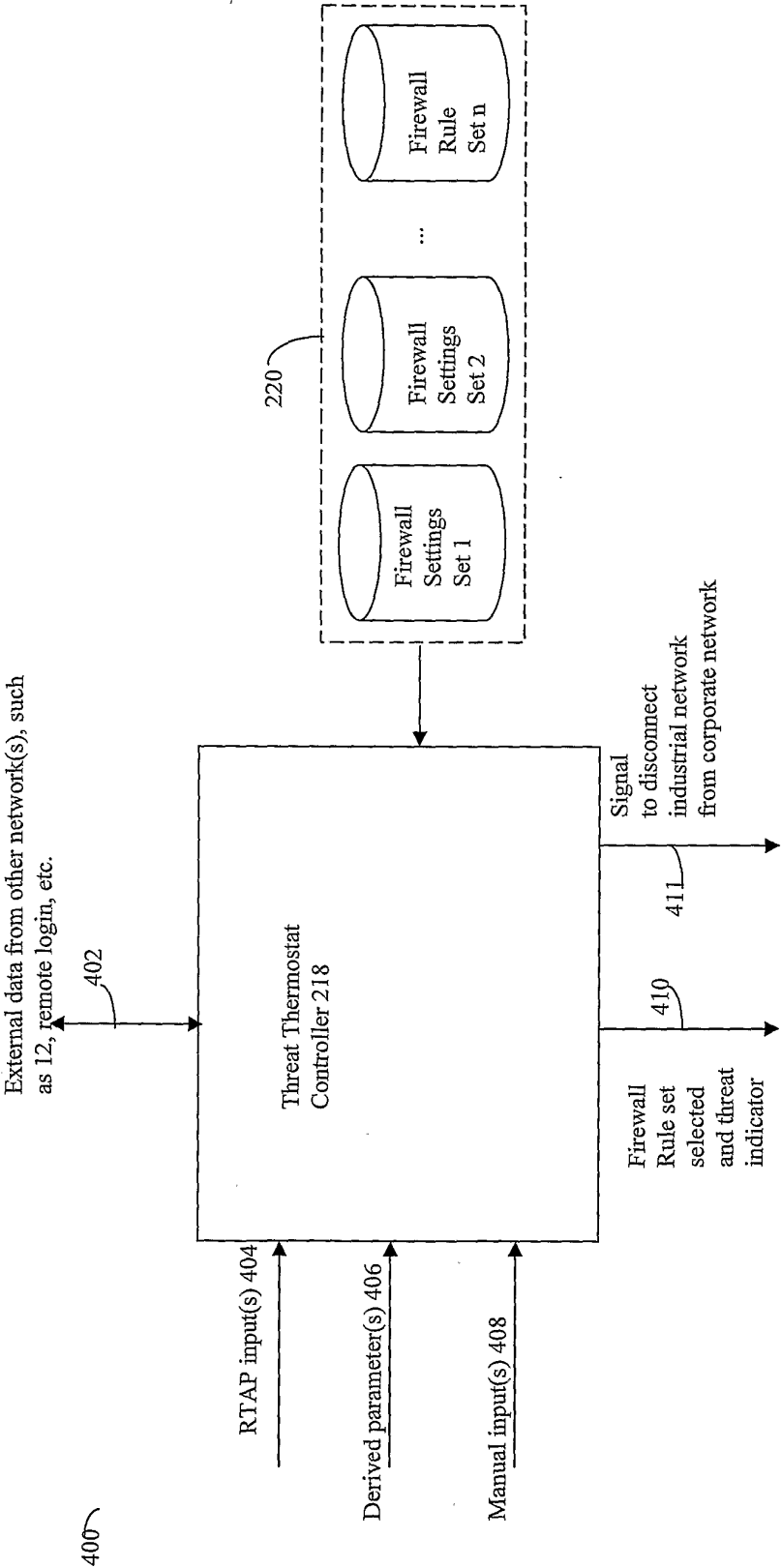


FIGURE 4A

300 ~

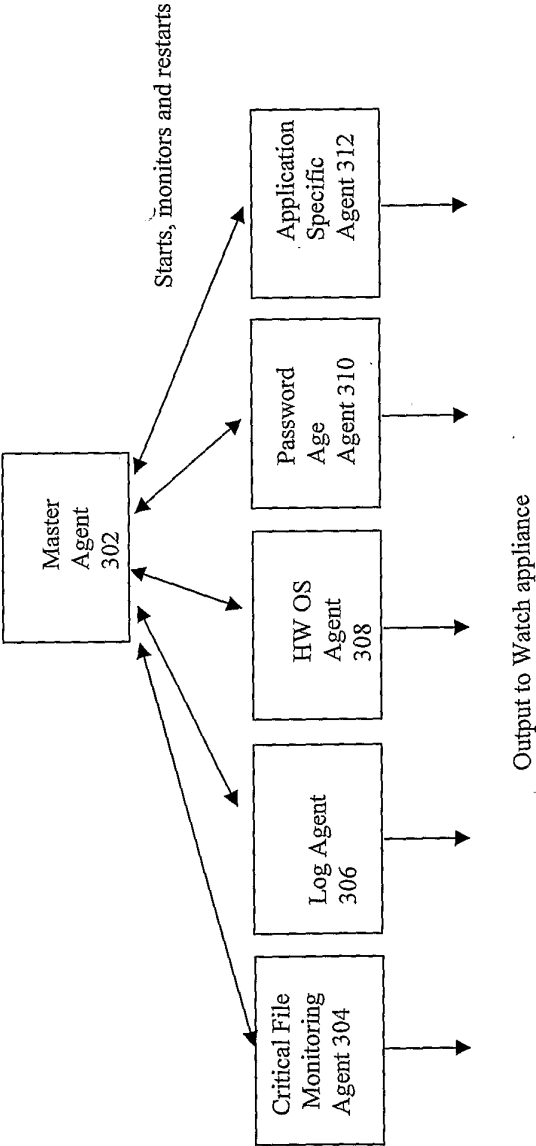


FIGURE 5

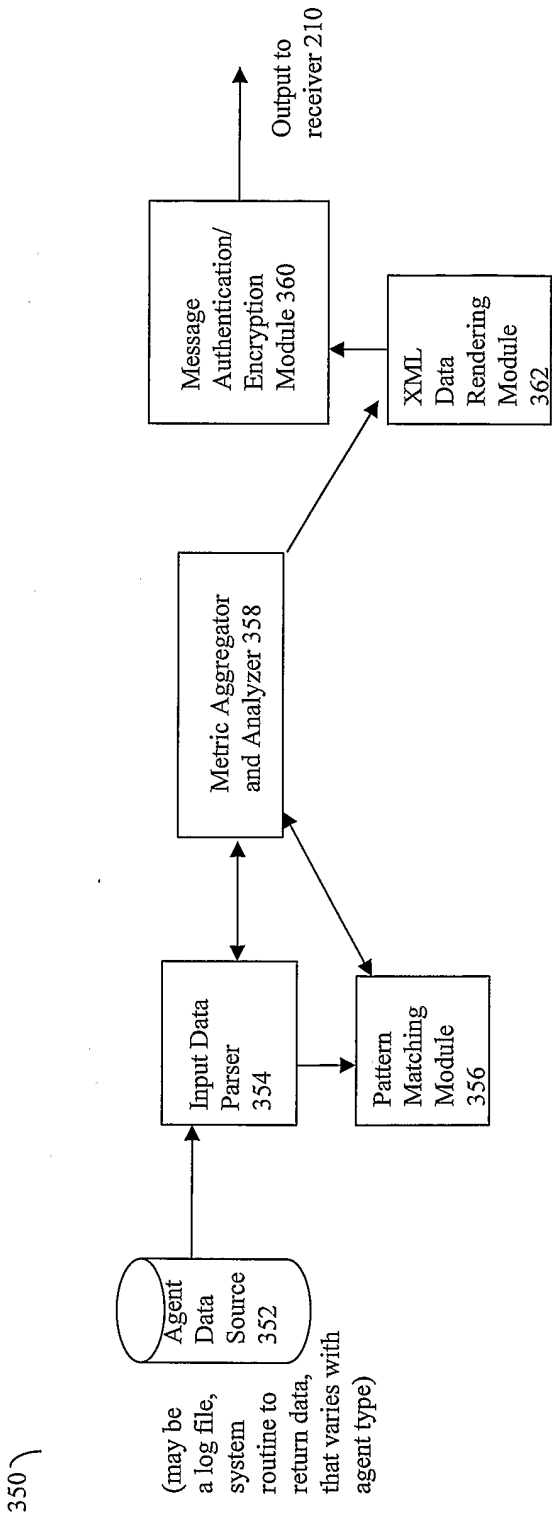


FIGURE 6

450

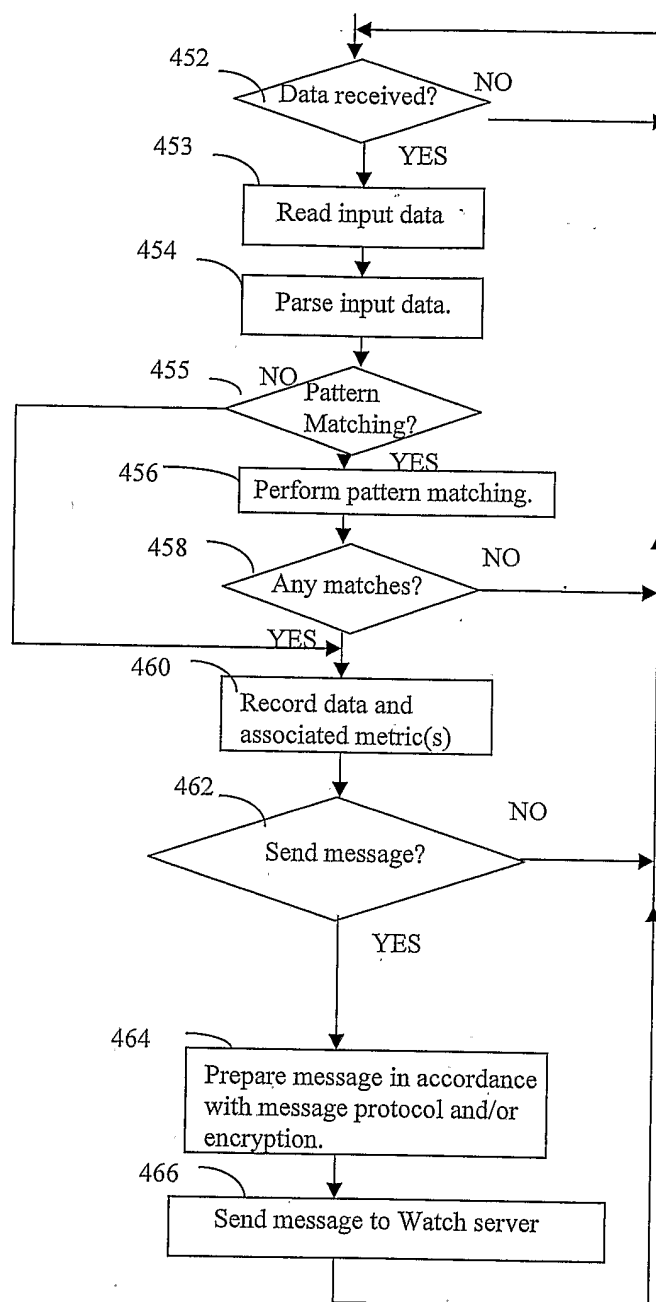


FIGURE 7

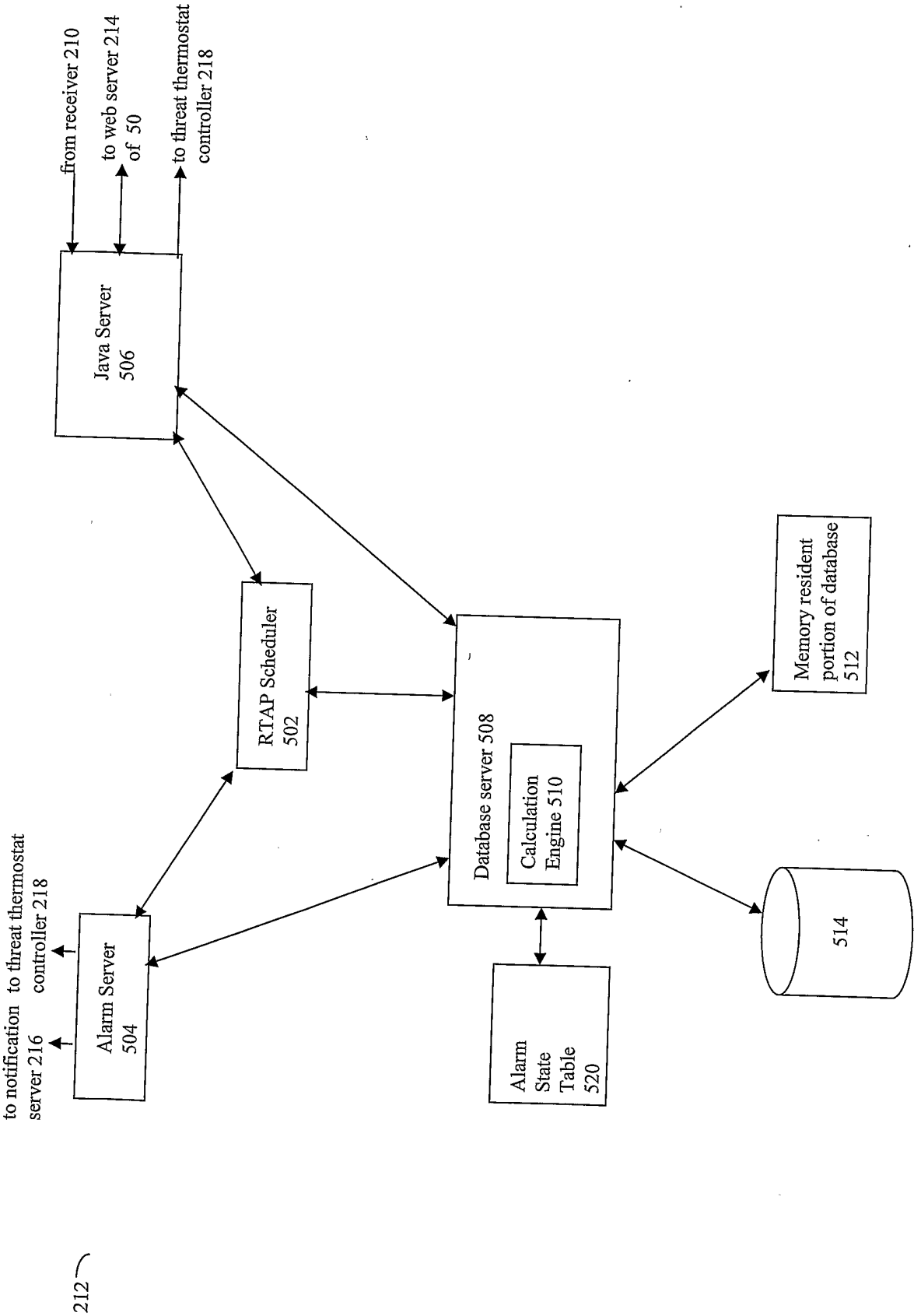


FIGURE 8

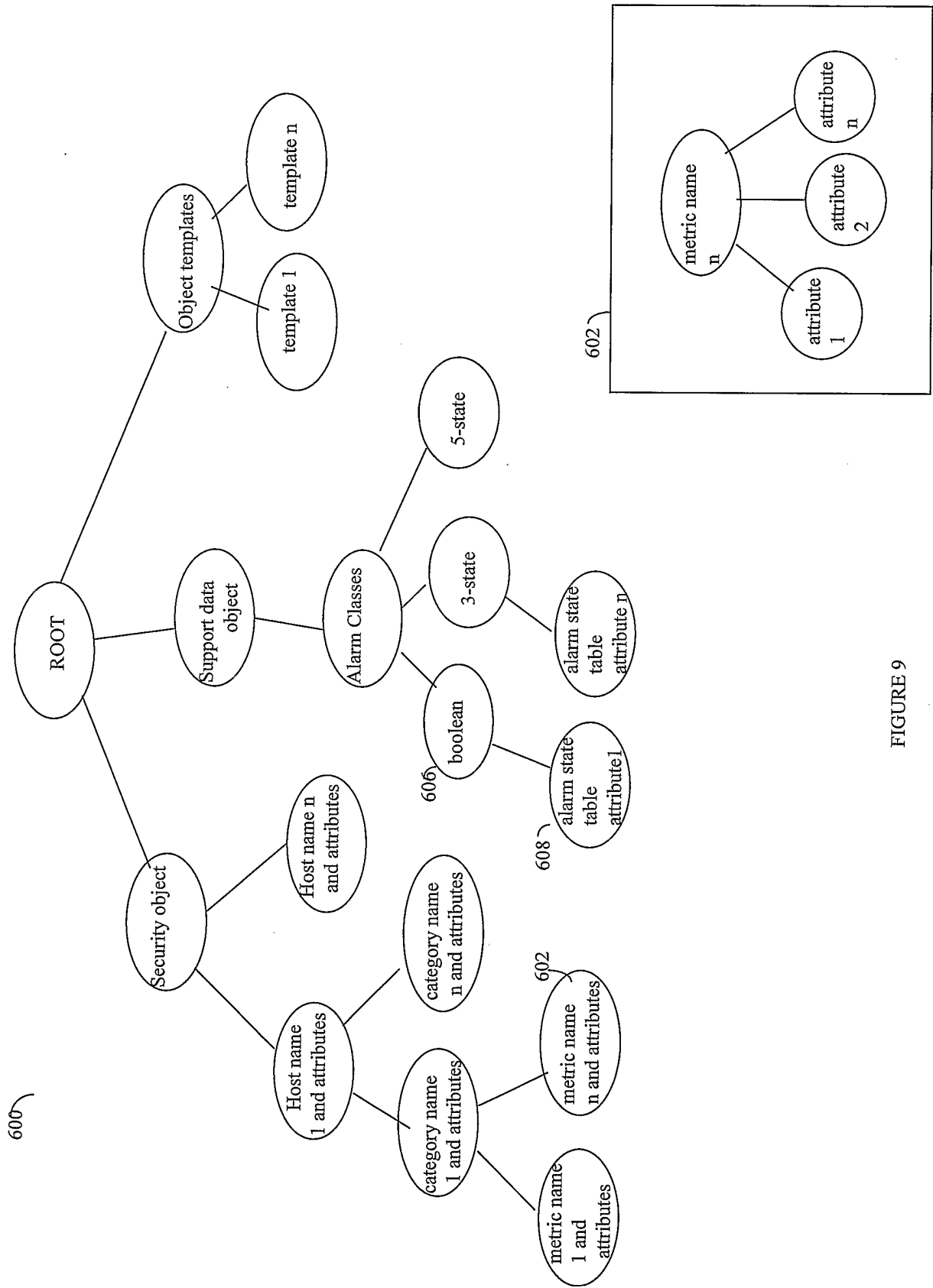


FIGURE 9

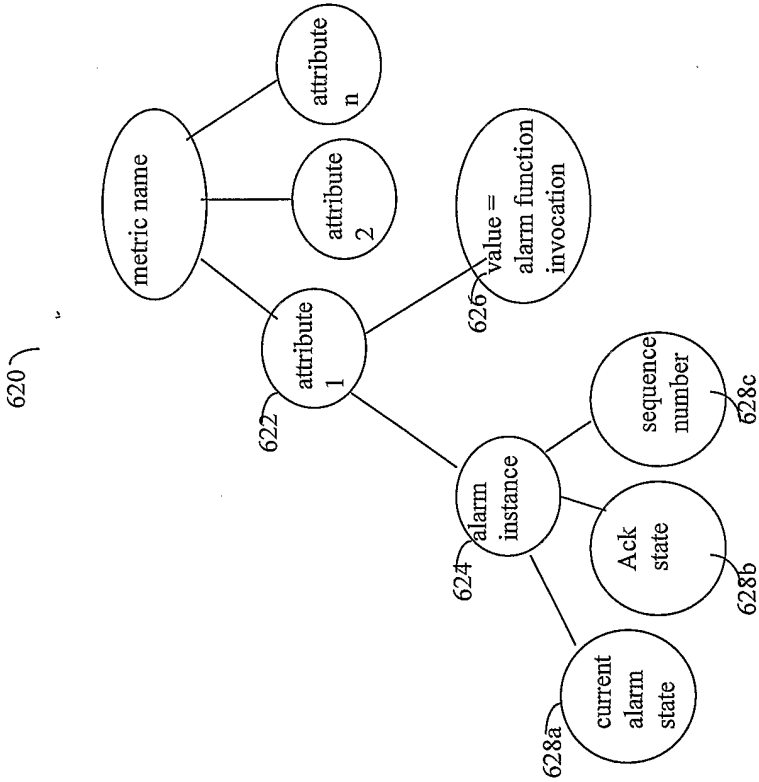


FIGURE 9A

520

Current level/state	ACK	input	next level/state	action
LEVEL 1 (NORMAL)	N/A	0-100	LEVEL 1	NONE
LEVEL 1	N/A	101-200	LEVEL 2	ALARM CONDITION
LEVEL 2	NO	0-100	LEVEL 2	ALARM CONDITION
LEVEL 2	YES	0-100	LEVEL 1	CLEAR ALARM
LEVEL 2	N/A	201-300	LEVEL 3	ALARM CONDITION
:	:	:	:	:

652

654

656

658

660

FIGURE 10

700

704

706

708

710

714

712

716

702

State	Name	Color	Ack Required	&ACK	&Range level	&Range >	&Range ==	&Range <
0	Error	Blue	0	0	0	2	0	0
1	Normal Acked	Green	0	1	0	4	1	0
2	Normal Unack Warning	Green Flashing	1	1	0	4	2	0
3	Acked	Yellow	0	3	1	6	3	2
4	Warning Unack	Yellow Flashing	1	3	1	6	4	2
5	Alert Acked Alert	Red Red	0	5	2	5	5	4
6	Unacked	Flashing	1	5	2	6	6	4

Alarm limits vector: 2 200 100 0

720

FIGURE 11

800

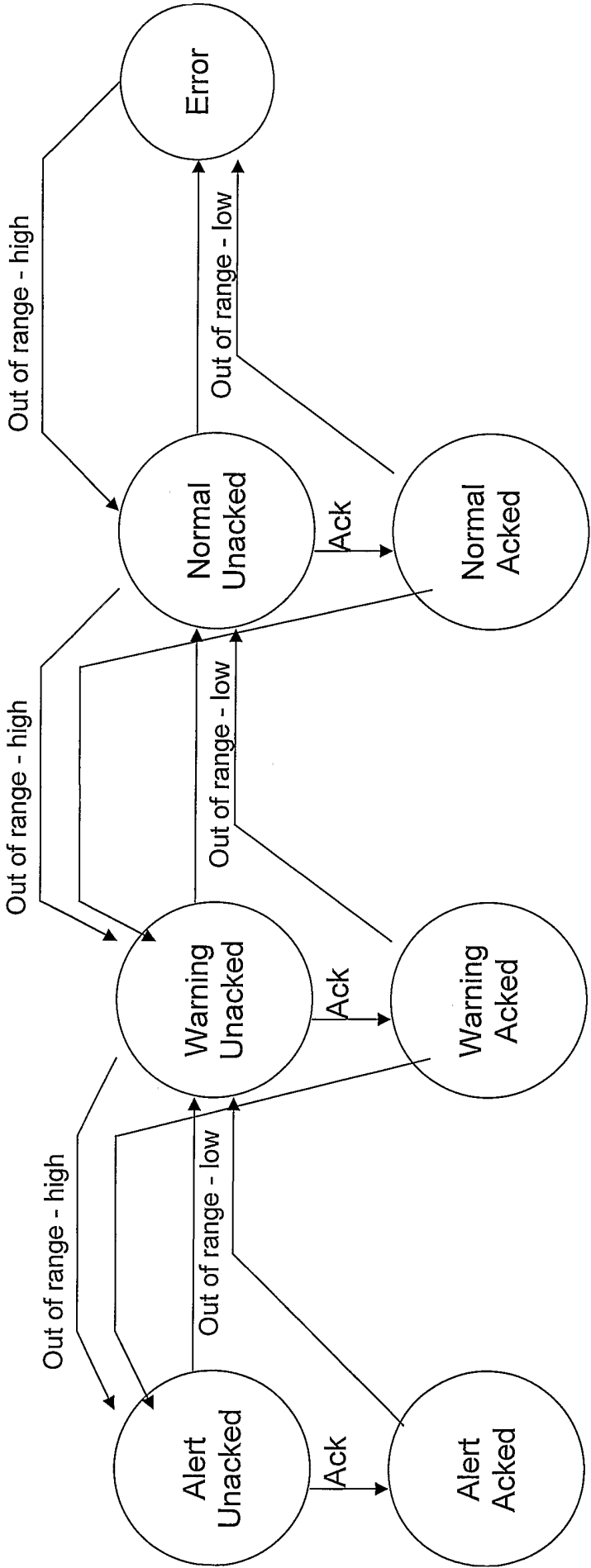


FIGURE 12

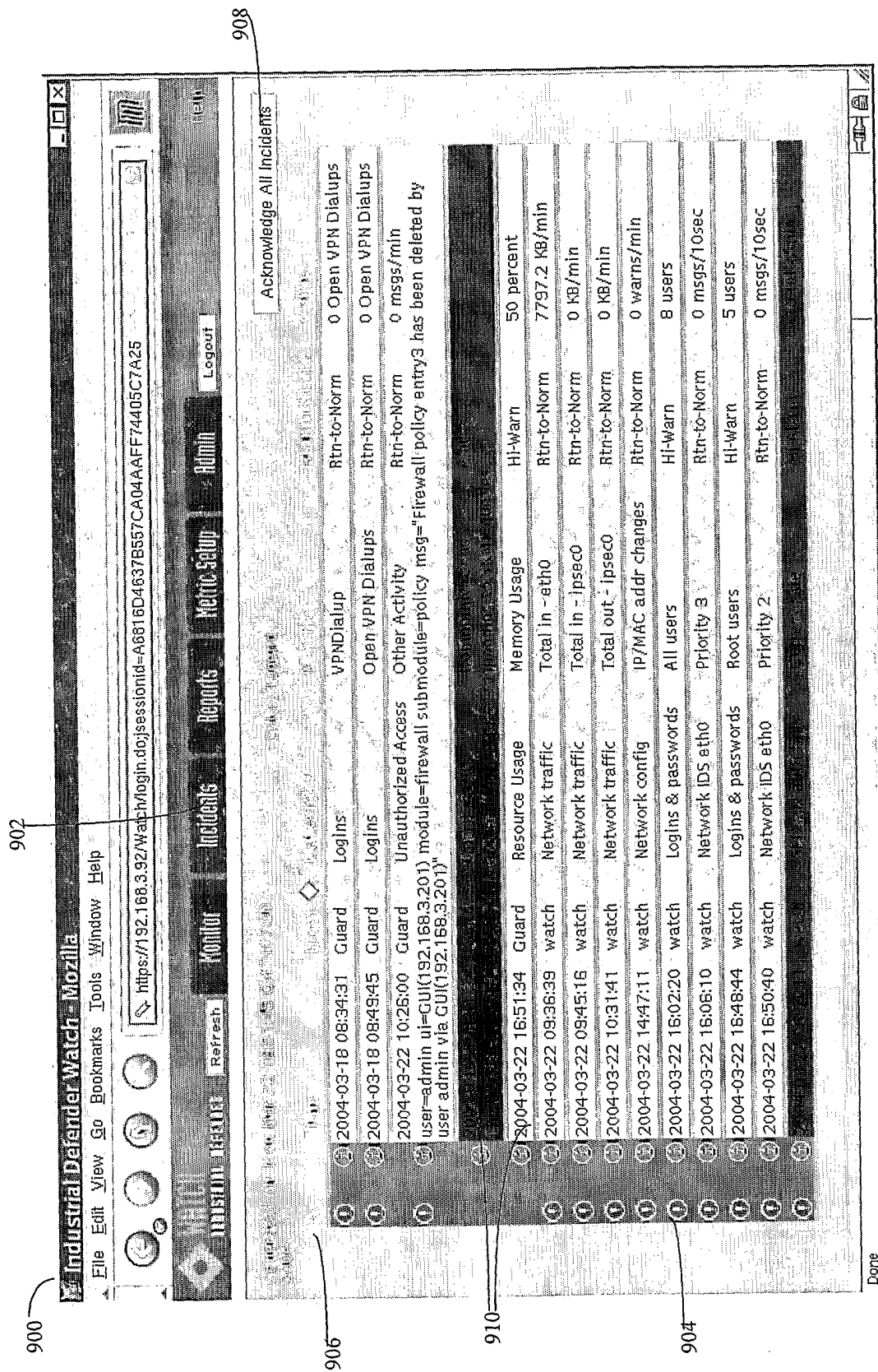


FIGURE 13

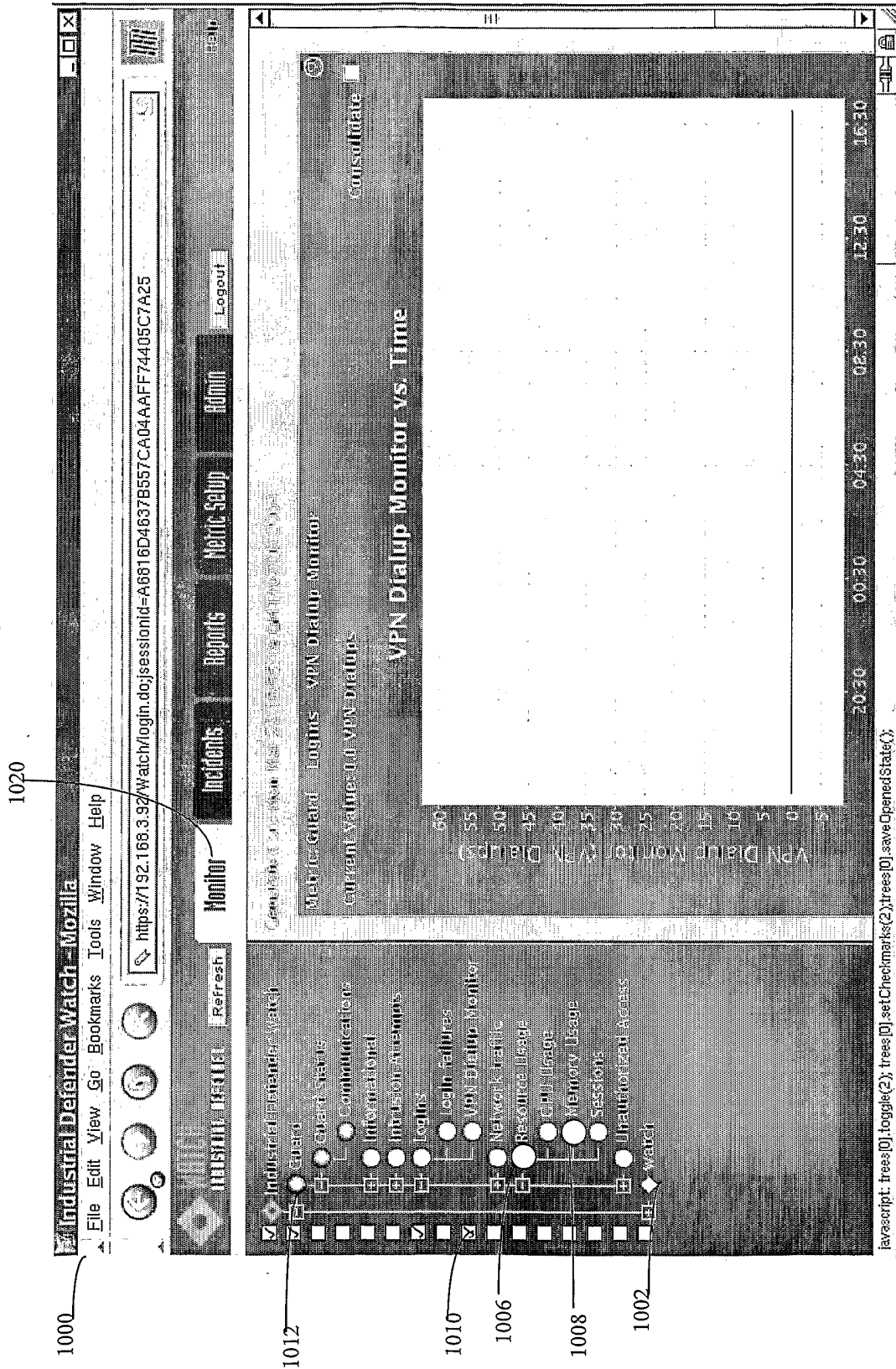


FIGURE 14

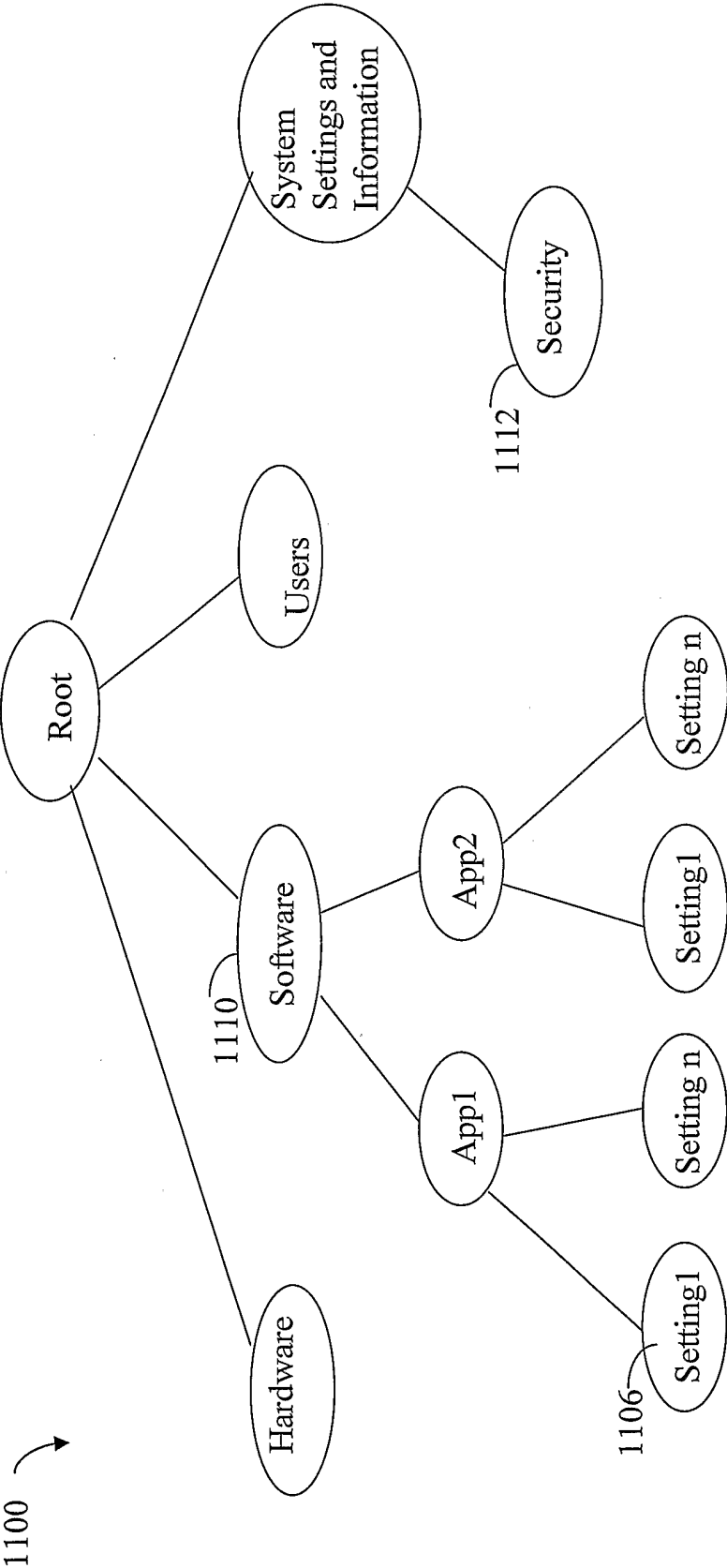


FIG. 15

1200

1202		1204	1206	1208	1210	1220	1222
Bucket #		Data store(s)	Inclusion Rules	Exclusion Rules	Information to be scanned	Resource 1 allocation	Resource N allocation
1	1230	File system Registry	CONFIG	*.TMP, *.LOG	access permissions	CPU, 10%	Disk I/Os, 10%
					hash of data file size :	CPU, 20%	:
2	1232	File system	*.SYS and pathname1		:		
:		:	:	:	:	:	:
:		:	:	:	:	:	:
:		:	:	:	:	:	:

FIG. 16

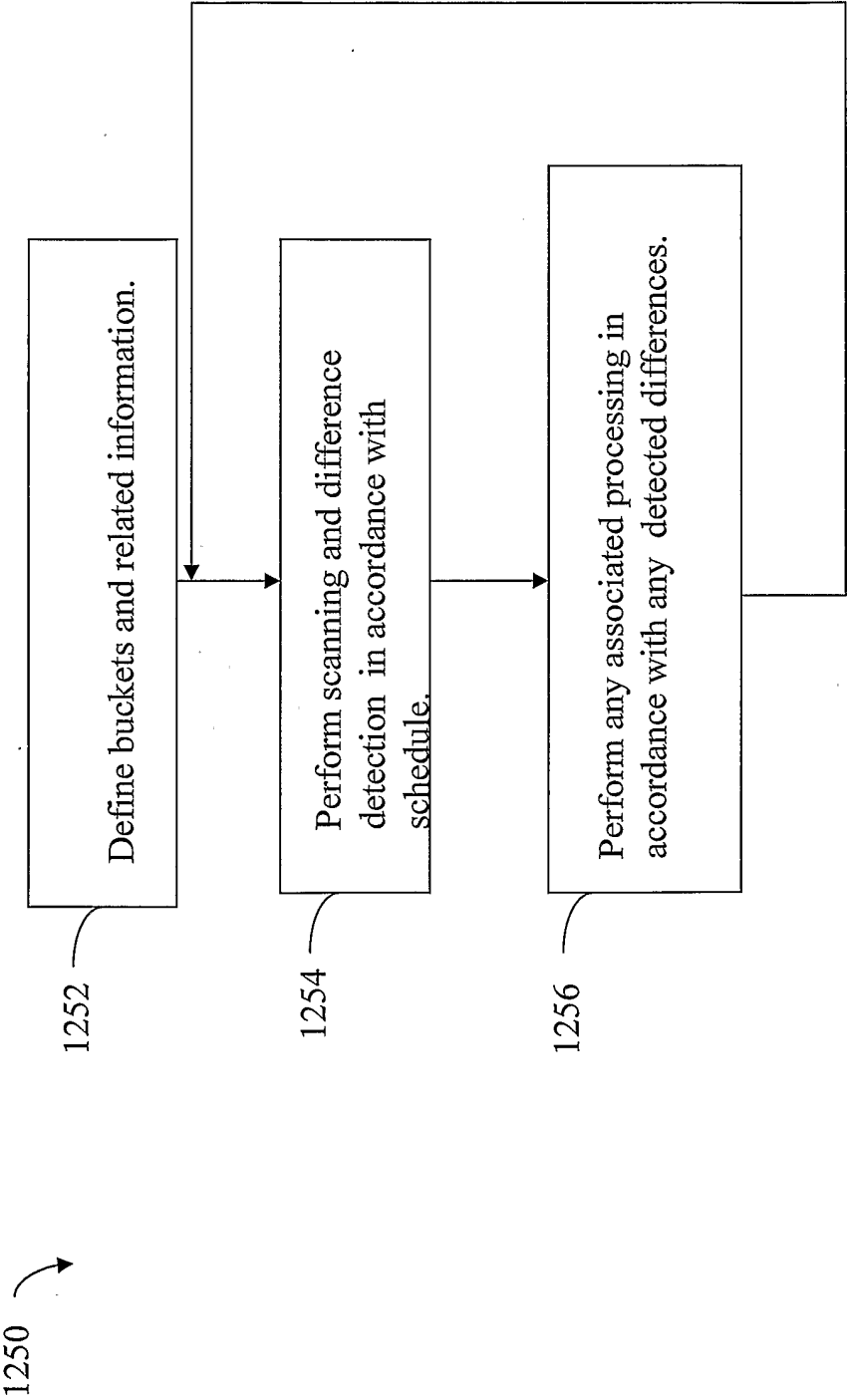


FIG. 17

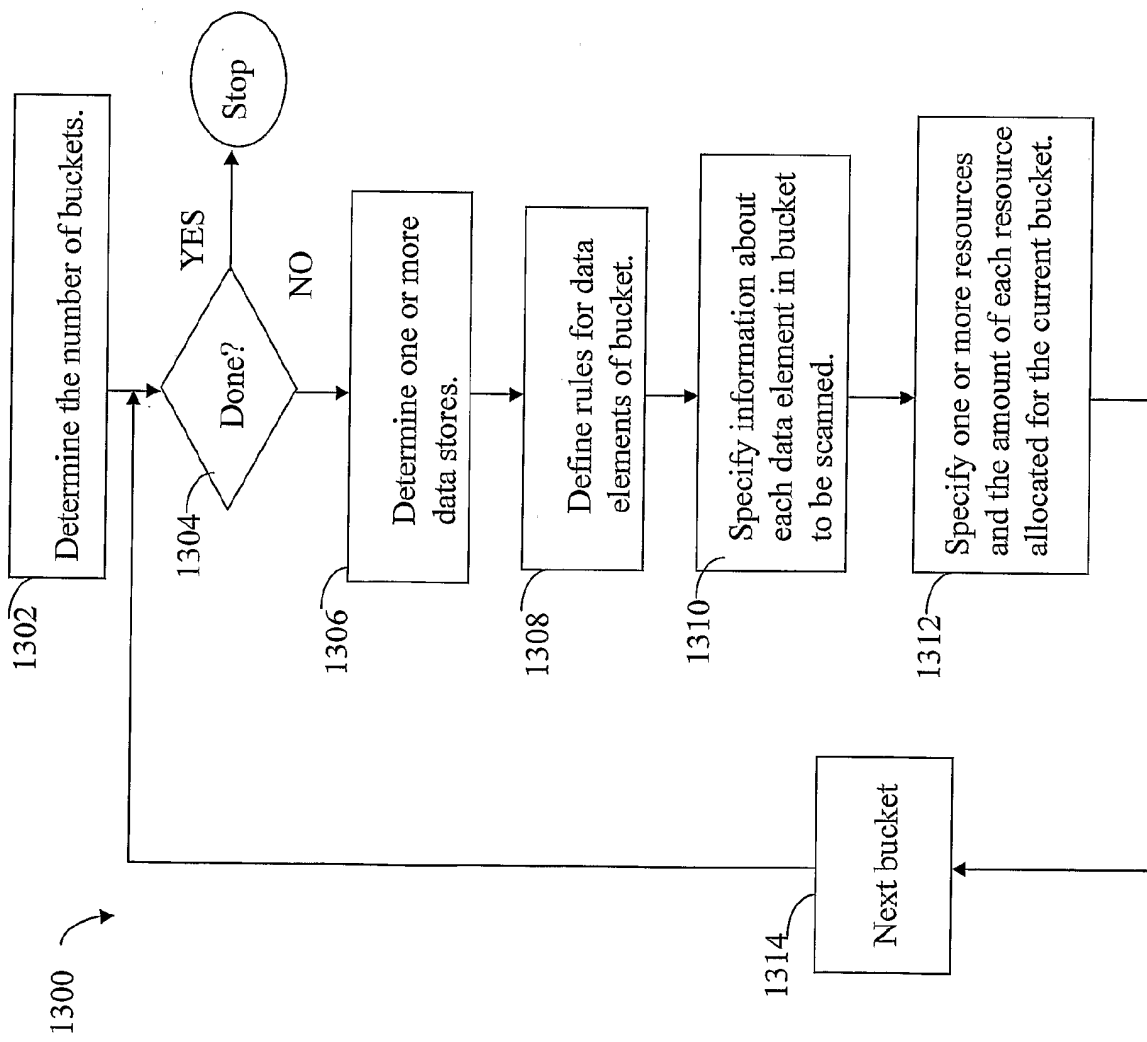


FIG. 18

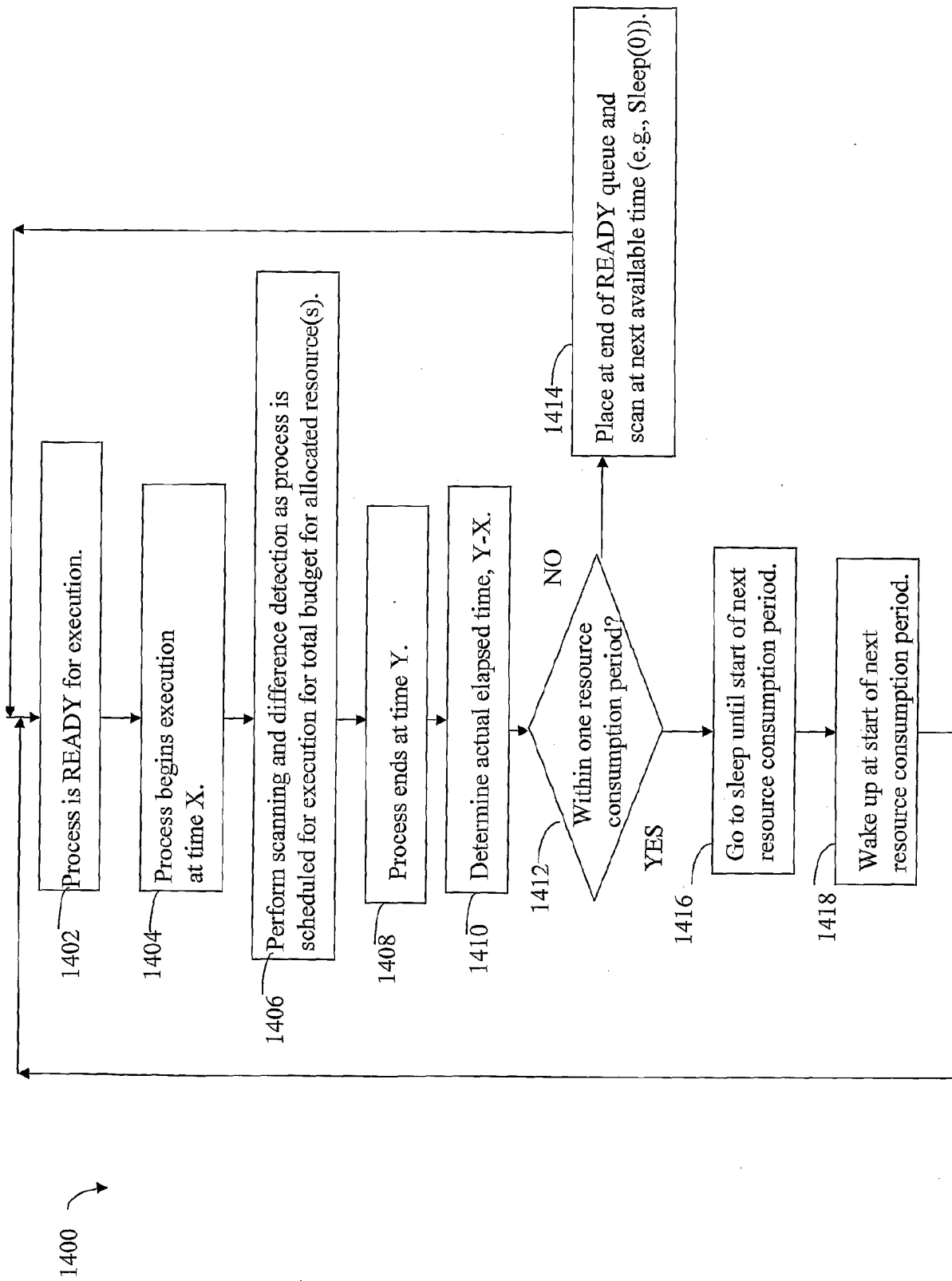


FIG. 19

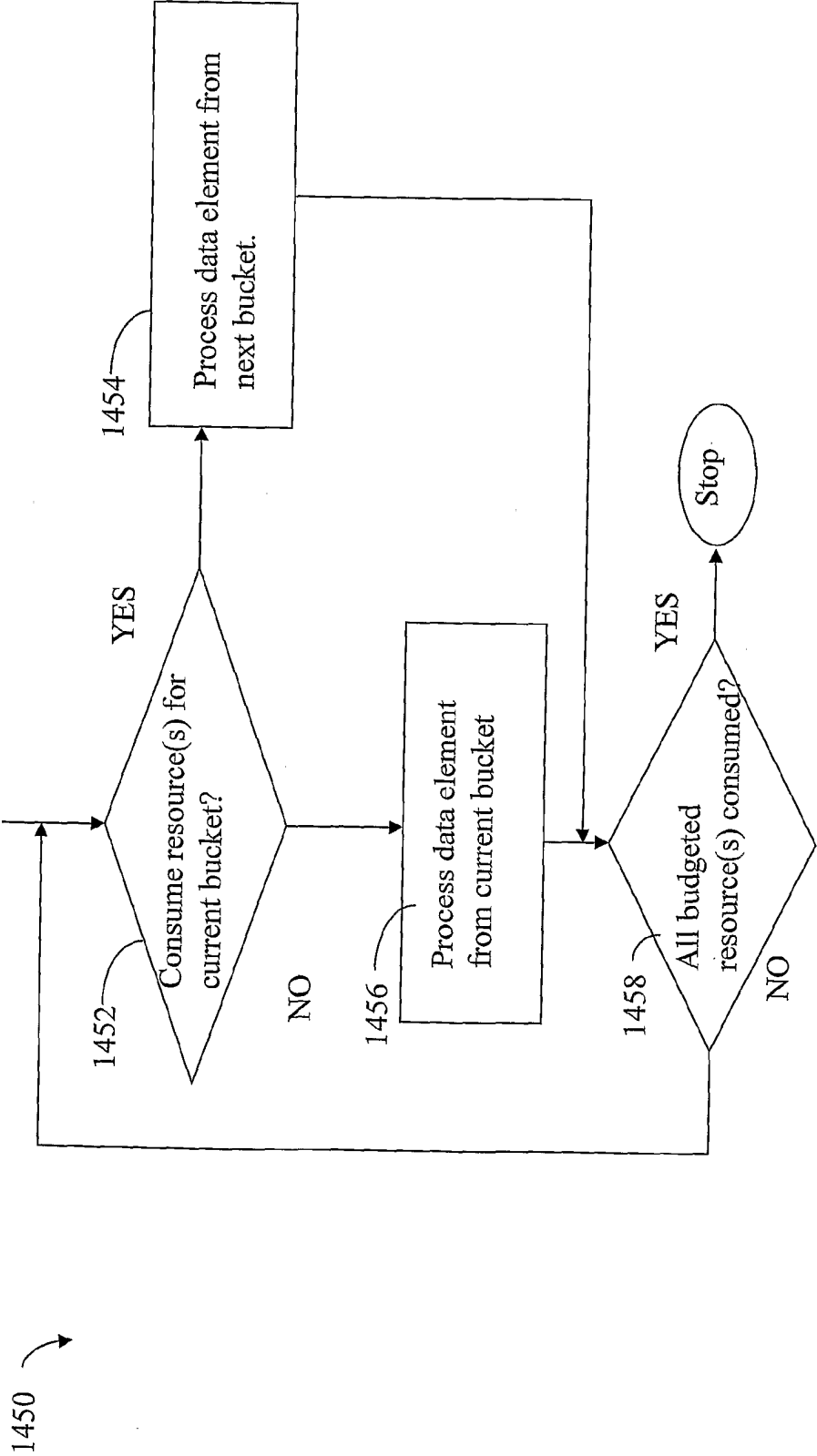


FIG. 20

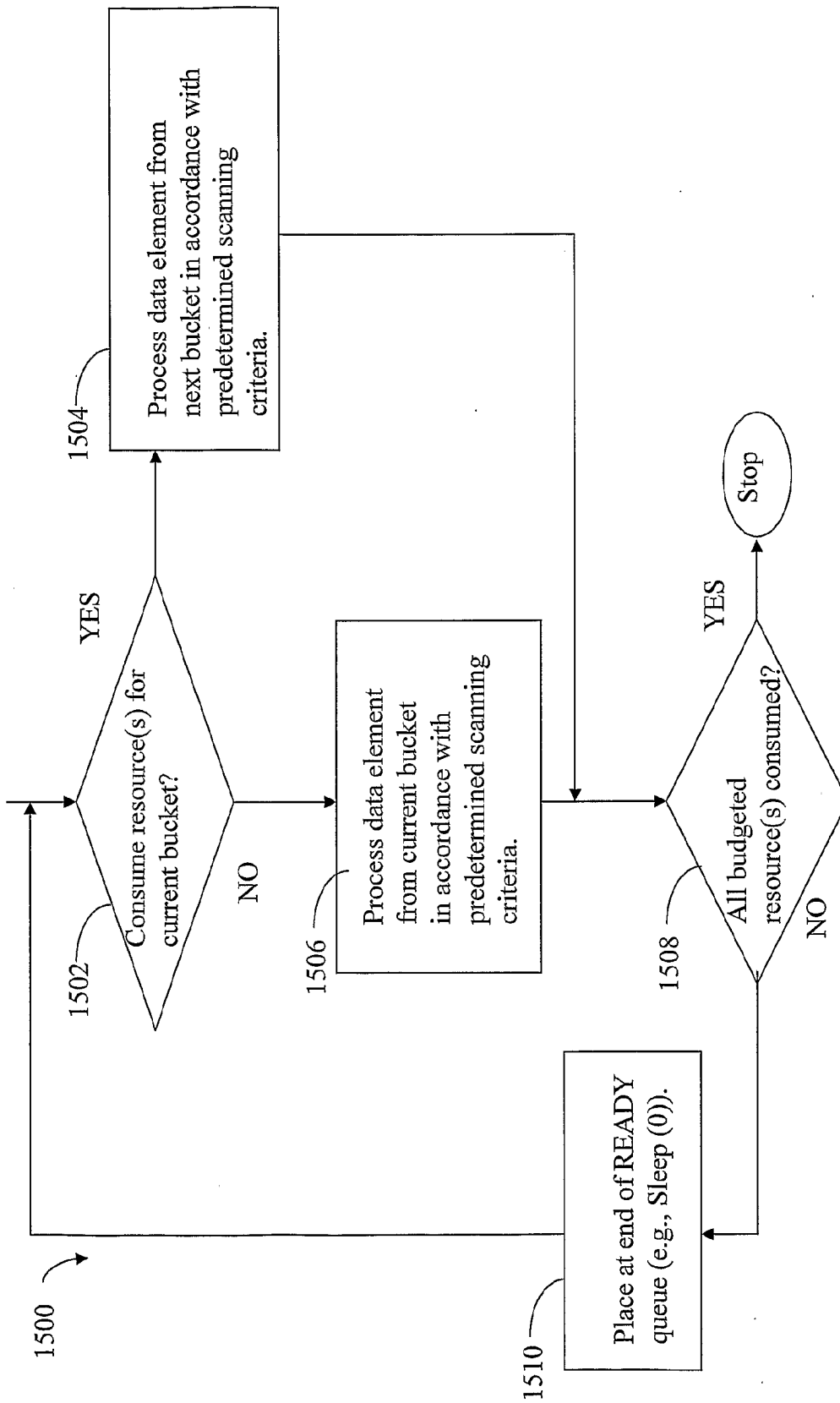


FIG. 21

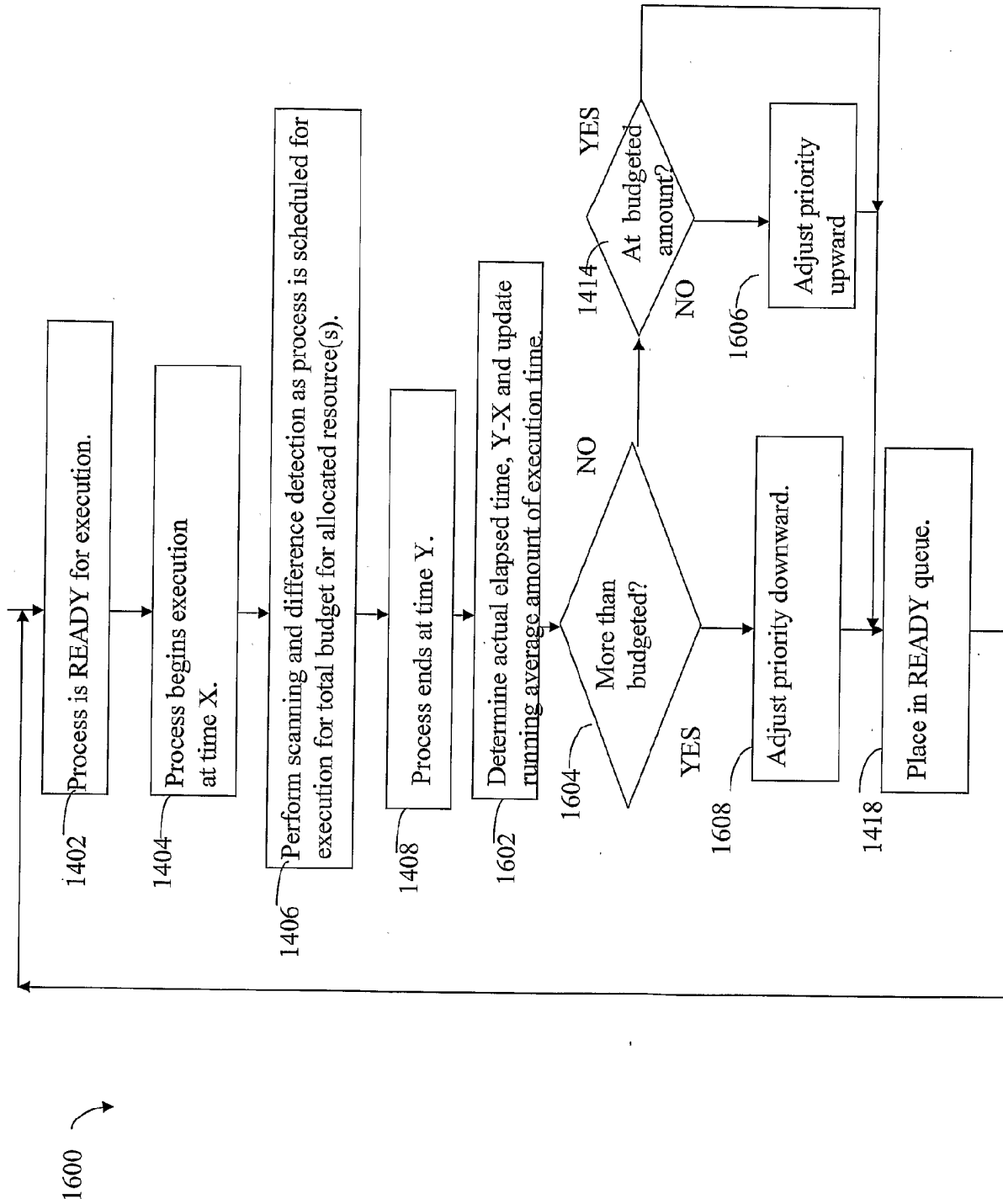
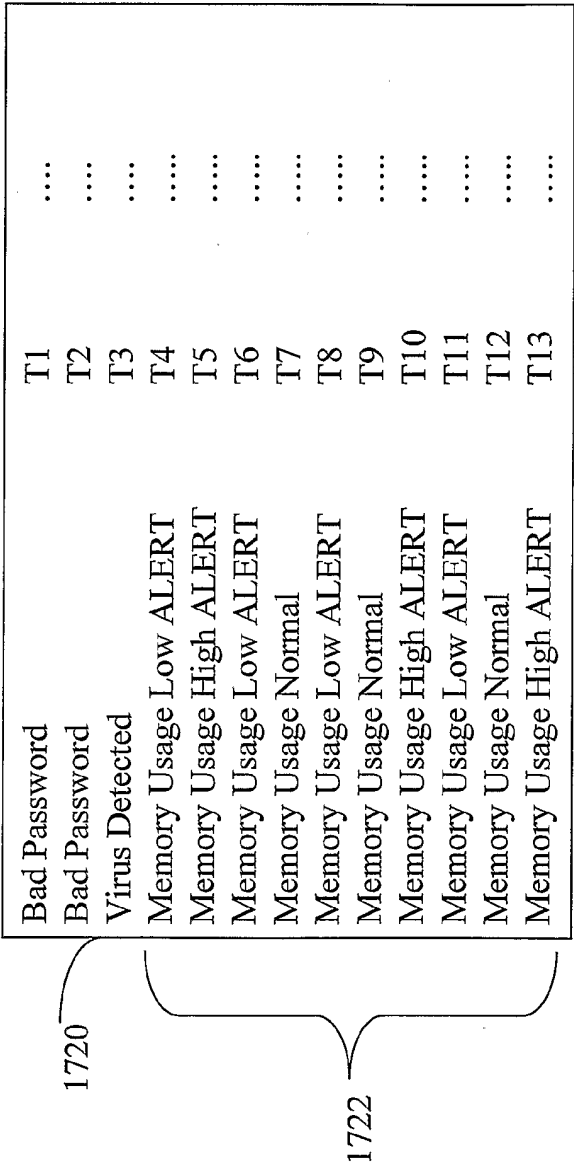
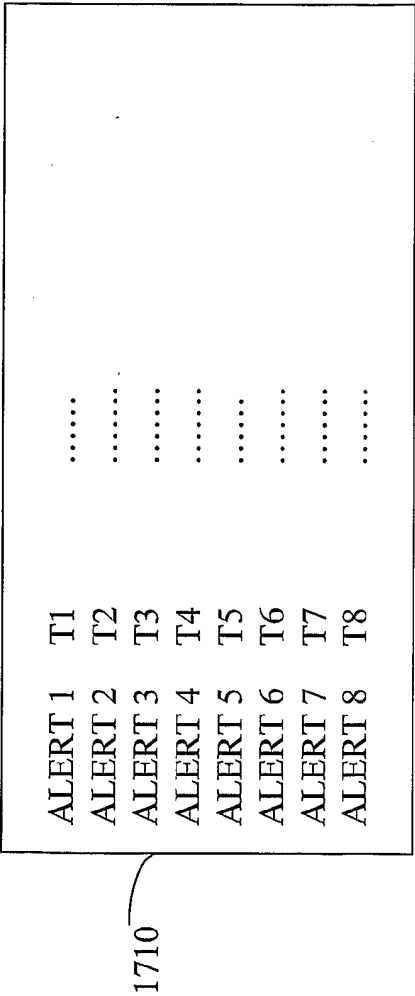


FIG. 22

1700



PRIOR ART

FIG. 23

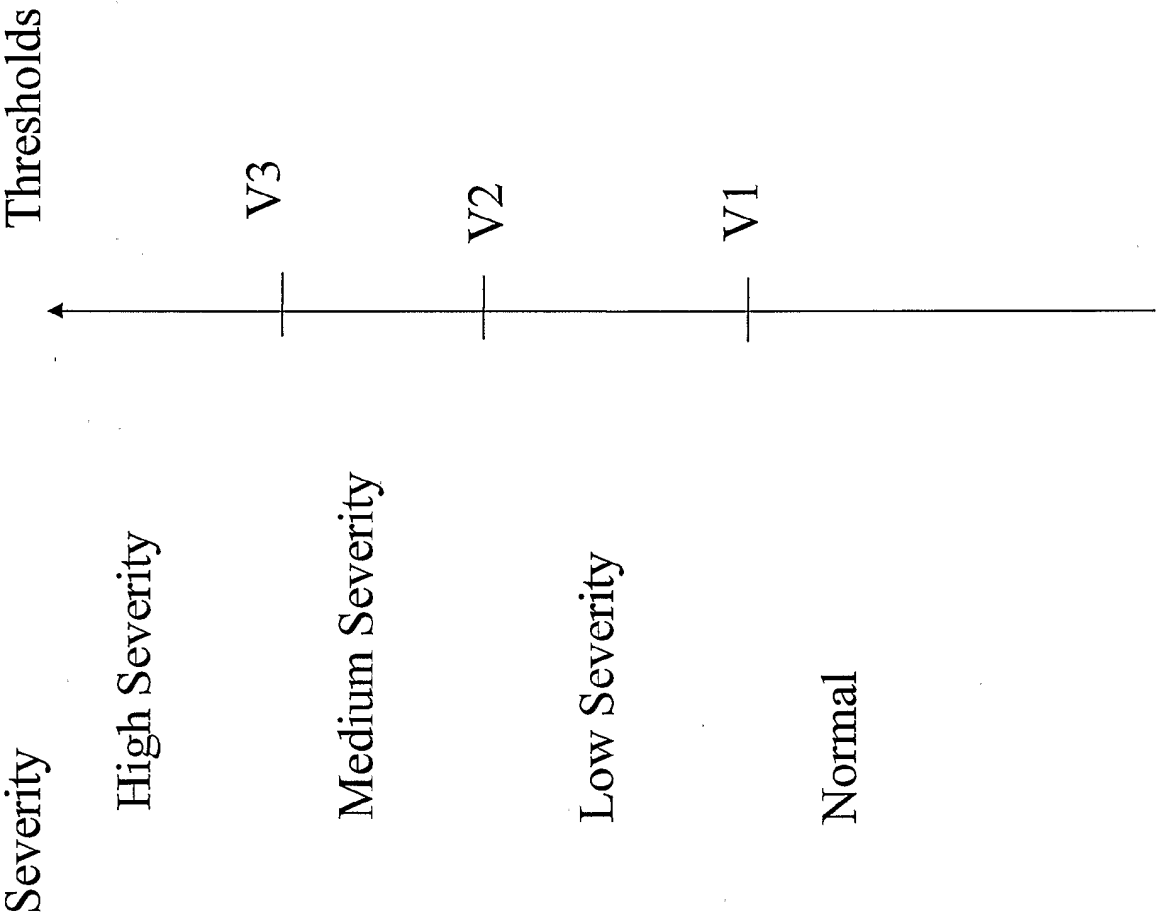


FIG. 24

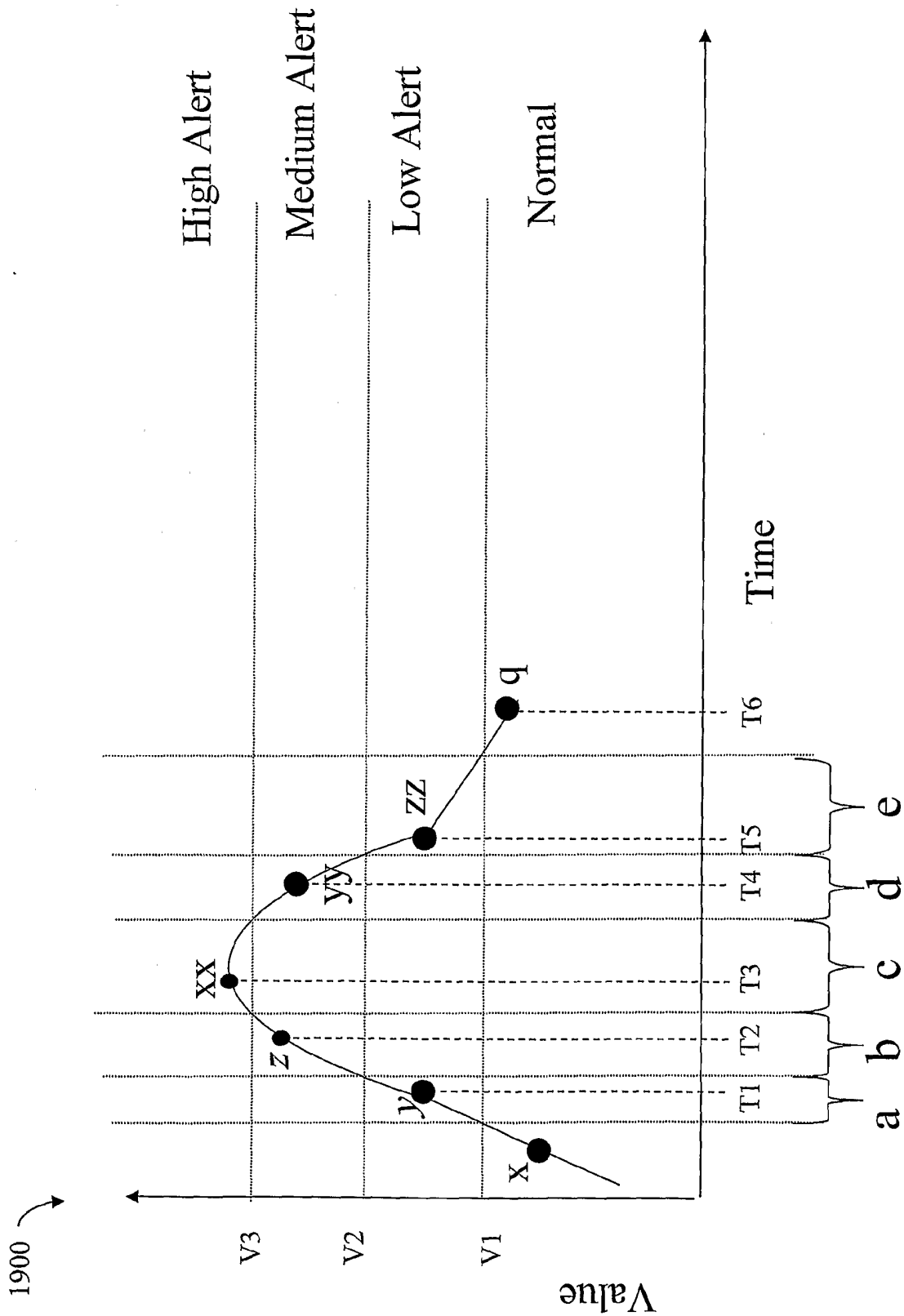


FIG. 25

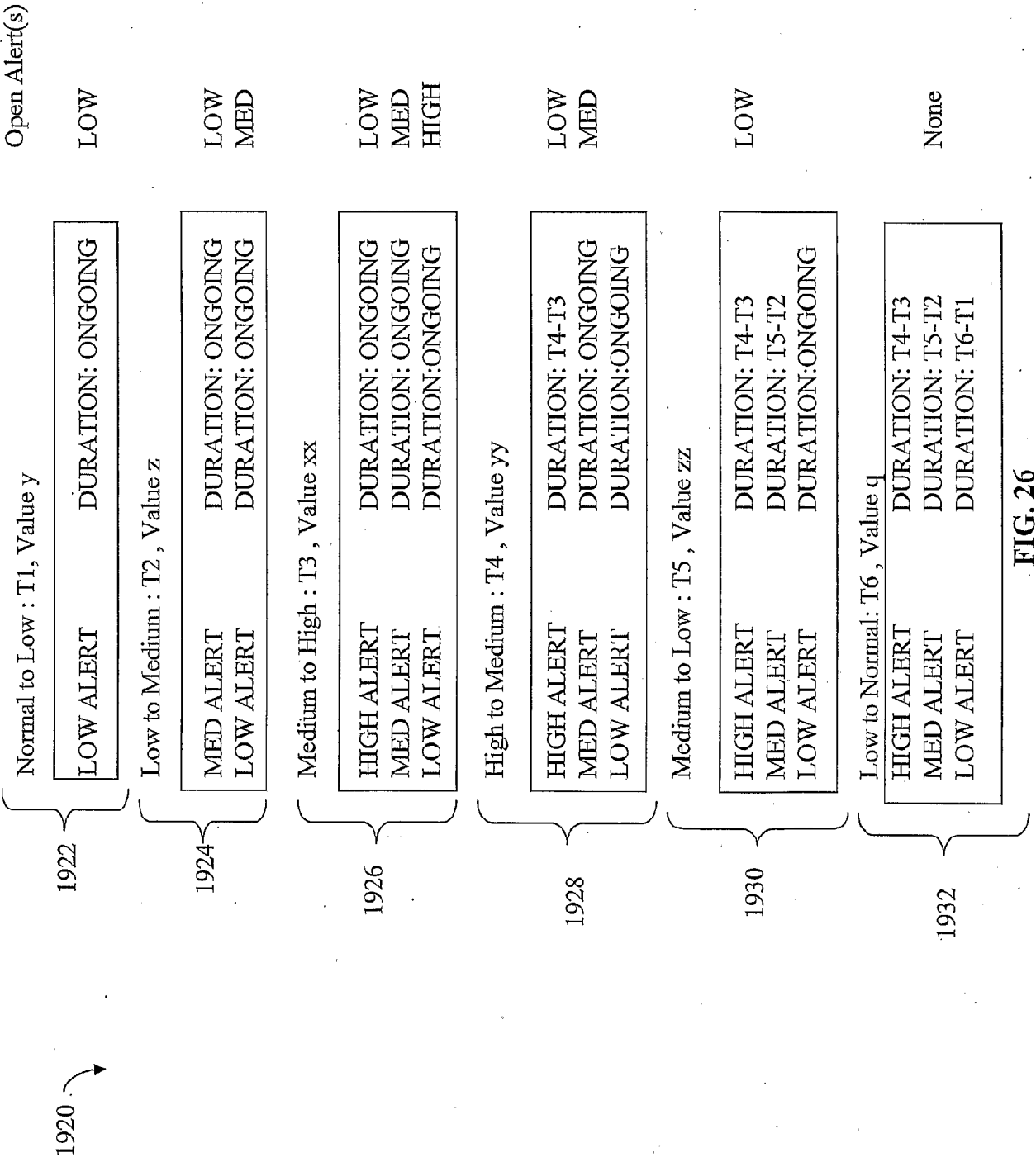


FIG. 26

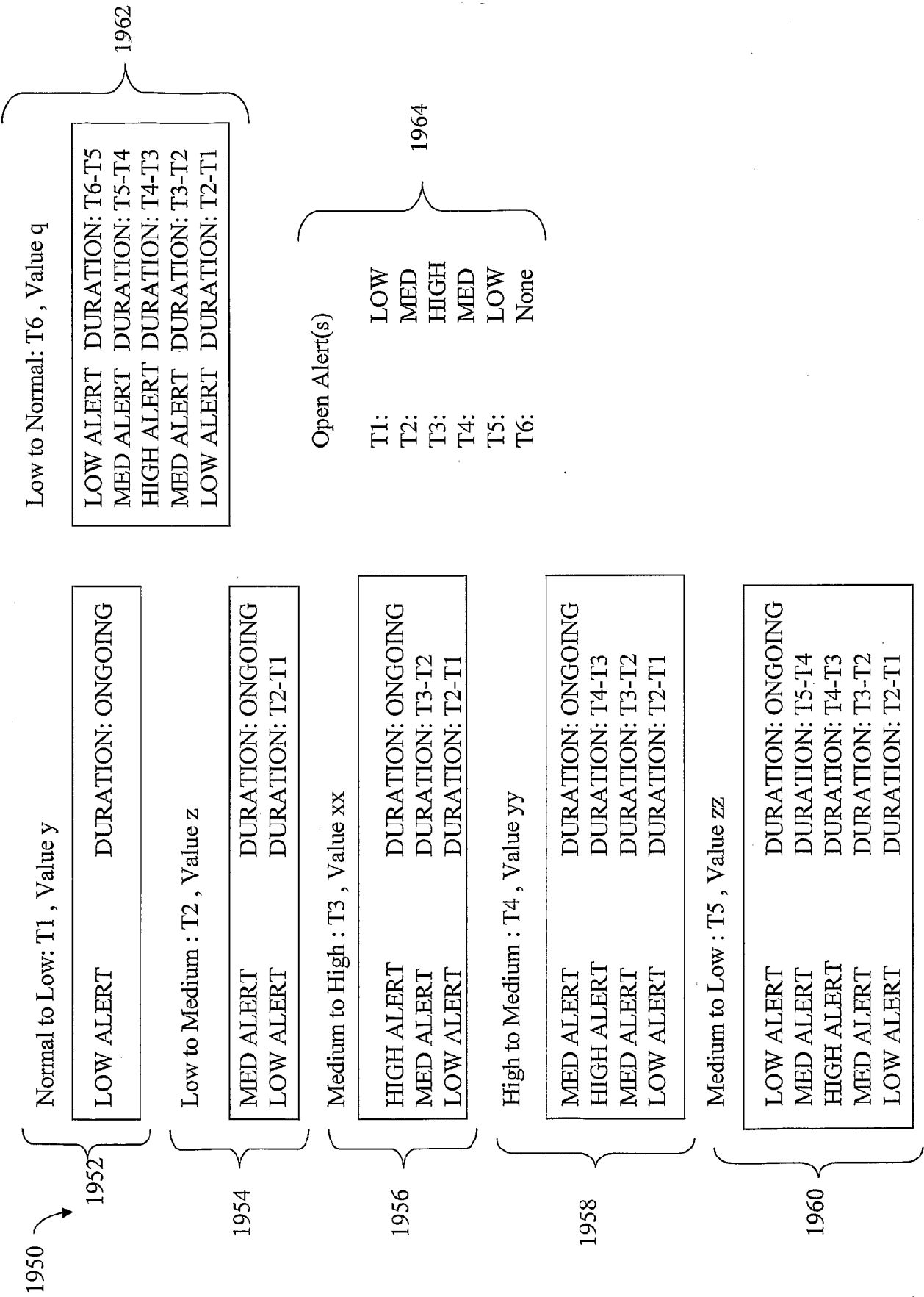


FIG. 27

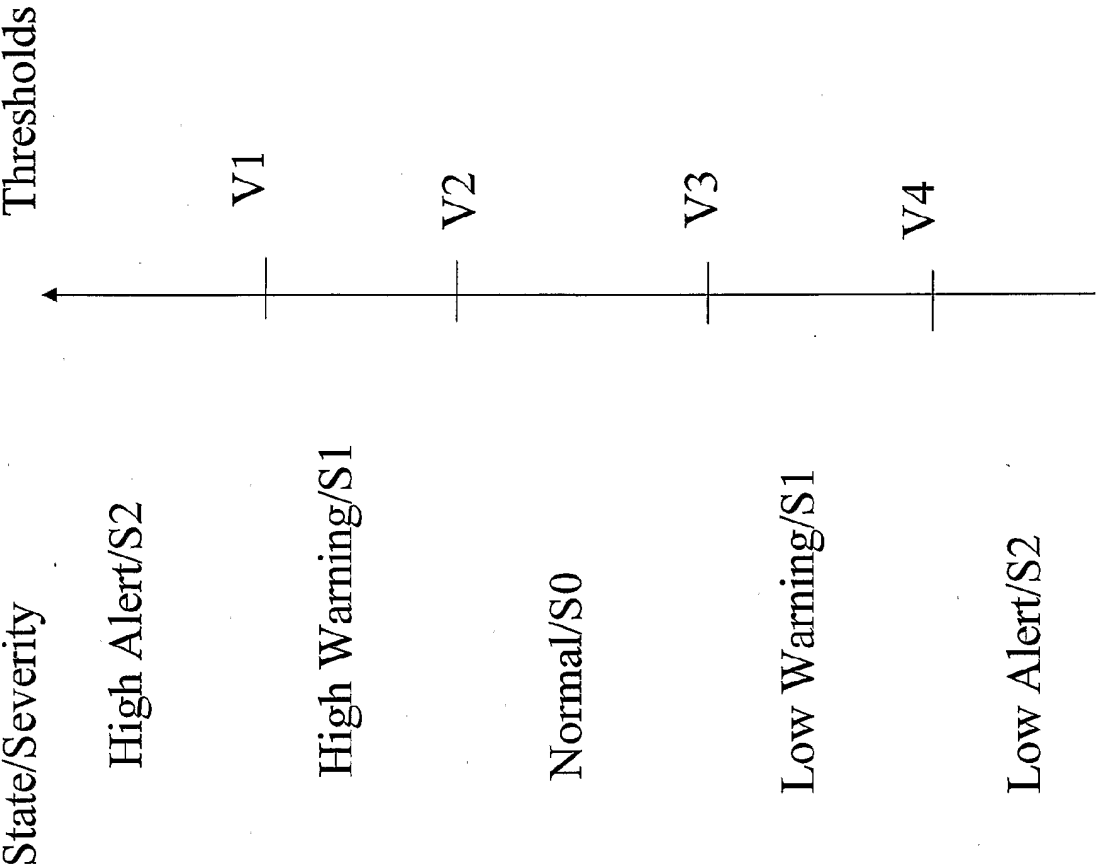


FIG. 28

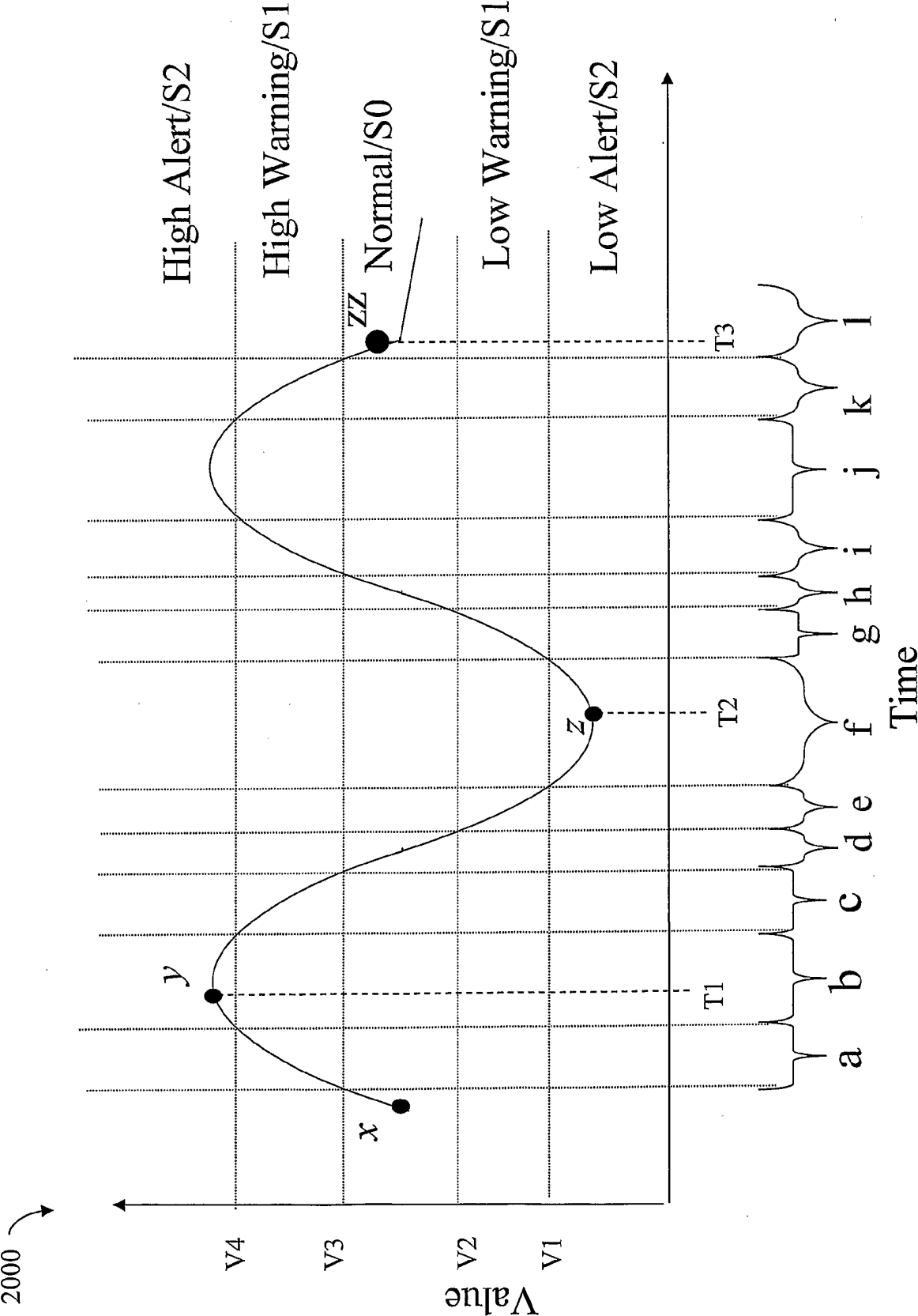


FIG. 29

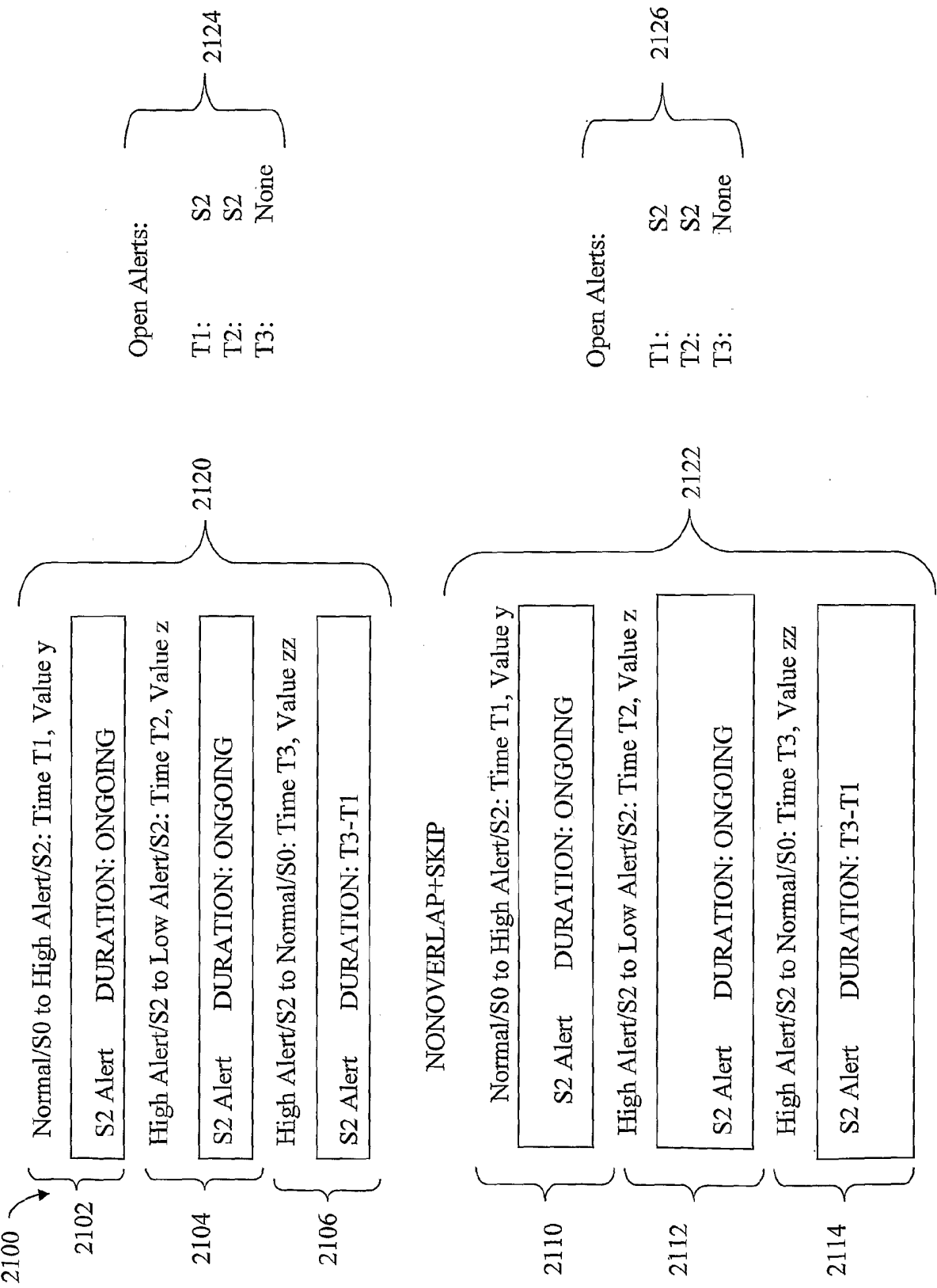


FIG. 30

2200 →

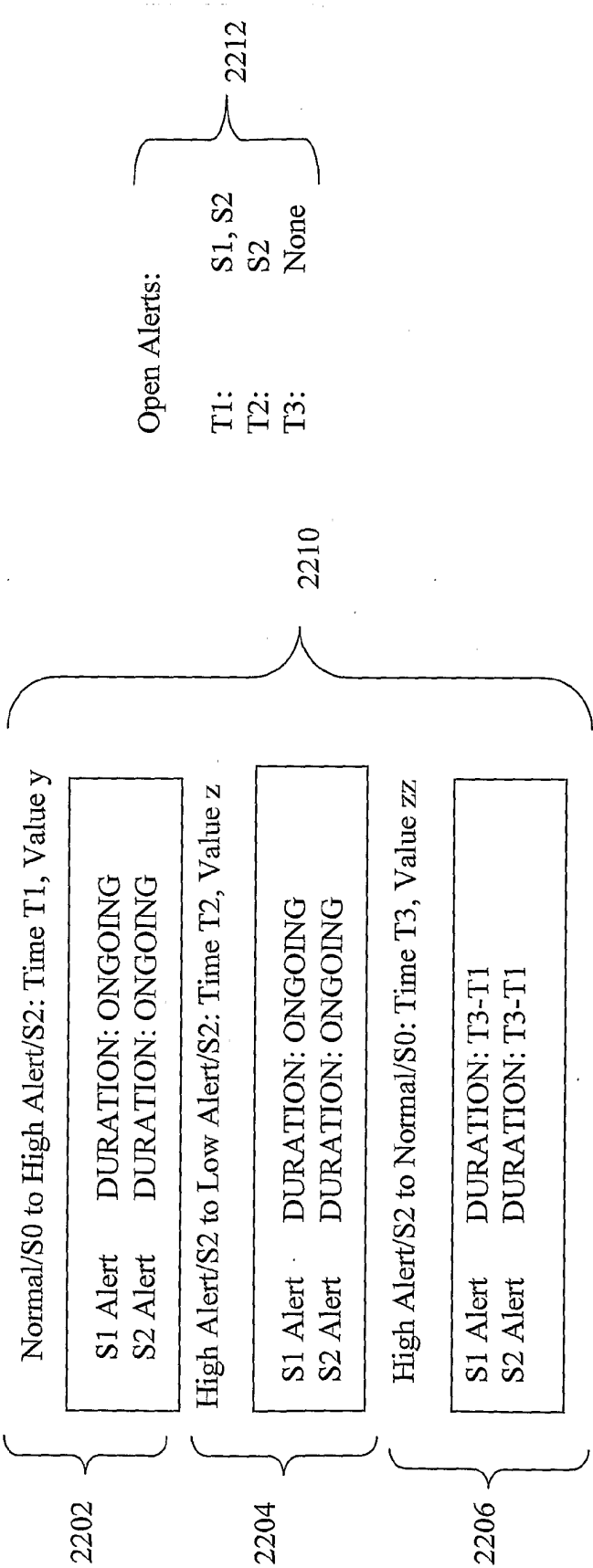


FIG. 31

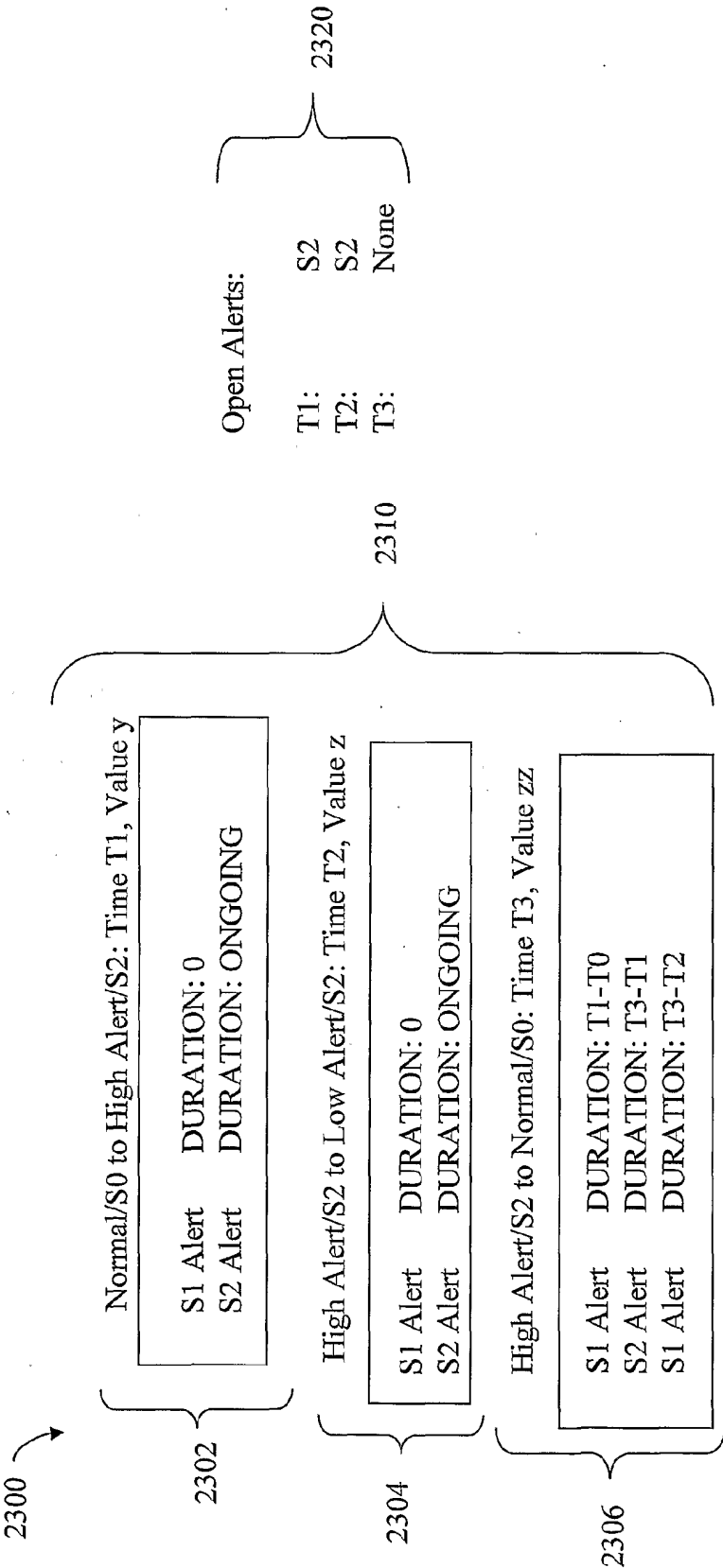


FIG. 32

2400 ↗

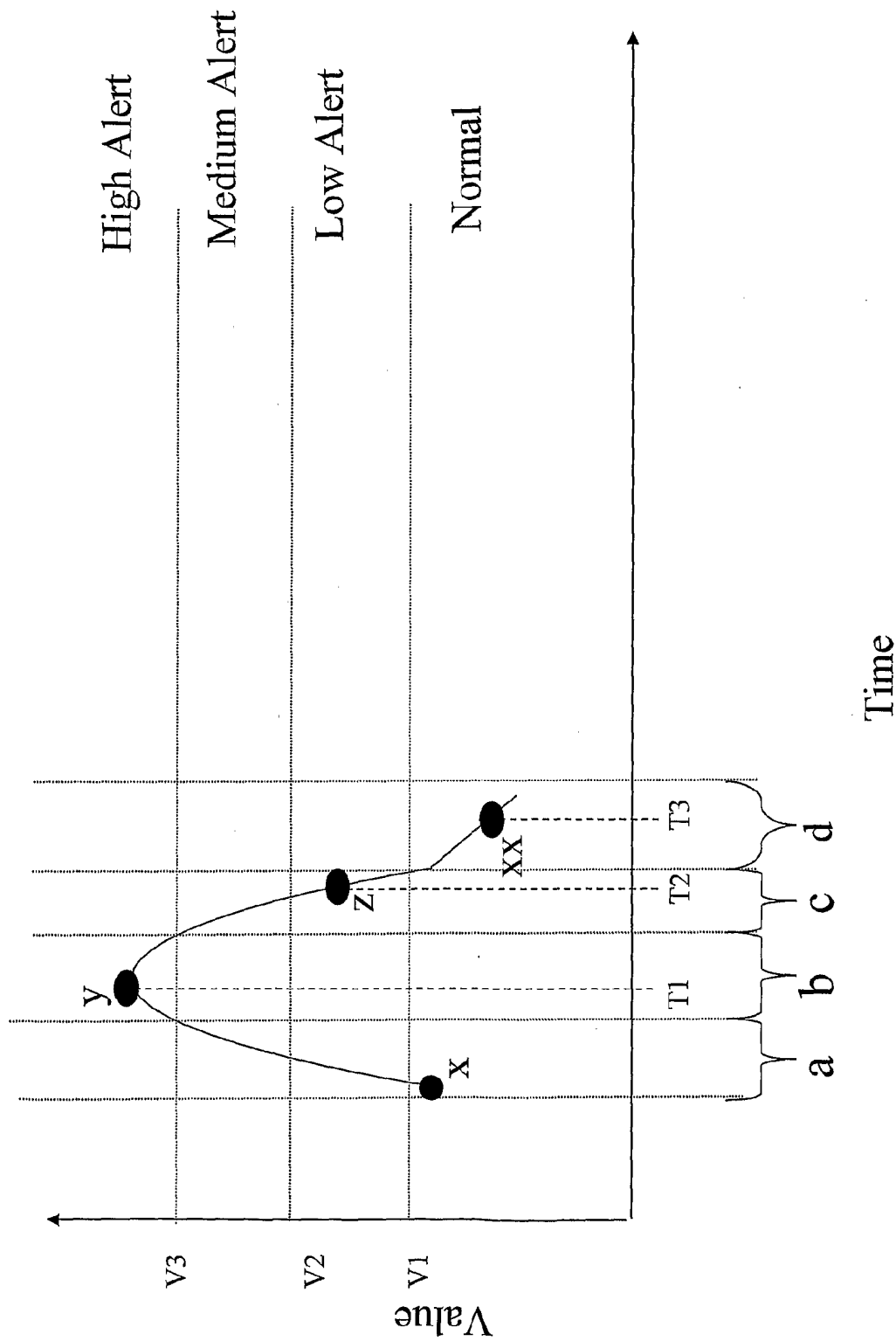


FIG. 33

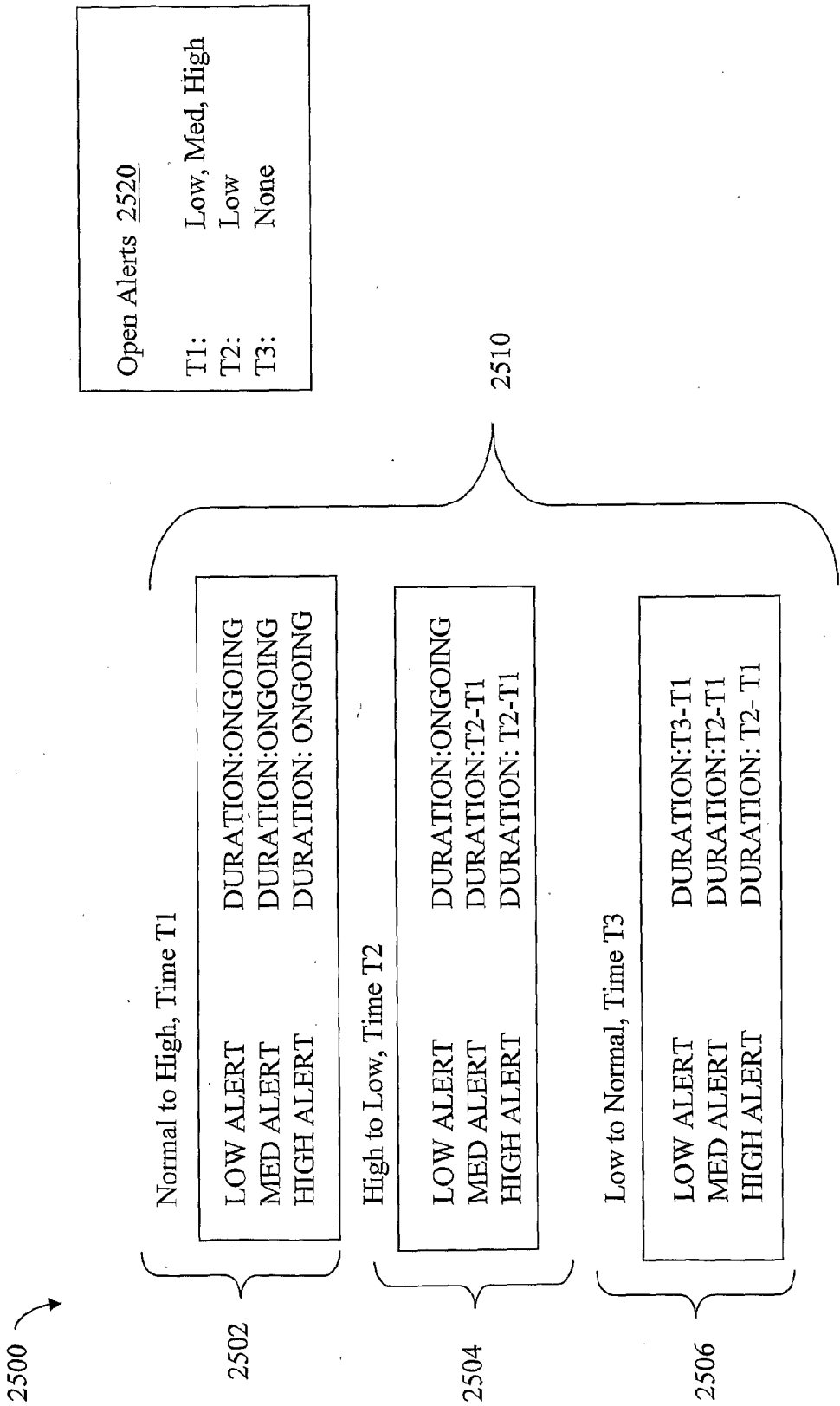


FIG. 34

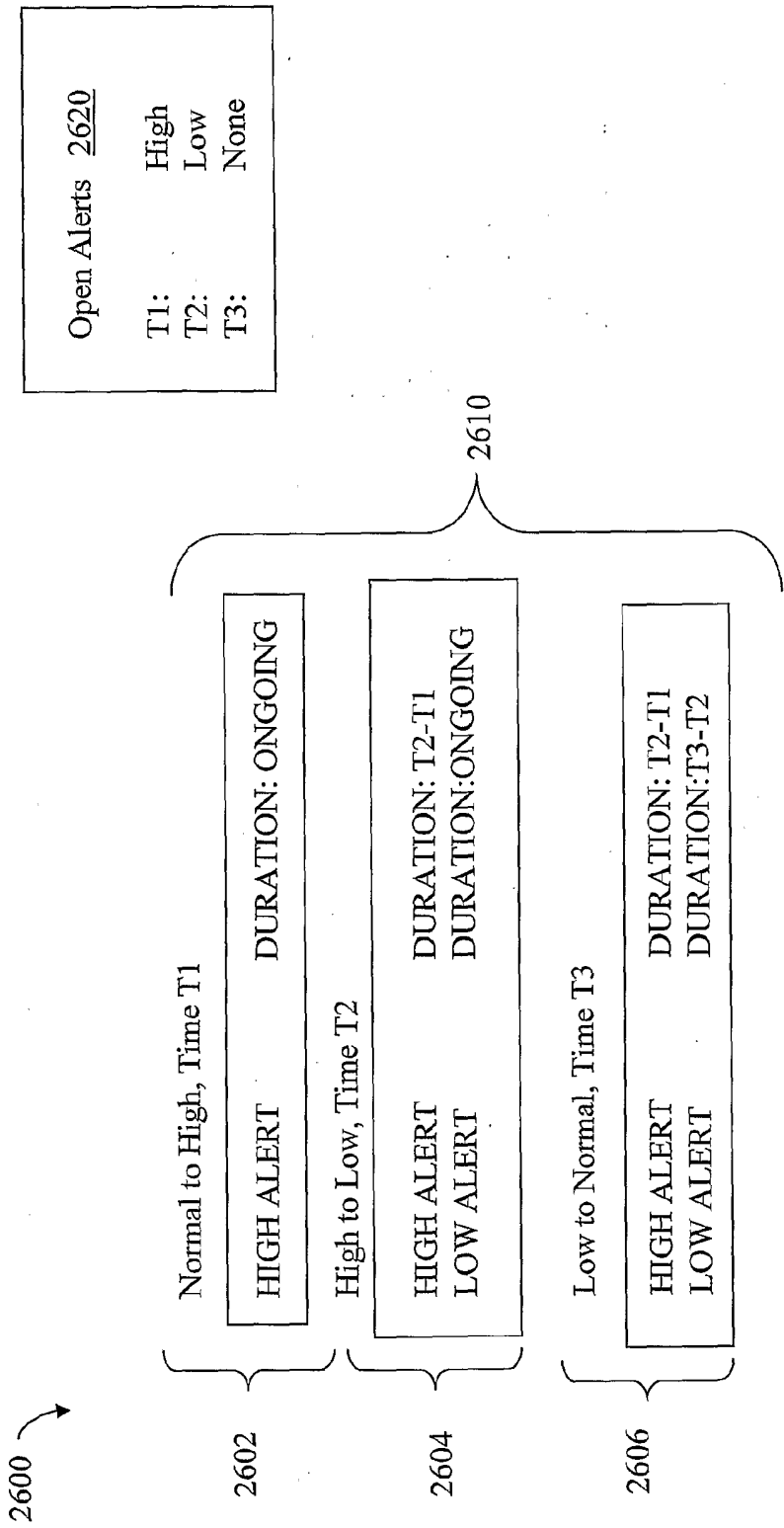


FIG. 35

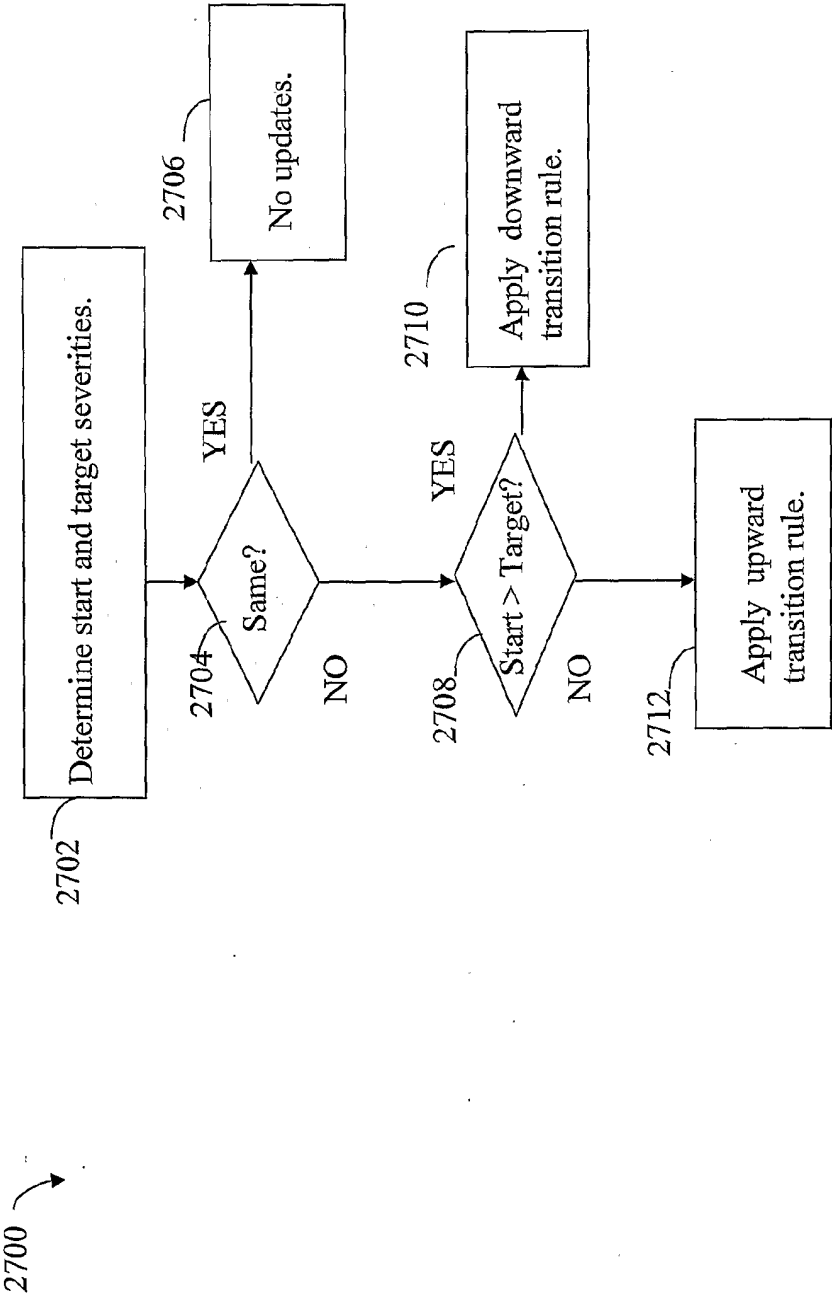


FIG. 36

2800 ↗

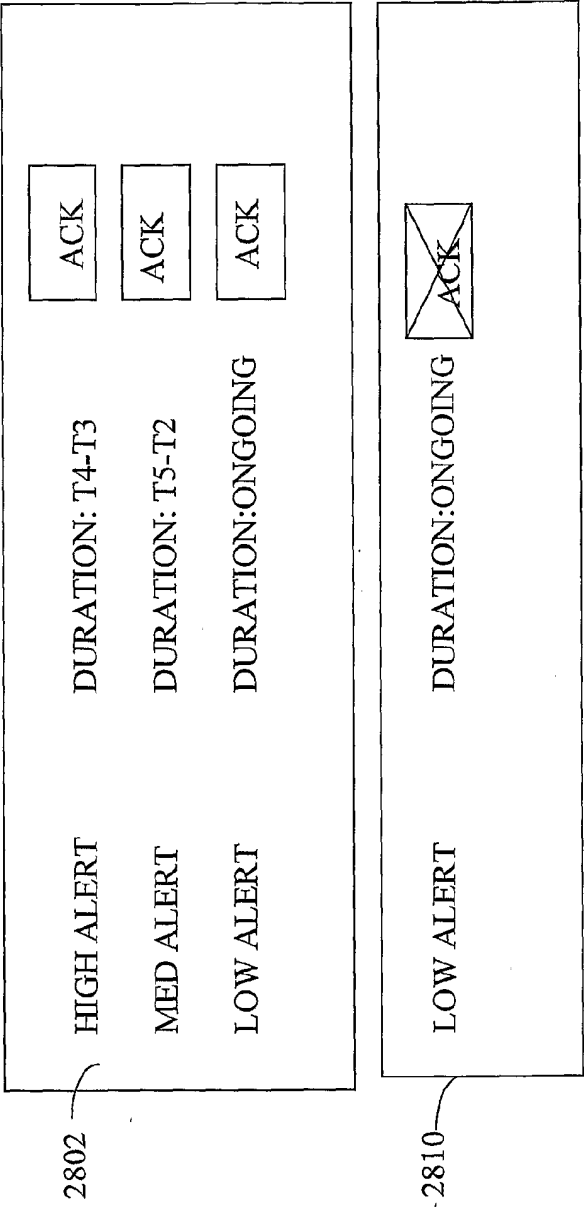


FIG. 37

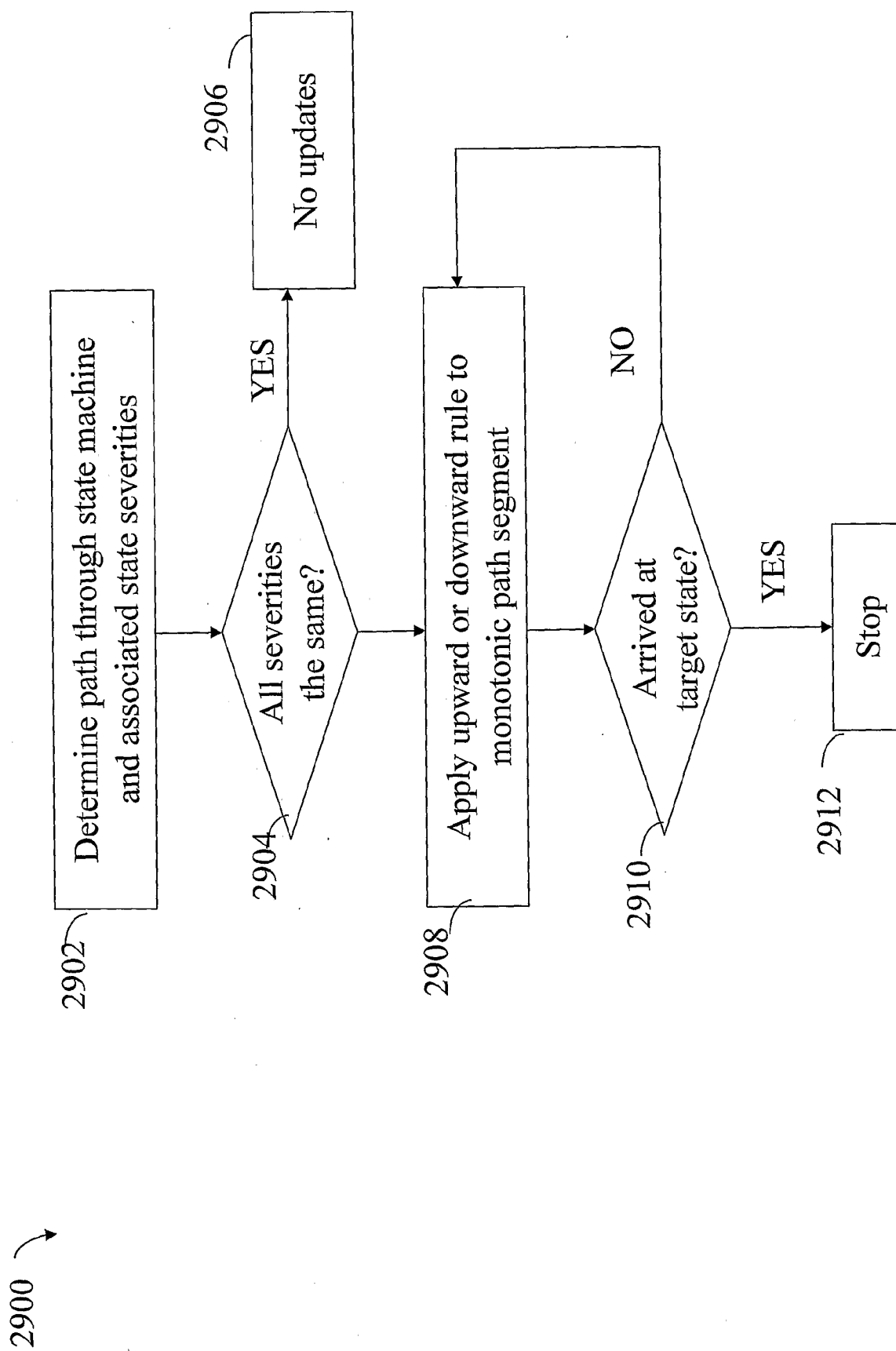


FIG. 38