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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 April 2021 (22.04.2021)

WIPO | PCT

(10) International Publication Number
WO 2021/076228 A1

(51) International Patent Classification:
G06Q 10/06 (2012.01) *B25J 9/16* (2006.01)
G06Q 10/10 (2012.01) *G06N 20/00* (2019.01)

(21) International Application Number:
PCT/US2020/046951

(22) International Filing Date:
19 August 2020 (19.08.2020)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/915,434 15 October 2019 (15.10.2019) US
16/708,083 09 December 2019 (09.12.2019) US

(71) Applicant: **UIPATH, INC.** [US/US]; 90 Park Avenue, 20th Floor, New York, NY 10016 (US).

(72) Inventors: **SINGH, Prabhdeep**; 90 Park Avenue, 20th Floor, New York, NY 10016 (US). **MCGONNELL, Anton**; 90 Park Avenue, 20th Floor, New York, NY 10016 (US).

(74) Agent: **LEONARD II, Michael, Aristo** et al.; LeonardPatel PC, 218 North Lee Street, Suite 300, Alexandria, VA 22314 (US).

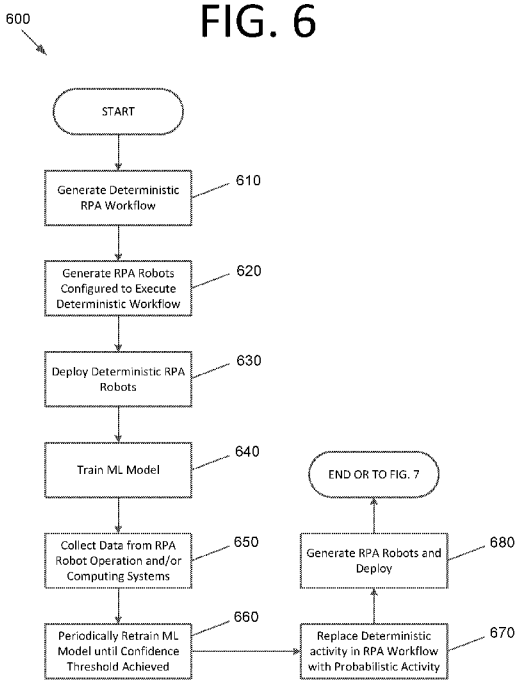
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS,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

FIG. 6



(57) **Abstract:** Probabilistic models may be used in a deterministic workflow for robotic process automation (RPA). Machine learning (ML) introduces a probabilistic framework where the outcome is not deterministic, and therefore, the steps are not deterministic. Deterministic workflows may be mixed with probabilistic workflows, or probabilistic activities may be inserted into deterministic workflows, in order to create more dynamic workflows. A supervisor system may be used to monitor an ML model and raise an alarm, disable an RPA robot, bypass an RPA robot, or roll back to a previous version of the ML model when an error is detected by a data drift detector, a concept drift detector, or both.

WO 2021/076228 A1

TITLE

INSERTING PROBABILISTIC MODELS IN DETERMINISTIC WORKFLOWS FOR ROBOTIC PROCESS AUTOMATION AND SUPERVISOR SYSTEM

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Nonprovisional Patent Application No. 16/708,083 filed December 9, 2019, and U.S. Provisional Patent Application No. 62/915,434 filed October 15, 2019. The subject matter of these earlier filed applications is hereby incorporated by reference in its entirety.

FIELD

[0002] The present invention generally relates to robotic process automation (RPA), and more specifically, to inserting probabilistic models in deterministic workflows for RPA and a supervisor system.

BACKGROUND

[0003] Current RPA systems work in a logistic fashion with decision points in a sequential workflow. However, such workflows are based on set logic and are not dynamic. Accordingly, an improved approach may be beneficial.

SUMMARY

[0004] Certain embodiments of the present invention may provide solutions to the problems and needs in the art that have not yet been fully identified, appreciated, or

solved by current RPA technologies. For example, some embodiments of the present invention pertain to inserting probabilistic models in deterministic workflows for RPA and a supervisor system.

[0005] In an embodiment, a computer-implemented method for implementing probabilistic models in a deterministic workflow for RPA includes generating a deterministic RPA workflow for RPA robots, generating a plurality of RPA robots configured to execute the deterministic RPA workflow, and deploying the plurality of RPA robots. The computer-implemented method also includes training a machine learning (ML) model associated with a probabilistic activity to replace a deterministic activity in the RPA workflow.

[0006] In another embodiment, a computer-implemented method for implementing probabilistic models in a deterministic workflow for RPA includes generating a plurality of RPA robots configured to execute a deterministic RPA workflow and deploying the plurality of RPA robots configured to execute the deterministic RPA workflow. The computer-implemented method also includes training an ML model associated with a probabilistic activity to replace a deterministic activity in the RPA workflow.

[0007] In yet another embodiment, a computer-implemented method includes generating an RPA robot configured to execute an RPA workflow including a probabilistic activity and deploying the RPA robot. The computer-implemented method also includes monitoring, by a supervisor system, an ML model called by the probabilistic activity in the RPA workflow to ensure that the ML model is operating correctly using metrics on an input side via a data drift detector and on an output side using a concept drift detector.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] In order that the advantages of certain embodiments of the invention will be readily understood, a more particular description of the invention briefly described above will be rendered by reference to specific embodiments that are illustrated in the appended drawings. While it should be understood that these drawings depict only typical embodiments of the invention and are not therefore to be considered to be limiting of its scope, the invention will be described and explained with additional specificity and detail through the use of the accompanying drawings, in which:

[0009] FIG. 1 is an architectural diagram illustrating an RPA system, according to an embodiment of the present invention.

[0010] FIG. 2 is an architectural diagram illustrating a deployed RPA system, according to an embodiment of the present invention.

[0011] FIG. 3 is an architectural diagram illustrating the relationship between a designer, activities, and drivers, according to an embodiment of the present invention.

[0012] FIG. 4 is an architectural diagram illustrating an RPA system, according to an embodiment of the present invention.

[0013] FIG. 5 is an architectural diagram illustrating a computing system configured to use probabilistic models in deterministic workflows for RPA, according to an embodiment of the present invention.

[0014] FIG. 6 is a flowchart illustrating a process for implementing probabilistic models in a deterministic workflow for RPA, according to an embodiment of the present invention.

[0015] FIG. 7 is a flowchart illustrating a process for a supervisor system, according to an embodiment of the present invention.

[0016] FIG. 8 is an architectural diagram illustrating a supervisor system, according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0017] Some embodiments pertain to using probabilistic models in a deterministic workflow for RPA. Such embodiments may make initial deployments faster, for example. Machine learning introduces a probabilistic framework where the outcome is not deterministic, and therefore, the steps are not deterministic. Deterministic workflows may be mixed with probabilistic workflows, or probabilistic activities may be inserted into deterministic workflows, in order to create more dynamic workflows. RPA workflows are usually in the form of logistic flow that is very deterministic with a fixed number of activities. However, probabilistic activities may be inserted in the place of one or more deterministic activities in the workflow in some embodiments. To insert a probabilistic activity, certain factors may be considered, such as a confidence level of a probabilistic activity. If a machine learning (ML) model associated with a probabilistic activity is trained such that it becomes accurate enough for use, a probabilistic activity may be used in the workflow in place of a deterministic activity. Thus, in some embodiments, workflows may begin as deterministic in nature and then be modified later to be probabilistic.

[0018] SUPERVISOR SYSTEM

[0019] Testing probabilistic systems can be difficult, however. Accordingly, in some embodiments, a supervisor system is used to monitor RPA robots executing probabilistic workflows to determine correct operation. For example, consider the case of an RPA implementation on an aircraft. A robot may be trained to perform certain controls, such as causing the aircraft to sacrifice altitude for speed if the robot determines that a stall may happen. However, this may not always be the desired action depending on flight conditions. For instance, performing this action shortly after takeoff may cause a crash.

[0020] The supervisor system may monitor the actions taken by robots and also monitor other data from the system. In the example above, the monitored data may include pilot actions. If the supervisor system determines that soon after the robot causes the aircraft to dive, the pilots pull back on the wheel to stop the dive. The supervisor system may then disable the robot or a portion of the robot's workflow and send the collected data so the robot can be retrained.

[0021] In some embodiments, an ML model may be monitored to ensure it is operating correctly with metrics on the input side and the output side. For instance, if the ML model requires sixteen data feeds, but one is broken, the ML model may not be suitable to use. This is monitoring for "data drift". A data drift detector may be included on the input side.

[0022] On the output side, it may be known what ranges to expect. For instance, if a hospital's return rate is known to not have been more than 10%, but it is determined that 290% of patients are coming back, something is wrong. This is called concept drift, which may be monitored by a concept drift detector. The supervisor system may thus

expect an ML model to perform a certain way and comply with certain parameters. The supervisor system may raise an alarm if a data drift detector indicates an error, a concept drift detector indicates an error, or both.

[0023] Consider the case of a supervisor system supervising ML models implemented in an aircraft. If a certain ML model affecting control surfaces is run and the nose is dipped aggressively or the descent is rapid when the autopilot is engaged, this could be flagged by the supervisor system, and robots calling the ML models could be shut off or bypassed by manual control. If the robot is not mission critical, or a previous version of an ML model is known to work with a high confidence in a mission critical system, the supervisor system may roll the ML model back to a previous version.

[0024] FIG. 1 is an architectural diagram illustrating an RPA system 100, according to an embodiment of the present invention. RPA system 100 includes a designer 110 that allows a developer to design and implement workflows. Designer 110 may provide a solution for application integration, as well as automating third-party applications, administrative Information Technology (IT) tasks, and business IT processes. Designer 110 may facilitate development of an automation project, which is a graphical representation of a business process. Simply put, designer 110 facilitates the development and deployment of workflows and robots.

[0025] The automation project enables automation of rule-based processes by giving the developer control of the execution order and the relationship between a custom set of steps developed in a workflow, defined herein as "activities." One commercial example of an embodiment of designer 110 is UiPath Studio™. Each activity may

include an action, such as clicking a button, reading a file, writing to a log panel, etc. In some embodiments, workflows may be nested or embedded.

[0026] Some types of workflows may include, but are not limited to, sequences, flowcharts, Finite State Machines (FSMs), and/or global exception handlers. Sequences may be particularly suitable for linear processes, enabling flow from one activity to another without cluttering a workflow. Flowcharts may be particularly suitable to more complex business logic, enabling integration of decisions and connection of activities in a more diverse manner through multiple branching logic operators. FSMs may be particularly suitable for large workflows. FSMs may use a finite number of states in their execution, which are triggered by a condition (i.e., transition) or an activity. Global exception handlers may be particularly suitable for determining workflow behavior when encountering an execution error and for debugging processes.

[0027] Once a workflow is developed in designer 110, execution of business processes is orchestrated by conductor 120, which orchestrates one or more robots 130 that execute the workflows developed in designer 110. One commercial example of an embodiment of conductor 120 is UiPath Orchestrator™. Conductor 120 facilitates management of the creation, monitoring, and deployment of resources in an environment. Conductor 120 may act as an integration point with third-party solutions and applications.

[0028] Conductor 120 may manage a fleet of robots 130, connecting and executing robots 130 from a centralized point. Types of robots 130 that may be managed include, but are not limited to, attended robots 132, unattended robots 134, development robots (similar to unattended robots 134, but used for development and testing purposes), and

nonproduction robots (similar to attended robots 132, but used for development and testing purposes). Attended robots 132 are triggered by user events and operate alongside a human on the same computing system. Attended robots 132 may be used with conductor 120 for a centralized process deployment and logging medium. Attended robots 132 may help the human user accomplish various tasks, and may be triggered by user events. In some embodiments, processes cannot be started from conductor 120 on this type of robot and/or they cannot run under a locked screen. In certain embodiments, attended robots 132 can only be started from a robot tray or from a command prompt. Attended robots 132 should run under human supervision in some embodiments.

[0029] Unattended robots 134 run unattended in virtual environments and can automate many processes. Unattended robots 134 may be responsible for remote execution, monitoring, scheduling, and providing support for work queues. Debugging for all robot types may be run in designer 110 in some embodiments. Both attended and unattended robots may automate various systems and applications including, but not limited to, mainframes, web applications, VMs, enterprise applications (e.g., those produced by SAP®, SalesForce®, Oracle®, etc.), and computing system applications (e.g., desktop and laptop applications, mobile device applications, wearable computer applications, etc.).

[0030] Conductor 120 may have various capabilities including, but not limited to, provisioning, deployment, configuration, queueing, monitoring, logging, and/or providing interconnectivity. Provisioning may include creating and maintenance of connections between robots 130 and conductor 120 (e.g., a web application).

Deployment may include assuring the correct delivery of package versions to assigned robots 130 for execution. Configuration may include maintenance and delivery of robot environments and process configurations. Queueing may include providing management of queues and queue items. Monitoring may include keeping track of robot identification data and maintaining user permissions. Logging may include storing and indexing logs to a database (e.g., an SQL database) and/or another storage mechanism (e.g., ElasticSearch®, which provides the ability to store and quickly query large datasets). Conductor 120 may provide interconnectivity by acting as the centralized point of communication for third-party solutions and/or applications.

[0031] Robots 130 are execution agents that run workflows built in designer 110. One commercial example of some embodiments of robot(s) 130 is UiPath Robots™. In some embodiments, robots 130 install the Microsoft Windows® Service Control Manager (SCM)-managed service by default. As a result, such robots 130 can open interactive Windows® sessions under the local system account, and have the rights of a Windows® service.

[0032] In some embodiments, robots 130 can be installed in a user mode. For such robots 130, this means they have the same rights as the user under which a given robot 130 has been installed. This feature may also be available for High Density (HD) robots, which ensure full utilization of each machine at its maximum potential. In some embodiments, any type of robot 130 may be configured in an HD environment.

[0033] Robots 130 in some embodiments are split into several components, each being dedicated to a particular automation task. The robot components in some embodiments include, but are not limited to, SCM-managed robot services, user mode

robot services, executors, agents, and command line. SCM-managed robot services manage and monitor Windows® sessions and act as a proxy between conductor 120 and the execution hosts (i.e., the computing systems on which robots 130 are executed). These services are trusted with and manage the credentials for robots 130. A console application is launched by the SCM under the local system.

[0034] User mode robot services in some embodiments manage and monitor Windows® sessions and act as a proxy between conductor 120 and the execution hosts. User mode robot services may be trusted with and manage the credentials for robots 130. A Windows® application may automatically be launched if the SCM-managed robot service is not installed.

[0035] Executors may run given jobs under a Windows® session (i.e., they may execute workflows. Executors may be aware of per-monitor dots per inch (DPI) settings. Agents may be Windows® Presentation Foundation (WPF) applications that display the available jobs in the system tray window. Agents may be a client of the service. Agents may request to start or stop jobs and change settings. The command line is a client of the service. The command line is a console application that can request to start jobs and waits for their output.

[0036] Having components of robots 130 split as explained above helps developers, support users, and computing systems more easily run, identify, and track what each component is executing. Special behaviors may be configured per component this way, such as setting up different firewall rules for the executor and the service. The executor may always be aware of DPI settings per monitor in some embodiments. As a result, workflows may be executed at any DPI, regardless of the configuration of the computing

system on which they were created. Projects from designer 110 may also be independent of browser zoom level in some embodiments. For applications that are DPI-unaware or intentionally marked as unaware, DPI may be disabled in some embodiments.

[0037] FIG. 2 is an architectural diagram illustrating a deployed RPA system 200, according to an embodiment of the present invention. In some embodiments, RPA system 200 may be, or may be a part of, RPA system 100 of FIG. 1. It should be noted that the client side, the server side, or both, may include any desired number of computing systems without deviating from the scope of the invention. On the client side, a robot application 210 includes executors 212, an agent 214, and a designer 216. However, in some embodiments, designer 216 may not be running on computing system 210. Executors 212 are running processes. Several business projects may run simultaneously, as shown in FIG. 2. Agent 214 (e.g., a Windows® service) is the single point of contact for all executors 212 in this embodiment. All messages in this embodiment are logged into conductor 230, which processes them further via database server 240, indexer server 250, or both. As discussed above with respect to FIG. 1, executors 212 may be robot components.

[0038] In some embodiments, a robot represents an association between a machine name and a username. The robot may manage multiple executors at the same time. On computing systems that support multiple interactive sessions running simultaneously (e.g., Windows® Server 2012), multiple robots may be running at the same time, each in a separate Windows® session using a unique username. This is referred to as HD robots above.

[0039] Agent 214 is also responsible for sending the status of the robot (e.g., periodically sending a "heartbeat" message indicating that the robot is still functioning) and downloading the required version of the package to be executed. The communication between agent 214 and conductor 230 is always initiated by agent 214 in some embodiments. In the notification scenario, agent 214 may open a WebSocket channel that is later used by conductor 230 to send commands to the robot (e.g., start, stop, etc.).

[0040] On the server side, a presentation layer (web application 232, Open Data Protocol (OData) Representative State Transfer (REST) Application Programming Interface (API) endpoints 234, and notification and monitoring 236), a service layer (API implementation / business logic 238), and a persistence layer (database server 240 and indexer server 250) are included. Conductor 230 includes web application 232, OData REST API endpoints 234, notification and monitoring 236, and API implementation / business logic 238. In some embodiments, most actions that a user performs in the interface of conductor 220 (e.g., via browser 220) are performed by calling various APIs. Such actions may include, but are not limited to, starting jobs on robots, adding/removing data in queues, scheduling jobs to run unattended, etc. without deviating from the scope of the invention. Web application 232 is the visual layer of the server platform. In this embodiment, web application 232 uses Hypertext Markup Language (HTML) and JavaScript (JS). However, any desired markup languages, script languages, or any other formats may be used without deviating from the scope of the invention. The user interacts with web pages from web application 232 via browser 220 in this embodiment in order to perform various actions to control conductor 230. For

instance, the user may create robot groups, assign packages to the robots, analyze logs per robot and/or per process, start and stop robots, etc.

[0041] In addition to web application 232, conductor 230 also includes service layer that exposes OData REST API endpoints 234. However, other endpoints may be included without deviating from the scope of the invention. The REST API is consumed by both web application 232 and agent 214. Agent 214 is the supervisor of one or more robots on the client computer in this embodiment.

[0042] The REST API in this embodiment covers configuration, logging, monitoring, and queueing functionality. The configuration endpoints may be used to define and configure application users, permissions, robots, assets, releases, and environments in some embodiments. Logging REST endpoints may be used to log different information, such as errors, explicit messages sent by the robots, and other environment-specific information, for instance. Deployment REST endpoints may be used by the robots to query the package version that should be executed if the start job command is used in conductor 230. Queueing REST endpoints may be responsible for queues and queue item management, such as adding data to a queue, obtaining a transaction from the queue, setting the status of a transaction, etc.

[0043] Monitoring REST endpoints may monitor web application 232 and agent 214. Notification and monitoring API 236 may be REST endpoints that are used for registering agent 214, delivering configuration settings to agent 214, and for sending/receiving notifications from the server and agent 214. Notification and monitoring API 236 may also use WebSocket communication in some embodiments.

[0044] The persistence layer includes a pair of servers in this embodiment – database server 240 (e.g., a SQL server) and indexer server 250. Database server 240 in this embodiment stores the configurations of the robots, robot groups, associated processes, users, roles, schedules, etc. This information is managed through web application 232 in some embodiments. Database server 240 may manages queues and queue items. In some embodiments, database server 240 may store messages logged by the robots (in addition to or in lieu of indexer server 250).

[0045] Indexer server 250, which is optional in some embodiments, stores and indexes the information logged by the robots. In certain embodiments, indexer server 250 may be disabled through configuration settings. In some embodiments, indexer server 250 uses ElasticSearch®, which is an open source project full-text search engine. Messages logged by robots (e.g., using activities like log message or write line) may be sent through the logging REST endpoint(s) to indexer server 250, where they are indexed for future utilization.

[0046] FIG. 3 is an architectural diagram illustrating the relationship 300 between a designer 310, activities 320, 330, and drivers 340, according to an embodiment of the present invention. Per the above, a developer uses designer 310 to develop workflows that are executed by robots. Workflows may include user-defined activities 320 and UI automation activities 330. Some embodiments are able to identify non-textual visual components in an image, which is called computer vision (CV) herein. Some CV activities pertaining to such components may include, but are not limited to, click, type, get text, hover, element exists, refresh scope, highlight, etc. Click in some embodiments identifies an element using CV, optical character recognition (OCR), fuzzy text

matching, and multi-anchor, for example, and clicks it. Type may identify an element using the above and types in the element. Get text may identify the location of specific text and scan it using OCR. Hover may identify an element and hover over it. Element exists may check whether an element exists on the screen using the techniques described above. In some embodiments, there may be hundreds or even thousands of activities that can be implemented in designer 310. However, any number and/or type of activities may be available without deviating from the scope of the invention.

[0047] UI automation activities 330 are a subset of special, lower level activities that are written in lower level code (e.g., CV activities) and facilitate interactions with the screen. UI automation activities 330 facilitate these interactions via drivers 340 that allow the robot to interact with the desired software. For instance, drivers 340 may include OS drivers 342, browser drivers 344, VM drivers 346, enterprise application drivers 348, etc.

[0048] Drivers 340 may interact with the OS at a low level looking for hooks, monitoring for keys, etc. They may facilitate integration with Chrome®, IE®, Citrix®, SAP®, etc. For instance, the "click" activity performs the same role in these different applications via drivers 340.

[0049] FIG. 4 is an architectural diagram illustrating an RPA system 400, according to an embodiment of the present invention. In some embodiments, RPA system 400 may be or include RPA systems 100 and/or 200 of FIGS. 1 and/or 2. RPA system 400 includes multiple client computing systems 410 running robots. Computing systems 410 are able to communicate with a conductor computing system 420 via a web

application running thereon. Conductor computing system 420, in turn, is able to communicate with a database server 430 and an optional indexer server 440.

[0050] With respect to FIGS. 1 and 3, it should be noted that while a web application is used in these embodiments, any suitable client/server software may be used without deviating from the scope of the invention. For instance, the conductor may run a server-side application that communicates with non-web-based client software applications on the client computing systems.

[0051] FIG. 5 is an architectural diagram illustrating a computing system 500 configured to use probabilistic models in deterministic workflows for RPA, according to an embodiment of the present invention. In some embodiments, computing system 500 may be one or more of the computing systems depicted and/or described herein. Computing system 500 includes a bus 505 or other communication mechanism for communicating information, and processor(s) 510 coupled to bus 505 for processing information. Processor(s) 510 may be any type of general or specific purpose processor, including a Central Processing Unit (CPU), an Application Specific Integrated Circuit (ASIC), a Field Programmable Gate Array (FPGA), a Graphics Processing Unit (GPU), multiple instances thereof, and/or any combination thereof. Processor(s) 510 may also have multiple processing cores, and at least some of the cores may be configured to perform specific functions. Multi-parallel processing may be used in some embodiments. In certain embodiments, at least one of processor(s) 510 may be a neuromorphic circuit that includes processing elements that mimic biological neurons. In some embodiments, neuromorphic circuits may not require the typical components of a Von Neumann computing architecture.

[0052] Computing system 500 further includes a memory 515 for storing information and instructions to be executed by processor(s) 510. Memory 515 can be comprised of any combination of Random Access Memory (RAM), Read Only Memory (ROM), flash memory, cache, static storage such as a magnetic or optical disk, or any other types of non-transitory computer-readable media or combinations thereof. Non-transitory computer-readable media may be any available media that can be accessed by processor(s) 510 and may include volatile media, non-volatile media, or both. The media may also be removable, non-removable, or both.

[0053] Additionally, computing system 500 includes a communication device 520, such as a transceiver, to provide access to a communications network via a wireless and/or wired connection. In some embodiments, communication device 520 may be configured to use Frequency Division Multiple Access (FDMA), Single Carrier FDMA (SC-FDMA), Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA), Orthogonal Frequency Division Multiplexing (OFDM), Orthogonal Frequency Division Multiple Access (OFDMA), Global System for Mobile (GSM) communications, General Packet Radio Service (GPRS), Universal Mobile Telecommunications System (UMTS), cdma2000, Wideband CDMA (W-CDMA), High-Speed Downlink Packet Access (HSDPA), High-Speed Uplink Packet Access (HSUPA), High-Speed Packet Access (HSPA), Long Term Evolution (LTE), LTE Advanced (LTE-A), 802.11x, Wi-Fi, Zigbee, Ultra-WideBand (UWB), 802.16x, 802.15, Home Node-B (HnB), Bluetooth, Radio Frequency Identification (RFID), Infrared Data Association (IrDA), Near-Field Communications (NFC), fifth generation (5G), New Radio (NR), any combination thereof, and/or any other currently existing or future-

implemented communications standard and/or protocol without deviating from the scope of the invention. In some embodiments, communication device 520 may include one or more antennas that are singular, arrayed, phased, switched, beamforming, beamsteering, a combination thereof, and or any other antenna configuration without deviating from the scope of the invention.

[0054] Processor(s) 510 are further coupled via bus 505 to a display 525, such as a plasma display, a Liquid Crystal Display (LCD), a Light Emitting Diode (LED) display, a Field Emission Display (FED), an Organic Light Emitting Diode (OLED) display, a flexible OLED display, a flexible substrate display, a projection display, a 4K display, a high definition display, a Retina® display, an In-Plane Switching (IPS) display, or any other suitable display for displaying information to a user. Display 525 may be configured as a touch (haptic) display, a three dimensional (3D) touch display, a multi-input touch display, a multi-touch display, etc. using resistive, capacitive, surface-acoustic wave (SAW) capacitive, infrared, optical imaging, dispersive signal technology, acoustic pulse recognition, frustrated total internal reflection, etc. Any suitable display device and haptic I/O may be used without deviating from the scope of the invention.

[0055] A keyboard 530 and a cursor control device 535, such as a computer mouse, a touchpad, etc., are further coupled to bus 505 to enable a user to interface with computing system. However, in certain embodiments, a physical keyboard and mouse may not be present, and the user may interact with the device solely through display 525 and/or a touchpad (not shown). Any type and combination of input devices may be used as a matter of design choice. In certain embodiments, no physical input device and/or

display is present. For instance, the user may interact with computing system 500 remotely via another computing system in communication therewith, or computing system 500 may operate autonomously.

[0056] Memory 515 stores software modules that provide functionality when executed by processor(s) 510. The modules include an operating system 540 for computing system 500. The modules further include a probabilistic model module 545 that is configured to perform all or part of the processes described herein or derivatives thereof. Computing system 500 may include one or more additional functional modules 550 that include additional functionality.

[0057] One skilled in the art will appreciate that a "system" could be embodied as a server, an embedded computing system, a personal computer, a console, a personal digital assistant (PDA), a cell phone, a tablet computing device, a quantum computing system, or any other suitable computing device, or combination of devices without deviating from the scope of the invention. Presenting the above-described functions as being performed by a "system" is not intended to limit the scope of the present invention in any way, but is intended to provide one example of the many embodiments of the present invention. Indeed, methods, systems, and apparatuses disclosed herein may be implemented in localized and distributed forms consistent with computing technology, including cloud computing systems.

[0058] It should be noted that some of the system features described in this specification have been presented as modules, in order to more particularly emphasize their implementation independence. For example, a module may be implemented as a hardware circuit comprising custom very large scale integration (VLSI) circuits or gate

arrays, off-the-shelf semiconductors such as logic chips, transistors, or other discrete components. A module may also be implemented in programmable hardware devices such as field programmable gate arrays, programmable array logic, programmable logic devices, graphics processing units, or the like.

[0059] A module may also be at least partially implemented in software for execution by various types of processors. An identified unit of executable code may, for instance, include one or more physical or logical blocks of computer instructions that may, for instance, be organized as an object, procedure, or function. Nevertheless, the executables of an identified module need not be physically located together, but may include disparate instructions stored in different locations that, when joined logically together, comprise the module and achieve the stated purpose for the module. Further, modules may be stored on a computer-readable medium, which may be, for instance, a hard disk drive, flash device, RAM, tape, and/or any other such non-transitory computer-readable medium used to store data without deviating from the scope of the invention.

[0060] Indeed, a module of executable code could be a single instruction, or many instructions, and may even be distributed over several different code segments, among different programs, and across several memory devices. Similarly, operational data may be identified and illustrated herein within modules, and may be embodied in any suitable form and organized within any suitable type of data structure. The operational data may be collected as a single data set, or may be distributed over different locations including over different storage devices, and may exist, at least partially, merely as electronic signals on a system or network.

[0061] FIG. 6 is a flowchart illustrating a process 600 for implementing probabilistic models in a deterministic workflow for RPA, according to an embodiment of the present invention. The process begins with generating a deterministic RPA workflow for RPA robots at 610. RPA robots configured to execute the deterministic RPA workflow are generated at 620. The RPA robots are then deployed at 630.

[0062] An ML model associated with a probabilistic activity to replace a deterministic activity in the deterministic RPA workflow is trained at 640. Data is collected from RPA robots implementing the deterministic workflow, their associated computing systems, or both, at 650. For instance, the data may include, but is not limited to, operations undertaken by the robots, user interactions with the computing systems (e.g., buttons pressed, mouse movements, applications used, etc.), processes run by the computing systems, corrections made by users, etc. This data is then used to periodically retrain the ML model until a desired confidence threshold is achieved at 660. A probabilistic workflow including the probabilistic activity that replaces the deterministic activity is then generated at 670 and RPA robots implementing the workflow are generated and deployed at 680.

[0063] FIG. 7 is a flowchart illustrating a process 700 for a supervisor system, according to an embodiment of the present invention. In some embodiments, process 700 may be implemented via supervisor system 810 of FIG. 8. The process begins (or continues from FIG. 6) with a supervisor system monitoring an ML model called by a probabilistic activity in an RPA workflow at 710 to ensure that the ML model is operating correctly using metrics on an input side via a data drift detector and on an output side using a concept drift detector. In some embodiments, the supervisor system

may monitor multiple ML models. In certain embodiments, the concept drift detector determines whether the output falls within a predetermined range.

[0064] When no error is detected at 720, the monitoring continues at 710. However, when the data drift detector, the concept drift detector, or both, indicate an error, an alarm is raised at 730 (e.g., sending a message to a computing system or a robot that an error has been detected). The supervisor system then takes remedial action at 740. The remedial action may include, but is not limited to, disabling an RPA robot, bypassing the RPA robot, rolling back to a previous version of the ML model, etc.

[0065] FIG. 8 is an architectural diagram 800 illustrating operation of a supervisor system 810, according to an embodiment of the present invention. Supervisor system includes a data drift detector that monitors input to an ML model 820 and a concept drift detector 814 that monitors output from ML model 820. RPA robot 830 calls ML model 820. When supervisor system 810 receives an error from data drift detector 812 and/or concept drift detector 814, supervisor system 810 may disable or bypass RPA robot 830, generate an alarm, etc.

[0066] The process steps performed in FIGS. 6 and 7 may be performed by a computer program, encoding instructions for the processor(s) to perform at least part of the process(es) described in FIGS. 6 and 7, in accordance with embodiments of the present invention. The computer program may be embodied on a non-transitory computer-readable medium. The computer-readable medium may be, but is not limited to, a hard disk drive, a flash device, RAM, a tape, and/or any other such medium or combination of media used to store data. The computer program may include encoded instructions for controlling processor(s) of a computing system (e.g., processor(s) 510

of computing system 500 of FIG. 5) to implement all or part of the process steps described in FIGS. 6 and 7, which may also be stored on the computer-readable medium.

[0067] The computer program can be implemented in hardware, software, or a hybrid implementation. The computer program can be composed of modules that are in operative communication with one another, and which are designed to pass information or instructions to display. The computer program can be configured to operate on a general purpose computer, an ASIC, or any other suitable device.

[0068] It will be readily understood that the components of various embodiments of the present invention, as generally described and illustrated in the figures herein, may be arranged and designed in a wide variety of different configurations. Thus, the detailed description of the embodiments of the present invention, as represented in the attached figures, is not intended to limit the scope of the invention as claimed, but is merely representative of selected embodiments of the invention.

[0069] The features, structures, or characteristics of the invention described throughout this specification may be combined in any suitable manner in one or more embodiments. For example, reference throughout this specification to "certain embodiments," "some embodiments," or similar language means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, appearances of the phrases "in certain embodiments," "in some embodiment," "in other embodiments," or similar language throughout this specification do not necessarily all refer to the same group of embodiments and the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0070] It should be noted that reference throughout this specification to features, advantages, or similar language does not imply that all of the features and advantages that may be realized with the present invention should be or are in any single embodiment of the invention. Rather, language referring to the features and advantages is understood to mean that a specific feature, advantage, or characteristic described in connection with an embodiment is included in at least one embodiment of the present invention. Thus, discussion of the features and advantages, and similar language, throughout this specification may, but do not necessarily, refer to the same embodiment.

[0071] Furthermore, the described features, advantages, and characteristics of the invention may be combined in any suitable manner in one or more embodiments. One skilled in the relevant art will recognize that the invention can be practiced without one or more of the specific features or advantages of a particular embodiment. In other instances, additional features and advantages may be recognized in certain embodiments that may not be present in all embodiments of the invention.

[0072] One having ordinary skill in the art will readily understand that the invention as discussed above may be practiced with steps in a different order, and/or with hardware elements in configurations which are different than those which are disclosed. Therefore, although the invention has been described based upon these preferred embodiments, it would be apparent to those of skill in the art that certain modifications, variations, and alternative constructions would be apparent, while remaining within the spirit and scope of the invention.

[0073] In an embodiment, a computer-implemented method includes generating a deterministic RPA workflow and training an ML model associated with a probabilistic

activity to replace a deterministic activity in the deterministic RPA workflow. The computer-implemented method also includes collecting data from RPA robots implementing the deterministic workflow and their associated computing systems and using the collected data to periodically retrain the ML model until a confidence threshold is achieved. Then, the computer-implemented method includes generating a probabilistic workflow including the probabilistic activity, replacing the deterministic activity.

CLAIMS

1. A computer-implemented method for implementing probabilistic models in a deterministic workflow for robotic process automation (RPA), comprising:

generating a deterministic RPA workflow for RPA robots;

generating a plurality of RPA robots configured to execute the deterministic RPA workflow;

deploying the plurality of RPA robots; and

training a machine learning (ML) model associated with a probabilistic activity to replace a deterministic activity in the RPA workflow.

2. The computer-implemented method of claim 1, further comprising:

collecting data from the plurality of deployed RPA robots executing the deterministic RPA workflow, collecting data from computing systems on which the RPA robots are running, or both.

3. The computer-implemented method of claim 2, further comprising:

periodically retraining the ML model using the collected data.

4. The computer-implemented method of claim 3, wherein when the ML model achieves a confidence threshold, the method further comprises:

replacing the deterministic activity in the RPA workflow with the probabilistic activity.

5. The computer-implemented method of claim 4, further comprising:

generating a plurality of RPA robots configured to execute the RPA workflow comprising the probabilistic activity; and

deploying the plurality of RPA robots that are configured to execute the RPA workflow comprising the probabilistic activity.

6. The computer-implemented method of claim 5, further comprising:

monitoring the ML model to ensure that the ML model is operating correctly using metrics on an input side via a data drift detector and on an output side using a concept drift detector.

7. The computer-implemented method of claim 6, wherein the concept drift detector determines whether the output falls within a predetermined range.

8. The computer-implemented method of claim 6, further comprising:

raising an alarm when the data drift detector, the concept drift detector, or both, indicate an error.

9. The computer-implemented method of claim 6, wherein when the data drift detector, the concept drift detector, or both, indicate an error, the method further comprises:

disabling or bypassing an RPA robot of the plurality of RPA robots.

10. The computer-implemented method of claim 6, wherein when the data drift detector, the concept drift detector, or both, indicate an error, the method further comprises:

rolling back to a previous version of the ML model.

11. A computer-implemented method for implementing probabilistic models in a deterministic workflow for robotic process automation (RPA), comprising:

generating a plurality of RPA robots configured to execute a deterministic RPA workflow;

deploying the plurality of RPA robots configured to execute the deterministic RPA workflow; and

training a machine learning (ML) model associated with a probabilistic activity to replace a deterministic activity in the RPA workflow.

12. The computer-implemented method of claim 11, further comprising:

collecting data from the plurality of deployed RPA robots executing the deterministic RPA workflow, collecting data from computing systems on which the RPA robots are running, or both.

13. The computer-implemented method of claim 12, further comprising:

periodically retraining the ML model using the collected data; and

when the ML model achieves a confidence threshold, replacing the deterministic activity in the RPA workflow with the probabilistic activity.

14. The computer-implemented method of claim 13, further comprising:

generating a plurality of RPA robots configured to execute the RPA workflow comprising the probabilistic activity; and

deploying the plurality of RPA robots that are configured to execute the RPA workflow comprising the probabilistic activity.

15. The computer-implemented method of claim 14, further comprising:

monitoring the ML model to ensure that the ML model is operating correctly using metrics on an input side via a data drift detector and on an output side using a concept drift detector.

16. The computer-implemented method of claim 15, wherein when the data drift detector, the concept drift detector, or both, indicate an error, the method further comprises:

disabling an RPA robot of the plurality of RPA robots, bypassing the RPA robot of the plurality of RPA robots, or rolling back to a previous version of the ML model.

17. A computer-implemented method, comprising:

generating a robotic process automation (RPA) robot configured to execute an RPA workflow comprising a probabilistic activity;

deploying the RPA robot; and

monitoring, by a supervisor system, a machine learning (ML) model called by the probabilistic activity in the RPA workflow to ensure that the ML model is operating

correctly using metrics on an input side via a data drift detector and on an output side using a concept drift detector.

18. The computer-implemented method of claim 17, wherein the concept drift detector determines whether the output falls within a predetermined range.

19. The computer-implemented method of claim 17, further comprising:

raising an alarm, by the supervisor system, when the data drift detector, the concept drift detector, or both, indicate an error.

20. The computer-implemented method of claim 17, wherein when the data drift detector, the concept drift detector, or both, indicate an error, the method further comprises:

disabling an RPA robot of the plurality of RPA robots, bypassing the RPA robot of the plurality of RPA robots, or rolling back to a previous version of the ML model, by the supervisor system.

FIG. 1

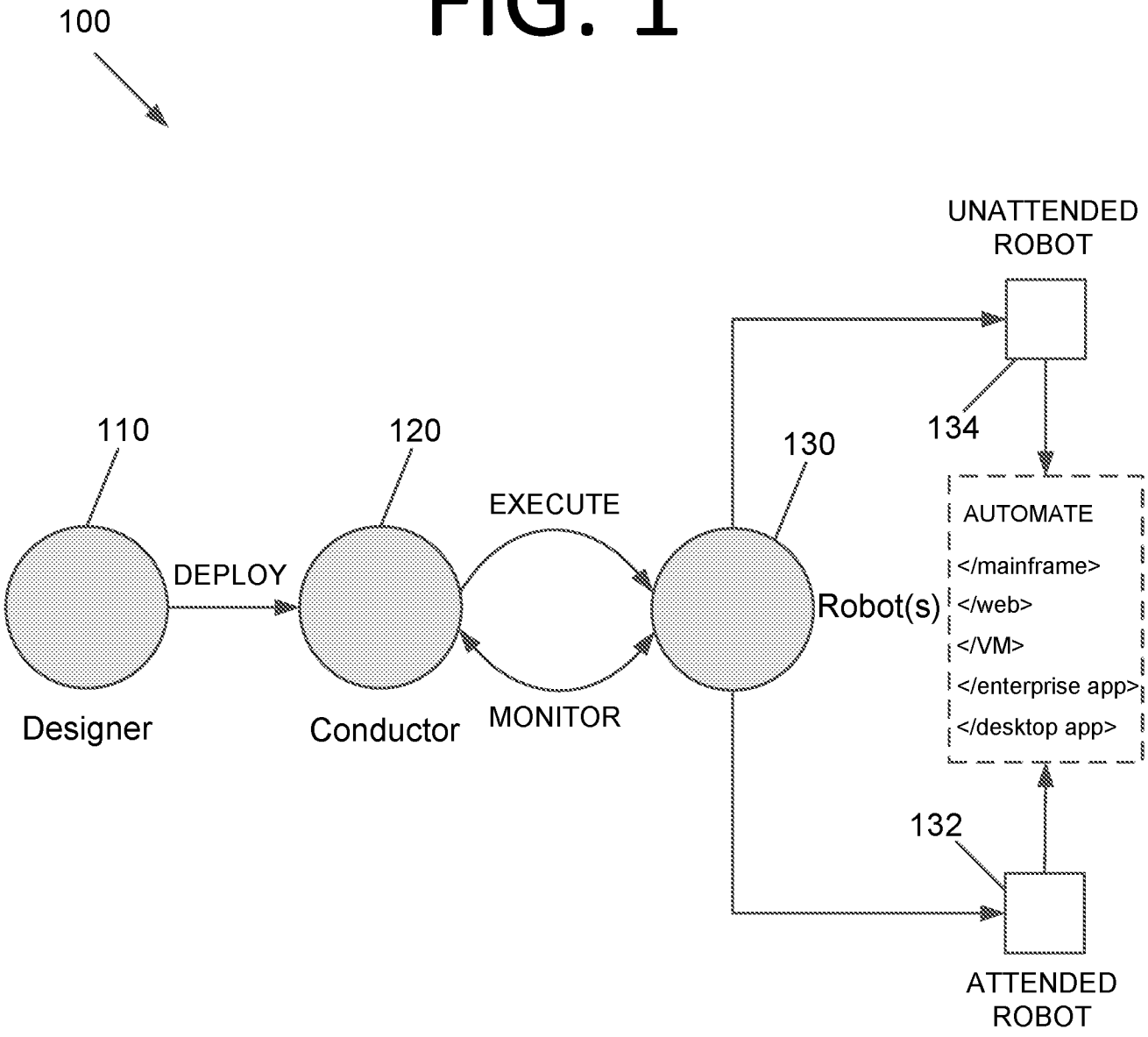

100
UNATTENDED ROBOT
110
120
130
134
EXECUTE
DEPLOY
Robot(s)
AUTOMATE
</mainframe>
</web>
</VM>
</enterprise app>
</desktop app>
Designer
Conductor
MONITOR
132
ATTENDED ROBOT

FIG. 2

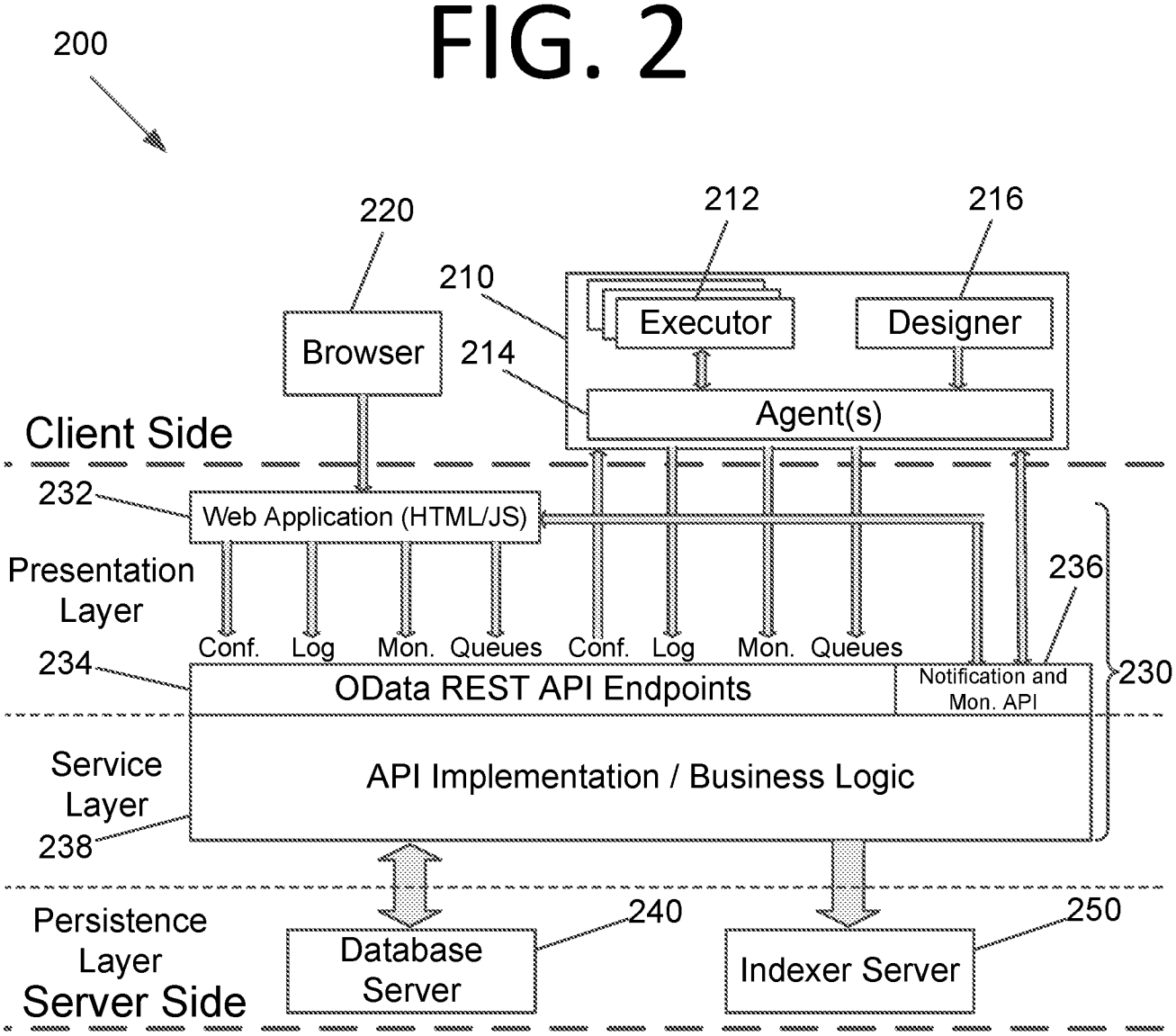

200
220
212
216
210
Executor
Designer
Browser
214
Agent(s)
Client Side
232
Web Application (HTML/JS)
Presentation Layer
236
Conf.
Log
Mon.
Queues
Conf.
Log
Mon.
Queues
234
OData REST API Endpoints
Notification and Mon. API
230
Service Layer
API Implementation / Business Logic
238
Persistence Layer
240
250
Database Server
Indexer Server
Server Side

FIG. 3

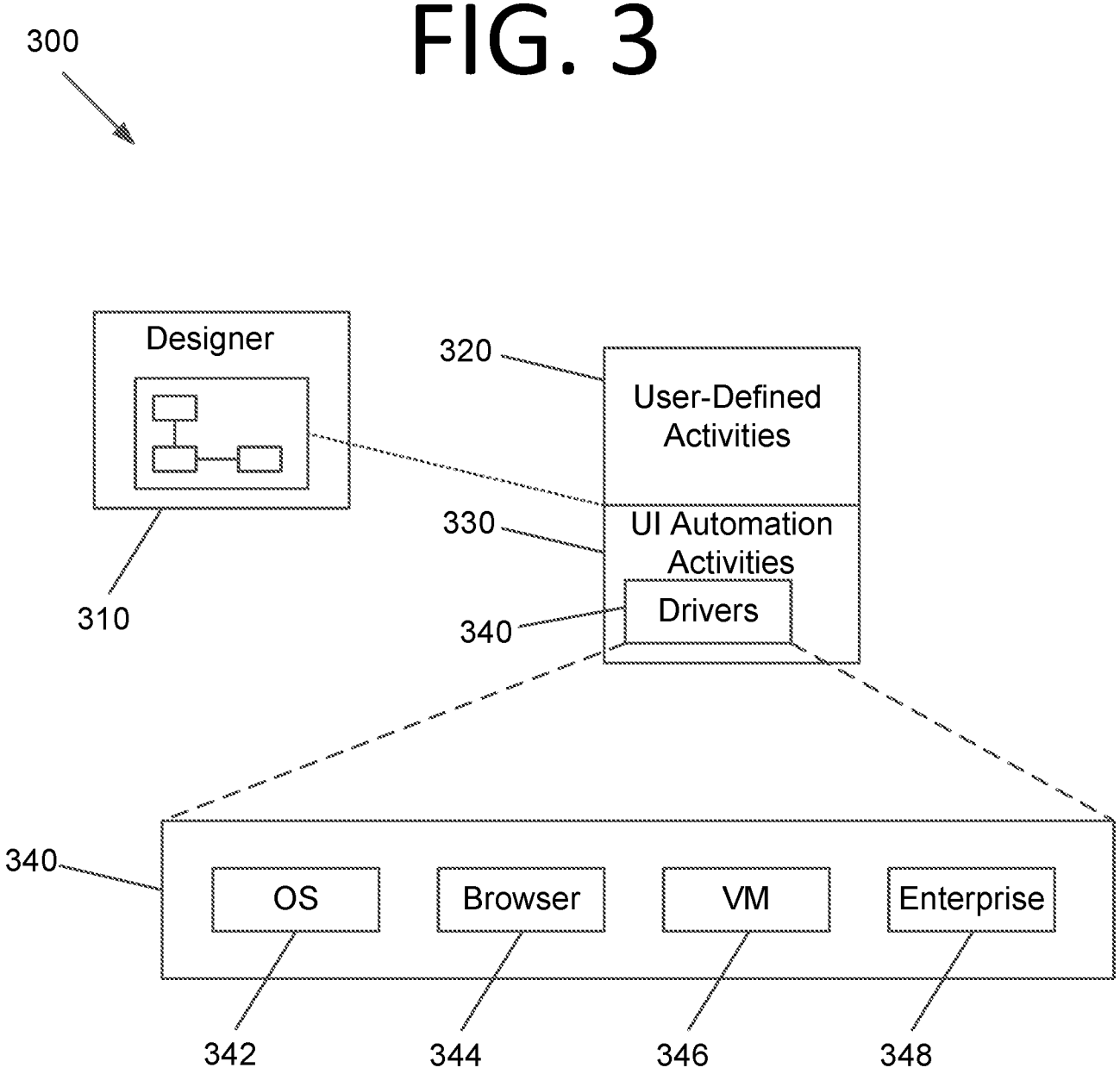

300
Designer
310
320
User-Defined Activities
330
UI Automation Activities
Drivers
340
340
OS
Browser
VM
Enterprise
342
344
346
348

FIG. 4

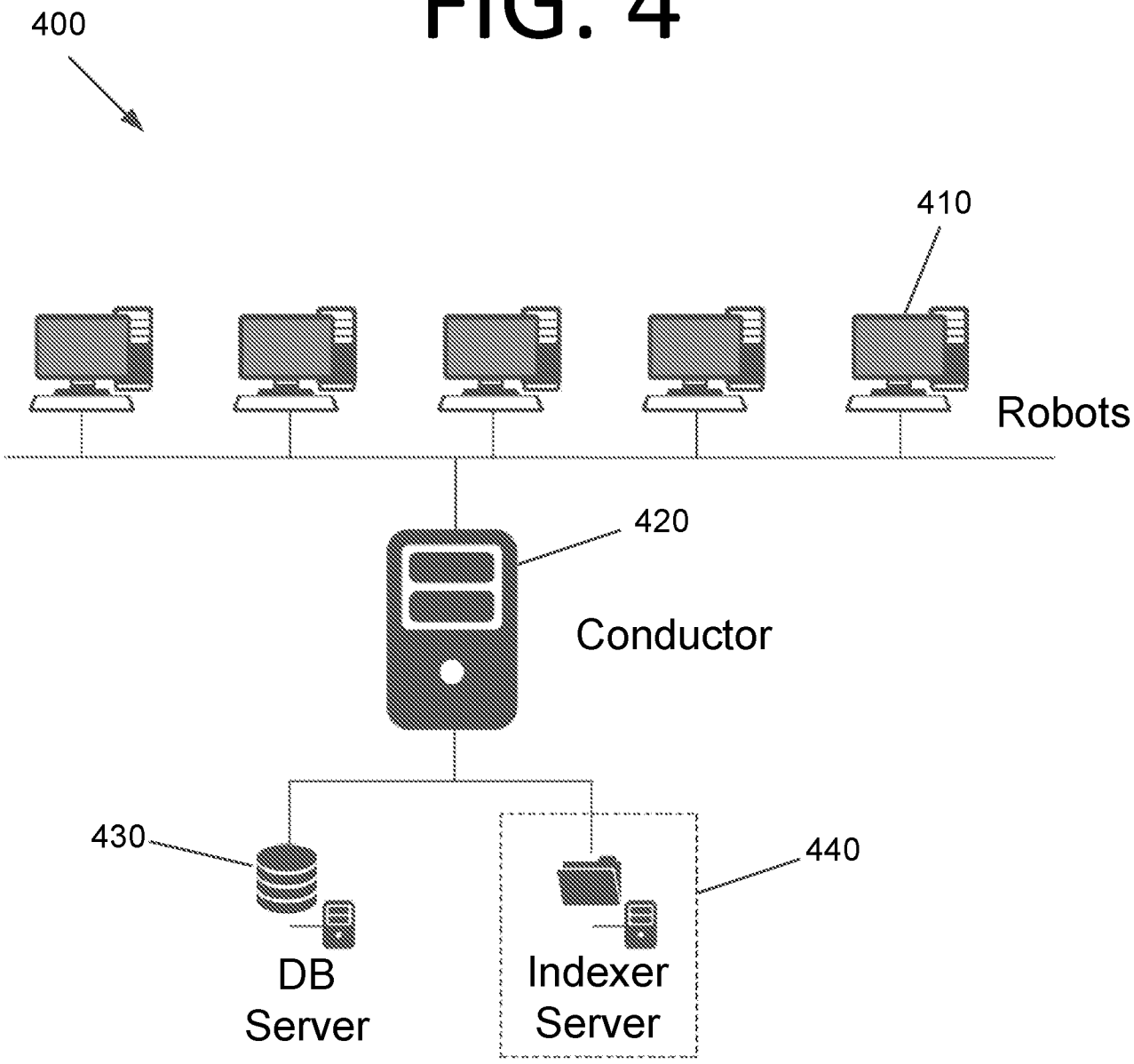

400
410
Robots
420
Conductor
430
DB
Server
440
Indexer
Server

FIG. 5

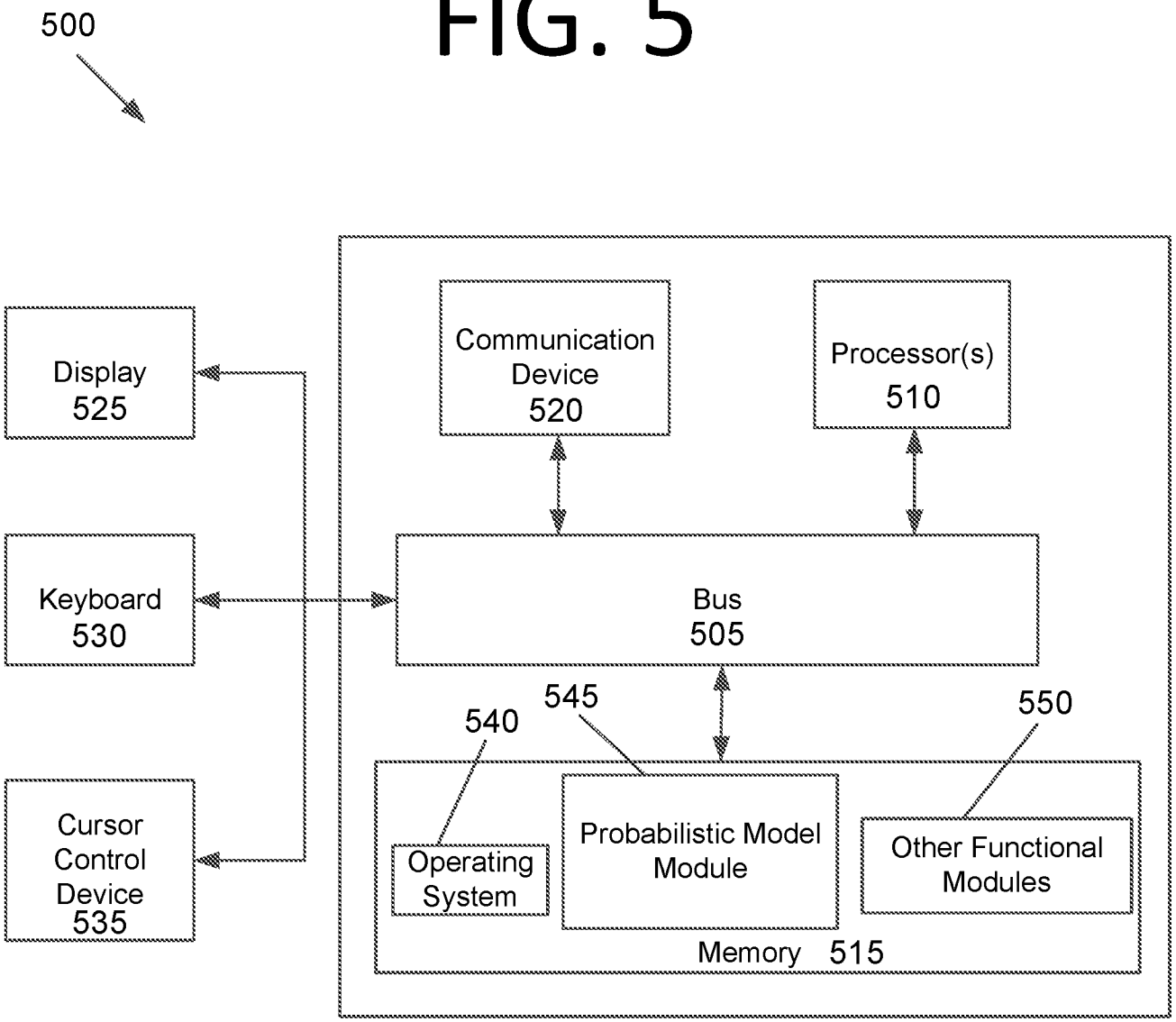

500
Display
525
Keyboard
530
Cursor
Control
Device
535
Communication
Device
520
Processor(s)
510
Bus
505
545
540
550
Operating
System
Probabilistic Model
Module
Other Functional
Modules
Memory 515

FIG. 6

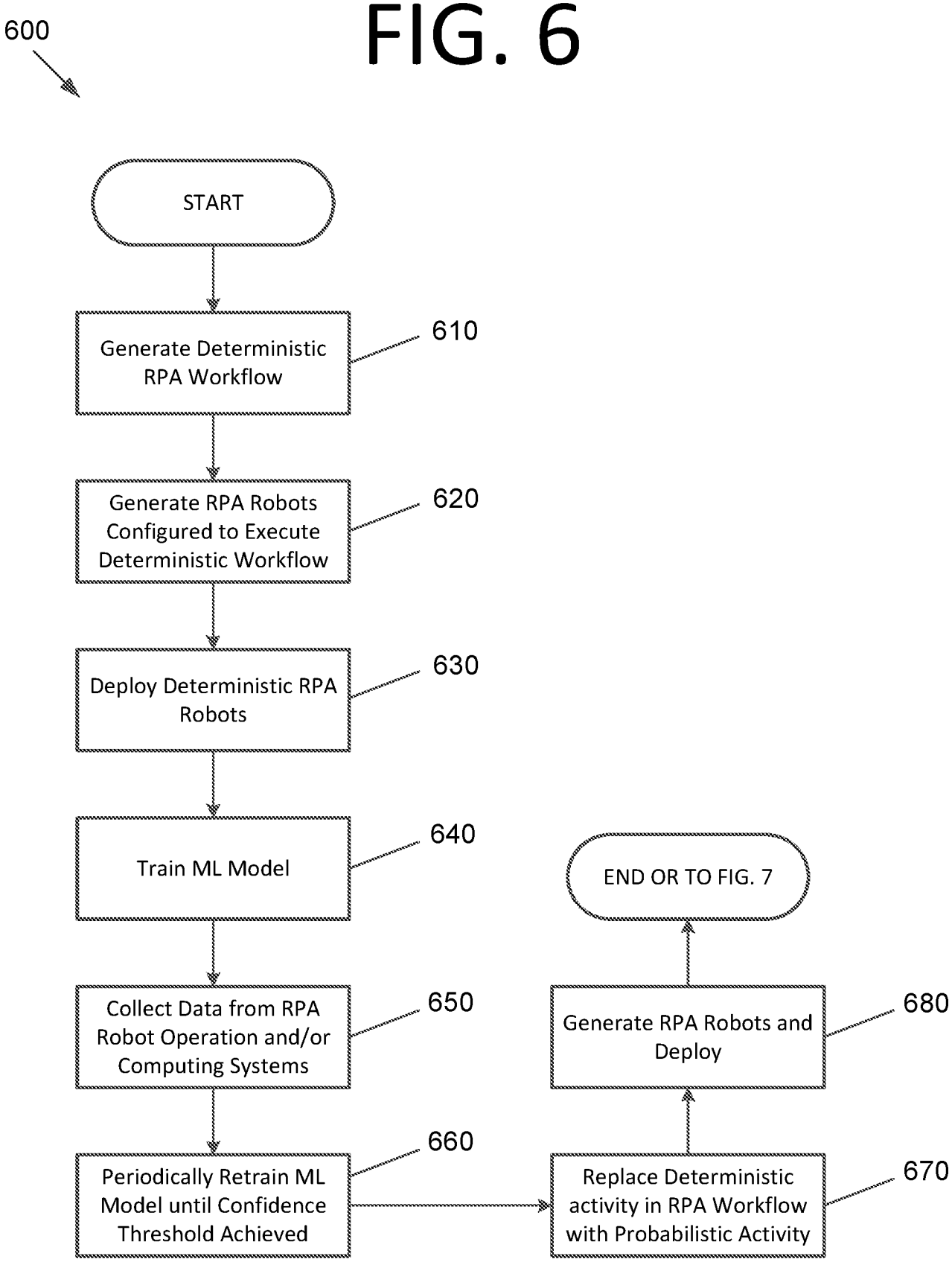

600
START
Generate Deterministic RPA Workflow
610
Generate RPA Robots Configured to Execute Deterministic Workflow
620
Deploy Deterministic RPA Robots
630
Train ML Model
640
Collect Data from RPA Robot Operation and/or Computing Systems
650
Periodically Retrain ML Model until Confidence Threshold Achieved
660
Replace Deterministic activity in RPA Workflow with Probabilistic Activity
670
Generate RPA Robots and Deploy
680
END OR TO FIG. 7

FIG. 7

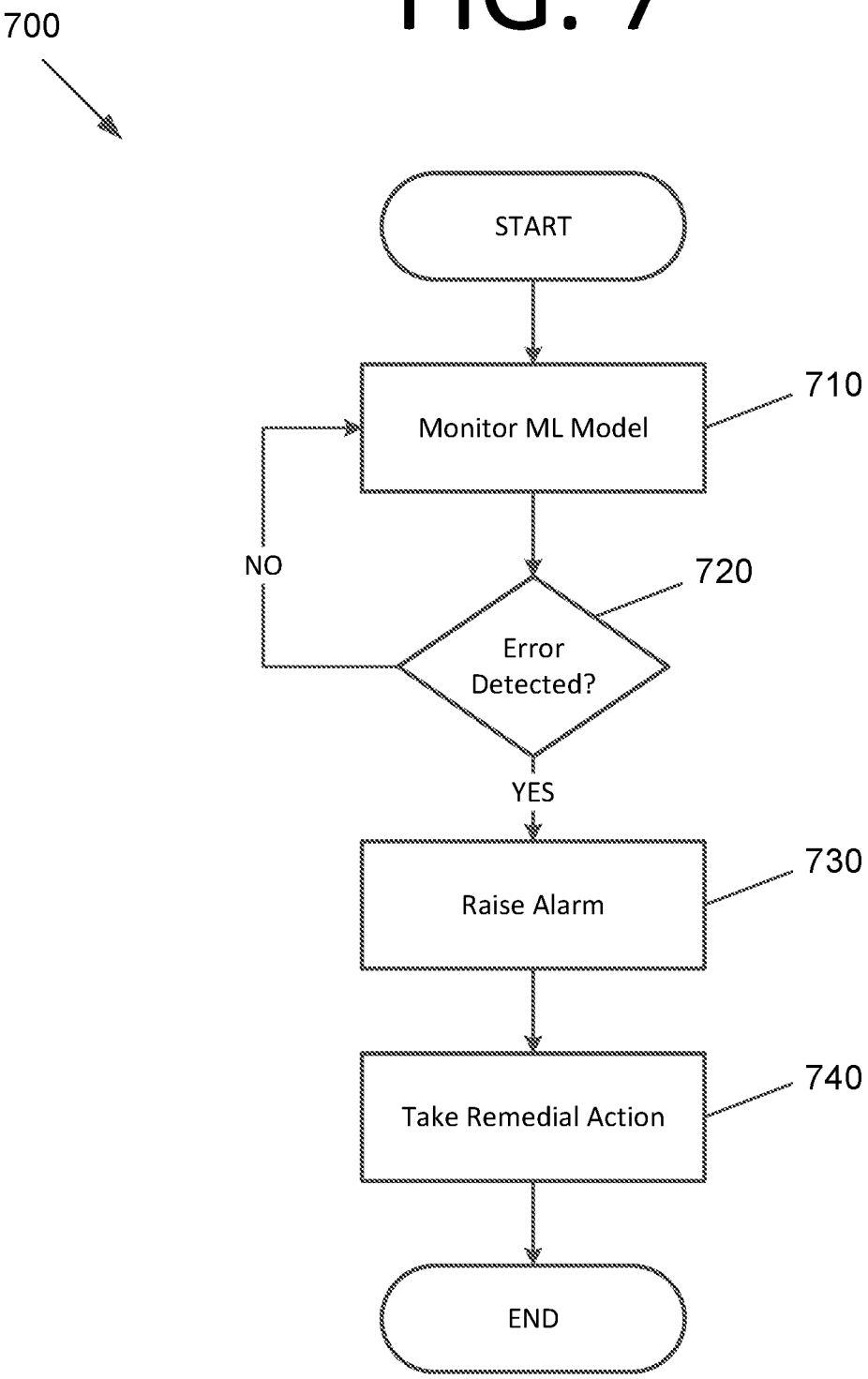
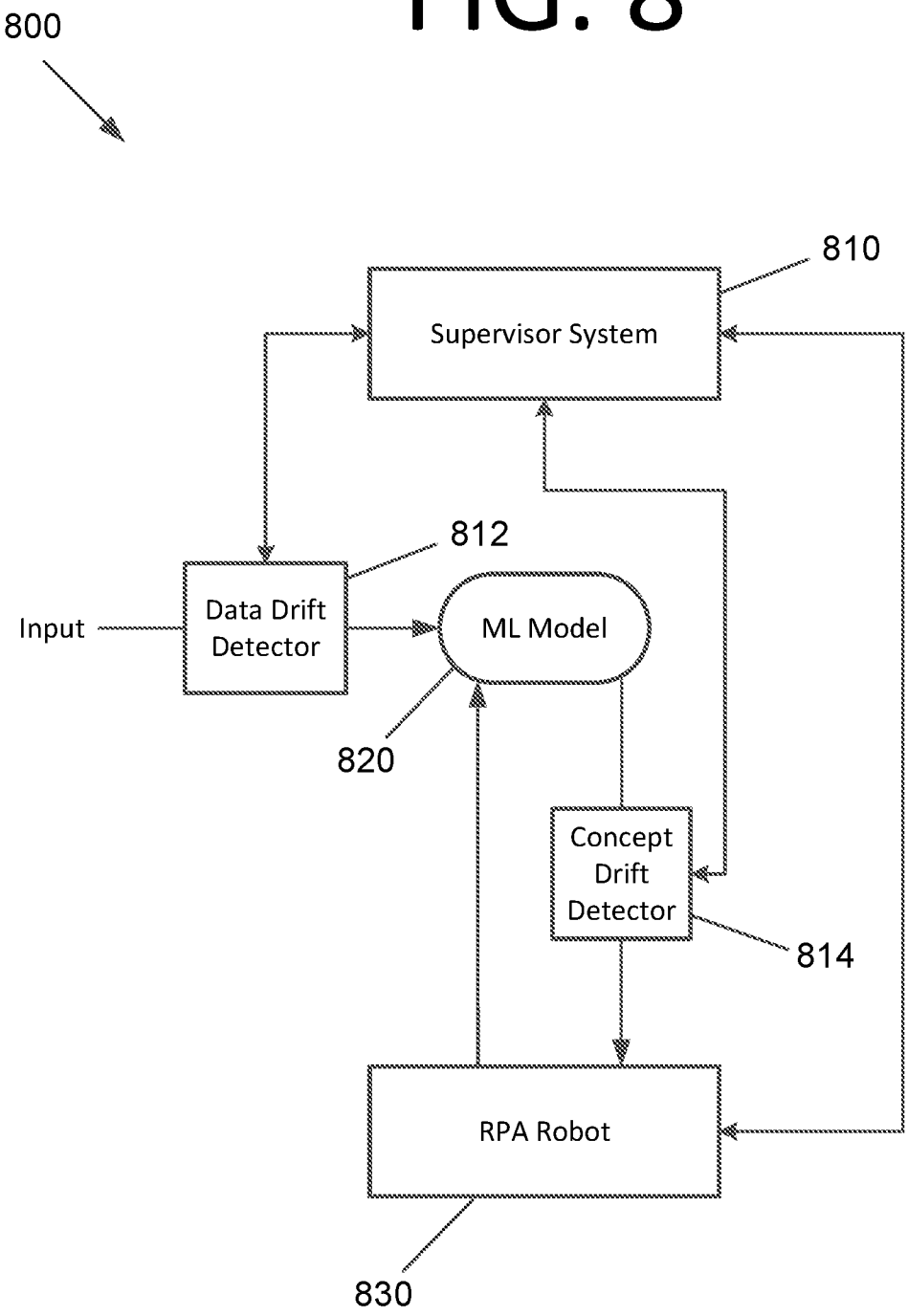

700
START
Monitor ML Model
710
NO
720
Error Detected?
YES
Raise Alarm
730
Take Remedial Action
740
END

FIG. 8


800
810
Supervisor System
812
Input
Data Drift Detector
ML Model
820
Concept Drift Detector
814
RPA Robot
830

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2020/046951

A. CLASSIFICATION OF SUBJECT MATTER

G06Q 10/06(2012.01)i, G06Q 10/10(2012.01)i, B25J 9/16(2006.01)i, G06N 20/00(2019.01)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)
G06Q 10/06; G06F 16/9032; G06F 17/30; G06F 19/00; G06N 5/02; G10L 15/22; G06Q 10/10; B25J 9/16; G06N 20/00

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: robotic process automation(RPA), probabilistic, workflow, machine learning, replace

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2018-0197123 A1 (SUTHERLAND GLOBAL SERVICES INC.) 12 July 2018 See paragraphs [0046], [0059], [0064], [0080], claims 1, 30, 33 and figures 1-4, 13.	1-20
Y	US 2019-0180746 A1 (ACCENTURE GLOBAL SOLUTIONS LIMITED) 13 June 2019 See paragraphs [0012], [0023], [0091], [0109], [0116]-[0117], claims 5, 17 and figure 7.	1-20
Y	US 2018-0341688 A1 (MPHASIS LIMITED) 29 November 2018 See paragraphs [0012], [0026]-[0036] and claims 1-5.	6-10,15-20
A	US 2019-0066013 A1 (ACCENTURE GLOBAL SOLUTIONS LIMITED) 28 February 2019 See paragraphs [0045]-[0053], claims 1, 4-5 and figures 1-5.	1-20
A	KR 10-2006-0063561 A (ELECTRONICS AND TELECOMMUNICATIONS RESEARCH INSTITUTE) 12 June 2006 See claims 1-2 and figures 1-2.	1-20

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

Date of the actual completion of the international search	Date of mailing of the international search report
27 November 2020 (27.11.2020)	**27 November 2020 (27.11.2020)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
KANG MIN JEONG
Telephone No. +82-42-481-8131

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2020/046951

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2018-0197123 A1	12/07/2018	GB 2558676 A	18/07/2018
		GB 2558676 B	21/10/2020
US 2019-0180746 A1	13/06/2019	AU 2018-274925 A1	27/06/2019
		AU 2020-204266 A1	16/07/2020
		US 10452674 B2	22/10/2019
US 2018-0341688 A1	29/11/2018	EP 3407217 A1	28/11/2018
		US 10726038 B2	28/07/2020
US 2019-0066013 A1	28/02/2019	CN 109426543 A	05/03/2019
KR 10-2006-0063561 A	12/06/2006	KR 10-0611101 B1	09/08/2006