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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
31 July 2014 (31.07.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/116294 A1

(51) International Patent Classification: *G06F 15/16* (2006.01)

(21) International Application Number: PCT/US2013/042788

(22) International Filing Date: 27 May 2013 (27.05.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/751,012 25 January 2013 (25.01.2013) US

(71) Applicant: **CONCURIX CORPORATION** [US/US]; 244 Market Street, Kirkland, Washington 98033 (US).

(72) Inventor: **KRAJEC, Russell S.**; 208 Newell Drive, Loveland, Colorado 80538 (US).

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

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

Published:

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

(54) Title: TRACING WITH A WORKLOAD DISTRIBUTOR

(57) Abstract: A load balanced system may incorporate instrumented systems within a group of managed devices and distribute workload among the devices to meet both load balancing and data collection. A workload distributor may communicate with and configure several managed devices, some of which may have instrumentation that may collect trace data for workload run on those devices. Authentication may be performed between the managed devices and the workload distributor to verify that the managed devices are able to receive the workloads and to verify the workloads prior to execution. The workload distributor may increase or decrease the amount of instrumentation in relation to the workload experienced at any given time.

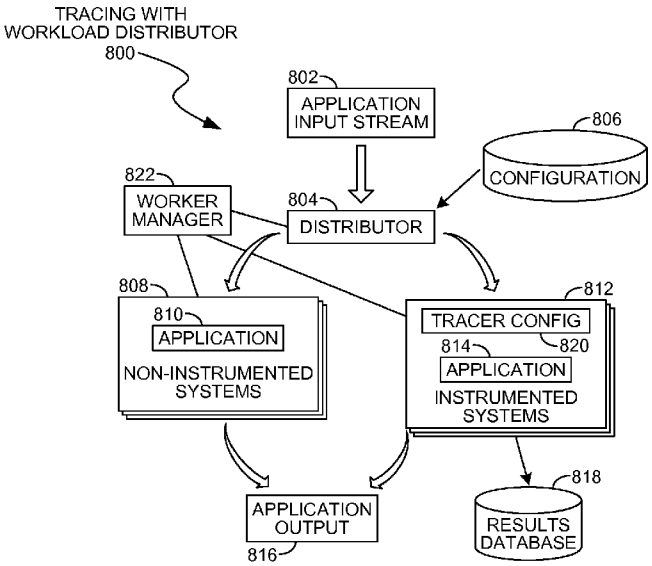

FIG. 8

WO 2014/116294 A1

Tracing with a Workload Distributor

Cross Reference to Related Applications

[0001] This application claims priority to and benefit of United States Patent Application serial number 13/751,012, filed January 25, 2013, entitled Tracing with a Workload Distributor, the entire contents of which are hereby expressly incorporated by reference.

Background

[0002] Tracing gathers information about how an application executes within a computer system. Trace data may include any type of data that may explain how the application operates, and such data may be analyzed by a developer during debugging or optimization of the application. In many cases, trace data may be used for debugging an application as well as understanding and optimizing the application. Trace data may also be used by an administrator during regular operation of the application to identify any problems.

Summary

[0003] An instrumented execution environment may connect to an execution environment to provide detailed tracing and logging of an application as it runs. The instrumented execution environment may be configured as a standalone service that can be configured and purchased. The instrumented execution environment may be deployed with various authentication systems, administrative user interfaces, and other components. The instrumented execution environment may engage a customer's system through a distributor that may manage an application workload to distribute work to the instrumented execution environment as well as other worker systems. A marketplace may provide multiple preconfigured execution environments that may be selected, further configured, and deployed to address specific data collection objectives.

[0004] A load balanced system may incorporate instrumented systems within a group of managed devices and distribute workload among the devices to meet both load balancing and data collection. A workload distributor may communicate with and configure several managed devices, some of which may have instrumentation that may collect trace data for workload run on those devices. Authentication may be performed between the managed devices and the workload distributor to verify that the managed devices are able to receive the workloads and to verify the workloads prior to execution. The workload distributor may increase or decrease the amount of instrumentation in relation to the workload experienced at any given time.

[0005] A parallel tracer may perform detailed or heavily instrumented analysis of an application in parallel with a performance or lightly instrumented version of the application. Both versions of the application may operate on the same input stream, but with the heavily instrumented version having different performance results than the lightly instrumented version. The tracing results may be used for various analyses, including optimization and debugging.

[0006] A tracer may obfuscate trace data such that the trace data may be used in an unsecure environment even though raw trace data may contain private, confidential, or other sensitive information. The tracer may obfuscate using irreversible or lossy hash functions, look up tables, or other mechanisms for certain raw trace data, rendering the obfuscated trace data acceptable for transmission, storage, and analysis. In the case of parameters passed to and from a function, trace data may be obfuscated as a group or as individual parameters. The obfuscated trace data may be transmitted to a remote server in some scenarios.

[0007] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

Brief Description of the Drawings

[0008] In the drawings,

[0009] FIGURE 1 is a diagram illustration of an embodiment showing a system for tracing applications that process confidential data.

[0010] FIGURE 2 is a diagram illustration of an embodiment showing a network environment for analyzing sensitive trace data.

[0011] FIGURE 3 is a timeline illustration of an embodiment showing interactions between a secure environment and a remote device.

[0012] FIGURE 4 is a diagram illustration of an embodiment showing a parallel tracing system.

[0013] FIGURE 5 is a diagram illustration of an embodiment showing a network environment with parallel tracing.

[0014] FIGURE 6 is a flowchart illustration of an embodiment showing a method for executing an application with parallel instrumentation.

[0015] FIGURE 7 is a flowchart illustration of an embodiment showing a method for combining results.

[0016] FIGURE 8 is a diagram illustration of an embodiment showing tracing with a workload distributor.

[0017] FIGURE 9 is a diagram illustration of an embodiment showing a network environment with a workload distributor.

[0018] FIGURE 10 is a flowchart illustration of an embodiment showing a method for load balancing with tracing.

[0019] FIGURE 11 is a diagram illustration of an embodiment showing tracing as a service.

[0020] FIGURE 12 is a diagram illustration of an embodiment showing a network environment with remote tracing.

[0021] FIGURE 13 is a flowchart illustration of an embodiment showing a method for configuring an instrumentation system.

[0022] FIGURE 14 is a flowchart illustration of an embodiment showing a method for deploying instrumented system.

Detailed Description

Tracing as a Service

[0023] An instrumented execution environment may be deployed as a service. After creating an account, a user may add the instrumented execution environment to a workload distributor within the user's own execution environment. The workload distributor may receive an incoming workload stream and direct some or all of the workload to the instrumented execution environment.

[0024] The instrumented execution environment may contain software, hardware, and other components that may capture various information while processing a workload. The instrumentation may collect various trace data that may be stored and analyzed. In some cases, the trace data may be analyzed after collection while in other cases, trace data may be analyzed on a real time basis.

[0025] A user may use an administrative user interface to configure the instrumented execution environment for use with a distributor. The user may be able to establish an account, determine a payment mechanism, and select various features of an instrumented execution environment.

[0026] The administrative activities may also include creating various authentication keys or other mechanisms that may authenticate the interacting systems to each other. The authentication system may be used to verify that the instrumented execution environment has been permitted to receive the output of a distributor, and that the distributor is authorized to send work items to the instrumented execution environment.

[0027] A distributor may be added to a user's execution environment to introduce a redirection point or decision point in an application. In some cases, the distributor may be an application that intercepts an input stream and applies both load balancing and instrumentation logic to identify an execution environment to process a given work item. In some cases, the distributor may be executable code that may be included in a library that may be called from within an application.

[0028] The instrumentation may be provided as a service to a developer or administrator of an application. The instrumentation may collect data using sophisticated tools and analysis that may be complex to install, configure, or operate. A user may pay for such a service using many different payment schemes, such as

paying based on part on how much processing, storage, or other resource may be consumed. In some cases, the payment may be a subscription for use over a period of time, such as a fixed fee payment for a month of service. Many other payment schemes may be deployed.

Tracing with a Workload Distributor

[0029] A load balanced or other managed computation environment may distribute work items to instrumented and non-instrumented systems. The load balancing or distribution may occur with consideration of instrumentation objectives for an application. In one example, the instrumentation may be performed when the load on the systems may allow, but instrumentation may be reduced or eliminated when load factors increase.

[0030] The workload distributor may have several instrumentation objectives that define conditions to collect data as well as the data to be collected. The objectives may include items such as sampling rates, events or conditions that start or stop instrumentation, quantity or quality of data to be collected, as well as the specific parameters or types of instrumentation to be applied.

[0031] The workload distributor may transmit instrumentation objectives to a worker system, where the objectives may cause the worker system to collect the described data. In such cases, the distributor may be able to create customized objectives for instrumenting each work item.

[0032] The computation environment may have multiple worker systems that execute work items as defined by a distributor. Each worker system may register with the distributor, which may include initial contact and registration as well as establishing an authentication mechanism between the devices. The status of the worker systems may be collected periodically and used to determine availability to perform a given work item.

Parallel Tracing

[0033] A heavily instrumented tracer may operate in parallel with a lightly instrumented tracer to capture both detailed and performance measurements of an application. In many cases, heavy instrumentation may adversely affect the performance of an application, thereby corrupting any performance metrics that may be collected at the same time. As the instrumentation becomes more detailed, the performance metrics generally may become more affected.

[0034] The tracing system may operate in parallel to trace in a performance environment that may capture only performance metrics, while another detailed tracer may capture detailed results. In some cases, the same application workload may be analyzed in parallel and the performance and detailed results may be aggregated together to produce a complete representation of the application.

[0035] The architecture of a parallel tracing system may be deployed in a distributed computing environment. In such an environment, multiple devices or processors may each perform some of the work of an application. A computing cluster may be one example of a distributed computing environment where multiple devices each execute a portion of an input stream.

[0036] A single device may be capable of parallel tracing. A multi-processor device may have some processors that may process an application workload in a high performance manner while other processors execute the application using a detailed tracer.

[0037] The concept of parallel tracing may be applied to a single system or single processor system. In such a system, the separate tracing operations may be performed sequentially. For example, a first run of an application workload may be performed using a performance level instrumentation followed by a second run of the workload using a detailed instrumentation system. In such an embodiment, the input stream may be captured for later execution in a detailed manner.

[0038] A distributor may identify units of work from an input stream to transmit to different components. A unit of work may be any computing workload that may be executed in a relatively independent fashion. For each program or application, the units of work may be different.

[0039] In some applications, a unit of work may be a function call that may include input parameters for the function. For functional languages, such as Erlang, Haskell, Scala, F#, or for non-functional languages that are written in a functional manner, portions of an application may be able to be computed independently. In some cases, the units of work may have dependencies or other interactions with other units of work.

[0040] A unit of work may be an input item or request made to an application. In an example of an application programming interface, a call to the interface may be considered a unit of work that may be executed by an instance of the application.

Such requests may or may not depend on other requests, but in general many such systems may have requests that may operate independently from other requests.

[0041] In some cases, a unit of work that may be executed by two differently instrumented systems may return different results. In a simple example, a unit of work may perform a function based on the exact time of day. In such an example, a unit of work executed on performance level instrumentation would return a different value than the same unit of work executed on a highly instrumented system that may be considerably slower.

[0042] When a unit of work executed on two different systems returns different values, an algorithm may be applied to determine a return value. In systems where latency, response time, or other performance related factors may adversely affect results, results from a performance level instrumented system may be used while results collected from highly instrumented system may be discarded. In some cases, the results from the performance level instrumented system may be discarded in favor of results from a highly instrumented system. In still other cases, averages or other summaries may be used to aggregate the application results from two separate runs of a unit of work.

Obfuscating Trace data

[0043] A tracer may obfuscate data collected from an application and transmit obfuscated data to an analysis engine. The analysis engine may perform all analyses on the obfuscated data, such that the analysis engine may not be exposed to any confidential, private, or otherwise sensitive data contained in the tracer-collected data.

[0044] A tracer may collect data during the execution of an application. The application may process data that may be sensitive. When the tracer encounters application data, such data may be obfuscated prior to analysis. The obfuscated data may then be analyzed to understand, categorize, optimize, or perform other functions relating to the application. In some cases, the analysis may generate recommendations or other results that may refer to specific data elements that may be obfuscated. In such cases, a mechanism for determining the underlying data values may be provided through a reverse obfuscation process.

[0045] The obfuscation process may consist of a hash function, which may be lossy or not. In other embodiments, the obfuscation process may be an encryption

process that may or may not be cryptographically secure. Still other embodiments may employ a lookup table maintained on a client device that translates a meaningful data value to an arbitrary value for analysis.

[0046] Multiple values may be hashed or otherwise considered as a single element in the tracer output. For example, a function may be called with three arguments. In some embodiments, the three arguments may be combined into a single argument and hashed or otherwise obfuscated into a single value. In other embodiments, the three arguments may be individually obfuscated and stored as three separate values.

[0047] Throughout this specification and claims, the terms "profiler", "tracer", and "instrumentation" are used interchangeably. These terms refer to any mechanism that may collect data when an application is executed. In a classic definition, "instrumentation" may refer to stubs, hooks, or other data collection mechanisms that may be inserted into executable code and thereby change the executable code, whereas "profiler" or "tracer" may classically refer to data collection mechanisms that may not change the executable code. The use of any of these terms and their derivatives may implicate or imply the other. For example, data collection using a "tracer" may be performed using non-contact data collection in the classic sense of a "tracer" as well as data collection using the classic definition of "instrumentation" where the executable code may be changed. Similarly, data collected through "instrumentation" may include data collection using non-contact data collection mechanisms.

[0048] Further, data collected through "profiling", "tracing", and "instrumentation" may include any type of data that may be collected, including performance related data such as processing times, throughput, performance counters, and the like. The collected data may include function names, parameters passed, memory object names and contents, messages passed, message contents, registry settings, register contents, error flags, interrupts, or any other parameter or other collectable data regarding an application being traced.

[0049] Throughout this specification and claims, the term "execution environment" may be used to refer to any type of supporting software used to execute an application. An example of an execution environment is an operating system. In some illustrations, an "execution environment" may be shown separately from an

operating system. This may be to illustrate a virtual machine, such as a process virtual machine, that provides various support functions for an application. In other embodiments, a virtual machine may be a system virtual machine that may include its own internal operating system and may simulate an entire computer system. Throughout this specification and claims, the term "execution environment" includes operating systems and other systems that may or may not have readily identifiable "virtual machines" or other supporting software.

[0050] Throughout this specification, like reference numbers signify the same elements throughout the description of the figures.

[0051] When elements are referred to as being "connected" or "coupled," the elements can be directly connected or coupled together or one or more intervening elements may also be present. In contrast, when elements are referred to as being "directly connected" or "directly coupled," there are no intervening elements present.

[0052] The subject matter may be embodied as devices, systems, methods, and/or computer program products. Accordingly, some or all of the subject matter may be embodied in hardware and/or in software (including firmware, resident software, micro-code, state machines, gate arrays, etc.) Furthermore, the subject matter may take the form of a computer program product on a computer-usable or computer-readable storage medium having computer-usable or computer-readable program code embodied in the medium for use by or in connection with an instruction execution system. In the context of this document, a computer-usable or computer-readable medium may be any medium that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

[0053] The computer-usable or computer-readable medium may be, for example but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, device, or propagation medium. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media.

[0054] Computer storage media includes volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM,

EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can accessed by an instruction execution system. Note that the computer-usable or computer-readable medium could be paper or another suitable medium upon which the program is printed, as the program can be electronically captured, via, for instance, optical scanning of the paper or other medium, then compiled, interpreted, of otherwise processed in a suitable manner, if necessary, and then stored in a computer memory.

[0055] When the subject matter is embodied in the general context of computer-executable instructions, the embodiment may comprise program modules, executed by one or more systems, computers, or other devices. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Typically, the functionality of the program modules may be combined or distributed as desired in various embodiments.

[0056] Figure 1 is a diagram of an embodiment 100 showing a tracing system for an application that processes sensitive data. Embodiment 100 is a simplified example of a system where tracing may occur within a secure environment, then trace data may be analyzed in an unsecure environment without compromising the integrity of the sensitive data.

[0057] Embodiment 100 illustrates an example of a tracing system that gathers data in a secure environment and obfuscates the data prior to sending the data to a remote system for analysis. The remote system may be in an unsecured environment, but because the sensitive data are obfuscated, any analysis on those data can be performed without compromising the security of the underlying data.

[0058] In an example, a secure environment 102 may have an application 104 that may process sensitive data. The sensitive data may be, for example, credit card numbers, social security numbers, employment data, healthcare data, classified military data, financial data, or any other type of data that may be sensitive from a legal, business, personal, or other perspective.

[0059] A tracer 106 may monitor the application 104 to collect performance and operational data while the application 104 executes. The tracer 106 may examine

individual functions, threads, processes, messages, data objects, and other information and that are part of the application or may be handled by the application. The tracer 106 may perform low-level inspection of various program elements, the results of which may be used to analyze the program execution. Such analysis may be for debugging, auditing, optimization, or other purposes.

[0060] In many cases, the tracer 106 may handle or come in contact with information that includes sensitive data. In such a circumstance, the data may be obfuscated prior to leaving the secure environment 102. In a simple example, a function may receive a data object, such as a raw credit card number. A trace of the function may collect the credit card number as a value passed to the function. In another example, a message passed from one thread to another may contain unfiltered financial information, a medical record object, or other sensitive information. During normal tracing operations, such sensitive data may be collected, but the sensitive data may be obfuscated before passing the data outside of the secure environment.

[0061] When sensitive data are collected and handled within the secure environment 102, the data may be considered safe. In a typical secure environment 102, data may be processed in a facility that complies with various privacy and security standards and procedures. Such facilities may maintain physical security that limits access to certain personnel, as well as network security that restricts access to the data.

[0062] An obfuscator 108 may obfuscate some or all of the trace data to create obfuscated trace data 110. The obfuscator 108 may use various hash functions, encryption algorithms, substitution schemes, or other techniques to make protect the sensitive data. In some cases, the obfuscator 108 may create a lookup database 112 that may contain the raw and obfuscated values for the traced data.

[0063] The obfuscated trace data 110 may be transmitted outside the secure environment 102. The obfuscated trace data 114 may be located on a remote device or other system that may have an analysis engine 116 that may perform analytics, optimizations, or other analyses on the obfuscated trace data 114. The results of the analysis engine 116 may reference individual data items or may contain references to data elements that remain obfuscated.

[0064] The results 118 may be passed back into the secure environment 102 and acted upon by a results processor 120. The results processor 120 may determine

the raw data values from the obfuscated data values. In some cases, such an operation may involve looking up the raw data value from the lookup database 112.

[0065] In one use scenario, a tracer 106 may analyze an application 104 that may handle bank account information, for example. The tracer 106 may identify a function to monitor, where the function receives a bank account number and returns a balance. In this example, the bank account number and the balance may be treated as private information.

[0066] The tracer 106 may detect that the function has been called and may capture the data sent to the function and the data returned by the function. In this case, the bank account information has been transferred to the function and the balance returned. The function may be called many times, and the tracer 106 may capture each time the function is called. After monitoring the application 104 for a period of time, the trace data may contain bank account information and balance information.

[0067] Before transmitting the trace data outside of the secure environment 102, an obfuscator 108 may obfuscate the trace data to create obfuscated trace data 110. Once obfuscated, the data may be analyzed by a remote system to determine performance metrics or debugging information for the application 104.

[0068] The analysis may handle each data object using the obfuscated value. In an example, a debugging analysis may determine that a specific input value to the function causes an unexpected behavior to the function. Because the analysis is performed on the obfuscated data, the analysis results may be transmitted back to the secure environment 102 where the original value for the trace data may be determined and action taken on the results.

[0069] In such a scenario, the sensitive data may be kept within the secure environment 102, yet the analysis may be performed in an environment that does not share the same level of security. For example, a trace analyzer or program optimizer may analyze applications that may be secret, secure, private, or otherwise sensitive, yet the results may be transmitted and analyzed in a relatively open environment. For example, the obfuscated trace data 110 may be transmitted in clear text with minimum or no encryption and stored in a facility that may not meet the high security standards of the application 104.

[0070] The analysis engine 116 may be an automated, semi-automated, or manual analysis of the obfuscated trace data 114. The results 118 may contain direct references to the obfuscated trace data 114, such as identifying the data values that caused an error condition, for example. However, because the analyses may be performed only on obfuscated trace data 114, the analysis engine 116 may not be exposed to the underlying raw data.

[0071] In some cases, the remote system may collect data from multiple users, each of which may produce obfuscated trace data. The data from each user may be combined into a single database containing trace data from many sources. In such cases, the remote system may store only obfuscated data and sensitive data may be kept within the secure environment 102.

[0072] The obfuscator 108 may create obfuscated trace data 110 using several different mechanisms. In some cases, the obfuscation mechanisms may or may not be lossy.

[0073] An example of a non-lossy system may be an encryption system that may use a key to encrypt the data. In such a system, the obfuscator 108 may encrypt the data items using the key and the analysis engine 116 may process the encrypted data objects. Once the results 118 are returned to the secure environment 102, the results processor 120 may decrypt the results 118 using the key. Such a system may not use a lookup database 112 to re-create the raw values from the obfuscated values.

[0074] Another example of a non-lossy system may be the use of a non-reversible secure hash, such as MD5, SHA, or other hash functions. Such functions may have none or very few collisions, but may be extremely difficult to extract the original value from the obfuscated value. In such systems, a lookup database 112 may be used to store the hashed and raw values, so that results 118 may be converted back to raw, un-hashed values.

[0075] An example of a lossy system may apply a lossy hash function, such as a checksum or other lossy compression technique to the raw data. Examples of such systems may be hash functions that have many collisions. Such a system may create obfuscated data elements that may not be reversed into a single value with a degree of certainty.

[0076] An example of another system, a lookup database 112 may be used to assign a random or sequential value to a raw value. For example, a record for each

new raw value may be assigned an incrementing index and the index may serve as the obfuscated value. Because the lookup database 112 may contain sensitive data, the lookup database 112 may be stored and protected within the secure environment 102.

[0077] In some embodiments, different types of obfuscation may be applied to different trace data. For example, highly sensitive data elements may be obfuscated with encryption while less sensitive data elements may be obfuscated with a simpler hash function. Such an embodiment may apply more computationally expensive obfuscation to more sensitive data and less computationally expensive obfuscation to less sensitive data.

[0078] Some trace data may be stored in cleartext or may not be obfuscated. For example, some embodiments may store function names in cleartext yet may obfuscate data passed to and from a function. In general, a function name, parameter name, variable name, or other hard coded descriptors within an application may describe operations of an application but not the underlying data that may be processed.

[0079] Cleartext descriptors of application elements may be extracted from a source code description of the application. Some embodiments may include a source code analyzer that extracts the descriptors of various application elements. In some embodiments, such analyzers may be built into a compiler, the output of which may include debugging or tagging information.

[0080] Cleartext descriptors of application elements, such as functions, variables, data objects, methods, or other elements may give a developer and administrator meaningful feedback regarding the performance of their application, even when the underlying data may be obfuscated. For example, an analysis engine 116 may identify function FOO has behaving in a certain manner and provide feedback that names function FOO. An application developer may recognize function FOO and be able to take action. Such an analysis may be performed in an unsecure manner using obfuscated data elements but with cleartext representations of program elements.

[0081] Cleartext descriptors of application elements may include function names, variable names, data object names, record descriptors, column descriptors, annotations, method names, class names, library names, file names, parameter names, tags, control flow diagrams, and other descriptors. Typically, such descriptors may be

created by a programmer or developer and may reflect the programmer's intent or logic.

[0082] In certain circumstances, such descriptors may reflect confidential information. The confidential information may be the underlying logic or program flow, which may be separate from the confidential nature of the data handled by the application. For example, an application that processes healthcare records may have a proprietary or trade secret method for analyzing a healthcare record. While the healthcare record itself may be confidential under HIPPA or other statutory or regulatory provisions, the methodology of the application may be a separate class of confidential information.

[0083] In cases where such descriptors reflect application logic, third party analysis may be performed under a nondisclosure agreement, privacy arrangement, or other confidentiality provision as the third party may be exposed to the underlying methodology in an application but not be exposed to the data handled by the application.

[0084] Figure 2 is a diagram of an embodiment 200 showing a computer system that transmits obfuscated trace data to a remote device for analysis. Embodiment 200 illustrates hardware components that may deliver the operations described in embodiment 100, as well as other embodiments.

[0085] The diagram of Figure 2 illustrates functional components of a system. In some cases, the component may be a hardware component, a software component, or a combination of hardware and software. Some of the components may be application level software, while other components may be execution environment level components. In some cases, the connection of one component to another may be a close connection where two or more components are operating on a single hardware platform. In other cases, the connections may be made over network connections spanning long distances. Each embodiment may use different hardware, software, and interconnection architectures to achieve the functions described.

[0086] Embodiment 200 illustrates a device 202 that may have a hardware platform 204 and various software components 206. The device 202 as illustrated represents a conventional computing device, although other embodiments may have different configurations, architectures, or components.

[0087] In many embodiments, the device 202 may be a server computer. In some embodiments, the device 202 may still also be a desktop computer, laptop computer, netbook computer, tablet or slate computer, wireless handset, cellular telephone, game console or any other type of computing device.

[0088] The hardware platform 204 may include a processor 208, random access memory 210, and nonvolatile storage 212. The hardware platform 204 may also include a user interface 214 and network interface 216.

[0089] The random access memory 210 may be storage that contains data objects and executable code that can be quickly accessed by the processors 208. In many embodiments, the random access memory 210 may have a high-speed bus connecting the memory 210 to the processors 208.

[0090] The nonvolatile storage 212 may be storage that persists after the device 202 is shut down. The nonvolatile storage 212 may be any type of storage device, including hard disk, solid state memory devices, magnetic tape, optical storage, or other type of storage. The nonvolatile storage 212 may be read only or read/write capable. In some embodiments, the nonvolatile storage 212 may be cloud based, network storage, or other storage that may be accessed over a network connection.

[0091] The user interface 214 may be any type of hardware capable of displaying output and receiving input from a user. In many cases, the output display may be a graphical display monitor, although output devices may include lights and other visual output, audio output, kinetic actuator output, as well as other output devices. Conventional input devices may include keyboards and pointing devices such as a mouse, stylus, trackball, or other pointing device. Other input devices may include various sensors, including biometric input devices, audio and video input devices, and other sensors.

[0092] The network interface 216 may be any type of connection to another computer. In many embodiments, the network interface 216 may be a wired Ethernet connection. Other embodiments may include wired or wireless connections over various communication protocols.

[0093] The client 202 may have an operating system 218 that may execute various applications 254. In some embodiments, an execution environment 220 may execute the applications 254. In either case, the operating system 218 or execution

environment 220 may manage execution of the applications 254 by managing resources consumed by the applications 254 as well as controlling the execution.

[0094] The resources managed by the operating system 218 or execution environment 220 may be memory resources, network resources, input/output resources, processor resources, and other resources. The operating system 218 or execution environment 220 may allocate memory, perform garbage collection, schedule processor availability, prioritize and allocate storage resources, and other functions. In some embodiments, the execution environment 220 may be referred to as a virtual machine.

[0095] Tracers 222 and 224 may operate within the operating system 218 or execution environment 220. The tracers 222 and 224 may monitor the execution of an application 254 and collect various information, including performance data, operational data, debugging data, and other types of information. In many cases, the tracers 222 and 224 may be exposed to sensitive data that may be processed by an application 254.

[0096] Raw trace data 226 may be the data as-collected by the tracers 222 or 224. The raw trace data 226 may include data elements processed by the applications 254, as well as references to application elements such as function names and other descriptors.

[0097] An obfuscator 228 may process the raw trace data 226 to create obfuscated trace data 230. The obfuscator 228 may obfuscate some or all of the raw trace data 226 using various mechanisms. In some cases, only certain elements may be obfuscated while other elements in the raw trace data 226 may remain in a cleartext format.

[0098] In some embodiments, a source code analyzer 252 may create a set of source code annotations 254. The source code annotations 254 may be used to decorate the raw trace data 226 with meaningful function names and other information.

[0099] The source code annotations 254 may be annotations, tags, labels, or other information that may be derived from source code. Such information may be created by a compiler, debugging tool, or other source. In some cases, source code annotations 254 may be created by a dedicated source code analyzer 252.

[00100] A communications agent 232 may transmit the obfuscated trace data 230 to a remote device 240 for processing. The remote device 240 may be located outside of a secure environment which may be protected by a firewall 236 as well as other security measures. The communications agent 232 may pass the obfuscated trace data 230 through a firewall 236 and network 238 to the remote device 240.

[00101] The remote device 240 may operate on a hardware platform 242. The hardware platform 242 may be similar to the hardware platform 204. In some instances, the hardware platform 242 may be a virtual machine, cloud computing system, computing cluster, or some other execution environment.

[00102] A communications agent 246 may receive obfuscated trace data 230 from the device 202 and store the obfuscated trace data 244. An analyzer 248 may perform analyses against the obfuscated trace data 244 to generate various analysis results, which may be debugging and performance information, optimization information, or any other type of analysis results.

[00103] The obfuscated trace data 244 may contain trace data from multiple devices 202. In such embodiments, the trace data from two or more devices may be combined to create a more comprehensive trace data set than what may be created from merely one device.

[00104] In some embodiments, results from the remote device 240 may be transmitted to the device 202 for further inspection and use. In such embodiments, a lookup database 234 may be populated with obfuscated and raw data elements. The lookup database 234 may be used to translate from obfuscated results to more meaningful results when results are received from a remote device 240.

[00105] Figure 3 is a flowchart illustration of an embodiment 300 showing a method for installing and monitoring executable code. Embodiment 300 illustrates the operations of a client device 302 in the left hand column and a remote device 304 in the right hand column.

[00106] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00107] Embodiment 300 illustrates an interaction between a device 302 in a secure location with a remote device 304 which may be outside the secure location. Trace data are gathered on the device 302, obfuscated, and transmitted to the remote device 304. The remote device 304 may process only the obfuscated data to create certain results, which are returned to the device 302 and de-obfuscated.

[00108] On the device 302, an application may be executed in block 306. While the application executes, trace data may be gathered in block 308. Some or all of the parameters may be obfuscated using hashing, encryption, lookup tables, randomization, or other obfuscation techniques.

[00109] The obfuscated data may be transmitted in block 312 to the remote device 304, which may receive the obfuscated data in block 314.

[00110] The remote device 304 may perform analysis on the obfuscated trace data in block 316 to generate analysis results in block 318. The analysis results may be transmitted in block 320 to the device 302, which may receive the analysis results in block 322.

[00111] The device 302 may de-obfuscate the data in block 324 and act on the analysis results in block 326. The de-obfuscating in block 324 may employ a mechanism that determines a raw, original value from results computed from obfuscated data.

[00112] Figure 4 is a diagram illustration of an embodiment 400 showing a parallel tracing system. Embodiment 400 illustrates a high level view of a system that has a distributor that may send units of work to different systems, which may include systems for performance measurements as well as detailed instrumentation or profiling.

[00113] Embodiment 400 may be an example of a system where units of work may be executed in different environments, which may include execution under no instrumentation, performance level instrumentation, and detailed instrumentation. In many cases, an increasing level of instrumentation may cause performance to degrade. However, a more complete understanding about an application may combine both performance and detailed instrumentation results. By collecting trace data from two different environments, the performance related data may be unaffected by the detailed tracing.

[00114] A single unit of work may be analyzed by two different systems. In such systems, the performance results and detailed tracing results may be combined for those units of work. Such systems may tag the tracing results with an identifier that may allow an aggregator to match the results to the same unit of work.

[00115] In other embodiments, a single unit of work may be analyzed only one time. In such systems, results from performance and detailed analyses may be combined to give an overall picture without being able to directly compare individual units of work. Such a picture may be statistically significant when the distribution of workloads to each type of analysis may have a statistically normal distribution, for example.

[00116] A requestor 402 may send a request 404 to a distributor 406. The request 404 may be a workload to be processed by a cluster of execution environments. In one example, the request 404 may be a call to an application programming interface, where the application programming interface may be executed by a computing cluster. In another example, a request 404 may be a workload within a high performance computing system. In still another example, the request 404 may be a function or method call within a computer application.

[00117] The distributor 406 may analyze the request 404 to determine how to route the request 404. In the example of embodiment 400, the request 404 may be routed to an non-instrumented system 410, a performance instrumented system 412, or a detailed instrumented system 414. Other embodiments may have more or fewer systems that may be able to perform the request 404.

[00118] The distributor 406 may have a configuration 408 that may define how the distributor 406 may perform its distribution functions. The configuration 408 may have conditions under which detailed or performance tracing may be performed, as well as conditions defining when no tracing may be performed.

[00119] The conditions 408 may contain filters that limit instrumentation to only a subset of available requests. The filters or objectives may define parameters relating to the input stream, a sampling frequency, or other parameters that may define how and when instrumentation may occur. The objectives may define instrumentation granularity, such a functional component, function, process, memory object, or other level of detail of the data collection. As an example of parameters relating to the input stream, the conditions 408 may indicate that instrumentation may

be performed on requests that contain a specific variable with a specific value or range of values. An example of other types of configuration definitions may define a sample frequency for requests that may be instrumented.

[00120] An non-instrumented system 410 may process the requests 404 with little or no tracing or instrumentation. In many cases, an non-instrumented system 410 may contain minimal instrumentation that may monitor the status of the system or other actions. The non-instrumented system 410 may not generate instrumentation results that may be analyzed with data gathered from other instrumented systems.

[00121] A performance instrumented system 412 may process a request 404 while gathering performance related metrics. The instrumentation may be designed to have minimal impact on the performance of system so that the performance metrics may be considered accurate.

[00122] A detailed instrumentation system 414 may process a request 404 while gathering detailed operational information. In many cases, such systems may trace function calls, gather data objects passed between processes and functions, gather object values as certain points during execution, and other data. In many cases, the detailed instrumented system 414 may be significantly slower than the performance system 412.

[00123] In the example of embodiment 400, three systems are illustrated as processing the requests 404. In some embodiments, such devices may be identical hardware devices with the same or different software components, while in other embodiments, different devices with different hardware or software components may be used. Such systems that are not similar may have specialized hardware or software components designed for high performance, detailed instrumentation, or other function.

[00124] The application output 416 may be received from the various systems that execute the request 404. The application output 416 may be returned to the requestor 402.

[00125] In cases where two systems execute the same request, the application output 416 may be compared to determine whether both systems generated the same output. In some embodiments, the output may be considered reliable or substantiated when two parallel devices generate the same output. When the output from two

systems is not the same, one of the output values may be selected, the values averaged, or some other action taken.

[00126] A results combiner 418 may collect trace data from both the performance instrumented system 412 and the detailed instrumented system 414 and store the results in a set of instrumentation results 420. In some cases, the results combiner 418 may match specific execution runs or characteristics between two or more different trace data. Examples of such matching may be found later in this specification.

[00127] Figure 5 is a diagram of an embodiment 500 showing a network environment in which different execution systems may generate different levels of trace data. Embodiment 500 illustrates hardware components that may implement some of the operations described in embodiment 400, as well as other embodiments.

[00128] The diagram of Figure 5 illustrates functional components of a system. In some cases, the component may be a hardware component, a software component, or a combination of hardware and software. Some of the components may be application level software, while other components may be execution environment level components. In some cases, the connection of one component to another may be a close connection where two or more components are operating on a single hardware platform. In other cases, the connections may be made over network connections spanning long distances. Each embodiment may use different hardware, software, and interconnection architectures to achieve the functions described.

[00129] Embodiment 500 illustrates an example of a system that processes requests received over an external network 504 from a requestor 502. A distributor system 506 may parse the incoming request stream and cause the various execution systems 520 to execute the requests. The application output or response to the request may be passed back to the requestor 502, while any trace data gathered from the execution systems 520 may be stored and used by an analysis system 552.

[00130] Each of the various devices illustrated in embodiment 500 may have a hardware platform. The respective hardware platforms may be similar to the hardware platform 104 in embodiment 100. The devices may be any type of hardware platform, such as a personal computer, server computer, game console, tablet computer, mobile telephone, or any other device with a programmable processor.

[00131] The distributor system 506 may receive and parse an input stream, then direct units of work to various execution systems 520. The distributor system 506 may operate on a hardware platform 508 and contain a distributor 510 that receives work requests via an external interface 512. The distributor 510 may have a tracing configuration 514 and distribution configuration 516 that define how the units of work may be distributed.

[00132] The tracing configuration 514 may define an algorithm, conditions, or other conditions that may define how and when to collect instrumentation data. The tracing configuration 514 may include granularity, sampling rates, sample sizes, data to be collected, and other information. The tracing configuration 514 may also include specific conditions when to trace or not to trace.

[00133] Such conditions may evaluate data elements in a request, as well as data elements from external sources. For example, a condition may cause detailed tracing to happen during evening hours when an input parameter is 'blue'.

[00134] The distribution configuration 516 may define conditions for allocating other units of work. The distribution configuration 516 may define a load balancing algorithm, for example, that allocates work to devices that are lightly loaded while avoiding sending work to devices that are heavily loaded.

[00135] A local network 518 may connect the distributor system 506 with various execution systems 520. The execution systems 520 may have a hardware platform 522 on which an operating system 524 or execution environment 530 may run. An application 534 may be executed on the execution systems 520 to respond to a unit of work, and various tracers may collect data while the application 534 processes the unit of work.

[00136] In some embodiments, the application 534 may execute directly on the operating system 524. In such embodiments, an operating system 524 may have a lightweight tracer 526 for collecting performance related measurements, as well as a detailed tracer 528 that may collect detailed information during application execution.

[00137] In other embodiments, the application 534 may execute in an execution environment 530. The execution environment 530 may be a virtual machine, such as a process virtual machine, that may manage execution and provide various support functions such as memory allocation, garbage collection, process

management, message passing, or other functions. Such execution environments may have a tracer 532.

[00138] The various tracers may be configured using a tracer configuration 536 that may define what information to collect and under what circumstances the information may be collected. In some embodiments, the tracer configuration 536 may be sufficient information to cause a single tracer to behave as a performance level tracer or as a detailed tracer.

[00139] The trace data 538 may be locally collected tracer output, which may be passed to the analysis system 552.

[00140] In some embodiments, the distributor 540 may be located within an execution system 520. In one such embodiment, the system 520 may execute an application 534, which may be executed in part by distributing workload items to multiple processing instances. One of the processing instances may be a detailed instrumented instance, while another processing instance may be a performance tracing instance. In such a case, the operation of embodiment 400 may be performed on a single device.

[00141] An analysis system 552 may collect the trace data 538 from various execution systems 520 to gather the results in a centralized trace data 544. The analysis system 552 may operate on a hardware platform 542 which may have a data store for the trace data 544, as well as a combiner 546, an analyzer 548, and an optimizer 550.

[00142] The analysis system 552 may gather and aggregate trace data from both performance and detailed tracers. The combiner 546 may create a joined set of results. The analyzer 548 may perform various analyses of the results, such as reports, alerts, or other output. The optimizer 550 may generate optimized settings for the application 534 or other optimizations.

[00143] Figure 6 is a flowchart illustration of an embodiment 600 showing a method for application execution and parallel instrumentation. Embodiment 600 illustrates the operations of a distributor and combiner 602 in the left hand column, a performance environment 604 in the center column, and a detailed instrumented environment 606 in the right hand column. The distributor and combiner 602 may represent the operations of a distributor 406 and combiner 418, while the performance environment 604 and detailed execution environment 606 may represent the

operations of a performance instrumented system 412 and detailed instrumented system 414 of embodiment 400, respectively.

[00144] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00145] In block 608, an application may be initiated by the distributor and combiner 602, which may cause the application to begin execution in the performance environment 604 in block 610 and in the detailed execution environment 606 in block 612.

[00146] The operations of blocks 608 through 612 illustrate an embodiment where an application may be configured to execute, then workload items may be transmitted to the application for processing. In such embodiments, the workload items may be data items consumed by the applications. In other embodiments, a workload item may be executable commands that may be passed to the various environments. In such embodiments, the operations of blocks 608 through 612 may not be performed.

[00147] An input stream may be received in block 614. The input stream may be parsed to identify a work unit in block 616. The work unit may be a block of data, executable code, or other workload that may be processed by an execution environment.

[00148] A distributor may analyze the work unit in block 618 to determine whether the work unit may be processed using detailed instrumentation or not. If the distributor selects detailed instrumentation in block 618, the workload may be transferred to the detailed instrumentation environment 606 to be executed in block 620. If the distributor does not select detailed instrumentation in block 618, the workload may be transferred to the performance environment 604 to be executed in block 622.

[00149] In some cases, the distributor and combiner 602 may send the same workload item to both the performance environment 604 and detailed execution environment 606.

[00150] The distributor may create an identifier for the work unit. In some cases, a work unit may include an identifier within the request, such as a sequence number or other identifier that may be used by a requestor to match application results with the request. In some cases, a timestamp, identification code, or other identifier may be created by the distributor and used to correlate results data from two or more trace data gather from different execution environments.

[00151] During execution in the performance environment 604, some tracing results may be collected, which may be transmitted in block 624 to the distributor and combiner 602 and received in block 626. Similarly, the detailed instrumented environment 606 may generate tracing results that may be transmitted in block 628 and received by the distributor and combiner 602 in block 630.

[00152] The results may be combined in block 632 and stored in block 634. The process may return to block 614 to handle another unit of work.

[00153] An example of a method to combine trace data from a performance environment 604 and detailed instrumented environment 606 may be illustrated in embodiment 700.

[00154] Figure 7 is a flowchart illustration of an embodiment 700 showing a method for combining results from trace runs with different levels of granularity. Embodiment 700 illustrates one example of the combining operations of block 632 of embodiment 600 or the results combiner 418 of embodiment 400.

[00155] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00156] Performance trace data may be received in block 702 and detailed trace data may be received in block 704. Within each set of results, one or more identifiers may be present. Such identifiers may be identified in block 706.

[00157] The identifiers may be any item that may be used to correlate the data between two different tracing operations of an application or work unit. In some cases, each unit of work may have an identifier, which may be used to match detailed

and performance trace data when the unit of work has been executed by both types of tracers in separate runs.

[00158] In other case, a time stamp, input data value, or other information may be used as an identifier.

[00159] For each identifier in block 708, the performance results may be gathered in block 710 and the detailed results may be gathered in block 712. The two sets of results may be combined in block 714 and stored in block 716.

[00160] After combining both sets of results, analyses may be performed on the larger set of data in block 718.

[00161] Figure 8 is a diagram illustration of an embodiment 800 showing a system that performs tracing with a workload distributor. Embodiment 800 illustrates a general process whereby a distributor may provide workload distribution, and may also determine how and when tracing may occur during execution.

[00162] Embodiment 800 may represent a managed computing environment, such as a cluster computing system or other system where multiple devices may be used to deliver an application. While a conventional cluster or load balanced environment may be used as the example in embodiment 800, the same principles may be applied to any computational system where workload may be partitioned and distributed to multiple instances, threads, processors, devices, or other compute elements.

[00163] An application input stream 802 may be sent to a distributor 804, which may partition out work items to various non-instrumented systems 808 and instrumented systems 812. The application output 816 may be produced by either type of systems, but the instrumented systems 812 may produce trace data that may be stored in a results database 818.

[00164] The worker systems include the non-instrumented systems 808 and instrumented systems 812. The worker systems may be capable of processing work units, which may be any element of an application. In some embodiments, the application may receive requests items on an application programming interface, then process each request as an individual work item. In such embodiments, incoming requests may be data items that are processed individually and independently.

[00165] In some embodiments, the application may be capable of parallel execution, with each work item being an independent computational element that may

or may not interact with other work items. In such embodiments, incoming requests may be executable code or a combination of executable code and data objects to be processed by the executable code.

[00166] The distributor 804 may use a configuration 806 to define how to allocate the work items across the worker systems. The configuration 806 may define load balancing algorithms and parameters, as well as the data collection configuration.

[00167] The data collection configuration may define how and when data items may be collected by any instrumentation on a worker system. In many cases, the data collection configuration may define specific objectives, such as data items to collect and conditions for collecting the data items.

[00168] The distributor 804 may create a tracer configuration 820 that may configure the instrumentation on an instrumented system 812 to collect specific data. Some embodiments may create specific or customized tracer configurations 820 for each work element. Such embodiments may allow the system to change the instrumentation with a tracer configuration 820 from run to run, allowing fine-tuned control over the data collection.

[00169] Because both the instrumentation and load balancing may be incorporated into the distributor 804, the instrumentation may become a factor in overall load balancing. For example, when the load on the system is heavy and there may be few resources available, the distributor 804 may be able to reduce the instrumentation so that the overall system performance may not suffer. Similarly, the distributor 804 may increase instrumentation during slack periods and there may be an excess of resources.

[00170] In systems that may implement a tracer configuration 820, each worker system may be configured as an instrumented or non-instrumented system merely by updating the tracer configuration 820 for a particular work item. In some such systems, all of the worker systems may be identically configured.

[00171] Some worker systems may have additional instrumentation capabilities that other, non-instrumented systems may not have. For example, an instrumented system 812 may have different or additional processors, memory, storage, network connectivity and even additional software resources that may support instrumentation. In such systems, the various worker systems may not be identical.

[00172] A worker manager 822 may manage the various worker systems by registering the worker systems, determining the availability of the worker systems, and other functions. In many embodiments, the worker manager 822 may deploy a two way authentication mechanism that may allow the distributor 804 to authenticate to the worker systems and vice versa.

[00173] The worker manager 822 may collect status information by periodically querying the worker systems or have other regular. The status information may include the capabilities of the worker system, which may include the hardware and software capabilities and configuration, as well as the current load or capacities of the worker system. Some elements may change in real time, such as the availability of the system to process a new request, while other elements may be more static, such as the hardware configuration.

[00174] The authentication mechanisms may help ensure that the various devices are supposed to be communicating with each other. When a worker device authenticates itself to a distributor, the distributor may rely on the authentication to assure that the worker is not a malicious device. When the distributor authenticates itself to the worker device, the worker device may rely on the authentication to assure that the distributor has the permission or authority to send work to the worker device. The authentication mechanisms may also be deployed for other scenarios, including instrumentation-as-a-service scenarios.

[00175] Figure 9 is a diagram of an embodiment 900 showing a network environment in which a workload distributor may send work units to various worker devices. Embodiment 900 illustrates hardware and software components that may implement some of the operations described in embodiment 800, as well as other embodiments.

[00176] The diagram of Figure 9 illustrates functional components of a system. In some cases, the component may be a hardware component, a software component, or a combination of hardware and software. Some of the components may be application level software, while other components may be execution environment level components. In some cases, the connection of one component to another may be a close connection where two or more components are operating on a single hardware platform. In other cases, the connections may be made over network

connections spanning long distances. Each embodiment may use different hardware, software, and interconnection architectures to achieve the functions described.

[00177] Embodiment 900 illustrates an environment in which multiple worker systems may perform portions of an application as determined by a distributor device. The distributor 902 may receive a stream of workload items, divide the incoming stream into units of work, determine which worker device may process each unit of work, and cause those units of work to be performed.

[00178] The distributor 902 may also determine what type of instrumentation may be performed on a given unit of work. The instrumentation may include various types of tracing, data collection, performance monitoring, or other data that may be used for diagnosis, debugging, administrative monitoring, optimization, or other uses.

[00179] The instrumentation may be configurable by the distributor 902. In some cases, the distributor 902 may be capable of routing work units to worker devices that may be preconfigured to perform specific types of tracing or instrumentation. For example, one of the instrumented systems 932 may be preconfigured to perform a specific set of data collection in addition to executing a work unit. In such an example, the distributor 902 may determine when to send a work unit to the instrumented system 932 and when to send another work unit to a worker 924 which may not have instrumentation configured.

[00180] In another example, the distributor 902 may transmit a configuration file or other descriptor to an instrumented system 932, where the configuration file may contain specific data items to collect, tests to perform, or other data collection activities. In such an embodiment, the instrumentation or tracers on the instrumented systems 932 may be configurable. In some cases, such a system may be able to execute a work unit with little or no instrumentation, then switch to a high level of instrumentation for the next work unit, as defined in a configuration file.

[00181] Each of the various devices illustrated in embodiment 900 may have a hardware platform. The respective hardware platforms may be similar to the hardware platform 104 in embodiment 100. The devices may be any type of hardware platform, such as a personal computer, server computer, game console, tablet computer, mobile telephone, or any other device with a programmable processor.

[00182] The distributor 902 may have a hardware platform 904 on which various software components may operate. A distributor 906 may receive an incoming workload stream from a load receiver 908 and determine which worker device will execute the work unit. The distributor 906 may use a load balancing configuration 910 that may define load balancing objectives, algorithms, or other definition for managing a quality of service or other factor. The distributor 906 may refer to a worker database 912 that may include the availability and status of the various worker devices.

[00183] The distributor 906 may balance the workload over multiple devices, including instrumented systems 932 and general workers 924. The workload balancing may use multiple devices in parallel to process a workload that may be larger than the capacity of a single device. For example, a large web-scale application may be processed by many devices, which may scale into several hundred server computers in some cases. The load balancing aspect of the distributor 906 may attempt to divide the workload and distribute the workload to available devices.

[00184] A tracer configuration 914 may define a set of instrumentation objectives, which may include data to be collected, sample rates, and many other factors.

[00185] In many cases, the instrumentation objectives may be in tension with the load balancing objectives. In general, instrumentation and data collection may come at some computational cost, meaning that as the instrumentation is increased, the performance of a system may decrease. In a high speed, high throughput environment, the distributor 906 may balance the instrumentation objectives against the processing capacity to handle the incoming workload. In some instances, the distributor 906 may scale back the instrumentation objectives during high loads so that a quality of service metric for the overall system may be met. In other instances, the distributor 906 may allow the quality of service metric to be missed so that instrumentation objectives may be met. The decisions made by the distributor 906 may be defined in the various configuration files.

[00186] A tracer manager 918 may manage the instrumentation to generate the overall objectives for data collection. The tracer manager 918 may cause different data elements to be collected from various work units or instrumented systems 932,

then collect the data. In some cases, the tracer manager 918 may perform first level aggregation, summaries, or other initial processing.

[00187] The tracer manager 918 may use an artificial load generator 916 to create or modify work units for testing and data collection. For example, the artificial load generator 916 may create a work unit that stresses a specific portion of an application. When such a work unit is performed, the instrumentation may collect data for the work unit. Such a work unit may produce application results that may be discarded by the instrumented system 932.

[00188] The distributor 902 may include a worker status collector 920 that may communicate with each worker device to determine a current status. The status may indicate whether the device may be available to accept work units. In some cases, the status may include statistics, such as excess capacity, current workload, or other performance metrics.

[00189] A worker manager 956 may manage available worker devices by adding and removing devices into a pool of managed devices. The worker manager 956 may allow new devices to connect, authenticate, and be added to the pool of managed devices. An administrator may use the worker manager 956 to monitor individual devices, configure devices, and add and remove devices to the pool.

[00190] In many embodiments, a worker manger 956 may issue authentication keys to worker devices. The authentication keys may be part of a public/private encryption key system, where the private key may be stored on a device and used to secure a communication, and a public key transmitted to a receiving device to decrypt the communication. Such systems may be one example of a system for authenticating between devices, and other embodiments may use other systems.

[00191] The workers 924 may operate on a hardware platform 926 to execute an application 930 within an execution environment 928. The application 930 may be a preinstalled set of executable code that processes work items from the distributor 906. In some cases, the application 930 may execute within an execution environment 928, which may be an operating system, virtual machine, framework, or other supporting software component.

[00192] The workers 924 may have an authentication key 954 which may be used to authenticate communication with the distributor 902. The key 954 may be

any type of authentication component, such as a public/private encryption key set or other component.

[00193] The instrumented system 932 may be similarly configured as the workers 924. A hardware platform 934 may support an execution environment 936 that executes the application 938. The instrumented system 932 may include a tracer 940, which may or may not be configurable by the distributor 902 during execution. The instrumented system 932 may also include a key 952 for authentication with the distributor 902.

[00194] In some embodiments, the instrumented system 932 may be identical hardware and software configuration as the workers 924. Other embodiments may have different hardware or software configurations between the instrumented system 932 and workers 924.

[00195] The tracer 940 may collect instrumentation or trace data and transmit those data to an analysis system 942.

[00196] The analysis system 942 may have a hardware platform 944 where a data collector 946 may collect trace data 948 from various tracers. An analyzer 950 may process the tracer for analysis, optimization, summarization, or other functions.

[00197] The example of the distributor 902 and other components illustrate devices that may have multiple functions. In different embodiments, each of the various illustrated components may be deployed on a separate device or group of devices. For example, the worker manager 956, worker status collection 920, tracer manager 918, and other functions may be deployed on individual devices or groups of devices.

[00198] Figure 10 is a flowchart illustration of an embodiment 1000 showing a method for configuring a pool of worker devices, then operating the worker devices in a load balanced and instrumentation balanced manner. Embodiment 1000 may illustrate one example of a process that may be implemented in the systems illustrated in embodiments 800 and 900, among others.

[00199] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous

manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00200] Embodiment 1000 illustrates an operating sequence for establishing and administering a pool of worker resources in block 1002, then an operational mode in block 1004 for parsing an input stream and distributing work units to the worker resources. In the example of embodiment 900, the worker resources may be individual devices, but in other embodiments, the worker resources may be any resource used to process work units. In some cases, the worker resources may be sub-device resources, such as processors, schedulers, threads, or other objects. In other cases, the worker resources may be multiple devices acting together, such as clusters, managed services, or other resource groups.

[00201] In an administrative mode in block 1002, the various execution environments may be identified in block 1006. The execution environment may refer to any resource used for processing a work unit. In many cases, the execution environment may be a device, processor, or other computation worker.

[00202] For each execution environment in block 1008, an attempt may be made to communicate and authenticate the environment in block 1010. If the authentication is not successful in block 1012, the environment may be marked as unavailable in block 1014.

[00203] After successfully authenticating in block 1012, the configuration and availability of the environment may be tested in block 1016. If the configuration and availability are not OK in block 1018, the environment may be marked as unavailable in block 1020. When the environment has successfully authenticated in block 1012 and the configuration and availability are OK in block 1018, the execution environment may be marked as available in block 1022.

[00204] The process of blocks 1006-1022 may represent an administrative function that may be performed prior to distributing work to the various execution environment. In some cases, the process of blocks 1006-1022 may be performed in parallel with the operational mode of block 1004. In such cases, the operations of the administrative mode may be an ongoing and repeated check of the various execution environments.

[00205] The operational mode of block 1004 may begin by receiving tracer configuration and load balancer configuration in block 1024.

[00206] The input stream may be received in block 1026. The input stream may be parsed in block 1028 to identify work items. For a given work item, a determination may be made in block 1030 identifying a device to process the work item. In some cases, a tracer configuration may be created in block 1030 that defines any data collection parameters. Once the determination is made in block 1030, the work item may be transmitted to the selected execution environment in block 1032. The process may return to block 1026 to process the next work item.

[00207] Figure 11 is a diagram illustration of an embodiment 1100 showing a tracing as a service. Embodiment 1100 illustrates one configuration of a system where a remote service 1104 may integrate with systems on a customer premise 1102, where the remote service 1104 may provide an instrumented execution environment to process workloads and generate trace data.

[00208] A customer premise 1102 may execute an application under control of a user. The customer premise 1102 may be a physical premise, such as a building or business to which the user may have access. In some cases, the customer premise 1102 may include one or more computers that may be owned by and located at a third party's premise, but under the control of the user. An example of such a system may be a cloud hosted execution system where a user may purchase computing resources. The resources may be owned by a third party, but the user may control how those resources may be deployed.

[00209] The remote service 1104 may receive work items and execute those work items using an instrumented environment 1114. The instrumented environment 1114 may have various hardware and software components that may capture various trace data while a work item executes. In some cases, the instrumented environment may include tools, measuring algorithms, probes, and other components that may be difficult or costly to install, manage, execute, or otherwise deploy.

[00210] Within the customer premise 1102, an application input stream 1106 may be passed to a distributor 1108. The distributor 1108 may parse work items from the input stream and pass the work items to various execution environments 1110. The output of the execution environments 1110 may be application output 1112.

[00211] The distributor 1108 may transfer some or all of the work items to an instrumented environment 1114, which may be part of a remote service 1104. In some cases, multiple instances of the instrumented environment 1114 may be used.

[00212] A load generator 1128 may create test loads that may be injected into the input stream 1106. The test loads may be data that may be processed by the instrumented environment 1114 to exercise an application. In many cases, the test loads may exercise an application in a more comprehensive manner than a typical or random input stream. Such test loads may perform unit tests or other tests that may be designed to exercise various corner cases and conditions. In some cases, the test loads may subject an application to large loads that may stress the performance of the system. Such test loads may help identify performance bottlenecks and measure overall throughput or response time during peak loading times.

[00213] The remote service 1104 may include an authentication, authorization, and accounting system 1116, which may manage various administrative aspects of the remote service 1104. An administrative user interface may permit a user to create an account, define a payment mechanism, and administer the instrumented environments 1114.

[00214] An instrumentation marketplace 1122 may be a website or other interface through which a user may browse preconfigured instrumented systems 1124 and preconfigured load generators 1126. Each of the various preconfigured systems may be customized for specific types of data collection. Some preconfigured systems may have additional features, algorithms, or capabilities that may not be available on other preconfigured systems. As such, some preconfigured systems may have different cost structures than other preconfigured systems.

[00215] A user may be able to select a preconfigured instrumented system 1124 and preconfigured load generator 1126 as a starting point for configuring a test regime for a given application. In some embodiments, a user may select a preconfigured system then add, remove, or edit various settings to achieve a specific objective.

[00216] In some embodiments, a user may be able to save a preconfigured instrumented system in the instrumentation marketplace 1122 for reuse. In some such embodiments, a third party may be able to upload their own instrumented system for sale in the instrumentation marketplace 1122.

[00217] Figure 12 is a diagram of an embodiment 1200 showing a network environment in which a remote service may provide instrumentation or tracing.

Embodiment 1200 illustrates hardware and software components that may implement some of the operations described in embodiment 1100, as well as other embodiments.

[00218] The diagram of Figure 12 illustrates functional components of a system. In some cases, the component may be a hardware component, a software component, or a combination of hardware and software. Some of the components may be application level software, while other components may be execution environment level components. In some cases, the connection of one component to another may be a close connection where two or more components are operating on a single hardware platform. In other cases, the connections may be made over network connections spanning long distances. Each embodiment may use different hardware, software, and interconnection architectures to achieve the functions described.

[00219] Each of the various devices illustrated in embodiment 1200 may have a hardware platform. The respective hardware platforms may be similar to the hardware platform 104 in embodiment 100. The devices may be any type of hardware platform, such as a personal computer, server computer, game console, tablet computer, mobile telephone, or any other device with a programmable processor.

[00220] Embodiment 1200 may illustrate an environment in which instrumentation systems may be provided from a remote service, then added to the computational pipeline of a user's system. The instrumentation system may include load generators as well as data collectors which may operate in concert with a user's application to collect various data about the application during execution.

[00221] A user may interact with the remote service through an administrative user interface. The user may be able to perform various administrative tasks, such as establishing an account and a method for payment, as well as to select and configure test components that may integrate into the user's application. The remote service may include an instrumentation marketplace in which a user may browse various preconfigured load generators and preconfigured instrumented workers. Once selected, the user may be able to configure or customize a component, then manage how the component may be deployed.

[00222] A user's system may include a workload distributor 1202 and various workers 1212 connected within an internal network 1210. The workload distributor 1202 may receive units of work for an application, then distribute the work units to

various workers 1212. An example of such a system may be a cluster work environment.

[00223] In the example of embodiment 1200, the user's system is illustrated as multiple devices that each may contribute to the execution of a large application. In other embodiments, the components may be deployed on a single device.

[00224] The distributor 1202 may operate on a hardware platform 1204 that may include a distributor 1206. The distributor 1206 may be a software component that may receive an input stream, parse the input stream into work units, then cause the work units to be executed on the various workers 1212.

[00225] The distributor 1202 may include a configuration 1207 which may include both load balancing and instrumentation objectives. The configuration 1207 may also include information that may be used to distribute some or all of the work units to one or more instrumented workers 1222, which may execute the workloads and collect data about the execution.

[00226] An authentication system 1208 may enable the distributor 1202 to establish trusted and, in some cases, secure communications with remote system components.

[00227] The workers 1212 may include a hardware platform 1214 and an execution environment 1216 that may execute the various work items. The workers 1212 in embodiment 1200 may be connected to the distributor 1202 through an internal network 1210. In many systems, connections within an internal network 1210 may be considered trusted and secure because of a firewall 1218 and other security measures. As such, the workers 1212 may be deployed without an authentication system.

[00228] The firewall 1218 may define a boundary between devices directly under a user's control and devices or services that may be provided by a third party. In some embodiments, the various instrumentation components may be available across an external network 1220, which may include the Internet. In many cases, the various remote services may be made available to many different users.

[00229] A third party may provide load generation and instrumentation services to the user by establishing a connection with a distributor 1206 within the user's system.

[00230] The distributor 1206 may operate at a location in an application where instrumentation may be desired. In some cases, the distributor 1206 may be a function call or other instruction that may be inserted into an application. Such an instruction may be added to a user's application by a programmer.

[00231] The instrumented workers 1222 may execute a portion of an application as defined by the distributor 1206. The instrumented workers 1222 may have a hardware platform 1224 on which an execution environment 1226 may execute work units from the distributor 1206. While executing the work unit, a tracer 1228 may collect data, which may be analyzed by an analyzer 1230. Not shown in embodiment 1200 may be a separate device that may collect and store trace data.

[00232] The instrumented workers 1222 may include an authentication system 1232, which may include a key 1234. The key 1234 may be any type of token, key, passphrase, or other item that may be used to authenticate the instrumented worker 1222 to the authentication system 1208 on the distributor 1202. In some cases, the key 1234 may be a set of pubic/private encryption keys.

[00233] A load generator 1260 may be another instrumentation component that may be configured and deployed as a remote service. The load generator 1260 may generate artificial loads or other inputs that may be performed by the application under test. In some cases, the load generator 1260 may create unit tests or other inputs that may exercise an application. In other cases, the load generator 1260 may generate large loads that may exercise an application to determine performance bottlenecks or other limits to performance.

[00234] The load generator 1260 may have a hardware platform 1262 with a load generator 1264. A configuration 1266 may define how the load generator 1264 may operate, including the type of data to generate along with the timing, frequency, and other operational aspects.

[00235] An authentication system 1268 may authenticate the load generator 1260 to the distributor 1202. The authentication system 1268 may include one or more keys 1270 for communicating with the distributor 1202.

[00236] An administrative server 1236 may perform many of the setup, configuration, and management operations to deploy various instrumented workers 1222 and load generators 1260. The administrative server 1236 is illustrated as

operating on a single hardware platform 1238, although other embodiments may deploy the various components on different platforms.

[00237] An administrative user interface 1240 may be a website, application, or other user interface through which a user may perform many administrative tasks. A user may establish an account and create various authentication components using an authentication and authorization system 1242. A key generator 1246 and key database 1244 may respectively create and store the various authentication keys that may be deployed to the instrumentation components and the user's system.

[00238] An accounting system 1240 and payment system 1250 may be a mechanism through which a remote service may collect monies for operation. The accounting system 1240 may identify each usage of the various instrumentation components, and the payment system 1250 may transfer money from the user to the service provider in exchange for the use of the system.

[00239] Many different payment schemes may be deployed to bill a user for the remote service. For example, a monthly subscription may pay for one or more instrumented systems. In another example, the instrumented systems 1222 and load generator 1260 may be billed on a processor cycle basis, per compute hour, per storage consumed, or other basis.

[00240] An instrumentation marketplace 1254 may be an interface through which a user may browse various preconfigured instrumented workers 1256 and preconfigured load generator 1258. The user may then be able to select and further configure a preconfigured component before deploying the component.

[00241] Figure 13 is a flowchart illustration of an embodiment 1300 showing a method for configuring instrumentation systems. Embodiment 1300 illustrates one example of a process of a user interacting with an administrative server to configure, deploy, and manage instrumentation systems.

[00242] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00243] Embodiment 1300 illustrates a method by which a user may set up and deploy instrumentation systems. The instrumentation systems may be instrumented worker devices, load generators, or other components.

[00244] A user may begin a session in block 1302, establish a user account in block 1304, and establish a payment method in block 1306. The user account may allow the user to log in at a later time and add, remove, and edit the operation of the various instrumentation systems. The payment method may be the mechanism through which payment may be made to a third party that provides the instrumentation services.

[00245] The user may browse preconfigured instrumentation systems in block 1308. The preconfigured systems may have varying capabilities. For example, one instrumented execution environment may have performance monitoring capabilities while another instrumented execution environment may have process or call tracing capabilities. In another example, one load generator may be configured for producing HTTP requests while another load generator may be configured for TCP/IP requests.

[00246] After selecting an instrumentation system in block 1310, data to be collected may be defined in block 1312, as well as various collection parameters and logic in block 1314. The configuration variables defined in blocks 1312 and 1314 may be stored in block 1316 to enable an instrumentation system to be deployed in block 1318.

[00247] The data to be collected in block 1312 may define specific parameters, types of parameters, or other information regarding data collection. In the case of a load generator, the parameters of block 1312 may define the load to be produced, which may be coordinated with the data collection performed by a corresponding instrumented execution environment.

[00248] The collection parameters and logic defined in block 1314 may define the conditions under which data may be collected. The conditions may be events, parameter values, timeframe, sampling rates, or other definitions that may define when data may be collected. In some cases, the conditions may be interpreted by a distributor to determine which work units to transmit to an instrumented execution environment.

[00249] After defining the data to be collected and when and how the data may be collected, the configuration may be stored in block 1316. In some cases, the stored configuration may be added as another preconfigured instrumented system in the instrumentation marketplace. Such a storage may be made accessible to the public at large or may be restricted to only the user who created the configuration.

[00250] The instrumented system may be deployed in block 1318 to interface with a user's system.

[00251] If another system is to be configured in block 1320, the process may return to block 1308. Once all systems are deployed, a user may browse and view deployed systems in block 1322. The user may select a system in block 1324 and, if the user wishes to make changes to the system in block 1326, the process may return to block 1312.

[00252] Figure 14 is a flowchart illustration of an embodiment 1400 showing a method for deploying an instrumented system. Embodiment 1400 illustrates one example of the deployment of an instrumented system, such as may be performed in block 1318 of embodiment 1300.

[00253] Other embodiments may use different sequencing, additional or fewer steps, and different nomenclature or terminology to accomplish similar functions. In some embodiments, various operations or set of operations may be performed in parallel with other operations, either in a synchronous or asynchronous manner. The steps selected here were chosen to illustrate some principles of operations in a simplified form.

[00254] Embodiment 1400 may illustrate one example of a process that may be performed to deploy an instrumented system. The instrumented system may be a load generator, instrumented worker, or other component.

[00255] A configuration file for the instrumented system may be created in block 1402, and the instrumented system may be instantiated in block 1404. In some embodiments, each instrumented system may be a virtual machine or other component that may be instantiated and managed within a datacenter environment.

[00256] Authentication keys may be created for the system in block 1406 and the keys may be transmitted to the system in block 1408. In blocks 1406 and 1408, any protocols or other configuration may be performed to connect to a distributor or to an administrative system.

[00257] An attempt may be made in block 1410 to connect to a distributor on a customer's system. If the connection is not successful in block 1412, the distributor may be installed and configured by identifying the instrumentation point in block 1414 and adding the distributor to the customer system in block 1416. In many cases, a programmer may add a function call or make other changes to the customer's application to add the distributor. The distributor may be configured in block 1418 to communicate with the instrumented system. The process may proceed to block 1420.

[00258] After the distributor is configured in block 1418 or there is success in connecting in block 1412, the instrumentation system may be added to the distributor as an available system in block 1420. The authentication mechanism may be established in block 1422 and the system may start operation in block 1424.

[00259] The foregoing description of the subject matter has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the subject matter to the precise form disclosed, and other modifications and variations may be possible in light of the above teachings. The embodiment was chosen and described in order to best explain the principles of the invention and its practical application to thereby enable others skilled in the art to best utilize the invention in various embodiments and various modifications as are suited to the particular use contemplated. It is intended that the appended claims be construed to include other alternative embodiments except insofar as limited by the prior art.

CLAIMS

What is claimed is:

1. A method performed by a computer processor, said method comprising:

receiving an input stream for an application, said application being executed on a plurality of execution environments, said input stream comprising work elements being discrete units of work for said application;

receiving a status from each of plurality of said execution environments, said status comprising availability to perform work;

receiving a configuration definition defining conditions for executing said discrete units of work on said second execution environment, said configuration definition comprising a load balancing objective and a data gathering objective; and

parsing said input stream to identify said discrete units of work, determining a first data gathering objective for a first work unit, and causing said first work unit to be executed on a first execution environment with said first data gathering objective.

2. The method of claim 1 further comprising:

transmitting said first data gathering objective to said first execution environment.

3. The method of claim 2 further comprising:

determining a second data gathering objective for a second work unit, said second data gathering objective;

determining that said plurality of execution environments are near capacity;

changing said second data gathering objective to a limited second data gathering objective; and

causing said second work unit to be executed with said limited second data gathering objective.

4. The method of claim 3, said second work unit being executed by said first execution environment.

5. The method of claim 4, said first data gathering objective being different from said limited second data gathering objective.

6. The method of claim 5 further comprising:

transmitting a first authentication mechanism to said first execution environment.

7. The method of claim 6 further comprising:

receiving a second authentication mechanism from said first execution environment.

8. The method of claim 7 further comprising:

said second authentication mechanism being received as part of said status.

9. The method of claim 2 further comprising:

determining a second data gathering objective for a second work unit, said second data gathering objective;

determining that said plurality of execution environments have excess capacity;

changing said second data gathering objective to an extensive second data gathering objective; and

causing said second work unit to be executed with said extensive second data gathering objective.

10. The method of claim 9, said second work unit being executed by said first execution environment.

11. The method of claim 2 further comprising:

causing said first work unit to be executed on a second execution environment with a second data gathering objective.

12. The method of claim 11, said second data gathering objective being different from said first data gathering objective.

13. The method of claim 1 further comprising:

determining that said first execution environment has preconfigured data gathering capabilities that are within said first data gathering objective.

14. The method of claim 13 further comprising:

determining that a second execution environment does not have said preconfigured data gathering capabilities and determining that said second execution environment will not execute said first work item.

15. The method of claim 14, said status for said first execution comprising an indicator for said preconfigured data gathering capabilities.

16. The method of claim 1, said configuration definition comprising a target quality of service.

17. A system comprising:

a processor;

a load receiver that receives an application workload to execute, said application workload comprising work units;

a worker status collector that receives status information from a plurality of execution environments;

a distributor executing on said processor that:

receives a configuration definition defining conditions for executing said discrete units of work on said second execution environment, said configuration definition comprising a load balancing objective and a data gathering objective;

parses said input stream to identify said discrete units of work;

determines a first data gathering objective for a first work unit; and

causes said first work unit to be executed on a first execution environment with said first data gathering objective.

18. The system of claim 17, said distributor that further:

transmits said first data gathering objective to said first execution environment.

19. The system of claim 18, said distributor that further:

determines a second data gathering objective for a second work unit, said second data gathering objective;

determines that said plurality of execution environments are near capacity;

changes said second data gathering objective to a limited second data gathering objective; and

causes said second work unit to be executed with said limited second data gathering objective.

20. The system of claim 19, said second work unit being executed by said first execution environment.

21. The system of claim 20, said first data gathering objective being different from said limited second data gathering objective.

22. The system of claim 21, said distributor that further:

transmits a first authentication mechanism to said first execution environment.

23. The system of claim 22, said distributor that further:

receives a second authentication mechanism from said first execution environment.

24. The system of claim 23, said second authentication mechanism being received as part of said status.

25. The system of claim 18, said distributor that further:

determines a second data gathering objective for a second work unit, said second data gathering objective;

determines that said plurality of execution environments have excess capacity;

changes said second data gathering objective to an extensive second data gathering objective; and

causes said second work unit to be executed with said extensive second data gathering objective.

26. The system of claim 25, said second work unit being executed by said first execution environment.

27. The system of claim 18, said distributor that further:

causes said first work unit to be executed on a second execution environment with a second data gathering objective.

28. The system of claim 27, said second data gathering objective being different from said first data gathering objective.

29. The system of claim 17, said distributor that further:

determines that said first execution environment has preconfigured data gathering capabilities that are within said first data gathering objective.

30. The system of claim 29, said distributor that further:

determines that a second execution environment does not have said preconfigured data gathering capabilities and determines that said second execution environment will not execute said first work item.

31. The system of claim 30 said status for said first execution comprising an indicator for said preconfigured data gathering capabilities.

32. The system of claim 17, said configuration definition comprising a target quality of service.

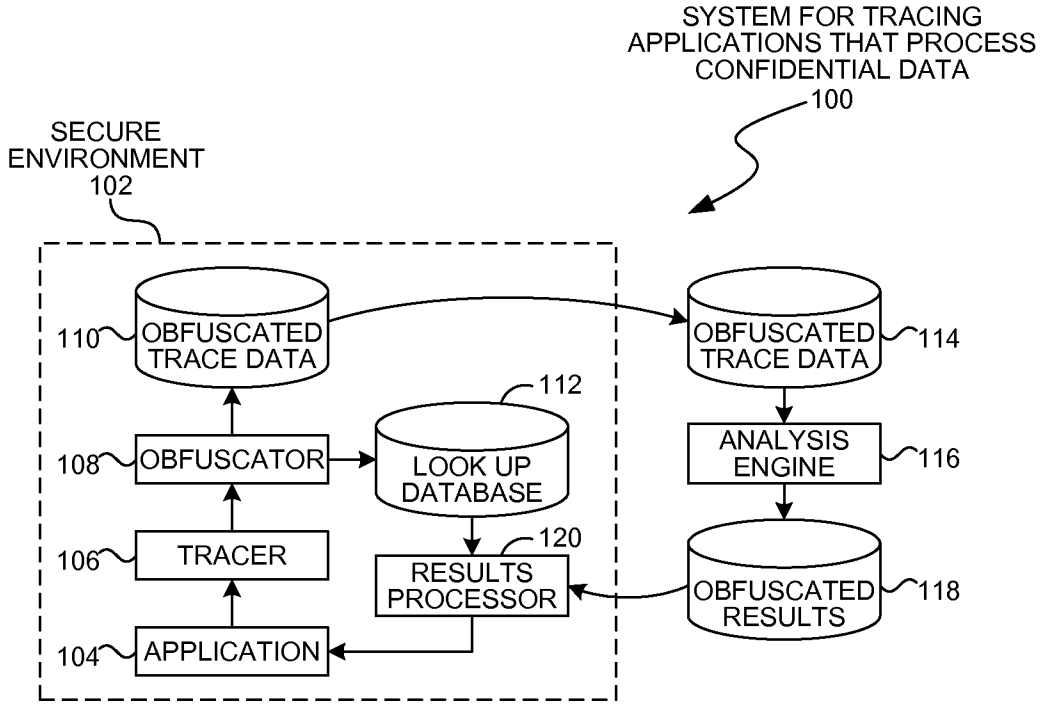
SYSTEM FOR TRACING APPLICATIONS THAT PROCESS CONFIDENTIAL DATA
100
SECURE ENVIRONMENT
102
110
OBFUSCATED TRACE DATA
112
LOOK UP DATABASE
108
OBFUSCATOR
106
TRACER
104
APPLICATION
120
RESULTS PROCESSOR
OBFUSCATED TRACE DATA
114
ANALYSIS ENGINE
116
OBFUSCATED RESULTS
118

FIG. 1

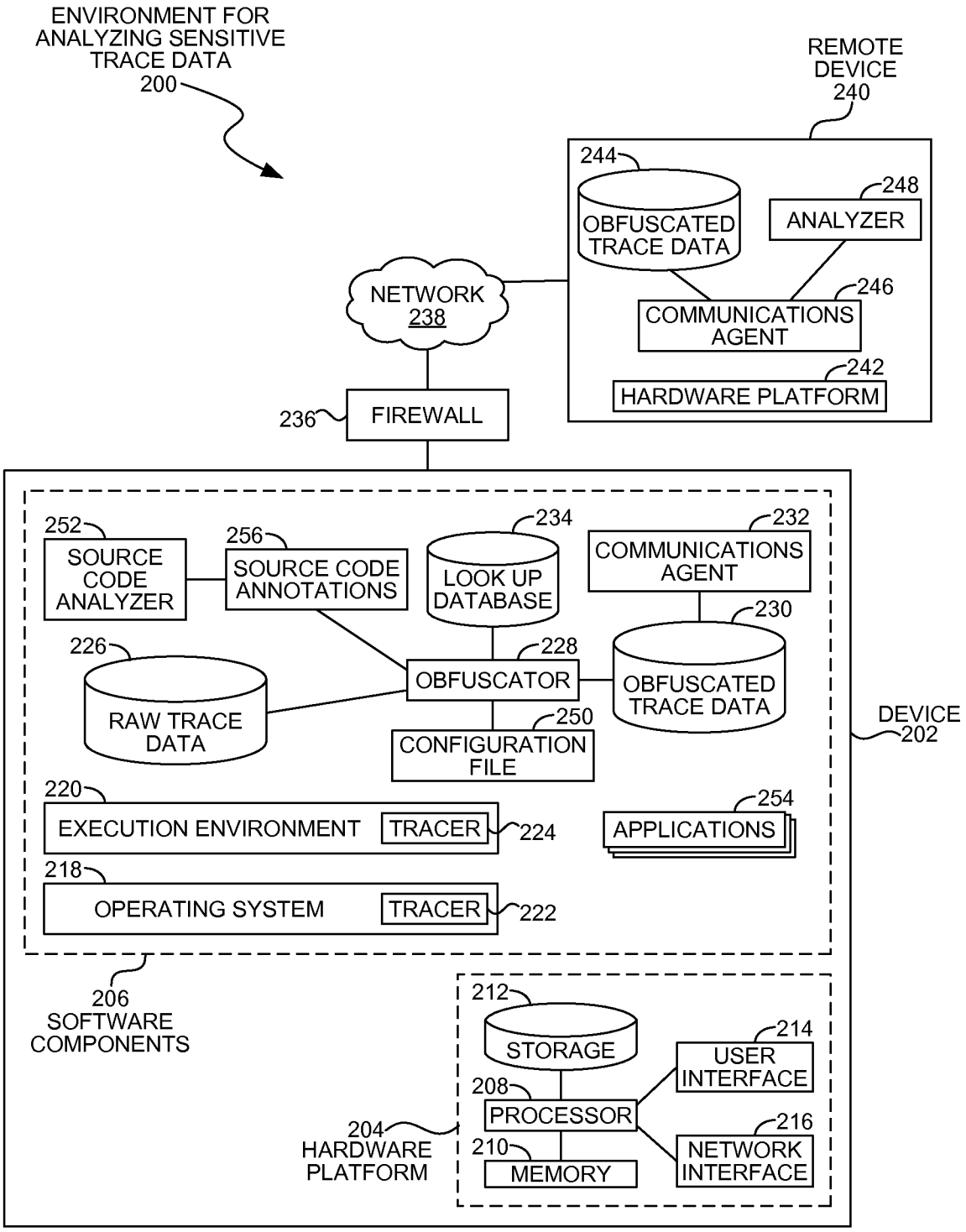
ENVIRONMENT FOR ANALYZING SENSITIVE TRACE DATA
200
REMOTE DEVICE
240
244
OBFUSCATED TRACE DATA
248
ANALYZER
246
COMMUNICATIONS AGENT
242
HARDWARE PLATFORM
NETWORK
238
236
FIREWALL
252
SOURCE CODE ANALYZER
256
SOURCE CODE ANNOTATIONS
234
LOOK UP DATABASE
232
COMMUNICATIONS AGENT
230
OBFUSCATED TRACE DATA
226
RAW TRACE DATA
228
OBFUSCATOR
250
CONFIGURATION FILE
DEVICE
202
220
EXECUTION ENVIRONMENT
TRACER
224
254
APPLICATIONS
218
OPERATING SYSTEM
TRACER
222
206
SOFTWARE COMPONENTS
212
STORAGE
214
USER INTERFACE
208
PROCESSOR
216
NETWORK INTERFACE
204
HARDWARE PLATFORM
210
MEMORY

FIG. 2

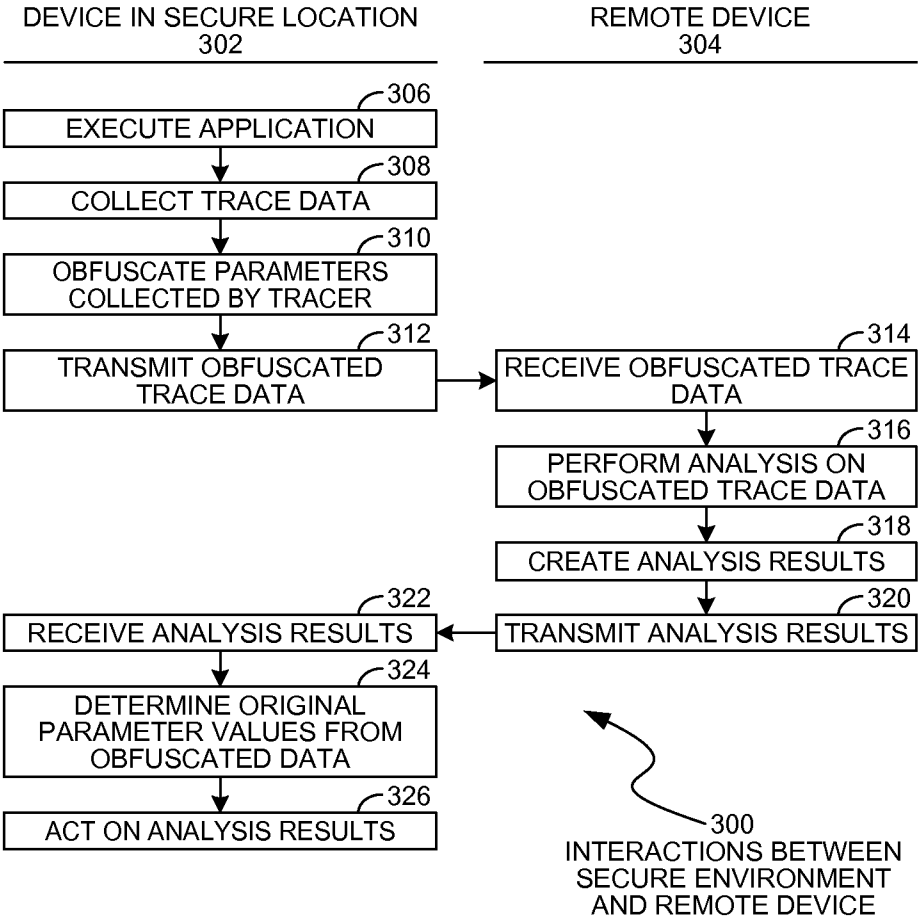
DEVICE IN SECURE LOCATION
302
REMOTE DEVICE
304
306
EXECUTE APPLICATION
308
COLLECT TRACE DATA
310
OBFUSCATE PARAMETERS COLLECTED BY TRACER
312
TRANSMIT OBFUSCATED TRACE DATA
314
RECEIVE OBFUSCATED TRACE DATA
316
PERFORM ANALYSIS ON OBFUSCATED TRACE DATA
318
CREATE ANALYSIS RESULTS
320
TRANSMIT ANALYSIS RESULTS
322
RECEIVE ANALYSIS RESULTS
324
DETERMINE ORIGINAL PARAMETER VALUES FROM OBFUSCATED DATA
326
ACT ON ANALYSIS RESULTS
300
INTERACTIONS BETWEEN SECURE ENVIRONMENT AND REMOTE DEVICE

FIG. 3

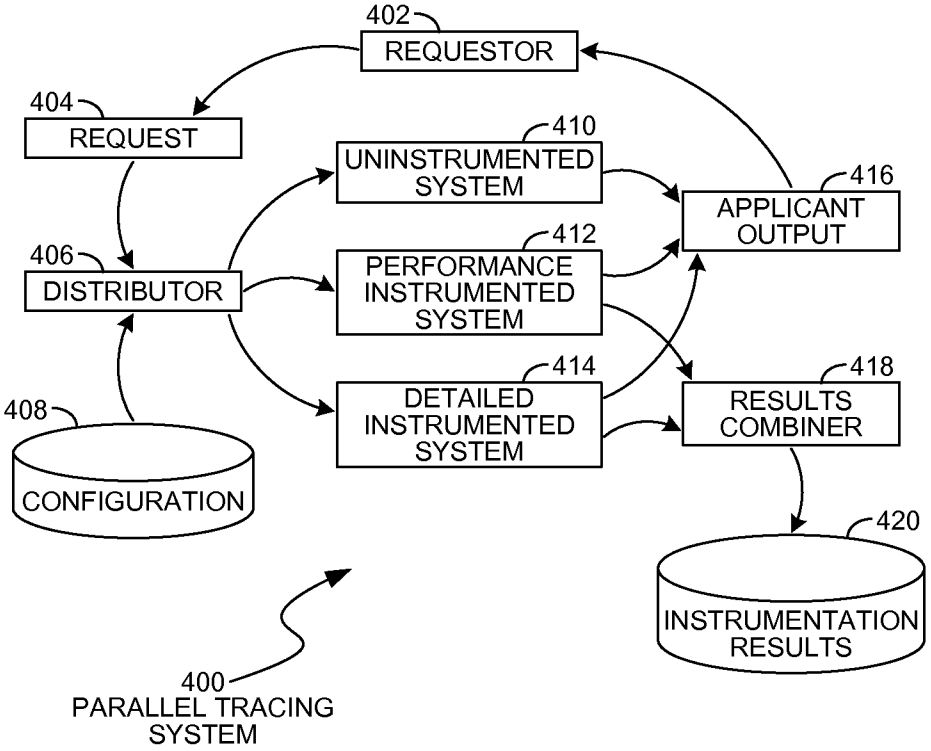
402
REQUESTOR
404
REQUEST
410
UNINSTRUMENTED SYSTEM
416
APPLICANT OUTPUT
406
DISTRIBUTOR
412
PERFORMANCE INSTRUMENTED SYSTEM
414
DETAILED INSTRUMENTED SYSTEM
418
RESULTS COMBINER
408
CONFIGURATION
420
INSTRUMENTATION RESULTS
400
PARALLEL TRACING SYSTEM

FIG. 4

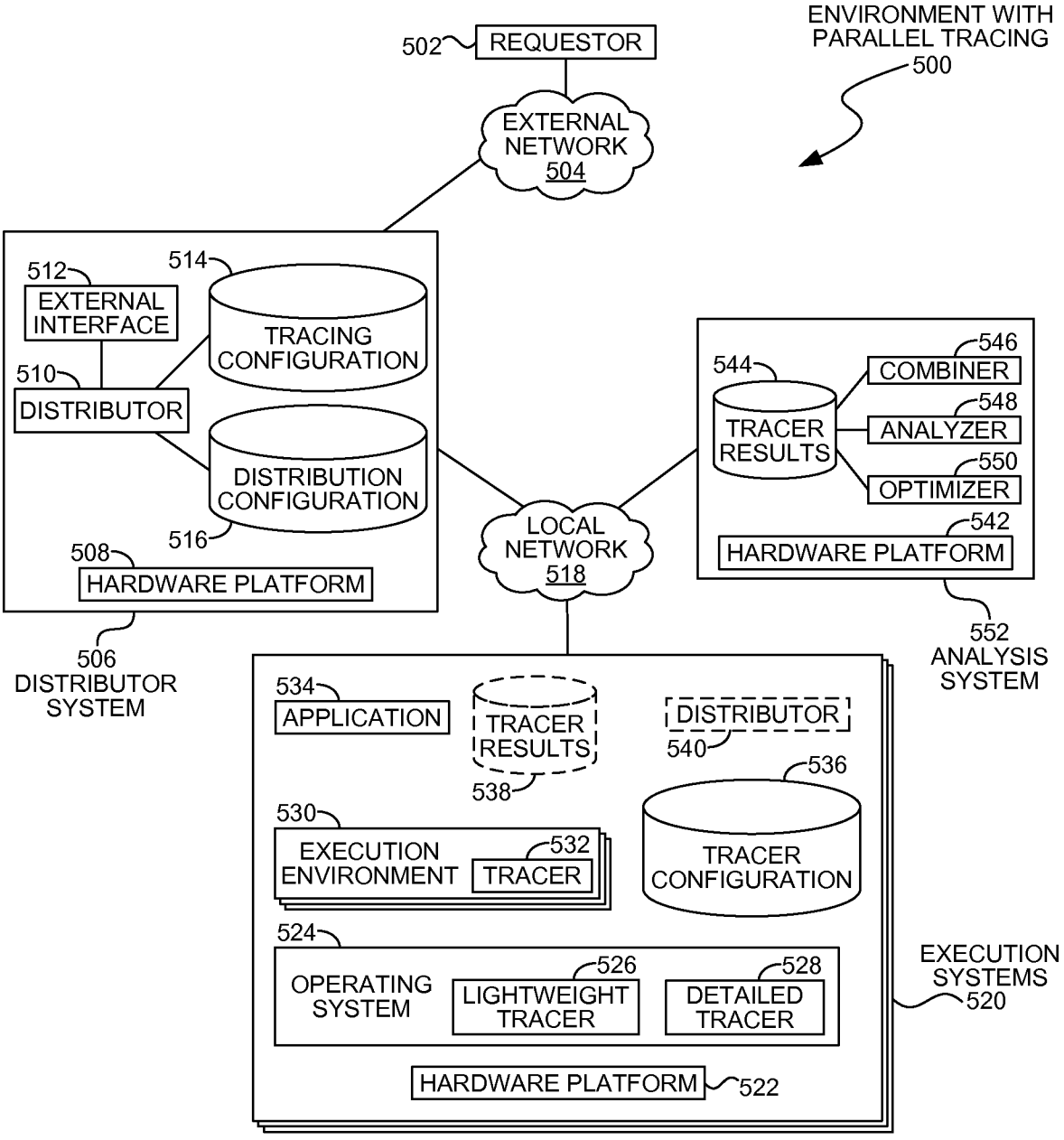
ENVIRONMENT WITH PARALLEL TRACING
500
502
REQUESTOR
EXTERNAL NETWORK 504
512
EXTERNAL INTERFACE
514
TRACING CONFIGURATION
510
DISTRIBUTOR
DISTRIBUTION CONFIGURATION
516
508
HARDWARE PLATFORM
506 DISTRIBUTOR SYSTEM
LOCAL NETWORK 518
544
TRACER RESULTS
546
COMBINER
548
ANALYZER
550
OPTIMIZER
542
HARDWARE PLATFORM
552 ANALYSIS SYSTEM
534
APPLICATION
TRACER RESULTS
538
DISTRIBUTOR
540
536
TRACER CONFIGURATION
530
EXECUTION ENVIRONMENT
532
TRACER
524
OPERATING SYSTEM
526
LIGHTWEIGHT TRACER
528
DETAILED TRACER
HARDWARE PLATFORM
522
EXECUTION SYSTEMS 520

FIG. 5

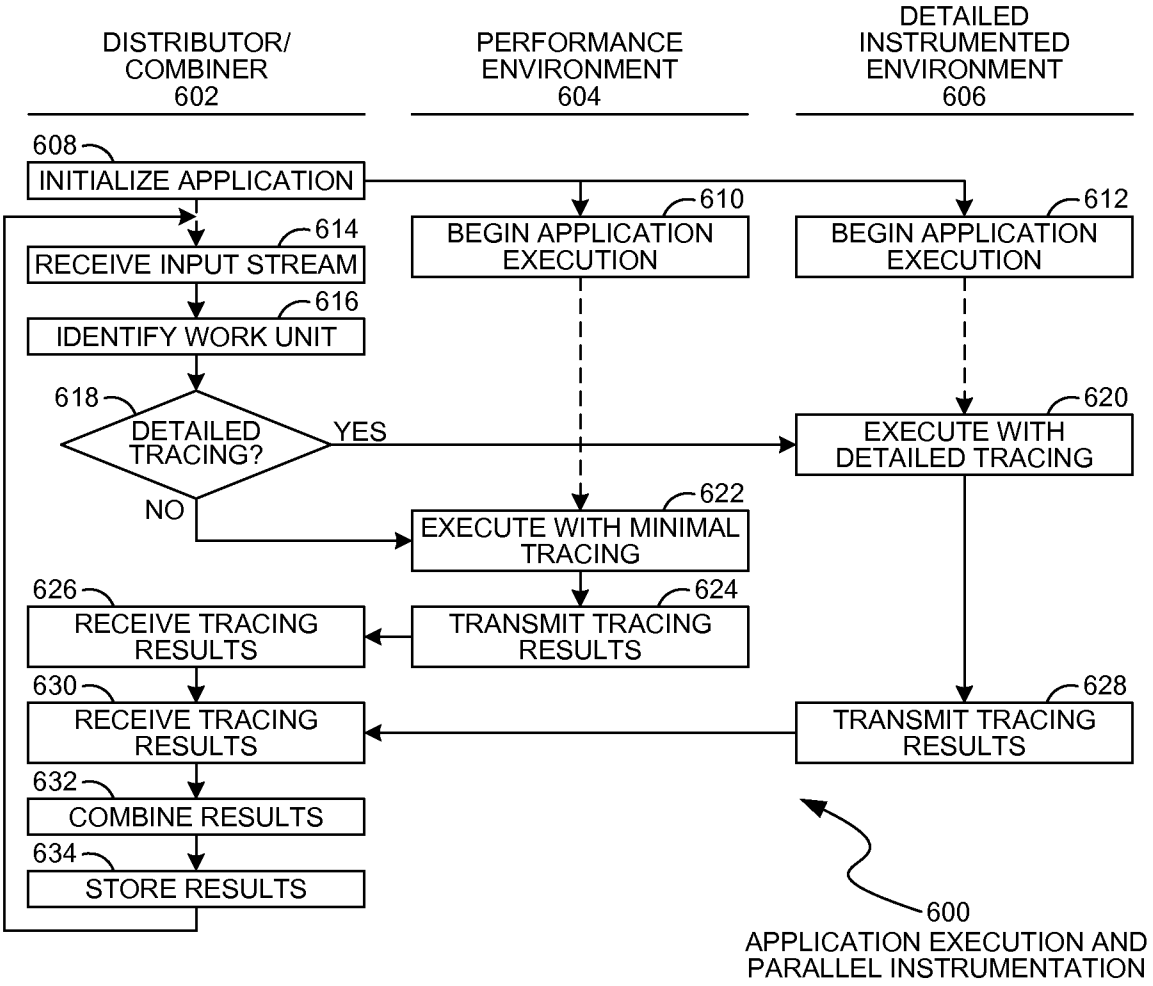
DISTRIBUTOR/ COMBINER 602
PERFORMANCE ENVIRONMENT 604
DETAILED INSTRUMENTED ENVIRONMENT 606
608
INITIALIZE APPLICATION
610
BEGIN APPLICATION EXECUTION
612
BEGIN APPLICATION EXECUTION
614
RECEIVE INPUT STREAM
616
IDENTIFY WORK UNIT
618
DETAILED TRACING?
YES
NO
620
EXECUTE WITH DETAILED TRACING
622
EXECUTE WITH MINIMAL TRACING
624
TRANSMIT TRACING RESULTS
626
RECEIVE TRACING RESULTS
628
TRANSMIT TRACING RESULTS
630
RECEIVE TRACING RESULTS
632
COMBINE RESULTS
634
STORE RESULTS
600
APPLICATION EXECUTION AND PARALLEL INSTRUMENTATION

FIG. 6

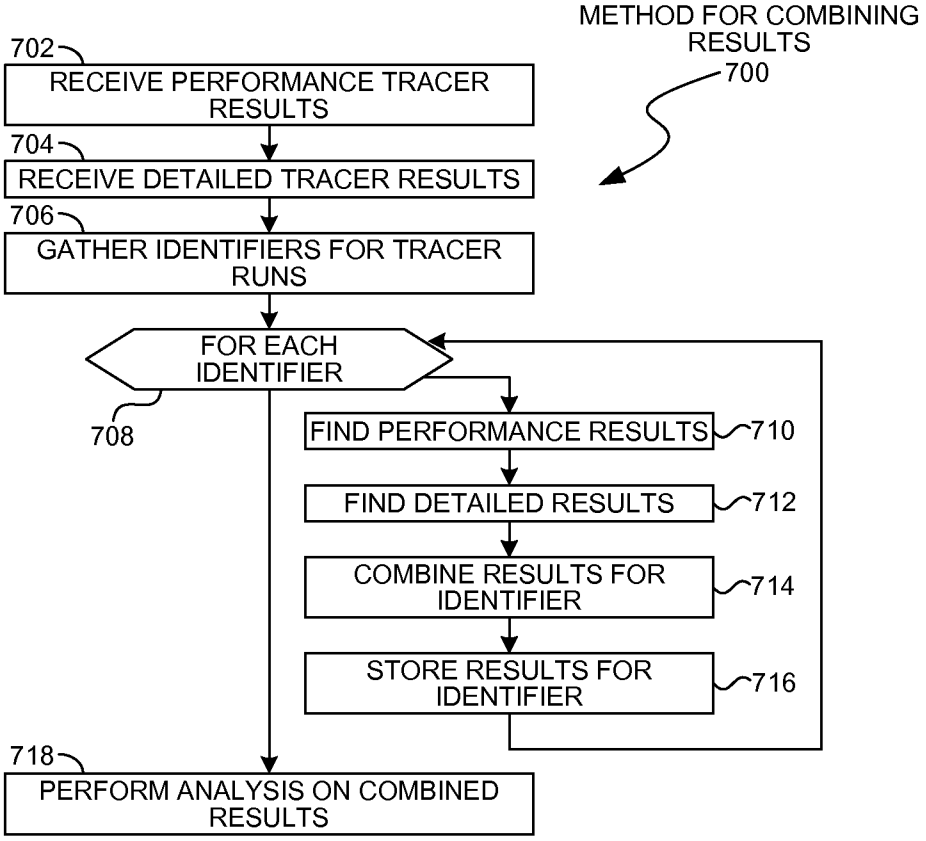
METHOD FOR COMBINING RESULTS
700
702
RECEIVE PERFORMANCE TRACER RESULTS
704
RECEIVE DETAILED TRACER RESULTS
706
GATHER IDENTIFIERS FOR TRACER RUNS
FOR EACH IDENTIFIER
708
FIND PERFORMANCE RESULTS
710
FIND DETAILED RESULTS
712
COMBINE RESULTS FOR IDENTIFIER
714
STORE RESULTS FOR IDENTIFIER
716
718
PERFORM ANALYSIS ON COMBINED RESULTS

FIG. 7

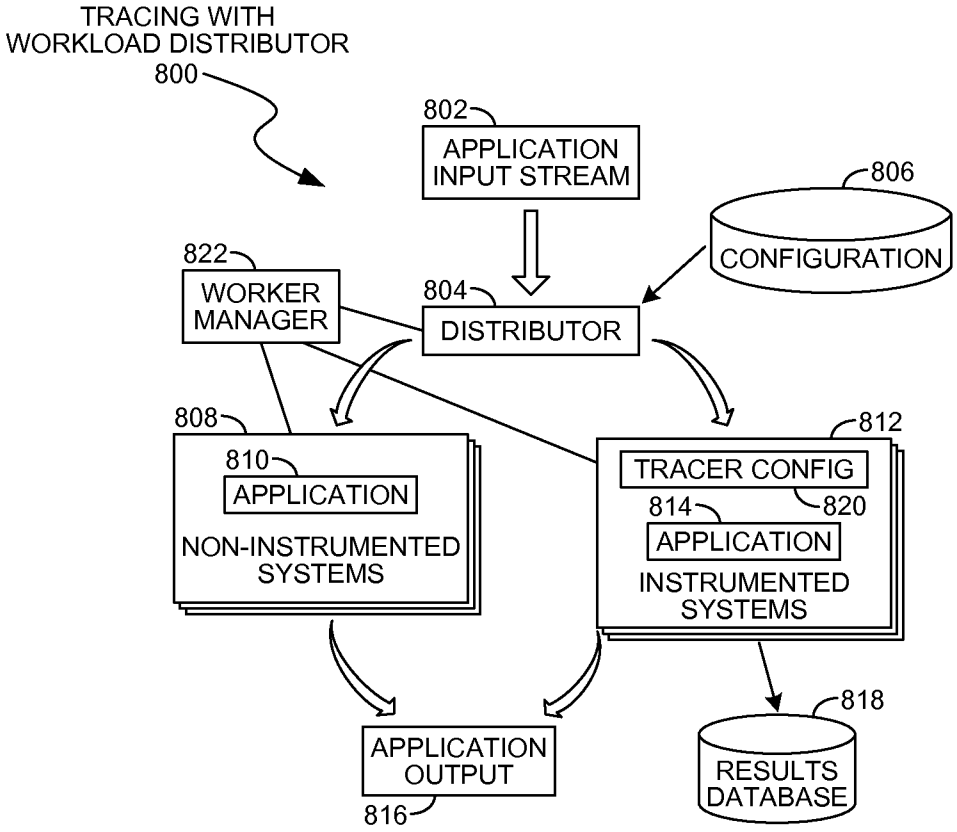
TRACING WITH
WORKLOAD DISTRIBUTOR
800
802
APPLICATION
INPUT STREAM
806
CONFIGURATION
822
WORKER
MANAGER
804
DISTRIBUTOR
808
810
APPLICATION
NON-INSTRUMENTED
SYSTEMS
812
TRACER CONFIG
820
814
APPLICATION
INSTRUMENTED
SYSTEMS
818
APPLICATION
OUTPUT
816
RESULTS
DATABASE

FIG. 8

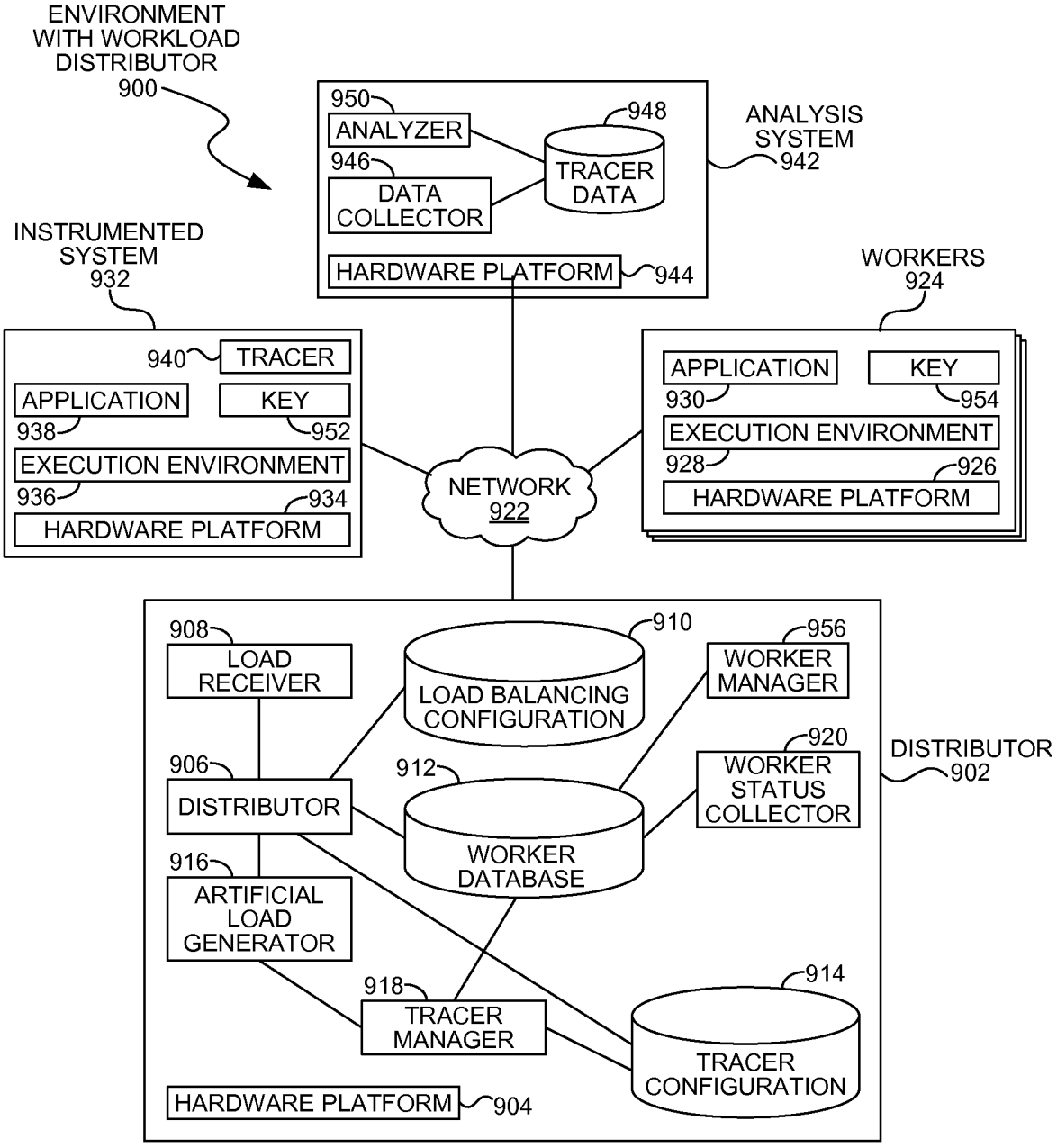
ENVIRONMENT WITH WORKLOAD DISTRIBUTOR
900
950
ANALYZER
946
DATA COLLECTOR
948
TRACER DATA
HARDWARE PLATFORM
944
ANALYSIS SYSTEM
942
INSTRUMENTED SYSTEM
932
940
TRACER
APPLICATION
938
KEY
952
EXECUTION ENVIRONMENT
936
934
HARDWARE PLATFORM
NETWORK
922
WORKERS
924
APPLICATION
930
KEY
954
EXECUTION ENVIRONMENT
928
926
HARDWARE PLATFORM
908
LOAD RECEIVER
910
LOAD BALANCING CONFIGURATION
956
WORKER MANAGER
906
DISTRIBUTOR
912
WORKER DATABASE
920
WORKER STATUS COLLECTOR
DISTRIBUTOR
902
916
ARTIFICIAL LOAD GENERATOR
918
TRACER MANAGER
914
TRACER CONFIGURATION
HARDWARE PLATFORM
904

FIG. 9

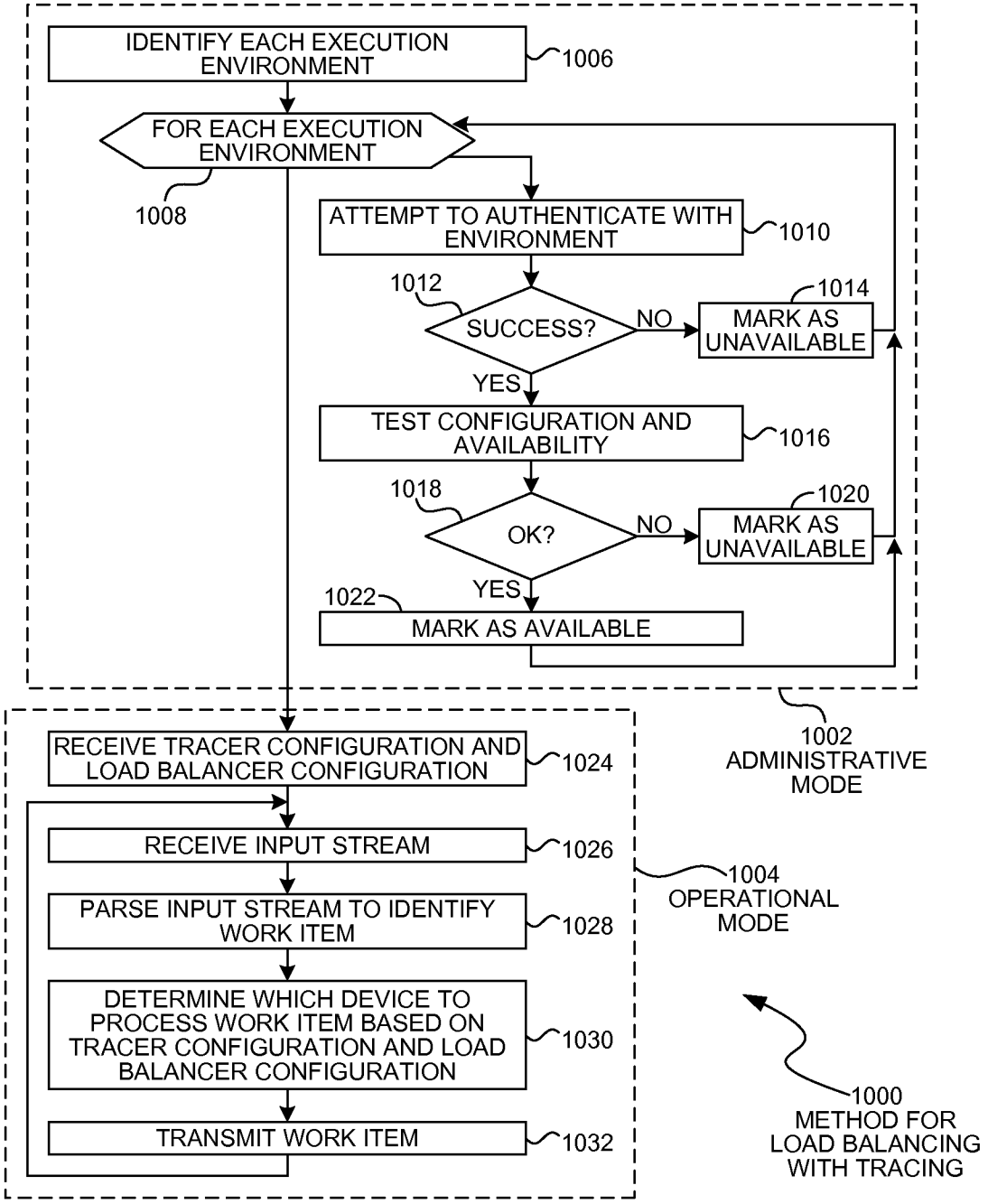
IDENTIFY EACH EXECUTION ENVIRONMENT
1006
FOR EACH EXECUTION ENVIRONMENT
1008
ATTEMPT TO AUTHENTICATE WITH ENVIRONMENT
1010
1012
SUCCESS?
NO
1014
MARK AS UNAVAILABLE
YES
TEST CONFIGURATION AND AVAILABILITY
1016
1018
OK?
NO
1020
MARK AS UNAVAILABLE
YES
1022
MARK AS AVAILABLE
1002
ADMINISTRATIVE MODE
RECEIVE TRACER CONFIGURATION AND LOAD BALANCER CONFIGURATION
1024
RECEIVE INPUT STREAM
1026
PARSE INPUT STREAM TO IDENTIFY WORK ITEM
1028
DETERMINE WHICH DEVICE TO PROCESS WORK ITEM BASED ON TRACER CONFIGURATION AND LOAD BALANCER CONFIGURATION
1030
TRANSMIT WORK ITEM
1032
1004
OPERATIONAL MODE
1000
METHOD FOR LOAD BALANCING WITH TRACING

FIG. 10

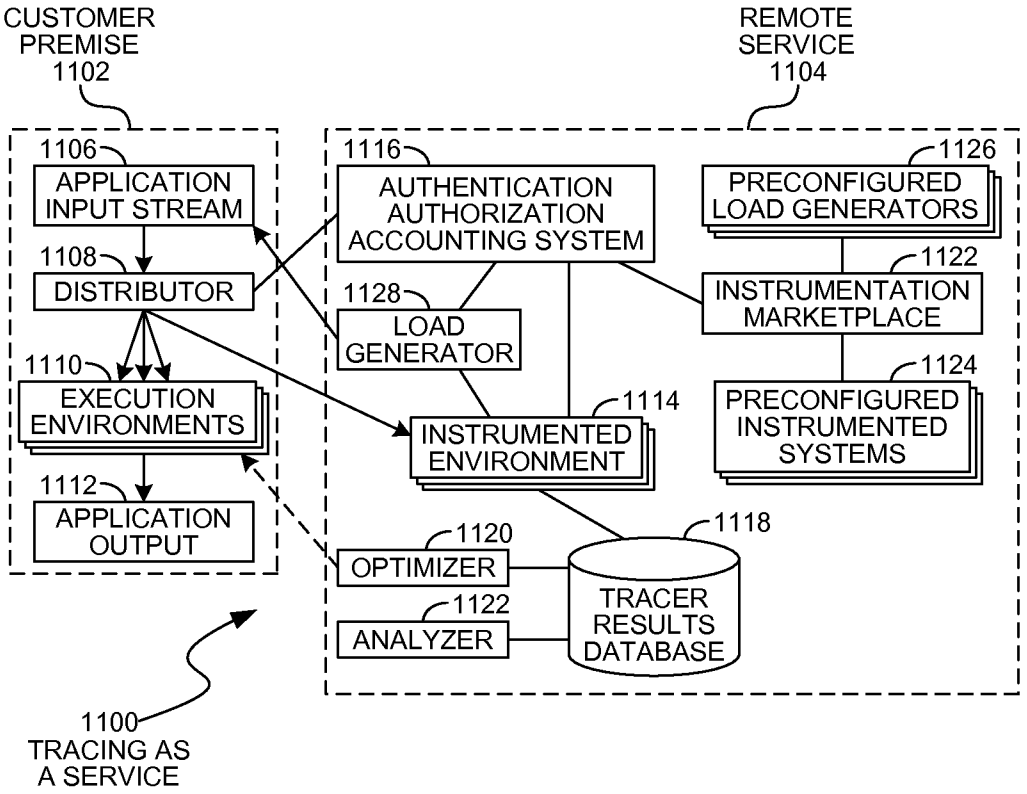
CUSTOMER PREMISE 1102
REMOTE SERVICE 1104
1106
APPLICATION INPUT STREAM
1108
DISTRIBUTOR
1110
EXECUTION ENVIRONMENTS
1112
APPLICATION OUTPUT
1116
AUTHENTICATION AUTHORIZATION ACCOUNTING SYSTEM
1126
PRECONFIGURED LOAD GENERATORS
1122
INSTRUMENTATION MARKETPLACE
1128
LOAD GENERATOR
1124
PRECONFIGURED INSTRUMENTED SYSTEMS
1114
INSTRUMENTED ENVIRONMENT
1118
1120
OPTIMIZER
1122
ANALYZER
TRACER RESULTS DATABASE
1100
TRACING AS A SERVICE

FIG. 11

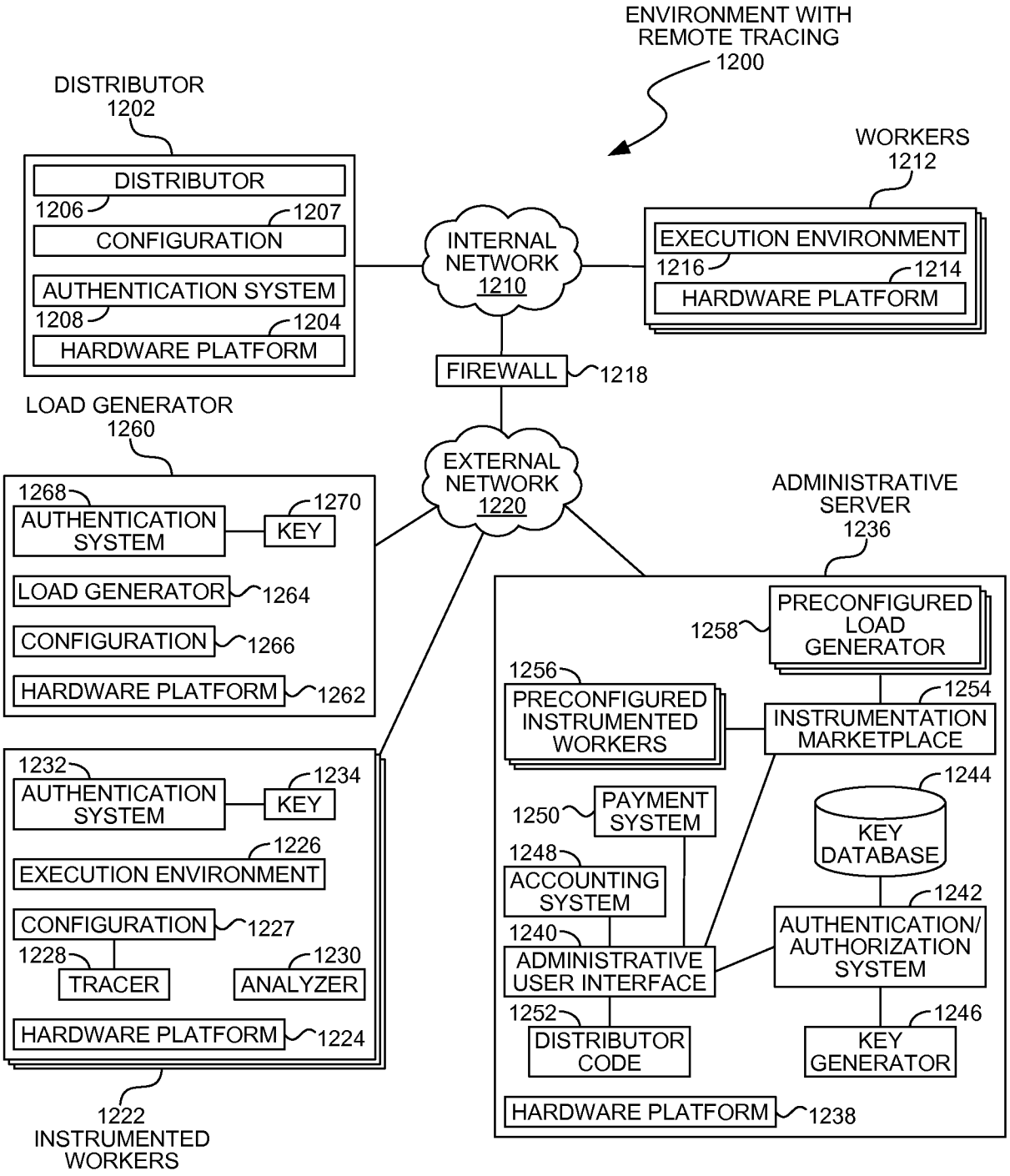
ENVIRONMENT WITH REMOTE TRACING
1200
DISTRIBUTOR
1202
DISTRIBUTOR
1206
1207
CONFIGURATION
AUTHENTICATION SYSTEM
1208
1204
HARDWARE PLATFORM
INTERNAL NETWORK
1210
WORKERS
1212
EXECUTION ENVIRONMENT
1216
1214
HARDWARE PLATFORM
FIREWALL
1218
LOAD GENERATOR
1260
EXTERNAL NETWORK
1220
1268
AUTHENTICATION SYSTEM
1270
KEY
LOAD GENERATOR
1264
CONFIGURATION
1266
HARDWARE PLATFORM
1262
ADMINISTRATIVE SERVER
1236
1258
PRECONFIGURED LOAD GENERATOR
1256
PRECONFIGURED INSTRUMENTED WORKERS
1254
INSTRUMENTATION MARKETPLACE
1232
AUTHENTICATION SYSTEM
1234
KEY
1244
KEY DATABASE
1250
PAYMENT SYSTEM
1226
EXECUTION ENVIRONMENT
1248
ACCOUNTING SYSTEM
1242
AUTHENTICATION/ AUTHORIZATION SYSTEM
CONFIGURATION
1227
1228
TRACER
1230
ANALYZER
1240
ADMINISTRATIVE USER INTERFACE
1252
DISTRIBUTOR CODE
1246
KEY GENERATOR
HARDWARE PLATFORM
1224
HARDWARE PLATFORM
1238
1222
INSTRUMENTED WORKERS

FIG. 12

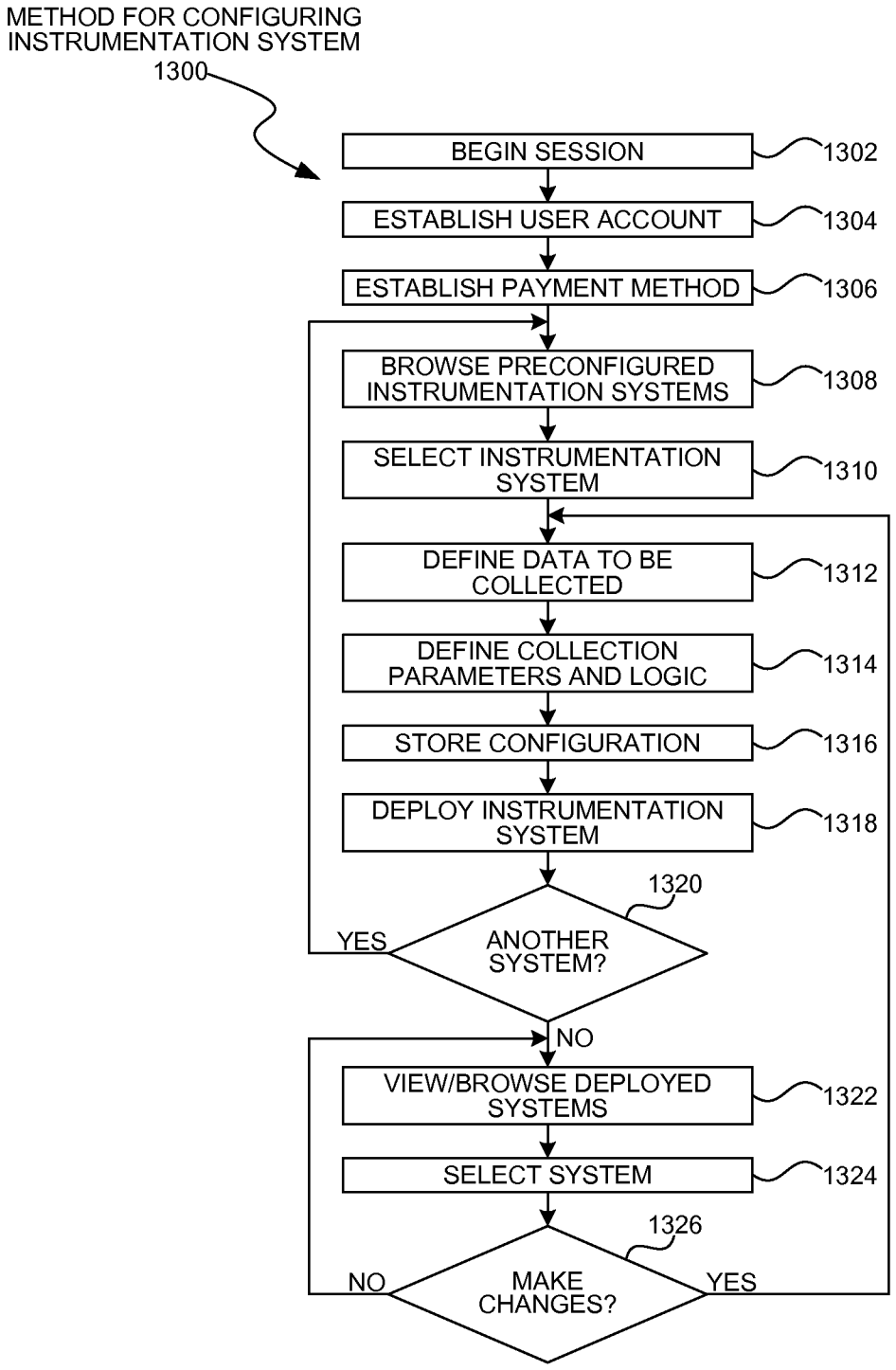
METHOD FOR CONFIGURING INSTRUMENTATION SYSTEM
1300
BEGIN SESSION
1302
ESTABLISH USER ACCOUNT
1304
ESTABLISH PAYMENT METHOD
1306
BROWSE PRECONFIGURED INSTRUMENTATION SYSTEMS
1308
SELECT INSTRUMENTATION SYSTEM
1310
DEFINE DATA TO BE COLLECTED
1312
DEFINE COLLECTION PARAMETERS AND LOGIC
1314
STORE CONFIGURATION
1316
DEPLOY INSTRUMENTATION SYSTEM
1318
1320
ANOTHER SYSTEM?
YES
NO
VIEW/BROWSE DEPLOYED SYSTEMS
1322
SELECT SYSTEM
1324
1326
MAKE CHANGES?
NO
YES

FIG. 13

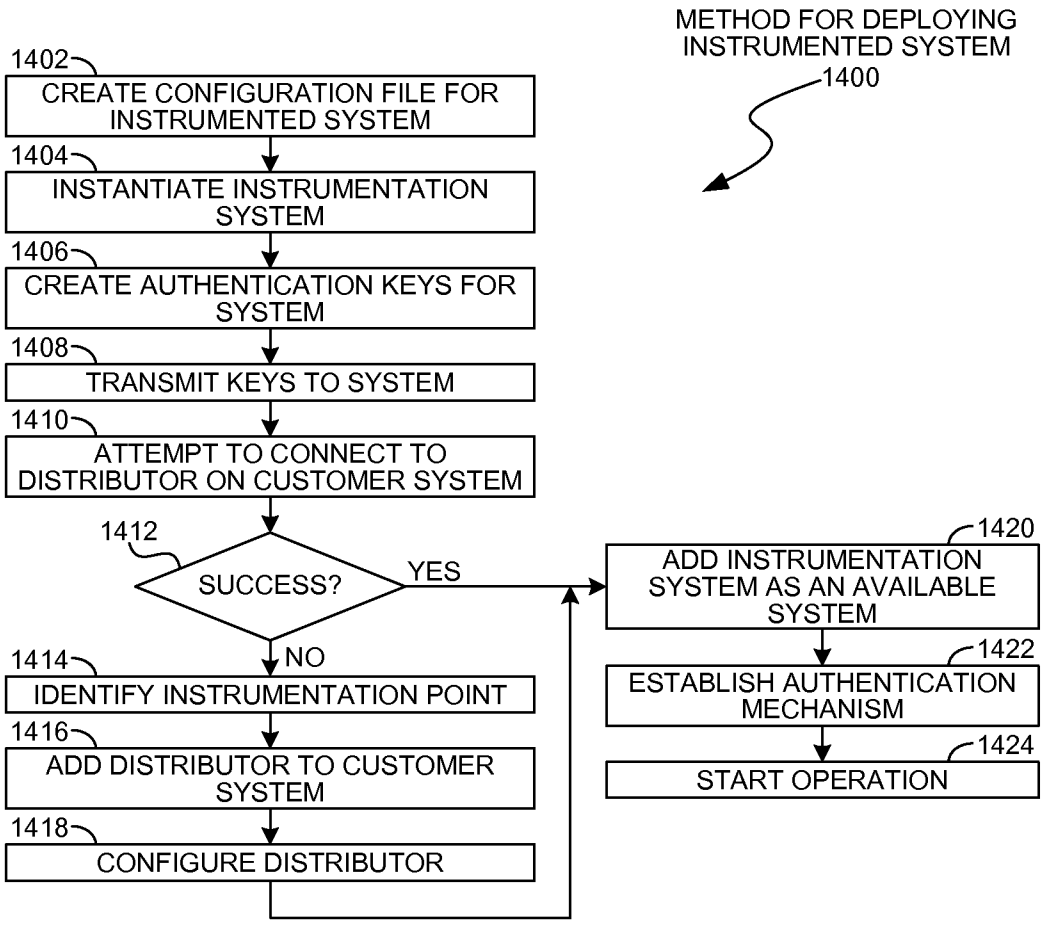
METHOD FOR DEPLOYING INSTRUMENTED SYSTEM
1400
1402
CREATE CONFIGURATION FILE FOR INSTRUMENTED SYSTEM
1404
INSTANTIATE INSTRUMENTATION SYSTEM
1406
CREATE AUTHENTICATION KEYS FOR SYSTEM
1408
TRANSMIT KEYS TO SYSTEM
1410
ATTEMPT TO CONNECT TO DISTRIBUTOR ON CUSTOMER SYSTEM
1412
SUCCESS?
YES
NO
1414
IDENTIFY INSTRUMENTATION POINT
1416
ADD DISTRIBUTOR TO CUSTOMER SYSTEM
1418
CONFIGURE DISTRIBUTOR
1420
ADD INSTRUMENTATION SYSTEM AS AN AVAILABLE SYSTEM
1422
ESTABLISH AUTHENTICATION MECHANISM
1424
START OPERATION

FIG. 14

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2013/042788

A. CLASSIFICATION OF SUBJECT MATTER

G06F 15/16(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 15/16

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: work load, balancing, tracing, file, data, status, debug, troubleshooting, error,debug,

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2008-0005281 A1 (HSUEH et al.) 03 January 2008 See abstract, paragraph 4, 30-44, figure 2, claims 1,10,19.	1-32
A	US 2006-0047752 A1 (HORNBY) 02 March 2006 See abstract, paragraph 6-12, 54-77, figure 2, claims 1,13,24.	1-32
A	US 2003-0217155 A1 (BERTRAM GRECK et al.) 20 November 2003 See abstract, paragraph 17-24, 50-66, figures 2,4, claims 1,7,19.	1-32
A	US 2010-0205583 A1 (GEBHARDT BORIS et al.) 12 August 2010 See abstract, paragraph 41-57, figures 2-4, claims 1,8,13.	1-32

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

☒ See patent family annex.

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

Date of the actual completion of the international search	Date of mailing of the international search report
04 September 2013 (04.09.2013)	**05 September 2013 (05.09.2013)**
Name and mailing address of the ISA/KR Korean Intellectual Property Office 189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea Facsimile No. +82-42-472-7140	Authorized officer HEO, Young Han Telephone No. +82-42-481-8456

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2013/042788

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008-0005281 A1	03/01/2008	None	
US 2006-0047752 A1	02/03/2006	None	
US 2003-0217155 A1	20/11/2003	US 7451206 B2	11/11/2008
US 2010-0205583 A1	12/08/2010	None	