US 20080294588A1

(19) United States

(12) Patent Application Publication

Morris et al.

(10) Pub. No.: US 2008/0294588 A1

(43) Pub. Date: Nov. 27, 2008

(54) EVENT CAPTURE, CROSS DEVICE EVENT CORRELATION, AND RESPONSIVE ACTIONS

(76) Inventors: **Stephen Jeffrey Morris**, Harvard, MA (US); **Richard Casey Clarkson**, Westborough, MA (US); **Steven Arnold Bolton**, Harvard, MA (US); **Robert Leo MacDonald, JR.**, Wrentham, MA (US); **Gregory Wayne Barrett**, Centreville, VA (US)

Correspondence Address:
BARRY W. CHAPIN, ESQ.
CHAPIN INTELLECTUAL PROPERTY LAW, LLC
WESTBOROUGH OFFICE PARK, 1700 WEST PARK DRIVE, SUITE 280
WESTBOROUGH, MA 01581 (US)

(21) Appl. No.: **12/125,115**

(22) Filed: **May 22, 2008**

Related U.S. Application Data

(60) Provisional application No. 60/939,517, filed on May 22, 2007, provisional application No. 60/939,503, filed on May 22, 2007, provisional application No. 60/939,521, filed on May 22, 2007, provisional application No. 60/939,528, filed on May 22, 2007.

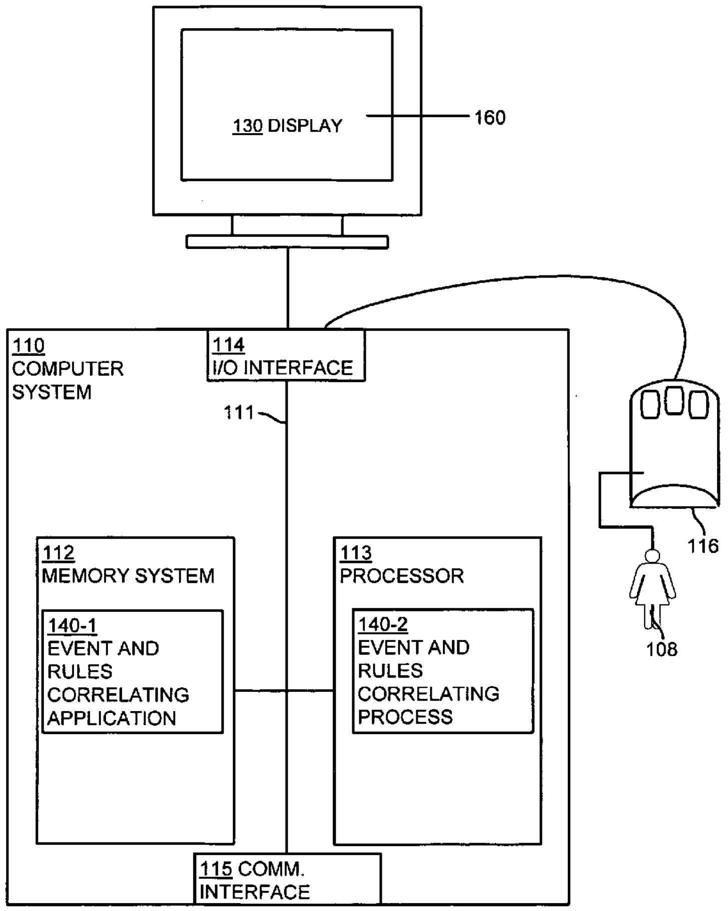
Publication Classification

(51) **Int. Cl.**
G06N 5/02 (2006.01)

(52) **U.S. Cl.** **706/47**

(57) **ABSTRACT**

Correlating security event data according to one or more rules to initiate one or more actions is disclosed. Security event data is received, via an event pipeline, that defines an occurrence of an event. Each type of source of event data may have its own pipeline. Event objects are created from the received security event data, and are gathered into a rules engine. The rules engine includes a plurality of rules. Each rule defines one or more conditions to be met and one or more actions to be taken in response. A rule is evaluated at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met. The one or more actions defined by the evaluated rule are initiated whenever the one or more conditions defined by that rule are met.



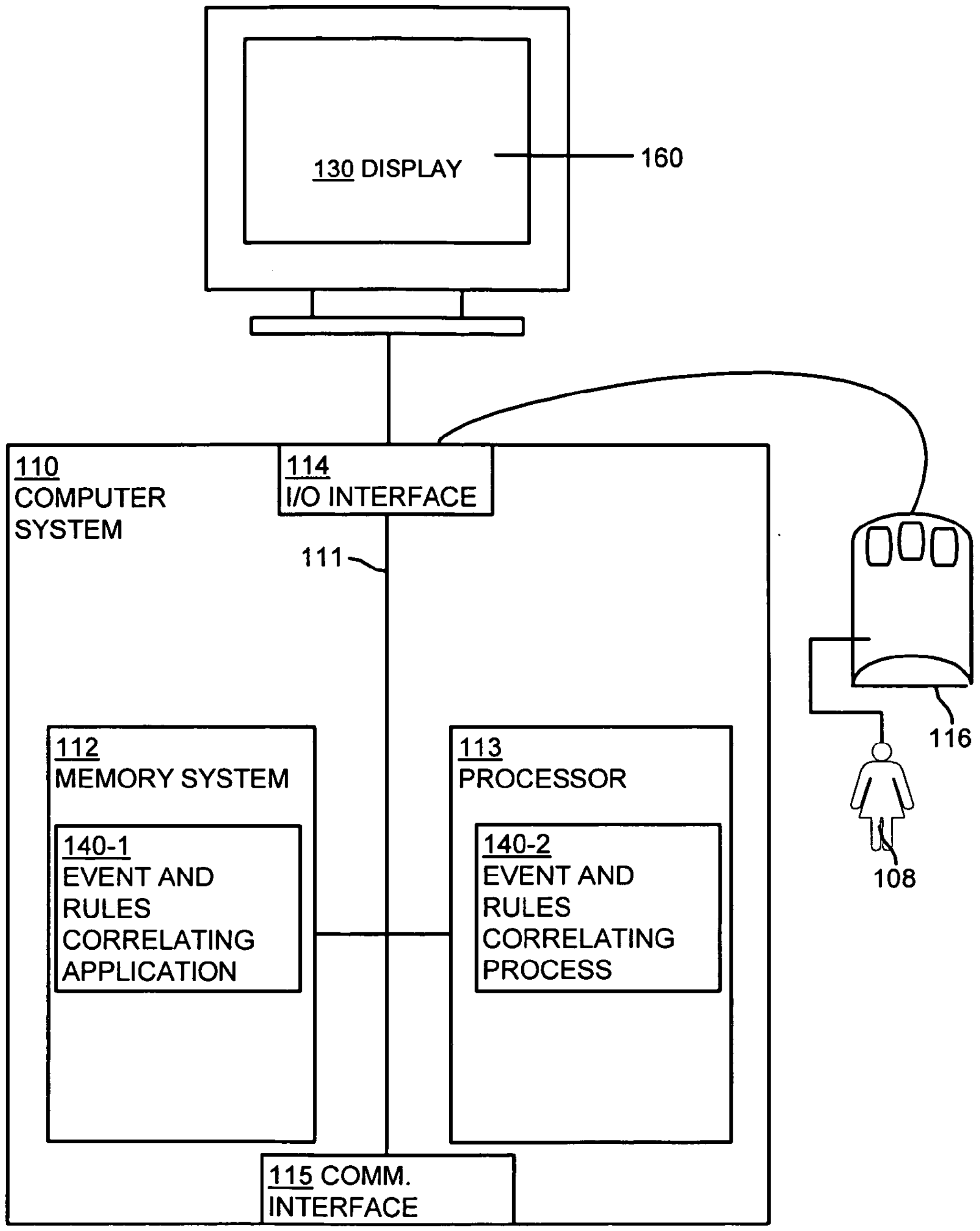

130 DISPLAY
160
110 COMPUTER SYSTEM
114 I/O INTERFACE
111
112 MEMORY SYSTEM
140-1 EVENT AND RULES CORRELATING APPLICATION
113 PROCESSOR
140-2 EVENT AND RULES CORRELATING PROCESS
115 COMM. INTERFACE
116
108

FIG. 1

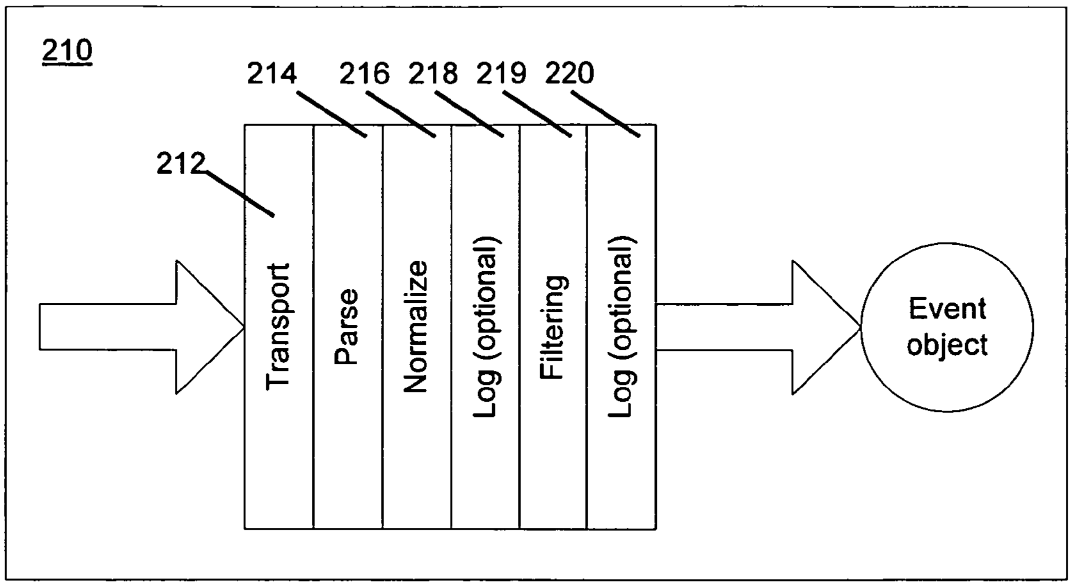

210
214
216
218
219
220
212
Transport
Parse
Normalize
Log (optional)
Filtering
Log (optional)
Event object

FIG. 2A

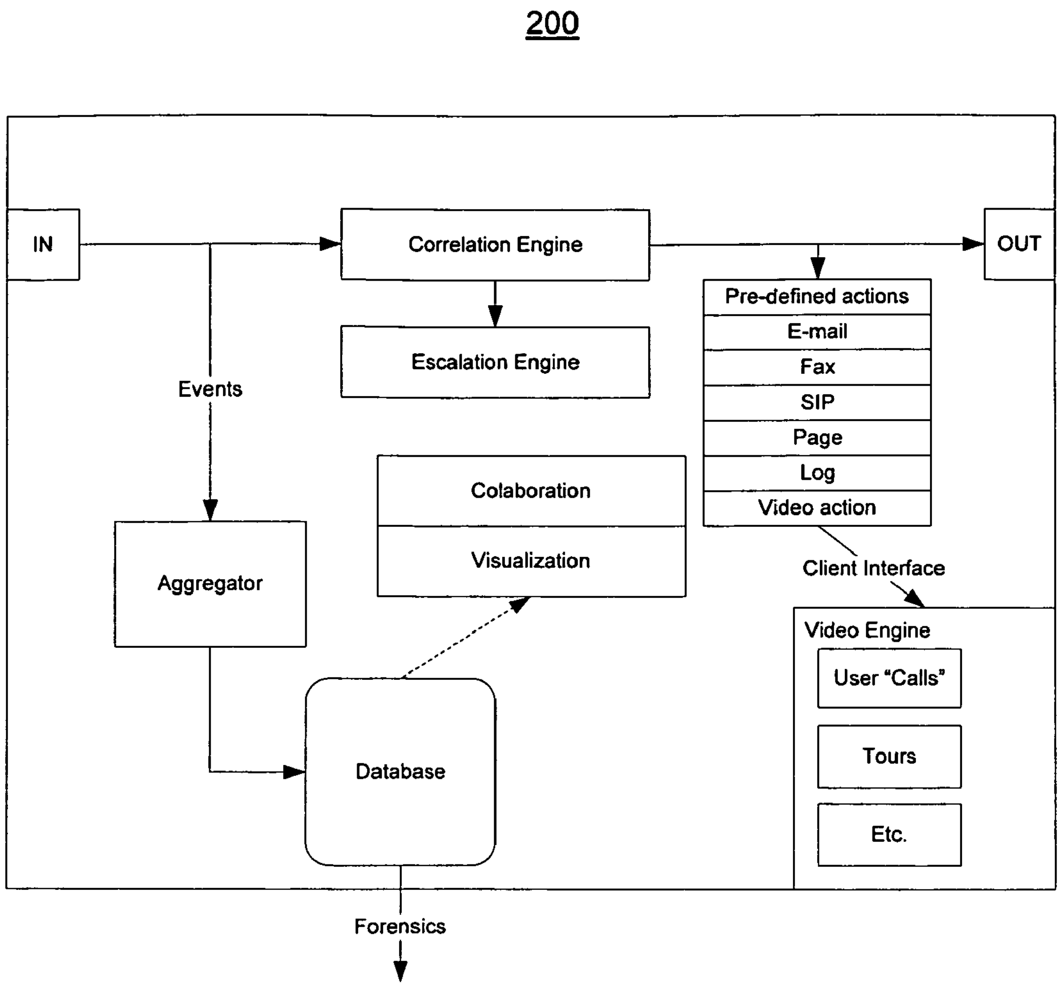

200
IN
Correlation Engine
OUT
Escalation Engine
Events
Pre-defined actions
E-mail
Fax
SIP
Page
Log
Video action
Colaboration
Visualization
Aggregator
Client Interface
Video Engine
User "Calls"
Tours
Etc.
Database
Forensics

FIG. 2B

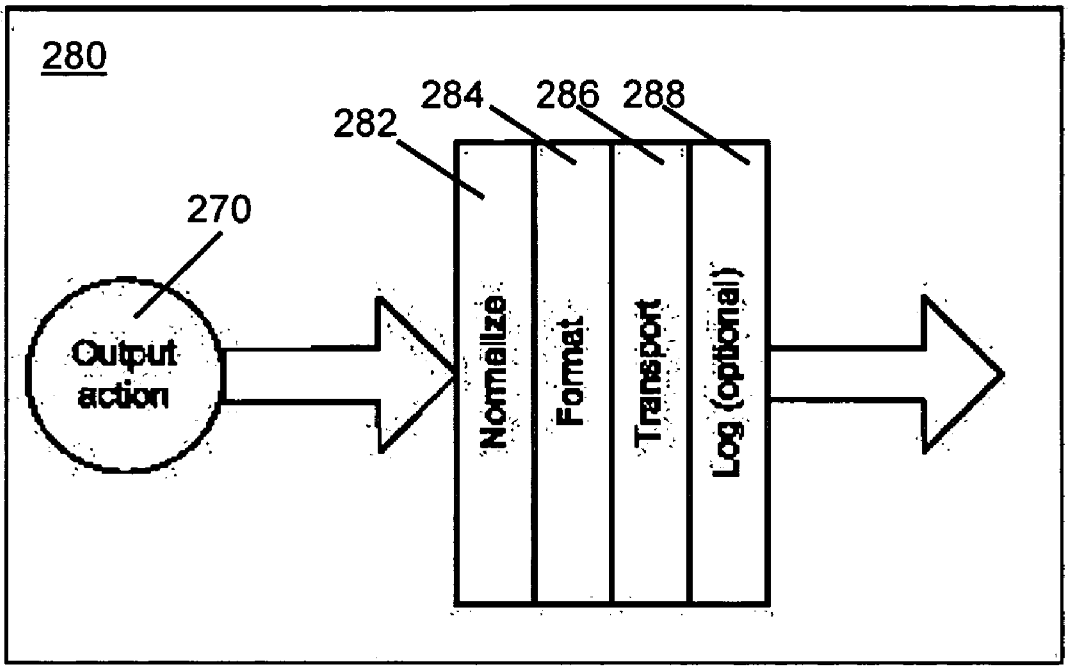

280
282
284
286
288
270
Output action
Normalize
Format
Transport
Log (optional)

FIG. 2C

```
<correlation-engine>
 <transports>
  <transport classname="com.vidsys.event.process.impl.HttpRequestTransport">
   <instance id="http-transport">
    <path>/alarm</path>
    <method>GET</method>
   </instance>
  </transport>
 </transports>
 <parsers>
  <parser id="urlencoded"classname="com.vidsys.event.process.impl.WwwFormUrlEncodedParser"/>
 </parsers>
 <filters>
 </filters>
 <loggers>
 </loggers>
 <modifiers>
 </modifiers>
 <normalizers>
 </normalizers>
 <formatters>
 </formatters>
 <actions>
 </actions>
 <event-pipelines>
  <event-pipeline id="http-pipeline">
   <transport idref="http-transport"/>
   <parser idref="urlencoded"/>
  </event-pipeline>
 </event-pipelines>
 <action-pipelines>
 </action-pipelines>
 <object-expiration>
 </object-expiration>
</correlation-engine>
```

FIG. 3A

```
when
        TimeOfDay( time ==      TimeOfDay.AFTER_HOURS )
        $location : Location( access == "restricted" )
        $person  : Person( location == $location, clearance != "restricted" )
then
        System.out.println("ALERT Unauthorized person " + $person.getName() + " in restricted location
        " + $location.getName() + " after hours.");
```

FIG. 3B

```
duration 3000; // 3 seconds
when
        $location : Location( access == "restricted" )
        $person : Person( location == $location, clearance != "restricted" )
        not Person( location == $location, clearance == "restricted" )
then
        System.out.println("ALERT Unattended person " +   $person.getName() + " in restricted location
        " +        $location.getName());
```

FIG. 3C

```
rule "Stopped Vehicle"
when
        $event   : Event( type == Event.STOPPED_VEHICLE, $evprox : proximity )
        $acc      : EventAccumulator( proximity == $evprox );
then
        $event.setWeight(2);
        $acc.add($event);
        modify($acc);

rule "Guard Late to Reach Next Checkpoint"
when
        $event   : Event( type == Event.LATE_GUARD, $evprox : proximity )
        $acc     : EventAccumulator( proximity == $evprox );
then
        $event.setWeight(3);
        $acc.add($event);
        modify($acc);
```

FIG. 3D

```
rule "Sum Of All Fears Exceeds Threshold"
when
        $ea : EventAccumulator()
        eval($ea.getWeight() > 10)
then
        System.out.println("ALERT Weight of cumulative events exceeds threshold\n    " +
        $ea.printDetail());
```

FIG. 3E

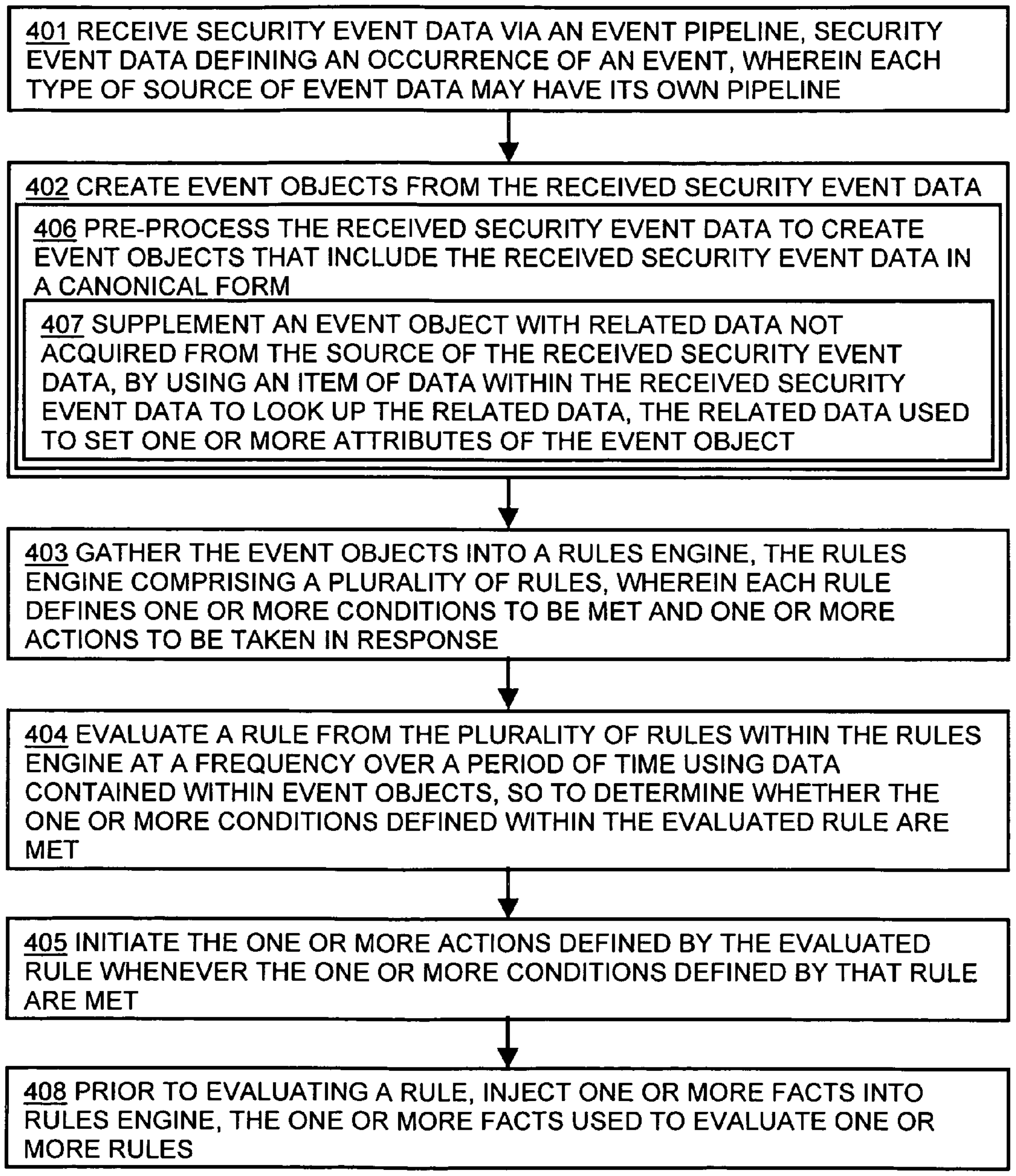
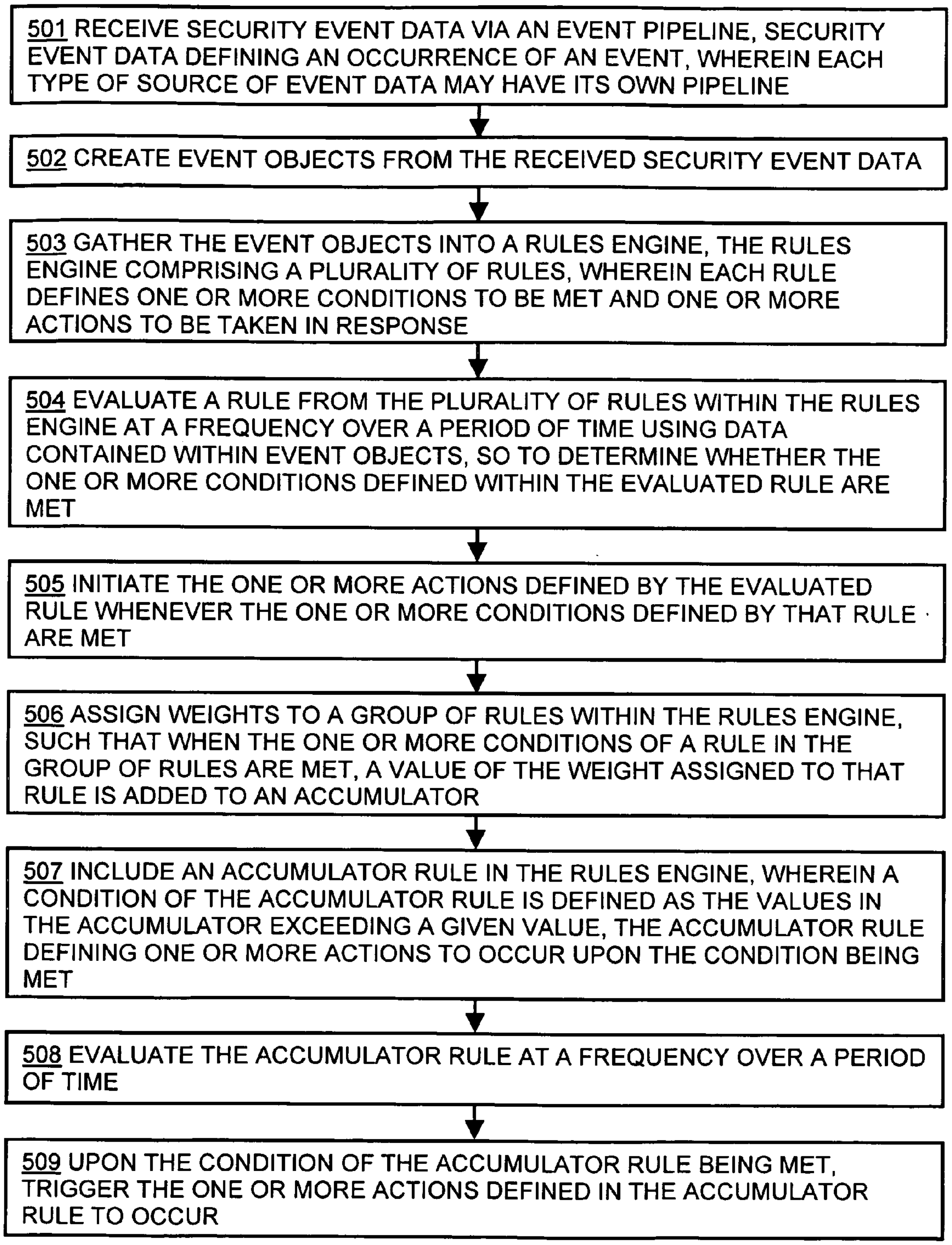

401 RECEIVE SECURITY EVENT DATA VIA AN EVENT PIPELINE, SECURITY EVENT DATA DEFINING AN OCCURRENCE OF AN EVENT, WHEREIN EACH TYPE OF SOURCE OF EVENT DATA MAY HAVE ITS OWN PIPELINE
402 CREATE EVENT OBJECTS FROM THE RECEIVED SECURITY EVENT DATA
406 PRE-PROCESS THE RECEIVED SECURITY EVENT DATA TO CREATE EVENT OBJECTS THAT INCLUDE THE RECEIVED SECURITY EVENT DATA IN A CANONICAL FORM
407 SUPPLEMENT AN EVENT OBJECT WITH RELATED DATA NOT ACQUIRED FROM THE SOURCE OF THE RECEIVED SECURITY EVENT DATA, BY USING AN ITEM OF DATA WITHIN THE RECEIVED SECURITY EVENT DATA TO LOOK UP THE RELATED DATA, THE RELATED DATA USED TO SET ONE OR MORE ATTRIBUTES OF THE EVENT OBJECT
403 GATHER THE EVENT OBJECTS INTO A RULES ENGINE, THE RULES ENGINE COMPRISING A PLURALITY OF RULES, WHEREIN EACH RULE DEFINES ONE OR MORE CONDITIONS TO BE MET AND ONE OR MORE ACTIONS TO BE TAKEN IN RESPONSE
404 EVALUATE A RULE FROM THE PLURALITY OF RULES WITHIN THE RULES ENGINE AT A FREQUENCY OVER A PERIOD OF TIME USING DATA CONTAINED WITHIN EVENT OBJECTS, SO TO DETERMINE WHETHER THE ONE OR MORE CONDITIONS DEFINED WITHIN THE EVALUATED RULE ARE MET
405 INITIATE THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE WHENEVER THE ONE OR MORE CONDITIONS DEFINED BY THAT RULE ARE MET
408 PRIOR TO EVALUATING A RULE, INJECT ONE OR MORE FACTS INTO RULES ENGINE, THE ONE OR MORE FACTS USED TO EVALUATE ONE OR MORE RULES

FIG. 4


501 RECEIVE SECURITY EVENT DATA VIA AN EVENT PIPELINE, SECURITY EVENT DATA DEFINING AN OCCURRENCE OF AN EVENT, WHEREIN EACH TYPE OF SOURCE OF EVENT DATA MAY HAVE ITS OWN PIPELINE

502 CREATE EVENT OBJECTS FROM THE RECEIVED SECURITY EVENT DATA

503 GATHER THE EVENT OBJECTS INTO A RULES ENGINE, THE RULES ENGINE COMPRISING A PLURALITY OF RULES, WHEREIN EACH RULE DEFINES ONE OR MORE CONDITIONS TO BE MET AND ONE OR MORE ACTIONS TO BE TAKEN IN RESPONSE

504 EVALUATE A RULE FROM THE PLURALITY OF RULES WITHIN THE RULES ENGINE AT A FREQUENCY OVER A PERIOD OF TIME USING DATA CONTAINED WITHIN EVENT OBJECTS, SO TO DETERMINE WHETHER THE ONE OR MORE CONDITIONS DEFINED WITHIN THE EVALUATED RULE ARE MET

505 INITIATE THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE WHENEVER THE ONE OR MORE CONDITIONS DEFINED BY THAT RULE ARE MET

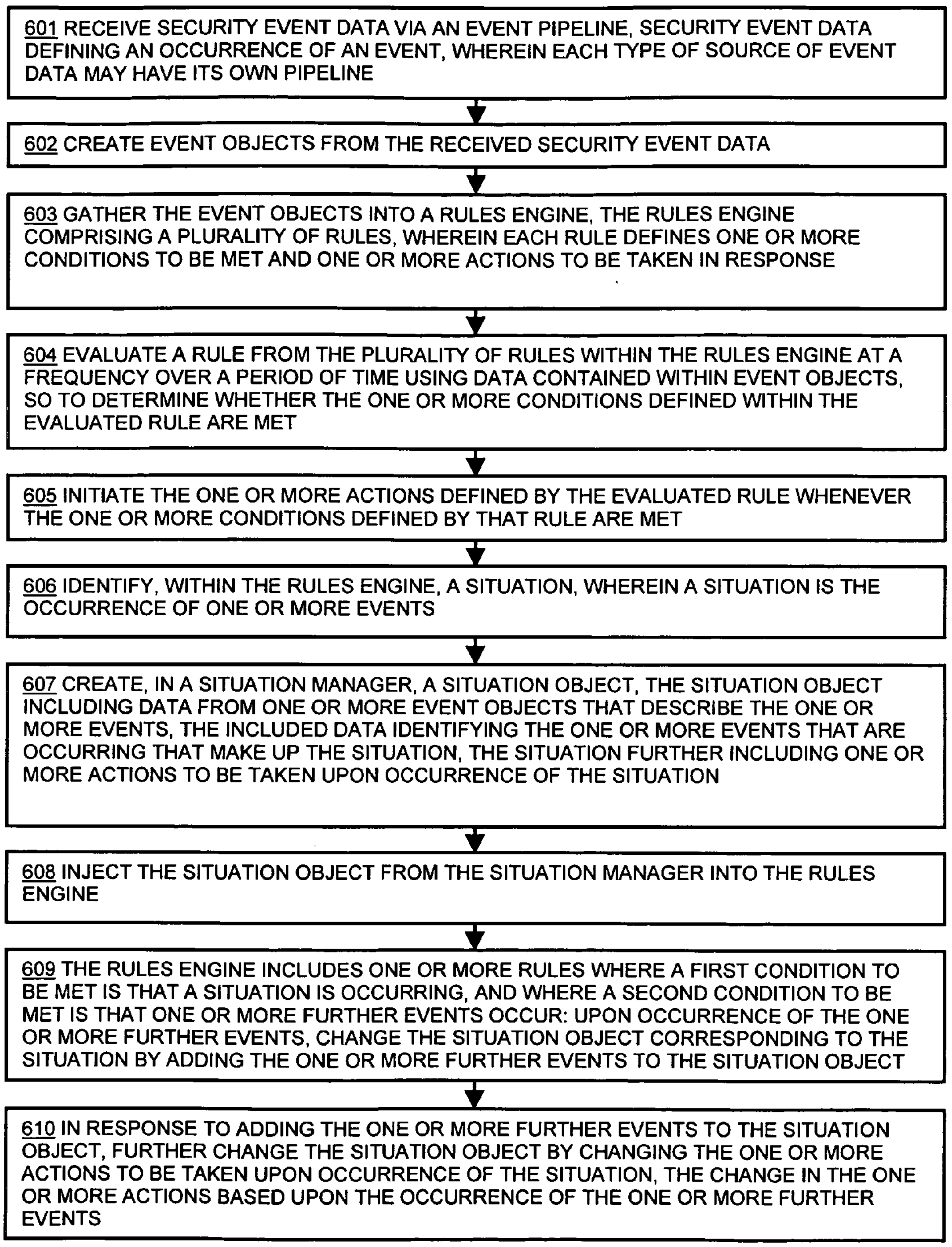
506 ASSIGN WEIGHTS TO A GROUP OF RULES WITHIN THE RULES ENGINE, SUCH THAT WHEN THE ONE OR MORE CONDITIONS OF A RULE IN THE GROUP OF RULES ARE MET, A VALUE OF THE WEIGHT ASSIGNED TO THAT RULE IS ADDED TO AN ACCUMULATOR

507 INCLUDE AN ACCUMULATOR RULE IN THE RULES ENGINE, WHEREIN A CONDITION OF THE ACCUMULATOR RULE IS DEFINED AS THE VALUES IN THE ACCUMULATOR EXCEEDING A GIVEN VALUE, THE ACCUMULATOR RULE DEFINING ONE OR MORE ACTIONS TO OCCUR UPON THE CONDITION BEING MET

508 EVALUATE THE ACCUMULATOR RULE AT A FREQUENCY OVER A PERIOD OF TIME

509 UPON THE CONDITION OF THE ACCUMULATOR RULE BEING MET, TRIGGER THE ONE OR MORE ACTIONS DEFINED IN THE ACCUMULATOR RULE TO OCCUR

FIG. 5



601 RECEIVE SECURITY EVENT DATA VIA AN EVENT PIPELINE, SECURITY EVENT DATA DEFINING AN OCCURRENCE OF AN EVENT, WHEREIN EACH TYPE OF SOURCE OF EVENT DATA MAY HAVE ITS OWN PIPELINE

602 CREATE EVENT OBJECTS FROM THE RECEIVED SECURITY EVENT DATA

603 GATHER THE EVENT OBJECTS INTO A RULES ENGINE, THE RULES ENGINE COMPRISING A PLURALITY OF RULES, WHEREIN EACH RULE DEFINES ONE OR MORE CONDITIONS TO BE MET AND ONE OR MORE ACTIONS TO BE TAKEN IN RESPONSE

604 EVALUATE A RULE FROM THE PLURALITY OF RULES WITHIN THE RULES ENGINE AT A FREQUENCY OVER A PERIOD OF TIME USING DATA CONTAINED WITHIN EVENT OBJECTS, SO TO DETERMINE WHETHER THE ONE OR MORE CONDITIONS DEFINED WITHIN THE EVALUATED RULE ARE MET

605 INITIATE THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE WHENEVER THE ONE OR MORE CONDITIONS DEFINED BY THAT RULE ARE MET

606 IDENTIFY, WITHIN THE RULES ENGINE, A SITUATION, WHEREIN A SITUATION IS THE OCCURRENCE OF ONE OR MORE EVENTS

607 CREATE, IN A SITUATION MANAGER, A SITUATION OBJECT, THE SITUATION OBJECT INCLUDING DATA FROM ONE OR MORE EVENT OBJECTS THAT DESCRIBE THE ONE OR MORE EVENTS, THE INCLUDED DATA IDENTIFYING THE ONE OR MORE EVENTS THAT ARE OCCURRING THAT MAKE UP THE SITUATION, THE SITUATION FURTHER INCLUDING ONE OR MORE ACTIONS TO BE TAKEN UPON OCCURRENCE OF THE SITUATION

608 INJECT THE SITUATION OBJECT FROM THE SITUATION MANAGER INTO THE RULES ENGINE

609 THE RULES ENGINE INCLUDES ONE OR MORE RULES WHERE A FIRST CONDITION TO BE MET IS THAT A SITUATION IS OCCURRING, AND WHERE A SECOND CONDITION TO BE MET IS THAT ONE OR MORE FURTHER EVENTS OCCUR: UPON OCCURRENCE OF THE ONE OR MORE FURTHER EVENTS, CHANGE THE SITUATION OBJECT CORRESPONDING TO THE SITUATION BY ADDING THE ONE OR MORE FURTHER EVENTS TO THE SITUATION OBJECT

610 IN RESPONSE TO ADDING THE ONE OR MORE FURTHER EVENTS TO THE SITUATION OBJECT, FURTHER CHANGE THE SITUATION OBJECT BY CHANGING THE ONE OR MORE ACTIONS TO BE TAKEN UPON OCCURRENCE OF THE SITUATION, THE CHANGE IN THE ONE OR MORE ACTIONS BASED UPON THE OCCURRENCE OF THE ONE OR MORE FURTHER EVENTS

FIG. 6

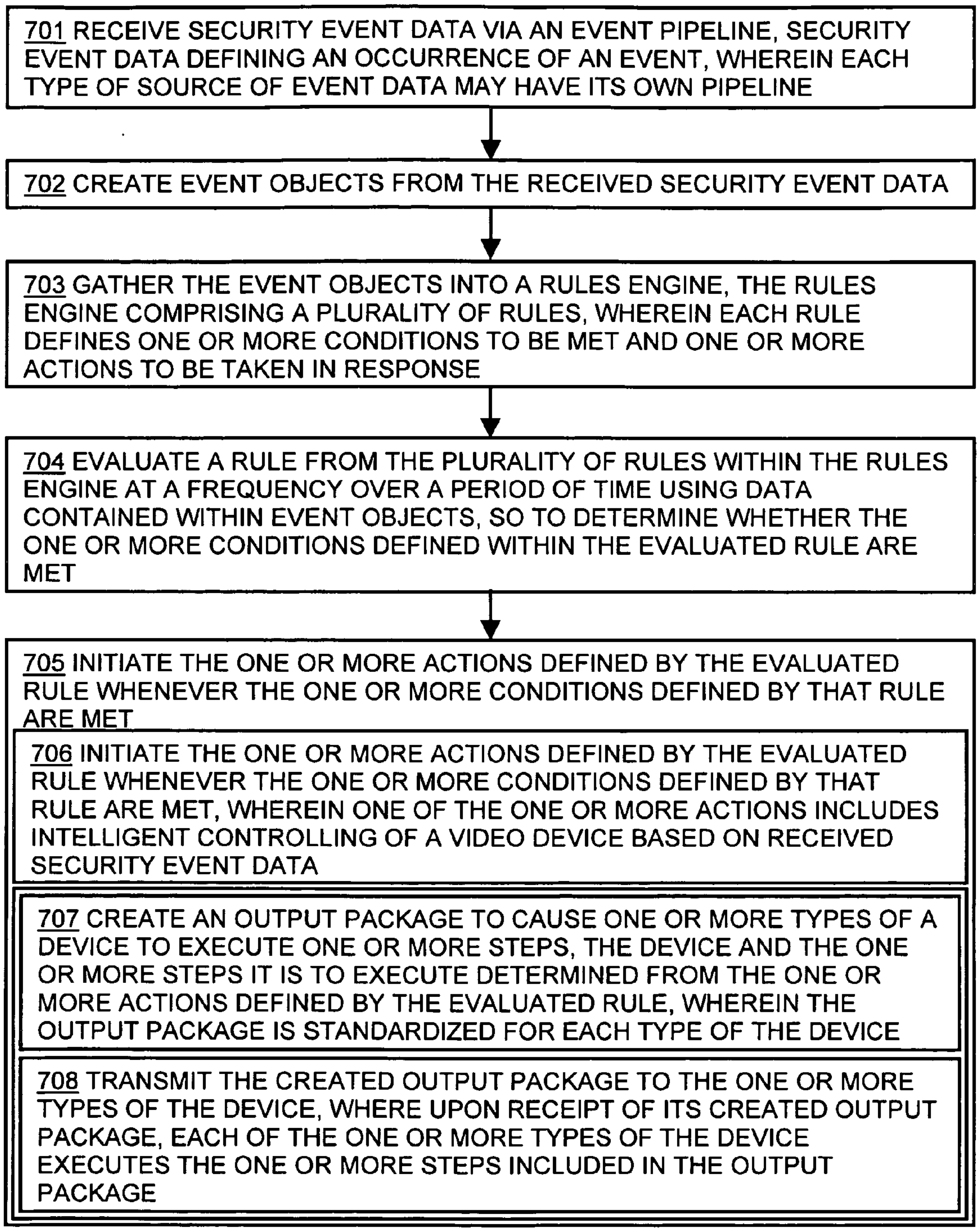

701 RECEIVE SECURITY EVENT DATA VIA AN EVENT PIPELINE, SECURITY EVENT DATA DEFINING AN OCCURRENCE OF AN EVENT, WHEREIN EACH TYPE OF SOURCE OF EVENT DATA MAY HAVE ITS OWN PIPELINE
702 CREATE EVENT OBJECTS FROM THE RECEIVED SECURITY EVENT DATA
703 GATHER THE EVENT OBJECTS INTO A RULES ENGINE, THE RULES ENGINE COMPRISING A PLURALITY OF RULES, WHEREIN EACH RULE DEFINES ONE OR MORE CONDITIONS TO BE MET AND ONE OR MORE ACTIONS TO BE TAKEN IN RESPONSE
704 EVALUATE A RULE FROM THE PLURALITY OF RULES WITHIN THE RULES ENGINE AT A FREQUENCY OVER A PERIOD OF TIME USING DATA CONTAINED WITHIN EVENT OBJECTS, SO TO DETERMINE WHETHER THE ONE OR MORE CONDITIONS DEFINED WITHIN THE EVALUATED RULE ARE MET
705 INITIATE THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE WHENEVER THE ONE OR MORE CONDITIONS DEFINED BY THAT RULE ARE MET
706 INITIATE THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE WHENEVER THE ONE OR MORE CONDITIONS DEFINED BY THAT RULE ARE MET, WHEREIN ONE OF THE ONE OR MORE ACTIONS INCLUDES INTELLIGENT CONTROLLING OF A VIDEO DEVICE BASED ON RECEIVED SECURITY EVENT DATA
707 CREATE AN OUTPUT PACKAGE TO CAUSE ONE OR MORE TYPES OF A DEVICE TO EXECUTE ONE OR MORE STEPS, THE DEVICE AND THE ONE OR MORE STEPS IT IS TO EXECUTE DETERMINED FROM THE ONE OR MORE ACTIONS DEFINED BY THE EVALUATED RULE, WHEREIN THE OUTPUT PACKAGE IS STANDARDIZED FOR EACH TYPE OF THE DEVICE
708 TRANSMIT THE CREATED OUTPUT PACKAGE TO THE ONE OR MORE TYPES OF THE DEVICE, WHERE UPON RECEIPT OF ITS CREATED OUTPUT PACKAGE, EACH OF THE ONE OR MORE TYPES OF THE DEVICE EXECUTES THE ONE OR MORE STEPS INCLUDED IN THE OUTPUT PACKAGE

FIG. 7

EVENT CAPTURE, CROSS DEVICE EVENT CORRELATION, AND RESPONSIVE ACTIONS

PRIORITY TO EARLIER FILED PROVISIONAL PATENT APPLICATIONS

[0001] This application claims the benefit of the filing date of the following earlier filed co-pending U.S. Provisional Patent Applications: Ser. No. 60/939,517, entitled "METHOD AND APPARATUS FOR EVENT CAPTURE, CROSS DEVICE EVENT CORRELATION, AND RESPONSIVE ACTIONS", having Attorney Docket No. VID07-02p, filed on May 22, 2007; Ser. No. 60/939,503, entitled "METHOD AND APPARATUS FOR TRACKING PEOPLE AND OBJECTS USING MULTIPLE LIVE AND RECORDED SURVEILLANCE CAMERA VIDEO FEEDS", having Attorney Docket VID07-01p, also filed on May 22, 2007; Ser. No. 60/939,521 entitled "METHOD AND APPARATUS FOR INTELLIGENT VIDEO TOURS", having Attorney Docket VID07-03p, also filed on May 22, 2007; and Ser. No. 60/939,528, entitled "METHOD AND APPARATUS FOR OPTIMAL ROUTING OF AUDIO & VIDEO SIGNALS THROUGH HETEROGENEOUS NETWORKS", having Attorney Docket VID07-04p, also filed on May 22, 2007. These all share co-inventorship with the present application. The entire teachings and contents of these Provisional Patent Applications are hereby incorporated by reference herein in their entirety.

BACKGROUND

[0002] Securing an area from threats, both internal to the facility and external to it, has long been a desire of those who have something of value that may be desirable to others located within the area. Early conventional security mechanisms involved placing guards at points of entry to an area to be secured. Locks of differing strengths may also have been deployed on points of entry to the area, if the area was surrounded by walls or gates. With increasing sophistication in technology, guards were also deployed within areas (i.e., inside buildings) to patrol, and badge readers and other electronic entry devices were used to supplement locks.

[0003] Guards, however, are expensive to pay, and also capable of error. Particularly for larger facilities/facilities with many possible points of entry, it may not be possible to hire enough guards to "watch" everything going on. Thus, automated devices such as security cameras have also been added to the mix of security measures. The addition of security cameras meant that security personnel could "see" all of the interesting areas of the facility (i.e., points of entry, locations where things of valued were stored, etc.). However, an increase in the number of cameras placed into a facility made it harder to watch all of the cameras simultaneously without hiring more personnel to watch the cameras. Doing so would remove the primary monetary advantage of using cameras, that is, not having to employ more personnel.

[0004] One conventional solution to the "too many cameras" problem is for security personnel to pay particular attention to a subset of all the available cameras. In this scenario, the question becomes which cameras to watch, and which cameras to ignore. Typically, the areas of high value and high risk (e.g., a vault, primary entry and exit points such as the doors going in/out of a building, etc.) are given primary focus, and other areas of lesser value and/or risk are given secondary focus. These measures have served as an adequate defense against "low tech" threats (i.e., breaking and entering by common criminals).

SUMMARY

[0005] Conventional security mechanisms such as those explained above suffer from a variety of deficiencies. One such deficiency is that, with an increase in the number of more active and more dangerous threats (i.e., terrorist activities, etc.), the cost of "watching" to see if something bad happens versus being more proactive has become very high. That is, conventional security systems allowed personnel to discover and deal with problems as they happened. Now it is of far more value to be able to have some amount of warning before a problem develops fully. Thus, instead of using video for watching high value/risk areas, and using video for forensics after the fact, embodiments disclosed herein provide for security mechanisms that provide situational awareness through rules-based processing of events.

[0006] Situational awareness includes an ability to take into account a variety of events that occur in and around a protected area or facility, including simultaneous or near-simultaneous events. For example, situational awareness may include, in some situations, being aware that: multiple people have gone through a door on a single badge swipe, through a combination of access control mechanism and video analytics); multiple doors are being tried, through door jiggle sensors; or perhaps something in a protected facility/area has gone out of an allowable range (i.e., level of fluid in a tank, acidity of a substance, etc.). By monitoring a number of events in and around a protected area/facility, and detecting unusual and/or unexpected events, embodiments bring such events to the attention of security personnel. Security personnel are then able to view areas of potential problems via traditional means (i.e., video cameras and network digital video recorders (NDVRs) versus waiting to be notified that a problem is in fact currently occurring.

[0007] Situational awareness as provided by embodiments described herein may include a number of phases. In a first phase, responses to individual alerts, or "singletons", occur. A singleton is a single event that causes one or more actions to take place, such as flipping cameras, sounding alarms, etc. Combining singletons results in more complicated scenarios that require a rules engines and correlation between events, as described herein. A second phase of situational awareness may be realizing that there is much more information to be mined by combining the results of many different types of events and correlating them. For example, it is not a particularly interesting event to know that someone has just logged into a computer located within a building, nor is it a particularly interesting event to know that someone has left the building. However, it is very interesting to know that the same person that just logged into the computer located within the building is also reported to not be in the building.

[0008] By combining fairly simple unrelated events into a model of what is "normal" (i.e., typical or expected behavior) and what is not normal (i.e., atypical or unexpected behavior), embodiments focus security personnel on unusual events, instead of the security personnel waiting for an external event (i.e., a phone call) to occur to warn that a problem is in progress. Continuing the above example, it is far more advantageous to know that a potential problem exists, based on the known information, then to know after the fact that the company's computer system was hacked into via the CEO's computer by a person at a remote location.

[0009] Embodiments of the invention thus include a rules engine used for correlating the events that come in from various statuses/alerting systems. The rules engine does not attempt to use any sort of Artificial Intelligence (AI) to learn what is normal, or to learn what is not normal. Rather, embodiments require that creators of rules, such as the programmers (system integrators) of the rules engine, determine what is not "normal", and create rules to detect those "abnormal" situations. Note that defining anything that is "abnormal" or otherwise unusual is not required. Rather, serious thought about the normal course of events is required. Those thoughts may then be embodied as rules within the rules engine by programming them into the engine, and then looking for patterns of actual events that are counter to those rules. Using the example from above, the pattern of a person being noted to have left a building but then logging in to a computer that is supposed to be located within the building, is noticed.

[0010] Use of a rules engine, and rules-based processing of event data, is in contrast to conventional systems that typically employ event-based processing. That is, conventional systems include thousands (or more) of lines of code, in an attempt to cover every possible event scenario that may be encountered. Every scenario that is thought of must have code to deal with just that particular scenario. The more complicated the scenario is, the more complex the code for that scenario also is. Thus, because the entirety of the code must be evaluated every time an event occurs, event-based processing results in a great deal of processing overhead being consumed. Further, conventional event-based processing systems may simply not include all possible scenarios. Rules-based processing through a rules engine, as described herein, is a more efficient way of not only accounting for, and processing, a variety of different scenarios, including increasingly complicated scenarios, but also is easier to formulate and less likely to leave scenarios out.

[0011] Determining what is "abnormal" or "unusual" may be difficult for security and surveillance personnel. Programming rules that look for specific "bad" events are the first line of defense. However, even rules that look for too many unusual things over a specified period of time may also be very valuable. In the example of someone logging in while out of the building, it was immediately clear that this was a pattern of a potential issue. There is another class or series of events that may be neither good nor significantly bad when treated independently, and that may not be correlated as a rule. Such a class or series of events that are an indicator that something is a bit amiss. For example, a light out in a parking lot, a door camera reported broken, and a heat sensor running higher than expected, are individual events. None of these singleton events represents an imminent threat, but in combination, with perhaps many other trivial rules that represent exceptions to the "normal", it is possible to weight and score these exceptions. Once these expectations reach a predefined threshold, it may be deduced that something might be wrong. This sort of early warning, in some scenarios, might be significant enough to cause a medium scale warning, or it might result in something as simple as an alert message (page, SIP call, etc.) being sent to a central area from which the security personnel may respond or otherwise act.

[0012] Embodiments of the invention thus use events coming into the system and the rules that correlate them to provide situational awareness. The actions that are taken as a result and the policies that dictate how this is all acted upon may be varied from customer to customer. Thus, embodiments provide a broad and flexible security system that may easily be modified to meet a particular customer's needs.

[0013] More particularly, in an embodiment, there is provided a method of correlating security event data according to one or more rules to initiate one or more actions. The method includes receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline, and creating event objects from the received security event data. The method also includes gathering the event objects into a rules engine, the rules engine may include a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response. The method also includes evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met, and initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

[0014] In a related embodiment, creating may include pre-processing the received security event data to create event objects that include the received security event data in a canonical form. In a further related embodiment, pre-processing may include supplementing an event object with related data not acquired from the source of the received security event data, by using an item of data within the received security event data to look up the related data, the related data used to set one or more attributes of the event object.

[0015] In another related embodiment, the method may include prior to evaluating a rule, injecting one or more facts into rules engine, the one or more facts used to evaluate one or more rules. In yet another related embodiment, the method may include assigning weights to a group of rules within the rules engine, such that when the one or more conditions of a rule in the group of rules are met, a value of the weight assigned to that rule is added to an accumulator; including an accumulator rule in the rules engine, wherein a condition of the accumulator rule is defined as the values in the accumulator exceeding a given value, the accumulator rule defining one or more actions to occur upon the condition being met; evaluating the accumulator rule at a frequency over a period of time; and upon the condition of the accumulator rule being met, triggering the one or more actions defined in the accumulator rule to occur.

[0016] In still yet another related embodiment, the method may include identifying, within the rules engine, a situation, wherein a situation is the occurrence of one or more events; creating, in a situation manager, a situation object, the situation object including data from one or more event objects that describe the one or more events, the included data identifying the one or more events that are occurring that make up the situation, the situation further including one or more actions to be taken upon occurrence of the situation; and injecting the situation object from the situation manager into the rules engine. In a further related embodiment, the rules engine may include one or more rules where a first condition to be met is that a situation is occurring, and where a second condition to be met is that one or more further events occur, and the method may include, upon occurrence of the one or more further events, changing the situation object corresponding to the situation by adding the one or more further events to the situation object. In another further related embodiment, the method may include, in response to adding the one or more further events to the situation object, further changing the situation object by changing the one or more actions to be taken upon occurrence of the situation, the change in the one or more actions based upon the occurrence of the one or more further events.

[0017] In yet another related embodiment, initiating may include initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met, wherein one of the one or more actions includes intelligent controlling of a video device based on received security event data. In still another related embodiment, initiating may include creating an output package to cause one or more types of a device to execute one or more steps, the device and the one or more steps it is to execute determined from the one or more actions defined by the evaluated rule, wherein the output package is standardized for each type of the device; and transmitting the created output package to the one or more types of the device, where upon receipt of its created output package, each of the one or more types of the device executes the one or more steps included in the output package.

[0018] In another embodiment, there is provided a computer program product, stored on computer readable medium, for correlating security event data according to one or more rules to initiate one or more actions. The computer program product includes computer program code for receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline, and computer program code for creating event objects from the received security event data. The computer program product also includes computer program code for gathering the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response, computer program code for evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met, and computer program code for initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

[0019] In another embodiment, there is provided a computer system. The computer system includes a memory; a processor; a network interface; and an interconnection mechanism coupling the memory, the processor, and the network interface, allowing communication there between. The memory is encoded with an event and rules correlating application, that when executed in the processor, provides a an event and rules correlating process that correlates security event data according to one or more rules to initiate one or more actions. The event and rules correlating process causes the computer system to perform the operations of: receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline; creating event objects from the received security event data; gathering the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response; evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met; and initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

[0020] Other arrangements of embodiments of the invention that are disclosed herein include software programs to perform the method embodiment steps and operations summarized above and disclosed in detail below. More particularly, a computer program product is one embodiment that has a computer-readable medium including computer program logic encoded thereon that when performed in a computerized device provides associated operations providing client management of download sequence of orchestrated content as explained herein. The computer program logic, when executed on at least one processor with a computing system, causes the processor to perform the operations (e.g., the methods) indicated herein as embodiments of the invention. Such arrangements of the invention are typically provided as software, code and/or other data structures arranged or encoded on a computer readable medium such as but not limited to an optical medium (e.g., CD-ROM, DVD-ROM, etc.), floppy or hard disk, a so-called “flash” (i.e., solid state) memory medium, or other physical medium, such as but not limited to firmware or microcode in one or more ROM or RAM or PROM chips, or as an Application Specific Integrated Circuit (ASIC), or as downloadable software images in one or more modules, shared libraries, etc. The software or firmware or other such configurations can be installed onto a computerized device to cause one or more processors in the computerized device to perform the techniques explained herein as embodiments of the invention. Software processes that operate in a collection of computerized devices, such as in a group of data communications devices or other entities may also provide the system of the invention. The system of the invention may be distributed between many software processes on several data communications devices, or all processes may run on a small set of dedicated computers, or on one computer alone.

[0021] It is to be understood that embodiments of the invention may be embodied strictly as a software program, as software and hardware, or as hardware and/or circuitry alone. The features disclosed and explained herein may be employed in computerized devices and software systems for such devices such as those manufactured by VidSys, Inc., of Vienna, Va.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] The foregoing will be apparent from the following description of particular embodiments disclosed herein, as illustrated in the accompanying drawings in which like reference characters refer to the same parts throughout the different views. The drawings are not necessarily to scale, emphasis instead being placed upon illustrating the principles disclosed herein.

[0023] FIG. **1** shows a high-level block diagram of a computer system according to one embodiment disclosed herein.

[0024] FIGS. **2**A-**2**C illustrate examples of system structures upon which a procedure performed by the system of FIG. **1** may operate.

[0025] FIGS. **3**A-**3**E illustrate example code of various components of embodiments described by FIGS. **4**-**7**.

[0026] FIG. **4** illustrates a flowchart of a procedure performed by the system of FIG. **1** when correlating events and rules so as to initiate actions related to security procedures.

[0027] FIG. **5** illustrates a flowchart of a procedure performed by the system of FIG. **1** when assigning weights to various events such that, as weights accumulate, representing a group of events occurring in a period of time, particular actions are taken.

[0028] FIG. **6** illustrates a flowchart of a procedure performed by the system of FIG. **1** when addressing situations and providing a situation object back to the rules engine so that the object is able to be modified over time as the situation changes.

[0029] FIG. **7** illustrates a flowchart of a procedure performed by the system of FIG. **1** when initiating actions through use of commands sent to devices.

DETAILED DESCRIPTION

[0030] Generally, disclosed embodiments provide methods and apparatus for correlating events to rules and in response, taking one or more actions. Data pertaining to events is continually received over time from a variety of sources, such as video cameras, badge readers, sensors, RFID tags, and so on. This data is collected and normalized into a standard format for processing. Data may be deemed unimportant and thus left out, or otherwise filtered out for any number of reasons. The data is processed into structures that may be supplemented with additional related data. The structures are then loaded into a rules engine that contains a plurality of rules defining different event scenarios (i.e., singleton events or combinations of events) that may occur, as well as one or more actions to be taken in response to the occurrence of the scenario(s) defined in the rule(s). The rules are continually evaluated over a period of time, and whenever the condition (s) for a rule is(are) met, the action(s) associated with that rule is(are) taken.

[0031] More particularly, FIG. **1** is a block diagram illustrating example architecture of a computer system **110** that executes, runs, interprets, operates or otherwise performs an event and rules correlation application **140-1** and an event and rules correlation process **140-2** suitable for use in explaining example configurations disclosed herein. The computer system **110** may be any type of computerized device such as a personal computer, workstation, portable computing device, console, laptop, network terminal or the like. As shown in this example, the computer system **110** includes an interconnection mechanism **111** such as a data bus or other circuitry that couples a memory system **112**, a processor **113**, an input/output interface **114**, and a communications interface **115**. An input device **116** (e.g., one or more user controlled devices such as a keyboard, mouse, touchpad, trackball, etc.) couples to the processor **113** through the I/O interface **114** and enables a user **108** such as a web page developer to provide input commands and generally control a graphical user interface **160** shown on a display **130**. The communications interface **115** enables the computer system **110** to communicate with other devices (e.g., other computers) on a network (not shown in FIG. **1**).

[0032] The memory system **112** is any type of computer readable medium and in this example is encoded with an event and rules correlation application **140-1**. The event and rules correlation application **140-1** may be embodied as software code such as data and/or logic instructions (e.g., code stored in the memory or on another computer readable medium such as a removable disk) that supports processing functionality according to different embodiments described herein. During operation of the computer system **110**, the processor **113** accesses the memory system **112** via the interconnection mechanism **111** in order to launch, run, execute, interpret or otherwise perform the logic instructions of the event and rules correlation application **140-1**. Execution of the event and rules correlation application **140-1** in this manner produces processing functionality in an event and rules correlation process **140-2**. In other words, the event and rules correlation process **140-2** represents one or more portions or runtime instances of the event and rules correlation application **140-1** performing or executing within or upon the processor **113** in the computer system **110** at runtime.

[0033] It is noted that example configurations disclosed herein include the event and rules correlation application **140-1** itself including the event and rules correlation process **140-2** (i.e., in the form of un-executed or non-performing logic instructions and/or data). The event and rules correlation application **140-1** may be stored on a computer readable medium (such as a floppy disk), hard disk, electronic, magnetic, optical or other computer readable medium. The event and rules correlation application **140-1** may also be stored in a memory system **112** such as in firmware, read only memory (ROM), or, as in this example, as executable code in, for example, Random Access Memory (RAM). In addition to these embodiments, it should also be noted that other embodiments herein include the execution of the event and rules correlation application **140-1** in the processor **113** as the event and rules correlation process **140-2**. Those skilled in the art will understand that the computer system **110** may include other processes and/or software and hardware components, such as an operating system not shown in this example.

[0034] The display **130** need not be coupled directly to computer system **110**. For example, the event and rules correlation application **140-1** may be executed on a remotely accessible computerized device via the network interface **115**. In this instance, the graphical user interface **160** may be displayed locally to a user of the remote computer and execution of the processing herein may be client-server based. In some embodiments, the graphical user interface **160** may be a customer interface through which a user, such as the user **108**, is able to learn various information and take various actions. The amount of features, and control thereof, may depend on a user level, such that a basic user has access to only a certain amount of features, while an administrator may have access to all available features. Key features of the graphical user interface **160** may include the ability to locate and activate cameras easily from a searchable directory; the ability to locate and activate cameras easily from a map representation; the ability to control Cameras and NDVRs to effect Pan-Tilt-Zoom (PTZ), Iris, Focus, Playback, etc.; the ability to list and get details on incoming events; the ability to locate people and use collaboration tools to interact with them; the ability to locate shared folders and reference and active files; the ability to control video walls; the ability to initiate video tours, which are described in detail in co-pending U.S. patent application Ser. No. ______, entitled "INTELLIGENT VIDEO TOURS" and given Attorney Docket VID08-04; and administrative features such as but not limited to create/modify/list/delete users, roles, tours, devices, etc.

[0035] FIGS. **2A-2C** illustrate examples of system structures upon which the event and rules correlating process

140-2 may operate. FIGS. **3**A-**3**E include example code of various components of embodiments described below, and are referenced throughout the discussion of FIGS. **4-7**. FIGS. **4-7** are flowcharts of various embodiments of the event and rules correlating process **140-2**. The rectangular elements are herein denoted "processing blocks" and represent computer software instructions or groups of instructions. Alternatively, the processing blocks represent steps performed by functionally equivalent circuits such as a digital signal processor circuit or an application specific integrated circuit (ASIC). The flowcharts do not depict the syntax of any particular programming language. Rather, the flowcharts illustrate the functional information one of ordinary skill in the art requires to fabricate circuits or to generate computer software to perform the processing required in accordance with the present invention. It should be noted that many routine program elements, such as initialization of loops and variables and the use of temporary variables are not shown. It will be appreciated by those of ordinary skill in the art that unless otherwise indicated herein, the particular sequence of steps described is illustrative only and may be varied without departing from the spirit of the invention. Thus, unless otherwise stated, the steps described below are unordered, meaning that, when possible, the steps may be performed in any convenient or desirable order.

[0036] FIG. **4** illustrates an embodiment of the event and rules correlating application **140-1**, executing as the event and rules correlating process **140-2**, to correlate security event data according to one or more rules to initiate one or more actions. The event and rules correlating process **140-2** first receives security event data via an event pipeline, step **401**. Security event data defines an occurrence of an event. An event is simply something that takes place and is, in some manner, recorded or otherwise noticed by a device to which the event and rules correlating process **140-2** has access. Thus, for example, when a person uses a security badge at a badge reader at the door to a particular room, to enter that room, the security event data may include the id number of the badge and the location of the badge reader. The event would be using the security badge. A further event may be the opening and closing of the door, if there is a sensor on the door capable of noticing that event. The event data for such a thing would be a reading from the sensor, first indicating the opening of the door, and then indicating the shutting of the door.

[0037] The number of devices that may serve as sources of event data for the event and rules correlating process **140-2** is large. The event and rules correlating process **140-2** is able to interface with any number of security-related devices (video cameras, sensors, lighting systems, badge readers, electronic locks, etc.). Those devices differ, not only in function, but the format of the data they provide and how they are controlled. Thus, the event and rules correlating process **140-2** must be flexible enough to deal with hundreds, perhaps thousands, of different data formats. The event and rules correlating process **140-2** achieves this flexibility through use of an input pipeline. An input pipeline as used by the event and rules correlating process **140-2** to receive event data, such as the example input pipeline shown in FIG. **2**A, may be thought of as an externally defined customizable stack **210** of various layers. Each layer in the stack **210** is relatively simple and is configured to do a single or small set of functions. The layers may include a transport layer **212**, a parser layer **214**, and a normalize layer **216**. The layers may also include optional log layers **218** and **220**, and an optional filtering layer **219**. Supporting any specific type of source of event data (i.e., device such as a camera, badge reader, etc.) is simply a matter of constructing a stack that includes appropriate layers (i.e., layers that are able to communicate with the device and format the event data into a form that is suitable for further processing by the event and rules correlating process **140-2** in the rules engine, as is discussed below). Example code of an example input pipeline including various layers in its stack is shown in FIG. **3**A.

[0038] Note that, in some embodiments, each type of source of event data has its own pipeline. That is, cameras may have their own pipeline (not necessarily for video but rather, for example, for analytic data), while sensors may have their own pipeline, badge readers may have their own pipeline, and so on. A type of source need not be simply the different devices that deliver event data to the event and rules correlating process **140-2**, but may also include variations within those devices. Thus, a first badge reader manufactured by one company may have a different data format and control mechanism than a second badge reader manufactured by another company. The event and rules correlating process **140-2** thus has a pipeline for the first badge reader (and any/all other badge readers of the same type) and a separate pipeline for the second badge reader (and any/all other badge readers of the same type). Alternatively, in some embodiments, different types of sources of event data may share a pipeline as is appropriate.

[0039] In the transport layer **212**, the event and rules correlating process **140-2**, through the pipeline on its external side, communicates with the source of the event data (i.e., a device) using the language and transport (i.e. protocol) that is defined by the source. Thus, examples of transports may include but are not limited to HTTP, File, SNMP, SMTP, SOAP, TCP, and the like. The transport layer **212** takes event data in and provides it as raw event data to the parser layer **214**. The event data, after coming out of a transport layer, is essentially an unpackaged byte stream.

[0040] The parser layer **214** may include rules for pulling fields from XML, or running regular expressions, or busting apart ASCII strings into a particular form. The parser layer **214** takes the raw event data and its attributes, as delivered by the transport layer **212**, and converts that raw event data into a structure that may be operated on by the remaining layers of the pipeline. Note that the parser layer **214** does not understand or know, or make judgments on, the raw event data received by the event and rules correlating process **140-2**. Rather, the parser layer **214** does a format specific conversion from the incoming byte stream (i.e., raw event data) to an implementation-specific format for the remainder of the pipeline. For example, if the incoming raw event data were an HTTP GET request with name value arguments in the query string, the parser layer **214** might convert those to the internal format, as perhaps an XML document. Alternatively, if the internal format was a name value string, the parser layer **214** may pass the ones provided along, but add to it the transport specific fields, original URL, arrival time, host, etc.

[0041] The event and rules correlating process **140-2** next creates event objects from the received security event data, step **402**. An event object is a data structure that may, and typically does, include one or more attributes. That is, an event object may be thought of a fact, with other related information included as well. In general, the more facts the event and rules correlating process **140-2** has, the better. As the raw security event data received by the event and rules correlating process **140-2** may be in any one of a number of formats, for the event and rules correlating process **140-2** to easily use that data, the event and rules correlating process **140-2** must first place it into a standardized format. That is, the event and rules correlating process **140-2** may create an event object by pre-processing the received security event data to create event objects that include the received security event data in a canonical form, step **406**. The pre-processing may include transforming, combining, and/or deleting fields or even events, among other actions. The event and rules correlating process **140-2** performs this pre-processing (and thus the creation of the event object) in at least the normalizer layer **216**.

[0042] The fact inside the event object, along with attributes, essentially defines what the event was. For example, using the badge strike example discussed above, the event object created by the event and rules correlating process **140-2** for that event data would include the badge id. The event object may also include related attributes, such as information that identifies the location of the badge reader, the owner of the badge with that badge id, and so on. The related attributes do not come from the source of the event data, but rather must be acquired from another source or sources. Thus, the event and rules correlating process **140-2** may supplement an event object with related data not acquired from the source of the received security event data, by using an item of data within the received security event data to look up the related data, the related data used to set one or more attributes of the event object, step **407**. Thus, the event and rules correlating process **140-2** may use the badge id to perform a lookup in a database to determine who the owner of the badge id is, and may place this information in an attribute of the event object. In some embodiments, the event and rules correlating process **140-2** may use a first attribute to acquire other attributes. That is, the event and rules correlating process **140-2** may use the name of the owner of the badge with the received badge id to lookup that person's permissions for entering various restricted areas of a protected facility. The event and rules correlating process **140-2** may also set those permissions as attributes within the event object, or in a separate event object.

[0043] In some embodiments, the event and rules correlating process **140-2** may use the optional filtering layer **219** to remove unnecessary data from an event object. For example, data from a sensor reporting its current value may arrive five times in a short duration of time (say, 10 seconds). The event and rules correlating process **140-2** may use the filtering layer **219** to determine that, because the value reported by the sensor is within an accepted range of error, the sensor has essentially reported the same value five times, and the event object need not contain that value five times over; once will suffice. Thus, the event and rules correlating process **140-2** may be programmed to store or otherwise note a current value of data for later comparisons. Alternatively, the event and rules correlating process **140-2** may use the filtering layer **219** to only remove certain pieces of data. For example, the same sensor may, over a ten minute period of time, occasionally report the same value, but viewed over the ten minutes, the value reported by the sensor shows a steady increase. The event and rules correlating process **140-2** may use the filtering layer **219** to remove only unnecessary repeats of a value, to thus make sure the event object contains the values showing that increase over time. In such a situation, the event and rules correlating process **140-2** is thus configured to not only maintain a current value, but also a number of recent values as well.

[0044] Note that an input pipeline as used by the event and rules correlating process **140-2** is generic, such that it may be extended to support more devices by adding support to the event and rules correlating process **140-2** for transports, parsers, and transforms for those devices. When sources of event data (i.e., devices) that the event and rules correlating process **140-2** communicates with via the pipeline have unusual characteristics, custom transports and/or customer parsers may have to be developed. Further note that event data in a pipeline may be logged at any time, by using either of the two optional log layers **218** and/or **220**.

[0045] The event and rules correlating process **140-2** then gathers the event objects into a rules engine, step **403**. The rules engine comprises a plurality of rules. Each rule defines one or more conditions to be met and one or more actions to be taken in response. Example rules are shown in FIGS. **3**B-**3**E. Rules are loaded from an external file and are thus not hard coded. Rules may be created by users, in some embodiments as XML files, or via a plug-in that supports a semi-natural language way to create syntax for proper rules, or via a graphical user interface. Some rules represent singleton events, that is, only one condition need occur. For example, a rule may be defined to send an alert message when a device including an RFID tag crosses a threshold point. Other rules may represent multiple events of varying complexity. Thus, such a rule may require that a device including an RFID tag crosses a threshold point and that prior to this, a badge was used to open the door at the threshold point. Patterns may important in some rules, while order may be important in others. The rules are able to define what the criteria are (i.e., the conditions) for when an action is to be initiated, and thus is easily able to cover a variety of constraints and scenarios.

[0046] The rules engine is not artificially intelligent, but rather is able to be programmed with rules to meet specific needs. Thus, it is possible to write rules to expect specific things or not expect specific things, but it is not possible to write a rule that says to 'watch for something unusual'. To watch for something unusual is really to define everything that is usual and then declare what the unusual patterns are. Since the device input drivers convert data from different devices into a canonical form, rules may be written that look for conditions across different devices.

[0047] The event and rules correlating process **140-2** evaluates a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met, step **404**. That is, the event and rules correlating process **140-2** causes the rules engine to look at the conditions defined by a rule and see if there are event objects, and in some cases other facts, containing data (i.e., describing events) that satisfy all of those conditions. Using the example rule shown in FIG. **3**B, the event and rules correlating process **140-2** looks for one or more event objects identifying location(s) to which access is restricted, and for one or more event objects identifying a person in such a restricted location even though that person does not have clearance to be in the restricted location. If the event and rules correlating process **140-2** finds such event objects (and the current time, determined as discussed below, is after hours), then the event and rules correlating process **140-2** will initiate the actions defined by the example rule, as discussed further below.

[0048] In some embodiments, prior to evaluating a rule, the event and rules correlating process **140-2** may inject one or more facts into rules engine, step **406**. The one or more facts may be used to evaluate one or more rules. For example, the example rule shown in FIG. **3**B requires knowledge of current time of day. The event and rules correlating process **140-2** is able to acquire that time from a source (a database or a clock), and then inject it into the rules engine. When the event and rules correlating process **140-2** causes the rules engine evaluates the rule shown in FIG. **3**B, the rules engine is able to find out the time, and evaluate the rule appropriately.

[0049] There are cases where rules located in the rules engine are not sufficient for the event and rules correlating process **140-2** to detect what needs to be detected. As real world events happen, a user might want to go back and use the event and rules correlating process **140-2** to run a set of rules against captured device events. In other words, the user may wish to user the event and rules correlating process **140-2** to perform forensics analysis of event data. To provide such functionality, in some embodiments the event and rules correlating process **140-2** records and time stamps all event data and/or event objects, so they may be re-evaluated against a new set of rules at a later time. The event and rules correlating process **140-2** may use a database aggregator to perform these actions. The data aggregator buffers out of band of the input pipeline events that are targeted for the database. The data aggregator performs the buffering for a short period of time (e.g., seconds) and then writes the event data to the database as a collective blob. When in forensics mode, the event and rules correlating process **140-2** pulls these blobs from the database in order, unpacks the individual events within the blobs, and serially (i.e., in order) inserts those events into the input pipeline stages, to be re-processed and re-entered into the rules engine for the forensics analysis. These components, as well as the input pipeline, the rules engine, and other components of a system upon which the event and rules correlating process **140-2** may operate, are shown in example system **200** of FIG. **2**B.

[0050] Finally, the event and rules correlating process **140-2** initiates the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met, step **405**. An action may be any command that an associated device is capable of carrying out. As the event and rules correlating process **140-2** may be associated with any number of security-related devices (video cameras, sensors of all types, microphones and other audio recording devices, NDVRs, lighting systems, badge readers, electronic locks, etc.) and is able to interface via the network interface **115** (shown in FIG. **1**) to other network-enabled devices and (computer systems, etc.) and networks themselves, the number of possible actions, as well as the depth of possible actions, is quite large. More particularly, actions may include, for example, sending messages using any number of transmission media (i.e., e-mail, phone call, facsimile, page, text message, video message, printout to a printing device, pop-up box or other alert mechanism displayed on a display device, etc.), and sending control commands to any associated devices, as is described in greater detail with regards to FIG. **7** below.

[0051] In FIG. **5**, the event and rules correlating process **140-2** addresses scenarios in which a number of small events, each not troubling on its own, is recognized as being a part of a larger event, such that when combined, the number of small events may represent a troubling problem. Thus, the event and rules correlating process **140-2** is capable of dealing with events that are not very interesting singly, or even in simple patterns, but are interesting if, for example, too many of them happen in too short a time period.

[0052] The event and rules correlating process **140-2** first receives security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data has its own pipeline, step **501**. Then the event and rules correlating process **140-2** creates event objects from the received security event data, step **502**. The event and rules correlating process **140-2** gathers the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response, step **503**.

[0053] The event and rules correlating process **140-2** may then assign weights to a group of rules within the rules engine, step **504**. Thus, when the one or more conditions of a rule in the group of rules are met, a value of the weight assigned to that rule is added to an accumulator. Any number of rules may be included in the group of rules. The group of rules represent a number of events that, when considered separately, are of little importance, but when they occur simultaneously or otherwise near each other, for example, may be of much greater importance. For example, an event such as a door handle to a building being jiggled is, by itself, a scenario perhaps worthy of little attention. The hand of a person attempting to enter that building may have simply slipped off the handle, or the handle might be slightly stuck, causing a person to jiggle it to get the door to open. However, when, in a short period of time, handles on multiple doors, all at entrance points to a building, are jiggled, and then a number of windows on the ground floor of the building are jiggled as well, this may well represent a scenario of importance.

[0054] The event and rules correlating process **140-2** may assign weights in any number of ways, and to any grouping of rules (and thus grouping of events). For example, the event and rules correlating process **140-2** may assign the jiggling of a door handle a lesser weight than the breaking of a window. Further, weights may be based on duration as well as time of day. That is, the event and rules correlating process **140-2** may assign weights such that, the same door handle being jiggled five times in a week is assigned a lesser weight than the same door handle being jiggled five times in a minute. Further, the event and rules correlating process **140-2** may assign weights such that, a door handle being jiggled at noon, while the building is full of people, has a lesser weight than a door handle being jiggled at midnight, after the building has been unoccupied for hours.

[0055] Note that the weight the event and rules correlating process **140-2** assigns to an event does not have to remain fixed. In addition to a weight being affected by time and/or occurrence of an event in a duration of time, a weight may also be affected by how long the event remains in the rules engine. Thus, in some cases, as time passes since the occurrence of an event, the event and rules correlating process **140-2** may calculate and assign a weight that is related to the time since the event and rules correlating process **140-2** created the event object representing that event, and that is further decremented as more time passes. Alternatively, the event and rules correlating process **140-2** may increase a weight as the time since an event object was created increases. For example, if a temperature sensor on a tank containing a liquid shows a value above a safe level at 12:00 AM, and the temperature sensor is still showing a value above a safe level at 2:00 AM, the event and rules correlating process **140-2** may increase the weight assigned to that event because the event has not been dealt with.

[0056] The event and rules correlating process **140-2** must, in addition to assigning weights to existing rules, also include an accumulator rule in the rules engine, step **505**. A condition of the accumulator rule is defined as the values in the accumulator associated with that rule exceeding a given value. The accumulator rule also defines one or more actions to occur upon the condition being met. The accumulator rule is what allows the event and rules correlating process **140-2** to monitor the value stored in an accumulator, and then take action if needed. The event and rules correlating process **140-2** may, of course, include a number of accumulator rules in the rules engine, all covering different rules from the same group of rules to which the event and rules correlating process **140-2** assigned weights. That is, for example, the group of rules to which the event and rules correlating process **140-2** assigned weights may include rules related to badge readers, rules related to video cameras, and rules related to infrared sensors. The event and rules correlating process **140-2** may include an accumulator rule that, for example, deals only with the badge reader rules, as well as an accumulator rule that deals with only the video camera rules, and an accumulator rule that deals with the badge reader rules, the video camera rules, and the infrared sensor rules. In other words, the variety of possible accumulator rules allows a user of the event and rules correlating process **140-2** to effectively define what combinations of events, and what combination of weights for those combination of events, result in one or more actions being taken.

[0057] The event and rules correlating process **140-2** will then evaluate an accumulator rule at a frequency over a period of time, step **506**, similar to how the event and rules correlating process **140-2** evaluates any rule stored within the rules engine. Finally, upon the condition(s) of an accumulator rule being met, the event and rules correlating process **140-2** triggers the one or more actions defined in the accumulator rule to occur, step **507**.

[0058] Note that, in some embodiments, an accumulator may have different levels associated with it. For example, an accumulator rule may indicate that, when the value in the accumulator passes a first given value, the events being accumulated may represent a low-level threat, and certain related actions result (i.e., providing an alert to security personnel). When the value in the accumulator passes a second given value, the events being accumulated may represent a mid-level threat, and certain other actions result (i.e., security personnel being summoned to a particular area). As many different types of levels as are desired may be defined, each with their own particular actions. Further, in some embodiments, the event and rules correlating process **140-2** may visualize each level on the graphical user interface **160** (shown in FIG. **1**), such as, for example, as a dial or bar, or using a color-coded scheme.

[0059] Further, in some embodiments, a plurality of accumulators may be associated with particular geographic locations. That is, a first accumulator may be associated with a first building, a second accumulator associated with a second building, and so on. Alternatively, or additionally, a plurality of accumulators may be associated with a hierarchy of geographic locations. Thus, a first accumulator may be associated with a first floor of a first building, a second accumulator may be associated with a first floor of a second building, while a third accumulator may be associated with the first building and the second building. This allows the event and rules correlating process **140-2** to deal with slightly troubling events that are spread out over a large area. In other words, any single event may contribute to multiple accumulators, including both localized accumulator(s) as well as accumulator(s) associated with a broader area.

[0060] In some embodiments, accumulators may count events themselves, instead of weights associated with events. Thus, an accumulator rule could count, for example, the number of people entering a room, and if that count exceeded a given value defined in the accumulator rule, an action would result.

[0061] FIG. **6** illustrates the event and rules correlating process **140-2** recognizing a number of events as a situation, and then injecting the situation back into the rules engine for further processing over time. The event and rules correlating process **140-2** first receives security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data has its own pipeline, step **601**. The event and rules correlating process **140-2** then creates event objects from the received security event data, step **602**, and gathers the event objects into a rules engine, step **603**, as described herein. The rules engine comprises a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response. Next, the event and rules correlating process **140-2** evaluates a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, step **604**, so to determine whether the one or more conditions defined within the evaluated rule are met.

[0062] Prior to any actions being taken, the event and rules correlating process **140-2** may identify, within the rules engine, a situation, wherein a situation is the occurrence of one or more events, step **606**. Thus, a situation is number of events that occur simultaneously or in a span of a period of time. Situations may include any number of events. Events comprising a situation may or may not be related. As discussed further below, a situation may change over time. Further, any number of situations may occur simultaneously, and multiple situations may include some of the same events.

[0063] In more detail, the event and rules correlating process **140-2** identifies a situation by detecting that a number of rules have evaluated positively and actions defined in those rules have occurred. That is, the event and rules correlating process **140-2** detects that the conditions for a number of rules, each rule corresponding to an event, the conditions corresponding to the occurrence of the event, have all been met, where the occurrence of those events constitutes the situation. For example, a situation may be a coordinated attempt to rob a bank. The events that make up the situation may include power being cut to video cameras, phone lines being cut, cellular phones being blocked, lights being turned off, incorrect combinations being given to a vault, and so on. The rules engine includes rules that describe all of these events through one or more conditions. When the event and rules correlating process **140-2** detects that all of these conditions for these rules have been met, such that actions defined in the rules are initiated, the event and rules correlating process **140-2** identifies a situation. The situation contains the events described by all the met conditions of the rules.

[0064] The event and rules correlating process **140-2** will then create, in a situation manager, a situation object, step

607. The situation object includes data from one or more event objects that describe the one or more events. The included data identifies the one or more events that are occurring that make up the situation. Note that the situation object further includes one or more actions to be taken upon occurrence of the situation.

[0065] The event and rules correlating process **140-2** injects the situation object from the situation manager into the rules engine, step **608**. By placing the situation object in the rules engine, the event and rules correlating process **140-2** is able to maintain state of all the event data that identifies the events that comprise the situation. That is, a situation tends to change over time. In the bank robbery example given above, the thieves may acquire the correct combination to a vault through coercion or other means, and thus they may be able to open that vault. The event and rules correlating process **140-2** is then able to update situation object, as described below, so that the event data shows that the correct combination was provided and thus the vault is now opened, as opposed to unsuccessful attempts being made to provide the correct combination to the vault.

[0066] In more detail, the rules engine will include one or more rules where a first condition to be met is that a situation is occurring, and where a second condition to be met is that one or more further events occur. In such a scenario, for example the bank vault opening scenario described above, upon occurrence of the one or more further events (i.e., the proper combination being provided to the bank vault so that is opens), the event and rules correlating process **140-2** changes the situation object corresponding to the situation by adding the one or more further events to the situation object, step **609**. In some embodiments, events may be deleted from a situation object, or otherwise modified (e.g., updated), instead of, or in addition to, being added.

[0067] In addition to changing the situation object by adding one or more further events, the event and rules correlating process **140-2** may also, in response to doing so, further change the situation object by changing the one or more actions to be taken upon occurrence of the situation, step **610**. The change in the one or more actions may be based upon the occurrence of the one or more further events. Thus, continuing to use the bank robbery example, the event and rules correlating process **140-2** may, upon the correct combination being provided to open the vault, change the situation object so that one of the actions taken in response to the situation is to activate alarms and warning lights associated with the vault, despite the vault being opened by the proper combination. The event and rules correlating process **140-2** would thus initiate the one or more actions defined by the situation object whenever the situation is occurring, and as the actions defined therein changed, the event and rules correlating process **140-2** would take appropriate steps to initiated those changes as well.

[0068] In FIG. **7**, the event and rules correlating process **140-2** initiates one or more actions defined in an evaluated rule by sending appropriate commands to control particular devices, and in some embodiments, to intelligently control video devices. The event and rules correlating process **140-2** first receives security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data has its own pipeline, step **701**. The event and rules correlating process **140-2** then creates event objects from the received security event data, step **702**, and gathers the event objects into a rules engine, step **703**, as described herein. The rules engine comprises a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response. Next, the event and rules correlating process **140-2** evaluates a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, step **704**, so to determine whether the one or more conditions defined within the evaluated rule are met. Finally, the event and rules correlating process **140-2** initiates the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met, step **705**. Actions may include any of the commands described herein.

[0069] In some embodiments, more particularly, the event and rules correlating process **140-2** initiates the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met, wherein one of the one or more actions includes intelligent controlling of a video device based on received security event data, step **706**. In some embodiments, intelligent controlling of a video device may include, for example, allowing a user to use a series of video cameras to track an individual both forwards and backwards through an area under surveillance of the series of video cameras. Such methods and apparatus are disclosed in co-pending U.S. patent application Ser. No. ______, entitled "TRACKING PEOPLE AND OBJECTS USING MULTIPLE LIVE AND RECORDED SURVEILLANCE CAMERA VIDEO FEEDS", given Attorney Docket No. VID08-03, the entirety of which is hereby incorporated by reference. Alternatively, in other embodiments, intelligent controlling of a video device may include intelligently controlling a sequence of cameras in order to provide a video tour of an area. Such methods and apparatus are disclosed in co-pending U.S. patent application Ser. No. ______, entitled "INTELLIGENT VIDEO TOURS", given Attorney Docket No. VID08-04, the entirety of which is hereby incorporated by reference. Thus, intelligently controlling a video device uses automated functionality to assist a user in deciding which video device(s) to use and how that (those) device(s) should be used (i.e., the direction in which the device should be aimed, etc.).

[0070] In some embodiments, the event and rules correlating process **140-2** provides a standardized way of controlling a video device, or any other controllable device, that may be programmed by a user. That is, in such embodiments, the event and rules correlating process **140-2** creates an output package to cause one or more types of a device to execute one or more steps, step **707**. The event and rules correlating process **140-2** creates an output package for a type of a device using information output by the rules engine and then processed in an output pipeline. An output pipeline is essentially an input pipeline discussed above with regards to FIGS. **4** and **2**B in reverse. Thus, an output pipeline is a textually defined stack used to control external devices, such as access control, lighting systems, etc. Similar to an input pipeline, the event and rules correlating process **140-2** defines a stack that formulates a command message using various layers, the result being a command specific message for an external device (i.e., output package). An example of an output pipeline is shown in FIG. **2**C.

[0071] As is seen in FIG. **2**C, an output action **270**, generated by the rules engine (shown in FIG. **2**B), is passed to a stack **280**. The stack **280** may contain any number of layers, but in the example shown, contains a normalize layer **282**, a format layer **284**, and a transmit layer **286**, as well as an optional log layer, **288**. The event and rules correlating process **140-2**, through the stack **280**, thus reformulates the output action **270** as an output package.

[0072] The event and rules correlating process **140-2** first determines, from the output action **270**, the device to which the output package is to be sent, and the one or more steps it is to execute (i.e., commands), from the one or more actions defined by the evaluated rule. This information, because it is generated by the event and rules correlating process **140-2** in the rules engine, is in the canonical form discussed above with regards to FIG. **4**. The device ultimately receiving the output package, of course, does not necessarily understand commands formatted in such a form. Thus, the event and rules correlating process **140-2**, in the normalize layer **282**, removes the standardization of the canonical form, and then, in the format layer **284**, formats the information for the particular type of the device that is receiving the output package. That is, for example, the event and rules correlating process **140-2** will format access control system commands (i.e., lock a door/window, unlock a door/window, etc.) according to the particular format of that brand of access control system that is receiving the output package. In other words, all access control systems manufactured by, for example, Honeywell®, will receive output packages containing commands formatted according to Honeywell® specifications, while all access control systems manufactured by Lenel® will receive output packages containing commands formatted according to Lenel® specifications. Alternatively, the event and rules correlating process **140-2** will format lighting system commands (i.e., lights on, lights off, intensity of the light(s), etc.) according to the particular format of the brand of lighting system that is receiving the output package, such that each different type of lighting system that is to receive commands receives its own formatted output package. At the completion of this layer, the event and rules correlating process **140-2** has created one or more output packages and now must send it/them to the appropriate device(s).

[0073] The event and rules correlating process **140-2** thus transmits the created output package to the one or more types of the device, where upon receipt of its created output package, each of the one or more types of the device executes the one or more steps included in the output package, step **708**. The event and rules correlating process **140-2** does this through the transport layer **286** of the output pipeline. The transport layer **286** includes all the various transport protocols that may be used by the device(s) receiving the output packages, and uses the particular transport protocol for a particular device to send the output package for that device to that device. Optionally, in some embodiments, the event and rules correlating process **140-2** may use the log layer **288** to record some or all of the output packages being sent to device(s) for later analysis.

[0074] The methods and systems described herein are not limited to a particular hardware or software configuration, and may find applicability in many computing or processing environments. The methods and systems may be implemented in hardware or software, or a combination of hardware and software. The methods and systems may be implemented in one or more computer programs, where a computer program may be understood to include one or more processor executable instructions. The computer program(s) may execute on one or more programmable processors, and may be stored on one or more storage medium readable by the processor (including volatile and non-volatile memory and/or storage elements), one or more input devices, and/or one or more output devices. The processor thus may access one or more input devices to obtain input data, and may access one or more output devices to communicate output data. The input and/or output devices may include one or more of the following: Random Access Memory (RAM), Redundant Array of Independent Disks (RAID), floppy drive, CD, DVD, magnetic disk, internal hard drive, external hard drive, memory stick, flash memory (i.e., solid state memory) device, or other storage device capable of being accessed by a processor as provided herein, where such aforementioned examples are not exhaustive, and are for illustration and not limitation.

[0075] The computer program(s) may be implemented using one or more high level procedural or object-oriented programming languages to communicate with a computer system; however, the program(s) may be implemented in assembly or machine language, if desired. The language may be compiled or interpreted.

[0076] As provided herein, the processor(s) may thus be embedded in one or more devices that may be operated independently or together in a networked environment, where the network may include, for example, a Local Area Network (LAN), wide area network (WAN), and/or may include an intranet and/or the internet and/or another network. The network(s) may be wired or wireless or a combination thereof and may use one or more communications protocols to facilitate communications between the different processors. The processors may be configured for distributed processing and may utilize, in some embodiments, a client-server model as needed. Accordingly, the methods and systems may utilize multiple processors and/or processor devices, and the processor instructions may be divided amongst such single- or multiple-processor/devices.

[0077] The device(s) or computer systems that integrate with the processor(s) may include, for example, a personal computer(s), workstation(s) (e.g., Sun, HP), personal digital assistant(s) (PDA(s)), handheld device(s) such as cellular telephone(s), laptop(s), handheld computer(s), or another device(s) capable of being integrated with a processor(s) that may operate as provided herein. Accordingly, the devices provided herein are not exhaustive and are provided for illustration and not limitation.

[0078] References to "a microprocessor" and "a processor", or "the microprocessor" and "the processor," may be understood to include one or more microprocessors that may communicate in a stand-alone and/or a distributed environment(s), and may thus be configured to communicate via wired or wireless communications with other processors, where such one or more processor may be configured to operate on one or more processor-controlled devices that may be similar or different devices. Use of such "microprocessor" or "processor" terminology may thus also be understood to include a central processing unit, an arithmetic logic unit, an application-specific integrated circuit (IC), and/or a task engine, with such examples provided for illustration and not limitation.

[0079] Furthermore, references to memory, unless otherwise specified, may include one or more processor-readable and accessible memory elements and/or components that may be internal to the processor-controlled device, external to the processor-controlled device, and/or may be accessed via a wired or wireless network using a variety of communications protocols, and unless otherwise specified, may be arranged to include a combination of external and internal memory devices, where such memory may be contiguous and/or partitioned based on the application. Accordingly, references to a database may be understood to include one or more memory associations, where such references may include commercially available database products (e.g., SQL, Informix, Oracle) and also proprietary databases, and may also include other structures for associating memory such as links, queues, graphs, trees, with such structures provided for illustration and not limitation.

[0080] References to a network, unless provided otherwise, may include one or more intranets and/or the internet. References herein to microprocessor instructions or microprocessor-executable instructions, in accordance with the above, may be understood to include programmable hardware.

[0081] Unless otherwise stated, use of the word "substantially" may be construed to include a precise relationship, condition, arrangement, orientation, and/or other characteristic, and deviations thereof as understood by one of ordinary skill in the art, to the extent that such deviations do not materially affect the disclosed methods and systems.

[0082] Throughout the entirety of the present disclosure, use of the articles "a" or "an" to modify a noun may be understood to be used for convenience and to include one, or more than one of the modified noun, unless otherwise specifically stated.

[0083] Elements, components, modules, and/or parts thereof that are described and/or otherwise portrayed through the figures to communicate with, be associated with, and/or be based on, something else, may be understood to so communicate, be associated with, and or be based on in a direct and/or indirect manner, unless otherwise stipulated herein.

[0084] Although the methods and systems have been described relative to a specific embodiment thereof, they are not so limited. Obviously many modifications and variations may become apparent in light of the above teachings. Those skilled in the art may make many additional changes in the details, materials, and arrangement of parts, herein described and illustrated.

What is claimed is:

1. A method of correlating security event data according to one or more rules to initiate one or more actions, the method comprising:

receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline;

creating event objects from the received security event data;

gathering the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response;

evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met; and

initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

2. The method of claim **1** wherein creating comprises:

pre-processing the received security event data to create event objects that include the received security event data in a canonical form.

3. The method of claim **2** wherein pre-processing comprises:

supplementing an event object with related data not acquired from the source of the received security event data, by using an item of data within the received security event data to look up the related data, the related data used to set one or more attributes of the event object.

4. The method of claim **1** comprising:

prior to evaluating a rule, injecting one or more facts into rules engine, the one or more facts used to evaluate one or more rules.

5. The method of claim **1** comprising:

assigning weights to a group of rules within the rules engine, such that when the one or more conditions of a rule in the group of rules are met, a value of the weight assigned to that rule is added to an accumulator;

including an accumulator rule in the rules engine, wherein a condition of the accumulator rule is defined as the values in the accumulator exceeding a given value, the accumulator rule defining one or more actions to occur upon the condition being met;

evaluating the accumulator rule at a frequency over a period of time; and

upon the condition of the accumulator rule being met, triggering the one or more actions defined in the accumulator rule to occur.

6. The method of claim **1** comprising:

identifying, within the rules engine, a situation, wherein a situation is the occurrence of one or more events;

creating, in a situation manager, a situation object, the situation object including data from one or more event objects that describe the one or more events, the included data identifying the one or more events that are occurring that make up the situation, the situation further including one or more actions to be taken upon occurrence of the situation; and

injecting the situation object from the situation manager into the rules engine.

7. The method of claim **6** wherein the rules engine includes one or more rules where a first condition to be met is that a situation is occurring, and where a second condition to be met is that one or more further events occur, the method comprising:

upon occurrence of the one or more further events, changing the situation object corresponding to the situation by adding the one or more further events to the situation object.

8. The method of claim **7** comprising:

in response to adding the one or more further events to the situation object, further changing the situation object by changing the one or more actions to be taken upon occurrence of the situation, the change in the one or more actions based upon the occurrence of the one or more further events.

9. The method of claim **1** wherein initiating comprises:

initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met, wherein one of the one or more actions includes intelligent controlling of a video device based on received security event data.

10. The method of claim **1** wherein initiating comprises:

creating an output package to cause one or more types of a device to execute one or more steps, the device and the one or more steps it is to execute determined from the one or more actions defined by the evaluated rule, wherein the output package is standardized for each type of the device; and

transmitting the created output package to the one or more types of the device, where upon receipt of its created output package, each of the one or more types of the device executes the one or more steps included in the output package.

11. A computer program product, stored on computer readable medium, for correlating security event data according to one or more rules to initiate one or more actions, comprising:

computer program code for receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline;

computer program code for creating event objects from the received security event data;

computer program code for gathering the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response;

computer program code for evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met; and

computer program code for initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

12. The computer program product of claim **11** wherein computer program code for creating comprises:

computer program code for pre-processing the received security event data to create event objects that include the received security event data in a canonical form, wherein computer program code for pre-processing comprises computer program code for supplementing an event object with related data not acquired from the source of the received security event data, by using an item of data within the received security event data to look up the related data, the related data used to set one or more attributes of the event object.

13. The computer program product of claim **1** comprising:

computer program code for assigning weights to a group of rules within the rules engine, such that when the one or more conditions of a rule in the group of rules are met, a value of the weight assigned to that rule is added to an accumulator;

computer program code for including an accumulator rule in the rules engine, wherein a condition of the accumulator rule is defined as the values in the accumulator exceeding a given value, the accumulator rule defining one or more actions to occur upon the condition being met;

computer program code for evaluating the accumulator rule at a frequency over a period of time; and

upon the condition of the accumulator rule being met, computer program code for triggering the one or more actions defined in the accumulator rule to occur.

14. The computer program product of claim **1** comprising:

computer program code for identifying, within the rules engine, a situation, wherein a situation is the occurrence of one or more events;

computer program code for creating, in a situation manager, a situation object, the situation object including data from one or more event objects that describe the one or more events, the included data identifying the one or more events that are occurring that make up the situation, the situation further including one or more actions to be taken upon occurrence of the situation; and

computer program code for injecting the situation object from the situation manager into the rules engine.

15. The computer program product of claim **14** wherein the rules engine includes one or more rules where a first condition to be met is that a situation is occurring, and where a second condition to be met is that one or more further events occur, the computer program product comprising:

upon occurrence of the one or more further events, computer program code for changing the situation object corresponding to the situation by adding the one or more further events to the situation object.

16. The computer program product of claim **15** comprising:

in response to adding the one or more further events to the situation object, computer program code for further changing the situation object by changing the one or more actions to be taken upon occurrence of the situation, the change in the one or more actions based upon the occurrence of the one or more further events.

17. A computer system comprising:

a memory;

a processor;

a network interface; and

an interconnection mechanism coupling the memory, the processor, and the network interface, allowing communication there between;

wherein the memory of the computer system is encoded with an event and rules correlating application, that when executed in the processor, provides an event and rules correlating process that correlates security event data according to one or more rules to initiate one or more actions, by causing the computer system to perform operations of:

receiving security event data via an event pipeline, security event data defining an occurrence of an event, wherein each type of source of event data may have its own pipeline;

creating event objects from the received security event data;

gathering the event objects into a rules engine, the rules engine comprising a plurality of rules, wherein each rule defines one or more conditions to be met and one or more actions to be taken in response;

evaluating a rule from the plurality of rules within the rules engine at a frequency over a period of time using data contained within event objects, so to determine whether the one or more conditions defined within the evaluated rule are met; and

initiating the one or more actions defined by the evaluated rule whenever the one or more conditions defined by that rule are met.

18. The computer system of claim **17** wherein creating comprises:

pre-processing the received security event data to create event objects that include the received security event data in a canonical form, wherein pre-processing comprises supplementing an event object with related data not acquired from the source of the received security event data, by using an item of data within the received security event data to look up the related data, the related data used to set one or more attributes of the event object.

19. The computer system of claim **17** comprising:

assigning weights to a group of rules within the rules engine, such that when the one or more conditions of a rule in the group of rules are met, a value of the weight assigned to that rule is added to an accumulator;

including an accumulator rule in the rules engine, wherein a condition of the accumulator rule is defined as the values in the accumulator exceeding a given value, the accumulator rule defining one or more actions to occur upon the condition being met;

evaluating the accumulator rule at a frequency over a period of time; and

upon the condition of the accumulator rule being met, triggering the one or more actions defined in the accumulator rule to occur.

20. The computer system of claim **17** comprising:

identifying, within the rules engine, a situation, wherein a situation is the occurrence of one or more events;

creating, in a situation manager, a situation object, the situation object including data from one or more event objects that describe the one or more events, the included data identifying the one or more events that are occurring that make up the situation, the situation further including one or more actions to be taken upon occurrence of the situation; and

injecting the situation object from the situation manager into the rules engine.

21. The computer system of claim **20** wherein the rules engine includes one or more rules where a first condition to be met is that a situation is occurring, and where a second condition to be met is that one or more further events occur, wherein the computer system performs operations of:

upon occurrence of the one or more further events, changing the situation object corresponding to the situation by adding the one or more further events to the situation object.

22. The computer system of claim **21** comprising:

in response to adding the one or more further events to the situation object, further changing the situation object by changing the one or more actions to be taken upon occurrence of the situation, the change in the one or more actions based upon the occurrence of the one or more further events.

* * * * *