



- (51) International Patent Classification:
H04L 67/12 (2022.01) *H04L 12/18* (2006.01)
- (21) International Application Number:
PCT/US2023/084872
- (22) International Filing Date:
19 December 2023 (19.12.2023)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
18/205,664 05 June 2023 (05.06.2023) US
- (63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 18/205,664 (CON)
Filed on 05 June 2023 (05.06.2023)
- (71) Applicant: **INTEL CORPORATION** [US/US]; 2200 Mission College Blvd., Santa Clara, California 95054 (US).
- (72) Inventors: **DOSHI, Kshitij Arun**; 2101 E. Balboa Drive, Tempe, Arizona 85282 (US). **GUIM BERNAT, Francesc**; Casonova 148, 08036 Barcelona Barcelona (ES). **CHERUVU, Sunil**; 1733 W. Myrna Ln., Tempe, Arizona 85284 (US). **SMITH, Ned M.**; 375 S.W. Delta Drive, Beaverton, Oregon 97006 (US). **CARRANZA, Marcos E.**; 6194 NW Snowlily Dr., Portland, Oregon 97229 (US).

(74) Agent: **PERDOK, Monique M.** et al.; Schwegman Lundberg & Woessner, P.A., P.O. Box 2938, Minneapolis, Minnesota 55402 (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, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, 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, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, 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, ME, 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: CONSENSUS-BASED NAMED FUNCTION EXECUTION

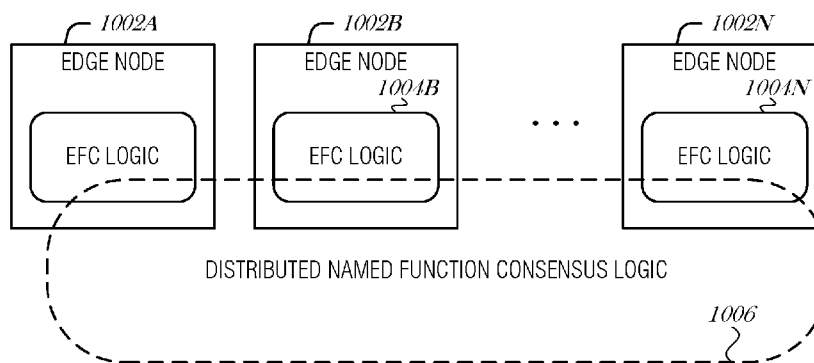


FIG. 10

(57) Abstract: Various systems and methods for providing consensus-based named function execution are described herein. A system is configured to access an interest packet received from a user device, the interest packet including a function name of a function and a data payload; broadcast the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are configured to execute a respective instance of the function; receive a plurality of responses from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function; analyze the plurality of responses using a consensus protocol to identify a consensus result; and transmit the consensus result to the user device.



CONSENSUS-BASED NAMED FUNCTION EXECUTION

PRIORITY APPLICATION

- 5 **[0001]** This application claims the priority to U.S. Application Serial Number 18/205,664, filed June 5, 2023, which is incorporated herein by reference in its entirety.

BACKGROUND

- 10 **[0002]** Edge computing, at a general level, refers to the transition of compute and storage resources closer to endpoint devices (e.g., consumer computing devices, user equipment, etc.) in order to optimize total cost of ownership, reduce application latency, improve service capabilities, and improve compliance with security or data privacy requirements. Edge computing may, in some scenarios, provide a cloud-like distributed service that offers orchestration and management for applications among many types of storage and compute
- 15 resources. As a result, some implementations of edge computing have been referred to as the “edge cloud” or the “fog”, as powerful computing resources previously available only in large remote data centers are moved closer to endpoints and made available for use by consumers at the “edge” of the network.

- 20 **[0003]** Edge computing and related technologies attempt to provide reduced latency, increased responsiveness, and more available computing power than offered in traditional cloud network services and wide area network connections. However, the integration of mobility and dynamically launched services to some mobile use and device processing use cases has led to limitations and concerns with orchestration, functional coordination, and resource management, especially in complex mobility settings where many participants
- 25 (devices, hosts, tenants, service providers, operators) are involved.

- In a similar manner, Internet of Things (IoT) networks and devices are designed to offer a distributed compute arrangement, from a variety of endpoints. IoT devices are physical or virtualized objects that may communicate on a network, and may include sensors, actuators, and other input/output components, which may be used to collect data or perform actions in a
- 30 real world environment. For example, IoT devices may include low-powered endpoint devices that are embedded or attached to everyday things, such as buildings, vehicles, packages, etc., to provide an additional level of artificial sensory perception of those things. Recently, IoT devices have become more popular and thus applications using these devices have proliferated.

[0004] The deployment of various Edge, Fog, and IoT networks, devices, and services have introduced a number of advanced use cases and scenarios occurring at and towards the edge of the network.

5

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] In the drawings, which are not necessarily drawn to scale, like numerals may describe similar components in different views. Like numerals having different letter suffixes may represent different instances of similar components. Some embodiments are illustrated by way of example, and not limitation, in the figures of the accompanying drawings in which:

10 [0006] FIG. 1 illustrates an overview of an Edge cloud configuration for Edge computing;

[0007] FIG. 2 illustrates operational layers among endpoints, an Edge cloud, and cloud computing environments;

[0008] FIG. 3 illustrates an example approach for networking and services in an Edge computing system;

15 [0009] FIG. 4 illustrates deployment of a virtual Edge configuration in an Edge computing system operated among multiple Edge nodes and multiple tenants;

[0010] FIG. 5 illustrates various compute arrangements deploying containers in an Edge computing system;

20 [0011] FIG. 6A provides an overview of example components for compute deployed at a compute node in an Edge computing system;

[0012] FIG. 6B provides a further overview of example components within a computing device in an Edge computing system;

25 [0013] FIG. 7 illustrates an example software distribution platform to distribute software, such as the example computer readable instructions of FIG. 6B, to one or more devices, according to an embodiment;

[0014] FIG. 8 illustrates an example information centric network (ICN), according to an embodiment;

[0015] FIG. 9 is a diagram illustrating an operating environment, according to an embodiment, according to an embodiment;

30 [0016] FIG. 10 is a diagram illustrating edge nodes participating in function consensus blockchain, according to an embodiment;

[0017] FIG. 11 is a block diagram illustrating edge function consensus (EFC) logic to implement function consensus blockchains, according to an embodiment;

[0018] FIG. 12 is a diagram illustrating a consensus-based named function networking system, according to an embodiment; and

[0019] FIG. 13 is a flowchart illustrating a method for coordinating named function execution in a named function network, according to an embodiment.

5

DETAILED DESCRIPTION

[0020] In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of some example embodiments. It will be evident, however, to one skilled in the art that the present disclosure may be practiced without these specific details.

[0021] NFN (Named Function Networking) is an extension of Information Centric Networking (ICN) that performs in-network resolution of functional expressions instead of mere content retrieval of a single name, as in ICN. In NFN, a service receives an interest packet with a function name, and the service provides name resolution to obtain the computational result. The name resolution is transparent to the client and allows for simpler interfaces. Use of NFN also disassociates functions from function providers so that the provider is not the sole basis for choosing a particular function. The function is executed by a node in the named function network and the results are returned to the client that issued the NFN interest packet.

[0022] In current NFN networks, entities (e.g., devices, services, user agents, etc.) that access servers that perform the named computations/functions must implicitly trust the servers. However, specific instances of named functions may differ in their security, performance, safety, reliability, or other properties based on how the named function is implemented by a given server. Additionally, variant versions of the same function may produce different results based on data precision of variables used in calculations, lack of testing, or other variances in hardware or software platforms at a server. As such, when a named function is accessed in an NFN and only one instance of the function is executed, there is no guarantee that the results of that particular execution are accurate or consistent with executions of other instances of the function that result from different named function calls. Hence, despite the named functions being executed by trusted servers in the NFN, the results may differ in accuracy. The present systems and methods address this by building in a mechanism to establish a consensus among various function servers about a function's results.

[0023] Blockchain technologies use various consensus protocols or consensus algorithms to ensure data validity. Consensus algorithms are used in computer science to find agreement on a

single result among distributed processes or systems. Consensus algorithms are used in distributed ledger technologies (e.g., blockchain). Example consensus algorithms include Proof of Work (PoW), Proof of Stake (PoS), Delegated Proof of Stake (DPoS), Proof of Burn (PoB), Proof of Capacity (PoC), or Proof of Elapsed Time (PoET), each of which provide security and prevent or disincentivize unauthorized entities from validating false transactions.

[0024] In a blockchain that uses a PoW consensus algorithm, such as Bitcoin, multiple nodes in the blockchain network (referred to as miners) simultaneously attempt to solve a difficult mathematical problem. Because of the design of the problem, the answer has to be calculated using brute force. The nodes continue to test unique values (known as nonces) until a suitable value is found that solves the problem. The node that manages to solve the problem announces the proposed solution to the rest of the nodes in the blockchain network. All of the other nodes in the blockchain network review the proposed solution to verify its accuracy. They may perform this review by recalculating the problem with the proposed solution. If more than 51% of the nodes agree that the proposed solution is correct (e.g., consensus), then the block is added to the blockchain and the node that solved the problem is rewarded (e.g., with Bitcoin currency).

[0025] In the present systems and methods, a function execution result is used as a proof of work from a node in a blockchain network. In order to verify the validity of the function's result, the function is executed on multiple nodes in a blockchain network. If the function's results are the same across a threshold number of nodes (or within some acceptable deviation), then the function's results are considered to be accurate and validated. The function's results may be stored in an immutable ledger (e.g., blockchain) after being verified.

[0026] As such, according to various embodiments described herein, multiple network server nodes are used to evaluate a set of instances of a named function $F = \{fn1, fn2, \dots, fnx\}$. Each of the instances of F are designed to analyze data or solve the same problem. The instances of F are masked behind a common interface that mimics NFN protocols. That is, the function F is called by name, but any number of instances of the function may be executed on respective nodes. This forms a consensus-based function result.

[0027] There are several advantages using the consensus-based function execution described herein. For example, consensus-based resolution provides the ability to overcome the debug/verification challenges in an edge NFN architecture. This benefit also applies to work that is performed in the FaaS (function-as-a-service) model of operation. This is essential for ensuring that the functions are correctly and completely tested on distributed, decentralized, and federated infrastructure.

[0028] There are also testing and verification benefits. Behavior of functions invoked as APIs (either in NFN or FaaS models) can be characterized in a controlled environment. This breaks the reliance on establishing trust in a single or small number of entities (to preclude possible collusion among untrusted entities). As a result, the contribution of a diverse
5 population of entities can be used to evaluate edge functions, which yields robustness and scale in testing.

[0029] Additionally, there are production deployment benefits. When the mechanism is used in a production network, the blockchain consensus also provides redundancy. This is vital when high reliability, availability or resiliency are needed.

10 [0030] Blockchain operations may add additional latency in exchange for added reliability. Nevertheless, this mechanism considers ways to accelerate edge functions that need to meet certain key performance indicators (KPIs) typical for low-latency solutions for real-time use cases such as vehicle-to-vehicle (V2V) or vehicle-to-everything (V2X) implementations.

[0031] Additionally, the mechanism provides attestation benefits. The mechanism uses
15 blockchain-type consensus to generate reference values for functions. These reference values can serve as an attestation set of ‘known good values’ that a verifier can use to check evidence for authenticity, and as substantiation of capabilities. A supplier in a supply chain may provide manufacturer ‘known good values’ as well, that the blockchain accepts based on a consensus of manufacturer authenticity or reputation. A duplicitous supplier may represent a different set
20 of ‘known good values’ to different verifiers resulting in different attestation results. The blockchain consensus can detect duplicitous ‘known good values.’

[0032] A verifier may further evaluate trustworthiness of a function based on known-good, expected, or anomalistic telemetry associated with the execution of the function. Further details are provided in the examples and figures described below.

25 [0033] FIG. 1 is a block diagram 100 showing an overview of a configuration for Edge computing, which includes a layer of processing referred to in many of the following examples as an “Edge cloud”. As shown, the Edge cloud 110 is co-located at an Edge location, such as an access point or base station 140, a local processing hub 150, or a central office 120, and thus may include multiple entities, devices, and equipment instances. The Edge cloud 110 is located
30 much closer to the endpoint (consumer and producer) data sources 160 (e.g., autonomous vehicles 161, user equipment 162, business and industrial equipment 163, video capture devices 164, drones 165, smart cities and building devices 166, sensors and IoT devices 167, etc.) than the cloud data center 130. Compute, memory, and storage resources which are offered at the edges in the Edge cloud 110 are critical to providing ultra-low latency response

times for services and functions used by the endpoint data sources 160 as well as reduce network backhaul traffic from the Edge cloud 110 toward cloud data center 130 thus improving energy consumption and overall network usages among other benefits.

[0034] Compute, memory, and storage are scarce resources, and generally decrease
5 depending on the Edge location (e.g., fewer processing resources being available at consumer endpoint devices, than at a base station, than at a central office). However, the closer that the Edge location is to the endpoint (e.g., user equipment (UE)), the more that space and power is often constrained. Thus, Edge computing attempts to reduce the amount of resources needed for network services, through the distribution of more resources which are located closer both
10 geographically and in network access time. In this manner, Edge computing attempts to bring the compute resources to the workload data where appropriate, or, bring the workload data to the compute resources.

[0035] The following describes aspects of an Edge cloud architecture that covers multiple potential deployments and addresses restrictions that some network operators or service
15 providers may have in their own infrastructures. These include, variation of configurations based on the Edge location (because edges at a base station level, for instance, may have more constrained performance and capabilities in a multi-tenant scenario); configurations based on the type of compute, memory, storage, fabric, acceleration, or like resources available to Edge locations, tiers of locations, or groups of locations; the service, security, and management and
20 orchestration capabilities; and related objectives to achieve usability and performance of end services. These deployments may accomplish processing in network layers that may be considered as “near Edge”, “close Edge”, “local Edge”, “middle Edge”, or “far Edge” layers, depending on latency, distance, and timing characteristics.

[0036] Edge computing is a developing paradigm where computing is performed at or
25 closer to the “Edge” of a network, typically through the use of a compute platform (e.g., x86 or ARM compute hardware architecture) implemented at base stations, gateways, network routers, or other devices which are much closer to endpoint devices producing and consuming the data. For example, Edge gateway servers may be equipped with pools of memory and storage resources to perform computation in real-time for low latency use-cases (e.g.,
30 autonomous driving or video surveillance) for connected client devices. Or as an example, base stations may be augmented with compute and acceleration resources to directly process service workloads for connected user equipment, without further communicating data via backhaul networks. Or as another example, central office network management hardware may be replaced with standardized compute hardware that performs virtualized network functions and

offers compute resources for the execution of services and consumer functions for connected devices. Within Edge computing networks, there may be scenarios in services which the compute resource will be “moved” to the data, as well as scenarios in which the data will be “moved” to the compute resource. Or as an example, base station compute, acceleration and network resources can provide services in order to scale to workload demands on an as needed basis by activating dormant capacity (subscription, capacity on demand) in order to manage corner cases, emergencies or to provide longevity for deployed resources over a significantly longer implemented lifecycle.

[0037] FIG. 2 illustrates operational layers among endpoints, an Edge cloud, and cloud computing environments. Specifically, FIG. 2 depicts examples of computational use cases 205, utilizing the Edge cloud 110 among multiple illustrative layers of network computing. The layers begin at an endpoint (devices and things) layer 200, which accesses the Edge cloud 110 to conduct data creation, analysis, and data consumption activities. The Edge cloud 110 may span multiple network layers, such as an Edge devices layer 210 having gateways, on-premise servers, or network equipment (nodes 215) located in physically proximate Edge systems; a network access layer 220, encompassing base stations, radio processing units, network hubs, regional data centers (DC), or local network equipment (equipment 225); and any equipment, devices, or nodes located therebetween (in layer 212, not illustrated in detail). The network communications within the Edge cloud 110 and among the various layers may occur via any number of wired or wireless mediums, including via connectivity architectures and technologies not depicted.

[0038] Examples of latency, resulting from network communication distance and processing time constraints, may range from less than a millisecond (ms) when among the endpoint layer 200, under 5 ms at the Edge devices layer 210, to even between 10 to 40 ms when communicating with nodes at the network access layer 220. Beyond the Edge cloud 110 are core network 230 and cloud data center 240 layers, each with increasing latency (e.g., between 50-60 ms at the core network layer 230, to 100 or more ms at the cloud data center layer). As a result, operations at a core network data center 235 or a cloud data center 245, with latencies of at least 50 to 100 ms or more, will not be able to accomplish many time-critical functions of the use cases 205. Each of these latency values are provided for purposes of illustration and contrast; it will be understood that the use of other access network mediums and technologies may further reduce the latencies. In some examples, respective portions of the network may be categorized as “close Edge”, “local Edge”, “near Edge”, “middle Edge”, or “far Edge” layers, relative to a network source and destination. For instance, from the

perspective of the core network data center 235 or a cloud data center 245, a central office or content data network may be considered as being located within a “near Edge” layer (“near” to the cloud, having high latency values when communicating with the devices and endpoints of the use cases 205), whereas an access point, base station, on-premise server, or network gateway may be considered as located within a “far Edge” layer (“far” from the cloud, having low latency values when communicating with the devices and endpoints of the use cases 205). It will be understood that other categorizations of a particular network layer as constituting a “close”, “local”, “near”, “middle”, or “far” Edge may be based on latency, distance, number of network hops, or other measurable characteristics, as measured from a source in any of the network layers 200-240.

[0039] The various use cases 205 may access resources under usage pressure from incoming streams, due to multiple services utilizing the Edge cloud. To achieve results with low latency, the services executed within the Edge cloud 110 balance varying requirements in terms of: (a) Priority (throughput or latency) and Quality of Service (QoS) (e.g., traffic for an autonomous car may have higher priority than a temperature sensor in terms of response time requirement; or, a performance sensitivity/bottleneck may exist at a compute/accelerator, memory, storage, or network resource, depending on the application); (b) Reliability and Resiliency (e.g., some input streams need to be acted upon and the traffic routed with mission-critical reliability, where as some other input streams may be tolerate an occasional failure, depending on the application); and (c) Physical constraints (e.g., power, cooling and form-factor).

[0040] The end-to-end service view for these use cases involves the concept of a service-flow and is associated with a transaction. The transaction details the overall service requirement for the entity consuming the service, as well as the associated services for the resources, workloads, workflows, and business functional and business level requirements. The services executed with the “terms” described may be managed at each layer in a way to assure real time, and runtime contractual compliance for the transaction during the lifecycle of the service. When a component in the transaction is missing its agreed to SLA, the system as a whole (components in the transaction) may provide the ability to (1) understand the impact of the SLA violation, and (2) augment other components in the system to resume overall transaction SLA, and (3) implement steps to remediate.

[0041] Thus, with these variations and service features in mind, Edge computing within the Edge cloud 110 may provide the ability to serve and respond to multiple applications of the use cases 205 (e.g., object tracking, video surveillance, connected cars, etc.) in real-time or near

real-time, and meet ultra-low latency requirements for these multiple applications. These advantages enable a whole new class of applications (Virtual Network Functions (VNFs), Function as a Service (FaaS), Edge as a Service (EaaS), standard processes, etc.), which cannot leverage conventional cloud computing due to latency or other limitations.

5 [0042] However, with the advantages of Edge computing comes the following caveats. The devices located at the Edge are often resource constrained and therefore there is pressure on usage of Edge resources. Typically, this is addressed through the pooling of memory and storage resources for use by multiple users (tenants) and devices. The Edge may be power and cooling constrained and therefore the power usage needs to be accounted for by the
10 applications that are consuming the most power. There may be inherent power-performance tradeoffs in these pooled memory resources, as many of them are likely to use emerging memory technologies, where more power requires greater memory bandwidth. Likewise, improved security of hardware and root of trust trusted functions are also required, because Edge locations may be unmanned and may even need permissioned access (e.g., when housed
15 in a third-party location). Such issues are magnified in the Edge cloud 110 in a multi-tenant, multi-owner, or multi-access setting, where services and applications are requested by many users, especially as network usage dynamically fluctuates and the composition of the multiple stakeholders, use cases, and services changes.

[0043] At a more generic level, an Edge computing system may be described to encompass
20 any number of deployments at the previously discussed layers operating in the Edge cloud 110 (network layers 200-240), which provide coordination from client and distributed computing devices. One or more Edge gateway nodes, one or more Edge aggregation nodes, and one or more core data centers may be distributed across layers of the network to provide an implementation of the Edge computing system by or on behalf of a telecommunication service
25 provider (“telco”, or “TSP”), internet-of-things service provider, cloud service provider (CSP), enterprise entity, or any other number of entities. Various implementations and configurations of the Edge computing system may be provided dynamically, such as when orchestrated to meet service objectives.

[0044] Consistent with the examples provided herein, a client compute node may be
30 embodied as any type of endpoint component, device, appliance, or other thing capable of communicating as a producer or consumer of data. Further, the label “node” or “device” as used in the Edge computing system does not necessarily mean that such node or device operates in a client or agent/minion/follower role; rather, any of the nodes or devices in the Edge computing system refer to individual entities, nodes, or subsystems which include

discrete or connected hardware or software configurations to facilitate or use the Edge cloud 110.

[0045] As such, the Edge cloud 110 is formed from network components and functional features operated by and within Edge gateway nodes, Edge aggregation nodes, or other Edge compute nodes among network layers 210-230. The Edge cloud 110 thus may be embodied as any type of network that provides Edge computing and/or storage resources which are proximately located to radio access network (RAN) capable endpoint devices (e.g., mobile computing devices, IoT devices, smart devices, etc.), which are discussed herein. In other words, the Edge cloud 110 may be envisioned as an “Edge” which connects the endpoint devices and traditional network access points that serve as an ingress point into service provider core networks, including mobile carrier networks (e.g., Global System for Mobile Communications (GSM) networks, Long-Term Evolution (LTE) networks, 5G/6G networks, etc.), while also providing storage and/or compute capabilities. Other types and forms of network access (e.g., Wi-Fi, long-range wireless, wired networks including optical networks) may also be utilized in place of or in combination with such 3GPP carrier networks.

[0046] The network components of the Edge cloud 110 may be servers, multi-tenant servers, appliance computing devices, and/or any other type of computing devices. For example, the Edge cloud 110 may include an appliance computing device that is a self-contained electronic device including a housing, a chassis, a case or a shell. In some circumstances, the housing may be dimensioned for portability such that it can be carried by a human and/or shipped. Example housings may include materials that form one or more exterior surfaces that partially or fully protect contents of the appliance, in which protection may include weather protection, hazardous environment protection (e.g., EMI, vibration, extreme temperatures), and/or enable submergibility. Example housings may include power circuitry to provide power for stationary and/or portable implementations, such as AC power inputs, DC power inputs, AC/DC or DC/AC converter(s), power regulators, transformers, charging circuitry, batteries, wired inputs and/or wireless power inputs. Example housings and/or surfaces thereof may include or connect to mounting hardware to enable attachment to structures such as buildings, telecommunication structures (e.g., poles, antenna structures, etc.) and/or racks (e.g., server racks, blade mounts, etc.). Example housings and/or surfaces thereof may support one or more sensors (e.g., temperature sensors, vibration sensors, light sensors, acoustic sensors, capacitive sensors, proximity sensors, etc.). One or more such sensors may be contained in, carried by, or otherwise embedded in the surface and/or mounted to the surface of the appliance. Example housings and/or surfaces thereof may support mechanical connectivity,

such as propulsion hardware (e.g., wheels, propellers, etc.) and/or articulating hardware (e.g., robot arms, pivotable appendages, etc.). In some circumstances, the sensors may include any type of input devices such as user interface hardware (e.g., buttons, switches, dials, sliders, etc.). In some circumstances, example housings include output devices contained in, carried
5 by, embedded therein and/or attached thereto. Output devices may include displays, touchscreens, lights, LEDs, speakers, I/O ports (e.g., USB), etc. In some circumstances, Edge devices are devices presented in the network for a specific purpose (e.g., a traffic light), but may have processing and/or other capacities that may be utilized for other purposes. Such Edge devices may be independent from other networked devices and may be provided with a
10 housing having a form factor suitable for its primary purpose; yet be available for other compute tasks that do not interfere with its primary task. Edge devices include Internet of Things devices. The appliance computing device may include hardware and software components to manage local issues such as device temperature, vibration, resource utilization, updates, power issues, physical and network security, etc. Example hardware for implementing
15 an appliance computing device is described in conjunction with FIG. 6B. The Edge cloud 110 may also include one or more servers and/or one or more multi-tenant servers. Such a server may include an operating system and implement a virtual computing environment. A virtual computing environment may include a hypervisor managing (e.g., spawning, deploying, destroying, etc.) one or more virtual machines, one or more containers, etc. Such virtual
20 computing environments provide an execution environment in which one or more applications and/or other software, code or scripts may execute while being isolated from one or more other applications, software, code or scripts.

[0047] In FIG. 3, various client endpoints 310 (in the form of mobile devices, computers, autonomous vehicles, business computing equipment, industrial processing equipment)
25 exchange requests and responses that are specific to the type of endpoint network aggregation. For instance, client endpoints 310 may obtain network access via a wired broadband network, by exchanging requests and responses 322 through an on-premise network system 332. Some client endpoints 310, such as mobile computing devices, may obtain network access via a wireless broadband network, by exchanging requests and responses 324 through an access
30 point (e.g., cellular network tower) 334. Some client endpoints 310, such as autonomous vehicles may obtain network access for requests and responses 326 via a wireless vehicular network through a street-located network system 336. However, regardless of the type of network access, the TSP may deploy aggregation points 342, 344 within the Edge cloud 110 to aggregate traffic and requests. Thus, within the Edge cloud 110, the TSP may deploy various

compute and storage resources, such as at Edge aggregation nodes 340, to provide requested content. The Edge aggregation nodes 340 and other systems of the Edge cloud 110 are connected to a cloud or data center 360, which uses a backhaul network 350 to fulfill higher-latency requests from a cloud/data center for websites, applications, database servers, etc.

5 Additional or consolidated instances of the Edge aggregation nodes 340 and the aggregation points 342, 344, including those deployed on a single server framework, may also be present within the Edge cloud 110 or other areas of the TSP infrastructure.

[0048] FIG. 4 illustrates deployment and orchestration for virtualized and container-based Edge configurations across an Edge computing system operated among multiple Edge nodes and multiple tenants (e.g., users, providers) which use such Edge nodes. Specifically, FIG. 4 depicts coordination of a first Edge node 422 and a second Edge node 424 in an Edge computing system 400, to fulfill requests and responses for various client endpoints 410 (e.g., smart cities / building systems, mobile devices, computing devices, business/logistics systems, industrial systems, etc.), which access various virtual Edge instances. Here, the virtual Edge instances 432, 434 provide Edge compute capabilities and processing in an Edge cloud, with access to a cloud/data center 440 for higher-latency requests for websites, applications, database servers, etc. However, the Edge cloud enables coordination of processing among multiple Edge nodes for multiple tenants or entities.

[0049] In the example of FIG. 4, these virtual Edge instances include: a first virtual Edge 432, offered to a first tenant (Tenant 1), which offers a first combination of Edge storage, computing, and services; and a second virtual Edge 434, offering a second combination of Edge storage, computing, and services. The virtual Edge instances 432, 434 are distributed among the Edge nodes 422, 424, and may include scenarios in which a request and response are fulfilled from the same or different Edge nodes. The configuration of the Edge nodes 422, 424 to operate in a distributed yet coordinated fashion occurs based on Edge provisioning functions 450. The functionality of the Edge nodes 422, 424 to provide coordinated operation for applications and services, among multiple tenants, occurs based on orchestration functions 460.

[0050] It should be understood that some of the devices in 410 are multi-tenant devices where Tenant 1 may function within a tenant1 'slice' while a Tenant 2 may function within a tenant2 slice (and, in further examples, additional or sub-tenants may exist; and each tenant may even be specifically entitled and transactionally tied to a specific set of features all the way day to specific hardware features). A trusted multi-tenant device may further contain a tenant specific cryptographic key such that the combination of key and slice may be considered

a “root of trust” (RoT) or tenant specific RoT. A RoT may further be computed dynamically composed using a DICE (Device Identity Composition Engine) architecture such that a single DICE hardware building block may be used to construct layered trusted computing base contexts for layering of device capabilities (such as a Field Programmable Gate Array (FPGA)). The RoT may further be used for a trusted computing context to enable a “fan-out” that is useful for supporting multi-tenancy. Within a multi-tenant environment, the respective Edge nodes 422, 424 may operate as security feature enforcement points for local resources allocated to multiple tenants per node. Additionally, tenant runtime and application execution (e.g., in instances 432, 434) may serve as an enforcement point for a security feature that creates a virtual Edge abstraction of resources spanning potentially multiple physical hosting platforms. Finally, the orchestration functions 460 at an orchestration entity may operate as a security feature enforcement point for marshalling resources along tenant boundaries.

[0051] Edge computing nodes may partition resources (memory, central processing unit (CPU), graphics processing unit (GPU), interrupt controller, input/output (I/O) controller, memory controller, bus controller, etc.) where respective partitionings may contain a RoT capability and where fan-out and layering according to a DICE model may further be applied to Edge Nodes. Cloud computing nodes often use containers, FaaS engines, Servlets, servers, or other computation abstraction that may be partitioned according to a DICE layering and fan-out structure to support a RoT context for each. Accordingly, the respective RoTs spanning devices 410, 422, and 440 may coordinate the establishment of a distributed trusted computing base (DTCB) such that a tenant-specific virtual trusted secure channel linking all elements end to end can be established.

[0052] Further, it will be understood that a container may have data or workload specific keys protecting its content from a previous Edge node. As part of migration of a container, a pod controller at a source Edge node may obtain a migration key from a target Edge node pod controller where the migration key is used to wrap the container-specific keys. When the container/pod is migrated to the target Edge node, the unwrapping key is exposed to the pod controller that then decrypts the wrapped keys. The keys may now be used to perform operations on container specific data. The migration functions may be gated by properly attested Edge nodes and pod managers (as described above).

[0053] In further examples, an Edge computing system is extended to provide for orchestration of multiple applications through the use of containers (a contained, deployable unit of software that provides code and needed dependencies) in a multi-owner, multi-tenant environment. A multi-tenant orchestrator may be used to perform key management, trust

anchor management, and other security functions related to the provisioning and lifecycle of the trusted ‘slice’ concept in FIG. 4. For instance, an Edge computing system may be configured to fulfill requests and responses for various client endpoints from multiple virtual Edge instances (and, from a cloud or remote data center). The use of these virtual Edge instances may support multiple tenants and multiple applications (e.g., augmented reality (AR)/virtual reality (VR), enterprise applications, content delivery, gaming, compute offload) simultaneously. Further, there may be multiple types of applications within the virtual Edge instances (e.g., normal applications; latency sensitive applications; latency-critical applications; user plane applications; networking applications; etc.). The virtual Edge instances may also be spanned across systems of multiple owners at different geographic locations (or, respective computing systems and resources which are co-owned or co-managed by multiple owners).

[0054] For instance, each Edge node 422, 424 may implement the use of containers, such as with the use of a container “pod” 426, 428 providing a group of one or more containers. In a setting that uses one or more container pods, a pod controller or orchestrator is responsible for local control and orchestration of the containers in the pod. Various Edge node resources (e.g., storage, compute, services, depicted with hexagons) provided for the respective Edge slices 432, 434 are partitioned according to the needs of each container.

[0055] With the use of container pods, a pod controller oversees the partitioning and allocation of containers and resources. The pod controller receives instructions from an orchestrator (e.g., orchestrator 460) that instructs the controller on how best to partition physical resources and for what duration, such as by receiving key performance indicator (KPI) targets based on SLA contracts. The pod controller determines which container requires which resources and for how long in order to complete the workload and satisfy the SLA. The pod controller also manages container lifecycle operations such as: creating the container, provisioning it with resources and applications, coordinating intermediate results between multiple containers working on a distributed application together, dismantling containers when workload completes, and the like. Additionally, a pod controller may serve a security role that prevents assignment of resources until the right tenant authenticates or prevents provisioning of data or a workload to a container until an attestation result is satisfied.

[0056] Also, with the use of container pods, tenant boundaries can still exist but in the context of each pod of containers. If each tenant specific pod has a tenant specific pod controller, there will be a shared pod controller that consolidates resource allocation requests to avoid typical resource starvation situations. Further controls may be provided to ensure

attestation and trustworthiness of the pod and pod controller. For instance, the orchestrator 460 may provision an attestation verification policy to local pod controllers that perform attestation verification. If an attestation satisfies a policy for a first tenant pod controller but not a second tenant pod controller, then the second pod could be migrated to a different Edge node that does
5 satisfy it. Alternatively, the first pod may be allowed to execute and a different shared pod controller is installed and invoked prior to the second pod executing.

[0057] FIG. 5 illustrates additional compute arrangements deploying containers in an Edge computing system. As a simplified example, system arrangements 510, 520 depict settings in which a pod controller (e.g., container managers 511, 521, and container orchestrator 531) is
10 adapted to launch containerized pods, functions, and functions-as-a-service instances through execution via compute nodes (515 in arrangement 510), or to separately execute containerized virtualized network functions through execution via compute nodes (523 in arrangement 520). This arrangement is adapted for use of multiple tenants in system arrangement 530 (using compute nodes 537), where containerized pods (e.g., pods 512), functions (e.g., functions 513,
15 VNFs 522, 536), and functions-as-a-service instances (e.g., FaaS instance 514) are launched within virtual machines (e.g., VMs 534, 535 for tenants 532, 533) specific to respective tenants (aside the execution of virtualized network functions). This arrangement is further adapted for use in system arrangement 540, which provides containers 542, 543, or execution of the various functions, applications, and functions on compute nodes 544, as coordinated by an
20 container-based orchestration system 541.

[0058] The system arrangements of depicted in FIG. 5 provides an architecture that treats VMs, Containers, and Functions equally in terms of application composition (and resulting applications are combinations of these three ingredients). Each ingredient may involve use of one or more accelerator (FPGA, ASIC) components as a local backend. In this manner,
25 applications can be split across multiple Edge owners, coordinated by an orchestrator.

[0059] In the context of FIG. 5, the pod controller/container manager, container orchestrator, and individual nodes may provide a security enforcement point. However, tenant isolation may be orchestrated where the resources allocated to a tenant are distinct from resources allocated to a second tenant, but Edge owners cooperate to ensure resource
30 allocations are not shared across tenant boundaries. Or, resource allocations could be isolated across tenant boundaries, as tenants could allow “use” via a subscription or transaction/contract basis. In these contexts, virtualization, containerization, enclaves and hardware partitioning schemes may be used by Edge owners to enforce tenancy. Other isolation environments may

include: bare metal (dedicated) equipment, virtual machines, containers, virtual machines on containers, or combinations thereof.

[0060] In further examples, aspects of software-defined or controlled silicon hardware, and other configurable hardware, may integrate with the applications, functions, and services an Edge computing system. Software defined silicon (SDSi) may be used to ensure the ability for some resource or hardware ingredient to fulfill a contract or service level agreement, based on the ingredient's ability to remediate a portion of itself or the workload (e.g., by an upgrade, reconfiguration, or provision of new features within the hardware configuration itself).

[0061] In further examples, any of the compute nodes or devices discussed with reference to the present Edge computing systems and environment may be fulfilled based on the components depicted in FIGS. 6A and 6B. Respective Edge compute nodes may be embodied as a type of device, appliance, computer, or other "thing" capable of communicating with other Edge, networking, or endpoint components. For example, an Edge compute device may be embodied as a personal computer, server, smartphone, a mobile compute device, a smart appliance, an in-vehicle compute system (e.g., a navigation system), a self-contained device having an outer case, shell, etc., or other device or system capable of performing the described functions.

[0062] In the simplified example depicted in FIG. 6A, an Edge compute node 600 includes a compute engine (also referred to herein as "compute circuitry") 602, an input/output (I/O) subsystem (also referred to herein as "I/O circuitry") 608, data storage (also referred to herein as "data storage circuitry") 610, a communication circuitry subsystem 612, and, optionally, one or more peripheral devices (also referred to herein as "peripheral device circuitry") 614. In other examples, respective compute devices may include other or additional components, such as those typically found in a computer (e.g., a display, peripheral devices, etc.). Additionally, in some examples, one or more of the illustrative components may be incorporated in, or otherwise form a portion of, another component.

[0063] The compute node 600 may be embodied as any type of engine, device, or collection of devices capable of performing various compute functions. In some examples, the compute node 600 may be embodied as a single device such as an integrated circuit, an embedded system, a field-programmable gate array (FPGA), a system-on-a-chip (SOC), or other integrated system or device. In the illustrative example, the compute node 600 includes or is embodied as a processor (also referred to herein as "processor circuitry") 604 and a memory (also referred to herein as "memory circuitry") 606. The processor 604 may be embodied as any type of processor(s) capable of performing the functions described herein (e.g., executing

an application). For example, the processor 604 may be embodied as a multi-core processor(s), a microcontroller, a processing unit, a specialized or special purpose processing unit, or other processor or processing/controlling circuit.

[0064] In some examples, the processor 604 may be embodied as, include, or be coupled to an FPGA, an application specific integrated circuit (ASIC), reconfigurable hardware or hardware circuitry, or other specialized hardware to facilitate performance of the functions described herein. Also in some examples, the processor 604 may be embodied as a specialized x-processing unit (xPU) also known as a data processing unit (DPU), infrastructure processing unit (IPU), or network processing unit (NPU). Such an xPU may be embodied as a standalone circuit or circuit package, integrated within an SOC, or integrated with networking circuitry (e.g., in a SmartNIC, or enhanced SmartNIC), acceleration circuitry, storage devices, storage disks, or AI hardware (e.g., GPUs or programmed FPGAs). Such an xPU may be designed to receive, retrieve and/or otherwise obtain programming to process one or more data streams and perform specific tasks and actions for the data streams (such as hosting microservices, performing service management or orchestration, organizing or managing server or data center hardware, managing service meshes, or collecting and distributing telemetry), outside of the CPU or general purpose processing hardware. However, it will be understood that a xPU, a SOC, a CPU, and other variations of the processor 604 may work in coordination with each other to execute many types of operations and instructions within and on behalf of the compute node 600.

[0065] The memory 606 may be embodied as any type of volatile (e.g., dynamic random access memory (DRAM), etc.) or non-volatile memory or data storage capable of performing the functions described herein. Volatile memory may be a storage medium that requires power to maintain the state of data stored by the medium. Non-limiting examples of volatile memory may include various types of random access memory (RAM), such as DRAM or static random access memory (SRAM). One particular type of DRAM that may be used in a memory module is synchronous dynamic random access memory (SDRAM).

[0066] In an example, the memory device (e.g., memory circuitry) is any number of block addressable memory devices, such as those based on NAND or NOR technologies (for example, Single-Level Cell ("SLC"), Multi-Level Cell ("MLC"), Quad-Level Cell ("QLC"), Tri-Level Cell ("TLC"), or some other NAND). In some examples, the memory device(s) includes a byte-addressable write-in-place three dimensional crosspoint memory device, or other byte addressable write-in-place non-volatile memory (NVM) devices, such as single or multi-level Phase Change Memory (PCM) or phase change memory with a switch (PCMS),

NVM devices that use chalcogenide phase change material (for example, chalcogenide glass), resistive memory including metal oxide base, oxygen vacancy base and Conductive Bridge Random Access Memory (CB-RAM), nanowire memory, ferroelectric transistor random access memory (FeTRAM), magneto resistive random access memory (MRAM) that
5 incorporates memristor technology, spin transfer torque (STT)-MRAM, a spintronic magnetic junction memory based device, a magnetic tunneling junction (MTJ) based device, a DW (Domain Wall) and SOT (Spin Orbit Transfer) based device, a thyristor based memory device, a combination of any of the above, or other suitable memory. A memory device may also include a three-dimensional crosspoint memory device (e.g., Intel® 3D XPoint™ memory), or
10 other byte addressable write-in-place nonvolatile memory devices. The memory device may refer to the die itself and/or to a packaged memory product. In some examples, 3D crosspoint memory (e.g., Intel® 3D XPoint™ memory) may include a transistor-less stackable cross point architecture in which memory cells sit at the intersection of word lines and bit lines and are individually addressable and in which bit storage is based on a change in bulk resistance. In
15 some examples, all or a portion of the memory 606 may be integrated into the processor 604. The memory 606 may store various software and data used during operation such as one or more applications, data operated on by the application(s), libraries, and drivers.

[0067] In some examples, resistor-based and/or transistor-less memory architectures include nanometer scale phase-change memory (PCM) devices in which a volume of phase-change
20 material resides between at least two electrodes. Portions of the example phase-change material exhibit varying degrees of crystalline phases and amorphous phases, in which varying degrees of resistance between the at least two electrodes can be measured. In some examples, the phase-change material is a chalcogenide-based glass material. Such resistive memory devices are sometimes referred to as memristive devices that remember the history of the
25 current that previously flowed through them. Stored data is retrieved from example PCM devices by measuring the electrical resistance, in which the crystalline phases exhibit a relatively lower resistance value(s) (e.g., logical “0”) when compared to the amorphous phases having a relatively higher resistance value(s) (e.g., logical “1”).

[0068] Example PCM devices store data for long periods of time (e.g., approximately 10
30 years at room temperature). Write operations to example PCM devices (e.g., set to logical “0”, set to logical “1”, set to an intermediary resistance value) are accomplished by applying one or more current pulses to the at least two electrodes, in which the pulses have a particular current magnitude and duration. For instance, a long low current pulse (SET) applied to the at least two electrodes causes the example PCM device to reside in a low-resistance crystalline state,

while a comparatively short high current pulse (RESET) applied to the at least two electrodes causes the example PCM device to reside in a high-resistance amorphous state.

[0069] In some examples, implementation of PCM devices facilitates non-von Neumann computing architectures that enable in-memory computing capabilities. Generally speaking, traditional computing architectures include a central processing unit (CPU) communicatively connected to one or more memory devices via a bus. As such, a finite amount of energy and time is consumed to transfer data between the CPU and memory, which is a known bottleneck of von Neumann computing architectures. However, PCM devices minimize and, in some cases, eliminate data transfers between the CPU and memory by performing some computing operations in-memory. Stated differently, PCM devices both store information and execute computational tasks. Such non-von Neumann computing architectures may implement vectors having a relatively high dimensionality to facilitate hyperdimensional computing, such as vectors having 10,000 bits. Relatively large bit width vectors enable computing paradigms modeled after the human brain, which also processes information analogous to wide bit vectors.

[0070] The compute circuitry 602 is communicatively coupled to other components of the compute node 600 via the I/O subsystem 608, which may be embodied as circuitry and/or components to facilitate input/output operations with the compute circuitry 602 (e.g., with the processor 604 and/or the main memory 606) and other components of the compute circuitry 602. For example, the I/O subsystem 608 may be embodied as, or otherwise include, memory controller hubs, input/output control hubs, integrated sensor hubs, firmware devices, communication links (e.g., point-to-point links, bus links, wires, cables, light guides, printed circuit board traces, etc.), and/or other components and subsystems to facilitate the input/output operations. In some examples, the I/O subsystem 608 may form a portion of a system-on-a-chip (SoC) and be incorporated, along with one or more of the processor 604, the memory 606, and other components of the compute circuitry 602, into the compute circuitry 602.

[0071] The one or more illustrative data storage devices/disks 610 may be embodied as one or more of any type(s) of physical device(s) configured for short-term or long-term storage of data such as, for example, memory devices, memory, circuitry, memory cards, flash memory, hard disk drives, solid-state drives (SSDs), and/or other data storage devices/disks. Individual data storage devices/disks 610 may include a system partition that stores data and firmware code for the data storage device/disk 610. Individual data storage devices/disks 610 may also

include one or more operating system partitions that store data files and executables for operating systems depending on, for example, the type of compute node 600.

[0072] The communication circuitry 612 may be embodied as any communication circuit, device, or collection thereof, capable of enabling communications over a network between the compute circuitry 602 and another compute device (e.g., an Edge gateway of an implementing Edge computing system). The communication circuitry 612 may be configured to use any one or more communication technology (e.g., wired or wireless communications) and associated protocols (e.g., a cellular networking protocol such a 3GPP 4G or 5G standard, a wireless local area network protocol such as IEEE 802.11/Wi-Fi®, a wireless wide area network protocol, Ethernet, Bluetooth®, Bluetooth Low Energy, a IoT protocol such as IEEE 802.15.4 or ZigBee®, low-power wide-area network (LPWAN) or low-power wide-area (LPWA) protocols, etc.) to effect such communication.

[0073] The illustrative communication circuitry 612 includes a network interface controller (NIC) 620, which may also be referred to as a host fabric interface (HFI). The NIC 620 may be embodied as one or more add-in-boards, daughter cards, network interface cards, controller chips, chipsets, or other devices that may be used by the compute node 600 to connect with another compute device (e.g., an Edge gateway node). In some examples, the NIC 620 may be embodied as part of a system-on-a-chip (SoC) that includes one or more processors, or included on a multichip package that also contains one or more processors. In some examples, the NIC 620 may include a local processor (not shown) and/or a local memory (not shown) that are both local to the NIC 620. In such examples, the local processor of the NIC 620 may be capable of performing one or more of the functions of the compute circuitry 602 described herein. Additionally, or alternatively, in such examples, the local memory of the NIC 620 may be integrated into one or more components of the client compute node at the board level, socket level, chip level, and/or other levels.

[0074] Additionally, in some examples, a respective compute node 600 may include one or more peripheral devices 614. Such peripheral devices 614 may include any type of peripheral device found in a compute device or server such as audio input devices, a display, other input/output devices, interface devices, and/or other peripheral devices, depending on the particular type of the compute node 600. In further examples, the compute node 600 may be embodied by a respective Edge compute node (whether a client, gateway, or aggregation node) in an Edge computing system or like forms of appliances, computers, subsystems, circuitry, or other components.

[0075] In a more detailed example, FIG. 6B illustrates a block diagram of an example of components that may be present in an Edge computing node 650 for implementing the techniques (e.g., operations, processes, methods, and methodologies) described herein. This Edge computing node 650 provides a closer view of the respective components of node 600 when implemented as or as part of a computing device (e.g., as a mobile device, a base station, server, gateway, etc.). The Edge computing node 650 may include any combination of the hardware or logical components referenced herein, and it may include or couple with any device usable with an Edge communication network or a combination of such networks. The components may be implemented as integrated circuits (ICs), portions thereof, discrete electronic devices, or other modules, instruction sets, programmable logic or algorithms, hardware, hardware accelerators, software, firmware, or a combination thereof adapted in the Edge computing node 650, or as components otherwise incorporated within a chassis of a larger system.

[0076] The Edge computing device 650 may include processing circuitry in the form of a processor 652, which may be a microprocessor, a multi-core processor, a multithreaded processor, an ultra-low voltage processor, an embedded processor, an xPU/DPU/IPU/NPU, special purpose processing unit, specialized processing unit, or other known processing elements. The processor 652 may be a part of a system on a chip (SoC) in which the processor 652 and other components are formed into a single integrated circuit, or a single package, such as the Edison™ or Galileo™ SoC boards from Intel Corporation, Santa Clara, California. As an example, the processor 652 may include an Intel® Architecture Core™ based CPU processor, such as a Quark™, an Atom™, an i3, an i5, an i7, an i9, or an MCU-class processor, or another such processor available from Intel®. However, any number other processors may be used, such as available from Advanced Micro Devices, Inc. (AMD®) of Sunnyvale, California, a MIPS®-based design from MIPS Technologies, Inc. of Sunnyvale, California, an ARM®-based design licensed from ARM Holdings, Ltd. or a customer thereof, or their licensees or adopters. The processors may include units such as an A5-A13 processor from Apple® Inc., a Snapdragon™ processor from Qualcomm® Technologies, Inc., or an OMAP™ processor from Texas Instruments, Inc. The processor 652 and accompanying circuitry may be provided in a single socket form factor, multiple socket form factor, or a variety of other formats, including in limited hardware configurations or configurations that include fewer than all elements shown in FIG. 6B.

[0077] The processor 652 may communicate with a system memory 654 over an interconnect 656 (e.g., a bus). Any number of memory devices may be used to provide for a

given amount of system memory. As examples, the memory 654 may be random access memory (RAM) in accordance with a Joint Electron Devices Engineering Council (JEDEC) design such as the DDR or mobile DDR standards (e.g., LPDDR, LPDDR2, LPDDR3, or LPDDR4). In particular examples, a memory component may comply with a DRAM standard promulgated by JEDEC, such as JESD79F for DDR SDRAM, JESD79-2F for DDR2 SDRAM, JESD79-3F for DDR3 SDRAM, JESD79-4A for DDR4 SDRAM, JESD209 for Low Power DDR (LPDDR), JESD209-2 for LPDDR2, JESD209-3 for LPDDR3, and JESD209-4 for LPDDR4. Such standards (and similar standards) may be referred to as DDR-based standards and communication interfaces of the storage devices that implement such standards may be referred to as DDR-based interfaces. In various implementations, the individual memory devices may be of any number of different package types such as single die package (SDP), dual die package (DDP) or quad die package (Q17P). These devices, in some examples, may be directly soldered onto a motherboard to provide a lower profile solution, while in other examples the devices are configured as one or more memory modules that in turn couple to the motherboard by a given connector. Any number of other memory implementations may be used, such as other types of memory modules, e.g., dual inline memory modules (DIMMs) of different varieties including but not limited to microDIMMs or MiniDIMMs.

[0078] To provide for persistent storage of information such as data, applications, operating systems and so forth, a storage 658 may also couple to the processor 652 via the interconnect 656. In an example, the storage 658 may be implemented via a solid-state disk drive (SSDD). Other devices that may be used for the storage 658 include flash memory cards, such as Secure Digital (SD) cards, microSD cards, eXtreme Digital (XD) picture cards, and the like, and Universal Serial Bus (USB) flash drives. In an example, the memory device may be or may include memory devices that use chalcogenide glass, multi-threshold level NAND flash memory, NOR flash memory, single or multi-level Phase Change Memory (PCM), a resistive memory, nanowire memory, ferroelectric transistor random access memory (FeTRAM), anti-ferroelectric memory, magnetoresistive random access memory (MRAM) memory that incorporates memristor technology, resistive memory including the metal oxide base, the oxygen vacancy base and the conductive bridge Random Access Memory (CB-RAM), or spin transfer torque (STT)-MRAM, a spintronic magnetic junction memory based device, a magnetic tunneling junction (MTJ) based device, a DW (Domain Wall) and SOT (Spin Orbit Transfer) based device, a thyristor based memory device, or a combination of any of the above, or other memory.

[0079] In low power implementations, the storage 658 may be on-die memory or registers associated with the processor 652. However, in some examples, the storage 658 may be implemented using a micro hard disk drive (HDD). Further, any number of new technologies may be used for the storage 658 in addition to, or instead of, the technologies described, such as resistance change memories, phase change memories, holographic memories, or chemical memories, among others.

[0080] The components may communicate over the interconnect 656. The interconnect 656 may include any number of technologies, including industry standard architecture (ISA), extended ISA (EISA), peripheral component interconnect (PCI), peripheral component interconnect extended (PCIx), PCI express (PCIe), or any number of other technologies. The interconnect 656 may be a proprietary bus, for example, used in an SoC based system. Other bus systems may be included, such as an Inter-Integrated Circuit (I2C) interface, a Serial Peripheral Interface (SPI) interface, point to point interfaces, and a power bus, among others.

[0081] The interconnect 656 may couple the processor 652 to a transceiver 666, for communications with the connected Edge devices 662. The transceiver 666 may use any number of frequencies and protocols, such as 2.4 Gigahertz (GHz) transmissions under the IEEE 802.15.4 standard, using the Bluetooth® low energy (BLE) standard, as defined by the Bluetooth® Special Interest Group, or the ZigBee® standard, among others. Any number of radios, configured for a particular wireless communication protocol, may be used for the connections to the connected Edge devices 662. For example, a wireless local area network (WLAN) unit may be used to implement Wi-Fi® communications in accordance with the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standard. In addition, wireless wide area communications, e.g., according to a cellular or other wireless wide area protocol, may occur via a wireless wide area network (WWAN) unit.

[0082] The wireless network transceiver 666 (or multiple transceivers) may communicate using multiple standards or radios for communications at a different range. For example, the Edge computing node 650 may communicate with close devices, e.g., within about 10 meters, using a local transceiver based on Bluetooth Low Energy (BLE), or another low power radio, to save power. More distant connected Edge devices 662, e.g., within about 50 meters, may be reached over ZigBee® or other intermediate power radios. Both communications techniques may take place over a single radio at different power levels or may take place over separate transceivers, for example, a local transceiver using BLE and a separate mesh transceiver using ZigBee®.

[0083] A wireless network transceiver 666 (e.g., a radio transceiver) may be included to communicate with devices or services in a cloud (e.g., an Edge cloud 695) via local or wide area network protocols. The wireless network transceiver 666 may be a low-power wide-area (LPWA) transceiver that follows the IEEE 802.15.4, or IEEE 802.15.4g standards, among
5 others. The Edge computing node 650 may communicate over a wide area using LoRaWAN™ (Long Range Wide Area Network) developed by Semtech and the LoRa Alliance. The techniques described herein are not limited to these technologies but may be used with any number of other cloud transceivers that implement long range, low bandwidth communications, such as Sigfox, and other technologies. Further, other communications
10 techniques, such as time-slotted channel hopping, described in the IEEE 802.15.4e specification may be used.

[0084] Any number of other radio communications and protocols may be used in addition to the systems mentioned for the wireless network transceiver 666, as described herein. For example, the transceiver 666 may include a cellular transceiver that uses spread spectrum
15 (SPA/SAS) communications for implementing high-speed communications. Further, any number of other protocols may be used, such as Wi-Fi® networks for medium speed communications and provision of network communications. The transceiver 666 may include radios that are compatible with any number of 3GPP (Third Generation Partnership Project) specifications, such as Long Term Evolution (LTE) and 5th Generation (5G) communication
20 systems, discussed in further detail at the end of the present disclosure. A network interface controller (NIC) 668 may be included to provide a wired communication to nodes of the Edge cloud 695 or to other devices, such as the connected Edge devices 662 (e.g., operating in a mesh). The wired communication may provide an Ethernet connection or may be based on other types of networks, such as Controller Area Network (CAN), Local Interconnect Network
25 (LIN), DeviceNet, ControlNet, Data Highway+, PROFIBUS, or PROFINET, among many others. An additional NIC 668 may be included to enable connecting to a second network, for example, a first NIC 668 providing communications to the cloud over Ethernet, and a second NIC 668 providing communications to other devices over another type of network.

[0085] Given the variety of types of applicable communications from the device to another
30 component or network, applicable communications circuitry used by the device may include or be embodied by any one or more of components 664, 666, 668, or 670. Accordingly, in various examples, applicable means for communicating (e.g., receiving, transmitting, etc.) may be embodied by such communications circuitry.

[0086] The Edge computing node 650 may include or be coupled to acceleration circuitry 664, which may be embodied by one or more artificial intelligence (AI) accelerators, a neural compute stick, neuromorphic hardware, an FPGA, an arrangement of GPUs, an arrangement of xPUs/DPUs/IPU/NPUs, one or more SoCs, one or more CPUs, one or more digital signal
5 processors, dedicated ASICs, or other forms of specialized processors or circuitry designed to accomplish one or more specialized tasks. These tasks may include AI processing (including machine learning, training, inferencing, and classification operations), visual data processing, network data processing, object detection, rule analysis, or the like. These tasks also may include the specific Edge computing tasks for service management and service operations
10 discussed elsewhere in this document.

[0087] The interconnect 656 may couple the processor 652 to a sensor hub or external interface 670 that is used to connect additional devices or subsystems. The devices may include sensors 672, such as accelerometers, level sensors, flow sensors, optical light sensors, camera sensors, temperature sensors, global navigation system (e.g., GPS) sensors, pressure
15 sensors, barometric pressure sensors, and the like. The hub or interface 670 further may be used to connect the Edge computing node 650 to actuators 674, such as power switches, valve actuators, an audible sound generator, a visual warning device, and the like.

[0088] In some optional examples, various input/output (I/O) devices may be present within or connected to, the Edge computing node 650. For example, a display or other output device
20 684 may be included to show information, such as sensor readings or actuator position. An input device 686, such as a touch screen or keypad may be included to accept input. An output device 684 may include any number of forms of audio or visual display, including simple visual outputs such as binary status indicators (e.g., light-emitting diodes (LEDs)) and multi-character visual outputs, or more complex outputs such as display screens (e.g., liquid crystal
25 display (LCD) screens), with the output of characters, graphics, multimedia objects, and the like being generated or produced from the operation of the Edge computing node 650. A display or console hardware, in the context of the present system, may be used to provide output and receive input of an Edge computing system; to manage components or services of an Edge computing system; identify a state of an Edge computing component or service; or to
30 conduct any other number of management or administration functions or service use cases.

[0089] A battery 676 may power the Edge computing node 650, although, in examples in which the Edge computing node 650 is mounted in a fixed location, it may have a power supply coupled to an electrical grid, or the battery may be used as a backup or for temporary

capabilities. The battery 676 may be a lithium ion battery, or a metal-air battery, such as a zinc-air battery, an aluminum-air battery, a lithium-air battery, and the like.

[0090] A battery monitor/charger 678 may be included in the Edge computing node 650 to track the state of charge (SoCh) of the battery 676, if included. The battery monitor/charger
5 678 may be used to monitor other parameters of the battery 676 to provide failure predictions, such as the state of health (SoH) and the state of function (SoF) of the battery 676. The battery monitor/charger 678 may include a battery monitoring integrated circuit, such as an LTC4020 or an LTC2990 from Linear Technologies, an ADT7488A from ON Semiconductor of Phoenix Arizona, or an IC from the UCD90xxx family from Texas Instruments of Dallas, TX. The
10 battery monitor/charger 678 may communicate the information on the battery 676 to the processor 652 over the interconnect 656. The battery monitor/charger 678 may also include an analog-to-digital (ADC) converter that enables the processor 652 to directly monitor the voltage of the battery 676 or the current flow from the battery 676. The battery parameters may be used to determine actions that the Edge computing node 650 may perform, such as
15 transmission frequency, mesh network operation, sensing frequency, and the like.

[0091] A power block 680, or other power supply coupled to a grid, may be coupled with the battery monitor/charger 678 to charge the battery 676. In some examples, the power block 680 may be replaced with a wireless power receiver to obtain the power wirelessly, for example, through a loop antenna in the Edge computing node 650. A wireless battery charging
20 circuit, such as an LTC4020 chip from Linear Technologies of Milpitas, California, among others, may be included in the battery monitor/charger 678. The specific charging circuits may be selected based on the size of the battery 676, and thus, the current required. The charging may be performed using the Airfuel standard promulgated by the Airfuel Alliance, the Qi wireless charging standard promulgated by the Wireless Power Consortium, or the Rezence
25 charging standard, promulgated by the Alliance for Wireless Power, among others.

[0092] The storage 658 may include instructions 682 in the form of software, firmware, or hardware commands to implement the techniques described herein. Although such instructions 682 are shown as code blocks included in the memory 654 and the storage 658, it may be understood that any of the code blocks may be replaced with hardwired circuits, for example,
30 built into an application specific integrated circuit (ASIC).

[0093] In an example, the instructions 682 provided via the memory 654, the storage 658, or the processor 652 may be embodied as a non-transitory, machine-readable medium 660 including code to direct the processor 652 to perform electronic operations in the Edge computing node 650. The processor 652 may access the non-transitory, machine-readable

medium 660 over the interconnect 656. For instance, the non-transitory, machine-readable medium 660 may be embodied by devices described for the storage 658 or may include specific storage units such as storage devices and/or storage disks that include optical disks (e.g., digital versatile disk (DVD), compact disk (CD), CD-ROM, Blu-ray disk), flash drives, floppy disks, hard drives (e.g., SSDs), or any number of other hardware devices in which information is stored for any duration (e.g., for extended time periods, permanently, for brief instances, for temporarily buffering, and/or caching). The non-transitory, machine-readable medium 660 may include instructions to direct the processor 652 to perform a specific sequence or flow of actions, for example, as described with respect to the flowchart(s) and block diagram(s) of operations and functionality depicted above. As used herein, the terms “machine-readable medium” and “computer-readable medium” are interchangeable. As used herein, the term “non-transitory computer-readable medium” is expressly defined to include any type of computer readable storage device and/or storage disk and to exclude propagating signals and to exclude transmission media.

[0094] Also in a specific example, the instructions 682 on the processor 652 (separately, or in combination with the instructions 682 of the machine readable medium 660) may configure execution or operation of a trusted execution environment (TEE) 690. In an example, the TEE 690 operates as a protected area accessible to the processor 652 for secure execution of instructions and secure access to data. Various implementations of the TEE 690, and an accompanying secure area in the processor 652 or the memory 654 may be provided, for instance, through use of Intel® Software Guard Extensions (SGX) or ARM® TrustZone® hardware security extensions, Intel® Management Engine (ME), or Intel® Converged Security Manageability Engine (CSME). Other aspects of security hardening, hardware roots-of-trust, and trusted or protected operations may be implemented in the device 650 through the TEE 690 and the processor 652.

[0095] While the illustrated examples of FIG. 6A and FIG. 6B include example components for a compute node and a computing device, respectively, examples disclosed herein are not limited thereto. As used herein, a “computer” may include some or all of the example components of FIGS. 6A and/or 6B in different types of computing environments. Example computing environments include Edge computing devices (e.g., Edge computers) in a distributed networking arrangement such that particular ones of participating Edge computing devices are heterogenous or homogeneous devices. As used herein, a “computer” may include a personal computer, a server, user equipment, an accelerator, etc., including any combinations thereof. In some examples, distributed networking and/or distributed computing includes any

number of such Edge computing devices as illustrated in FIGS. 6A and/or 6B, each of which may include different sub-components, different memory capacities, I/O capabilities, etc. For example, because some implementations of distributed networking and/or distributed computing are associated with particular desired functionality, examples disclosed herein

5 include different combinations of components illustrated in FIGS. 6A and/or 6B to satisfy functional objectives of distributed computing tasks. In some examples, the term “compute node” or “computer” only includes the example processor 604, memory 606 and I/O subsystem 608 of FIG. 6A. In some examples, one or more objective functions of a distributed computing task(s) rely on one or more alternate devices/structure located in different parts of an Edge

10 networking environment, such as devices to accommodate data storage (e.g., the example data storage 610), input/output capabilities (e.g., the example peripheral device(s) 614), and/or network communication capabilities (e.g., the example NIC 620).

[0096] In some examples, computers operating in a distributed computing and/or distributed networking environment (e.g., an Edge network) are structured to accommodate particular

15 objective functionality in a manner that reduces computational waste. For instance, because a computer includes a subset of the components disclosed in FIGS. 6A and 6B, such computers satisfy execution of distributed computing objective functions without including computing structure that would otherwise be unused and/or underutilized. As such, the term “computer” as used herein includes any combination of structure of FIGS. 6A and/or 6B that is capable of

20 satisfying and/or otherwise executing objective functions of distributed computing tasks. In some examples, computers are structured in a manner commensurate to corresponding distributed computing objective functions in a manner that downscales or upscales in connection with dynamic demand. In some examples, different computers are invoked and/or otherwise instantiated in view of their ability to process one or more tasks of the distributed

25 computing request(s), such that any computer capable of satisfying the tasks proceed with such computing activity.

[0097] In the illustrated examples of FIGS. 6A and 6B, computing devices include operating systems. As used herein, an “operating system” is software to control example computing devices, such as the example Edge compute node 600 of FIG. 6A and/or the

30 example Edge compute node 650 of FIG. 6B. Example operating systems include, but are not limited to consumer-based operating systems (e.g., Microsoft® Windows® 10, Google® Android® OS, Apple® Mac® OS, etc.). Example operating systems also include, but are not limited to industry-focused operating systems, such as real-time operating systems, hypervisors, etc. An example operating system on a first Edge compute node may be the same

or different than an example operating system on a second Edge compute node. In some examples, the operating system invokes alternate software to facilitate one or more functions and/or operations that are not native to the operating system, such as particular communication protocols and/or interpreters. In some examples, the operating system instantiates various functionalities that are not native to the operating system. In some examples, operating systems include varying degrees of complexity and/or capabilities. For instance, a first operating system corresponding to a first Edge compute node includes a real-time operating system having particular performance expectations of responsivity to dynamic input conditions, and a second operating system corresponding to a second Edge compute node includes graphical user interface capabilities to facilitate end-user I/O.

[0098] The instructions 682 may further be transmitted or received over a communications network using a transmission medium via the wireless network transceiver 666 utilizing any one of a number of wireless local area network (WLAN) transfer protocols (e.g., frame relay, internet protocol (IP), transmission control protocol (TCP), user datagram protocol (UDP), hypertext transfer protocol (HTTP), etc.). Example communication networks may include a local area network (LAN), a wide area network (WAN), a packet data network (e.g., the Internet), mobile telephone networks (e.g., cellular networks), Plain Old Telephone (POTS) networks, and wireless data networks. Communications over the networks may include one or more different protocols, such as Institute of Electrical and Electronics Engineers (IEEE) 802.11 family of standards known as Wi-Fi, IEEE 802.16 family of standards, IEEE 802.15.4 family of standards, a Long Term Evolution (LTE) family of standards, a Universal Mobile Telecommunications System (UMTS) family of standards, peer-to-peer (P2P) networks, a next generation (NG)/5th generation (5G) standards among others.

[0099] Note that the term “circuitry” as used herein refers to, is part of, or includes hardware components such as an electronic circuit, a logic circuit, a processor (shared, dedicated, or group) and/or memory (shared, dedicated, or group), an Application Specific Integrated Circuit (ASIC), a field-programmable device (FPD) (e.g., a field-programmable gate array (FPGA), a programmable logic device (PLD), a complex PLD (CPLD), a high-capacity PLD (HCPLD), a structured ASIC, or a programmable SoC), digital signal processors (DSPs), etc., that are configured to provide the described functionality. In some embodiments, the circuitry may execute one or more software or firmware programs to provide at least some of the described functionality. The term “circuitry” may also refer to a combination of one or more hardware elements (or a combination of circuits used in an electrical or electronic system) with the program code used to carry out the functionality of that program code. In

these embodiments, the combination of hardware elements and program code may be referred to as a particular type of circuitry.

[0100] The term “processor circuitry” or “processor” as used herein thus refers to, is part of, or includes circuitry capable of sequentially and automatically carrying out a sequence of arithmetic or logical operations, or recording, storing, and/or transferring digital data. The term “processor circuitry” or “processor” may refer to one or more application processors, one or more baseband processors, a physical central processing unit (CPU), a single- or multi-core processor, and/or any other device capable of executing or otherwise operating computer-executable instructions, such as program code, software modules, and/or functional processes.

[0101] Any of the radio links described herein may operate according to any one or more of the following radio communication technologies and/or standards including but not limited to: a Global System for Mobile Communications (GSM) radio communication technology, a General Packet Radio Service (GPRS) radio communication technology, an Enhanced Data Rates for GSM Evolution (EDGE) radio communication technology, and/or a Third Generation Partnership Project (3GPP) radio communication technology, for example Universal Mobile Telecommunications System (UMTS), Freedom of Multimedia Access (FOMA), 3GPP Long Term Evolution (LTE), 3GPP Long Term Evolution Advanced (LTE Advanced), Code division multiple access 2000 (CDMA2000), Cellular Digital Packet Data (CDPD), Mobitex, Third Generation (3G), Circuit Switched Data (CSD), High-Speed Circuit-Switched Data (HSCSD), Universal Mobile Telecommunications System (Third Generation) (UMTS (3G)), Wideband Code Division Multiple Access (Universal Mobile Telecommunications System) (W-CDMA (UMTS)), High Speed Packet Access (HSPA), High-Speed Downlink Packet Access (HSDPA), High-Speed Uplink Packet Access (HSUPA), High Speed Packet Access Plus (HSPA+), Universal Mobile Telecommunications System-Time-Division Duplex (UMTS-TDD), Time Division-Code Division Multiple Access (TD-CDMA), Time Division-Synchronous Code Division Multiple Access (TD-CDMA), 3rd Generation Partnership Project Release 8 (Pre-4th Generation) (3GPP Rel. 8 (Pre-4G)), 3GPP Rel. 9 (3rd Generation Partnership Project Release 9), 3GPP Rel. 10 (3rd Generation Partnership Project Release 10), 3GPP Rel. 11 (3rd Generation Partnership Project Release 11), 3GPP Rel. 12 (3rd Generation Partnership Project Release 12), 3GPP Rel. 13 (3rd Generation Partnership Project Release 13), 3GPP Rel. 14 (3rd Generation Partnership Project Release 14), 3GPP Rel. 15 (3rd Generation Partnership Project Release 15), 3GPP Rel. 16 (3rd Generation Partnership Project Release 16), 3GPP Rel. 17 (3rd Generation Partnership Project Release 17) and subsequent Releases (such as Rel. 18, Rel. 19, etc.), 3GPP 5G, 5G, 5G New Radio (5G NR), 3GPP 5G

New Radio, 3GPP LTE Extra, LTE-Advanced Pro, LTE Licensed-Assisted Access (LAA), MuLTEfire, UMTS Terrestrial Radio Access (UTRA), Evolved UMTS Terrestrial Radio Access (E-UTRA), Long Term Evolution Advanced (4th Generation) (LTE Advanced (4G)), cdmaOne (2G), Code division multiple access 2000 (Third generation) (CDMA2000 (3G)),

5 Evolution-Data Optimized or Evolution-Data Only (EV-DO), Advanced Mobile Phone System (1st Generation) (AMPS (1G)), Total Access Communication System/Extended Total Access Communication System (TACS/ETACS), Digital AMPS (2nd Generation) (D-AMPS (2G)), Push-to-talk (PTT), Mobile Telephone System (MTS), Improved Mobile Telephone System (IMTS), Advanced Mobile Telephone System (AMTS), OLT (Norwegian for Offentlig

10 Landmobil Telefoni, Public Land Mobile Telephony), MTD (Swedish abbreviation for Mobiltelefonisystem D, or Mobile telephony system D), Public Automated Land Mobile (Autotel/PALM), ARP (Finnish for Autoradiopuhelin, “car radio phone”), NMT (Nordic Mobile Telephony), High capacity version of NTT (Nippon Telegraph and Telephone) (Hicap), Cellular Digital Packet Data (CDPD), Mobitex, DataTAC, Integrated Digital

15 Enhanced Network (iDEN), Personal Digital Cellular (PDC), Circuit Switched Data (CSD), Personal Handy-phone System (PHS), Wideband Integrated Digital Enhanced Network (WiDEN), iBurst, Unlicensed Mobile Access (UMA), also referred to as also referred to as 3GPP Generic Access Network, or GAN standard), Zigbee, Bluetooth®, Wireless Gigabit Alliance (WiGig) standard, mmWave standards in general (wireless systems operating at 10-

20 300 GHz and above such as WiGig, IEEE 802.11ad, IEEE 802.11ay, etc.), technologies operating above 300 GHz and THz bands, (3GPP/LTE based or IEEE 802.11p or IEEE 802.11bd and other) Vehicle-to-Vehicle (V2V) and Vehicle-to-X (V2X) and Vehicle-to-Infrastructure (V2I) and Infrastructure-to-Vehicle (I2V) communication technologies, 3GPP cellular V2X, DSRC (Dedicated Short Range Communications) communication systems such

25 as Intelligent-Transport-Systems and others (typically operating in 5850 MHz to 5925 MHz or above (typically up to 5935 MHz following change proposals in CEPT Report 71)), the European ITS-G5 system (i.e. the European flavor of IEEE 802.11p based DSRC, including ITS-G5A (i.e., Operation of ITS-G5 in European ITS frequency bands dedicated to ITS for safety related applications in the frequency range 5,875 GHz to 5,905 GHz), ITS-G5B (i.e.,

30 Operation in European ITS frequency bands dedicated to ITS non- safety applications in the frequency range 5,855 GHz to 5,875 GHz), ITS-G5C (i.e., Operation of ITS applications in the frequency range 5,470 GHz to 5,725 GHz)), DSRC in Japan in the 700 MHz band (including 715 MHz to 725 MHz), IEEE 802.11bd based systems, etc.

[0102] Aspects described herein can be used in the context of any spectrum management scheme including dedicated licensed spectrum, unlicensed spectrum, license exempt spectrum, (licensed) shared spectrum (such as LSA = Licensed Shared Access in 2.3-2.4 GHz, 3.4-3.6 GHz, 3.6-3.8 GHz and further frequencies and SAS = Spectrum Access System / CBRs =

5 Citizen Broadband Radio System in 3.55-3.7 GHz and further frequencies). Applicable spectrum bands include IMT (International Mobile Telecommunications) spectrum as well as other types of spectrum/bands, such as bands with national allocation (including 450 - 470 MHz, 902-928 MHz (note: allocated for example in the US (FCC Part 15)), 863-868.6 MHz (note: allocated for example in European Union (ETSI EN 300 220)), 915.9-929.7 MHz (note:

10 allocated for example in Japan), 917-923.5 MHz (note: allocated for example in South Korea), 755-779 MHz and 779-787 MHz (note: allocated for example in China), 790 - 960 MHz, 1710 - 2025 MHz, 2110 - 2200 MHz, 2300 - 2400 MHz, 2.4-2.4835 GHz (note: it is an ISM band with global availability and it is used by Wi-Fi technology family (11b/g/n/ax) and also by Bluetooth®), 2500 - 2690 MHz, 698-790 MHz, 610 - 790 MHz, 3400 - 3600 MHz, 3400 -

15 3800 MHz, 3800 - 4200 MHz, 3.55-3.7 GHz (note: allocated for example in the US for Citizen Broadband Radio Service), 5.15-5.25 GHz and 5.25-5.35 GHz and 5.47-5.725 GHz and 5.725-5.85 GHz bands (note: allocated for example in the US (FCC part 15), consists four U-NII bands in total 500 MHz spectrum), 5.725-5.875 GHz (note: allocated for example in EU (ETSI EN 301 893)), 5.47-5.65 GHz (note: allocated for example in South Korea, 5925-7085 MHz

20 and 5925-6425MHz band (note: under consideration in US and EU, respectively. Next generation Wi-Fi system is expected to include the 6 GHz spectrum as operating band, but it is noted that, as of December 2017, Wi-Fi system is not yet allowed in this band. Regulation is expected to be finished in 2019-2020 time frame), IMT-advanced spectrum, IMT-2020 spectrum (expected to include 3600-3800 MHz, 3800 - 4200 MHz, 3.5 GHz bands, 700 MHz

25 bands, bands within the 24.25-86 GHz range, etc.), spectrum made available under FCC's "Spectrum Frontier" 5G initiative (including 27.5 - 28.35 GHz, 29.1 - 29.25 GHz, 31 - 31.3 GHz, 37 - 38.6 GHz, 38.6 - 40 GHz, 42 - 42.5 GHz, 57 - 64 GHz, 71 - 76 GHz, 81 - 86 GHz and 92 - 94 GHz, etc.), the ITS (Intelligent Transport Systems) band of 5.9 GHz (typically 5.85-5.925 GHz) and 63-64 GHz, bands currently allocated to WiGig such as WiGig Band 1

30 (57.24-59.40 GHz), WiGig Band 2 (59.40-61.56 GHz) and WiGig Band 3 (61.56-63.72 GHz) and WiGig Band 4 (63.72-65.88 GHz), 57-64/66 GHz (note: this band has near-global designation for Multi-Gigabit Wireless Systems (MGWS)/WiGig. In US (FCC part 15) allocates total 14 GHz spectrum, while EU (ETSI EN 302 567 and ETSI EN 301 217-2 for fixed P2P) allocates total 9 GHz spectrum), the 70.2 GHz - 71 GHz band, any band between

65.88 GHz and 71 GHz, bands currently allocated to automotive radar applications such as 76-81 GHz, and future bands including 94-300 GHz and above. Furthermore, the scheme can be used on a secondary basis on bands such as the TV White Space bands (typically below 790 MHz) where in particular the 400 MHz and 700 MHz bands are promising candidates. Besides
5 cellular applications, specific applications for vertical markets may be addressed such as PMSE (Program Making and Special Events), medical, health, surgery, automotive, low-latency, drones, etc. applications.

[0103] FIG. 7 illustrates an example software distribution platform 705 to distribute software, such as the example computer readable instructions 682 of FIG. 6B, to one or more
10 devices, such as example processor platform(s) 710 and/or example connected Edge devices. The example software distribution platform 705 may be implemented by any computer server, data facility, cloud service, etc., capable of storing and transmitting software to other computing devices (e.g., third parties, the example connected Edge devices). Example connected Edge devices may be customers, clients, managing devices (e.g., servers), third
15 parties (e.g., customers of an entity owning and/or operating the software distribution platform 705). Example connected Edge devices may operate in commercial and/or home automation environments. In some examples, a third party is a developer, a seller, and/or a licensor of software such as the example computer readable instructions 682 of FIG. 6B. The third parties may be consumers, users, retailers, OEMs, etc. that purchase and/or license the software for
20 use and/or re-sale and/or sub-licensing. In some examples, distributed software causes display of one or more user interfaces (UIs) and/or graphical user interfaces (GUIs) to identify the one or more devices (e.g., connected Edge devices) geographically and/or logically separated from each other (e.g., physically separated IoT devices chartered with the responsibility of water distribution control (e.g., pumps), electricity distribution control (e.g., relays), etc.).

[0104] In the illustrated example of FIG. 7, the software distribution platform 705 includes one or more servers and one or more storage devices. The storage devices store the computer readable instructions 682. The one or more servers of the example software distribution
platform 705 are in communication with a network 715, which may correspond to any one or more of the Internet and/or any of the example networks described above. In some examples,
30 the one or more servers are responsive to requests to transmit the software to a requesting party as part of a commercial transaction. Payment for the delivery, sale and/or license of the software may be handled by the one or more servers of the software distribution platform and/or via a third-party payment entity. The servers enable purchasers and/or licensors to download the computer readable instructions 682 from the software distribution platform 605.

For example, the software, which may correspond to the example computer readable instructions, may be downloaded to the example processor platform(s) 700 (e.g., example connected Edge devices), which is/are to execute the computer readable instructions 682 to implement the content insertion at a switch. In some examples, one or more servers of the software distribution platform 705 are communicatively connected to one or more security domains and/or security devices through which requests and transmissions of the example computer readable instructions 682 must pass. In some examples, one or more servers of the software distribution platform 705 periodically offer, transmit, and/or force updates to the software (e.g., the example computer readable instructions 682 of FIG. 6B) to ensure improvements, patches, updates, etc. are distributed and applied to the software at the end user devices.

[0105] In the illustrated example of FIG. 7, the computer readable instructions 682 are stored on storage devices of the software distribution platform 705 in a particular format. A format of computer readable instructions includes, but is not limited to a particular code language (e.g., Java, JavaScript, Python, C, C#, SQL, HTML, etc.), and/or a particular code state (e.g., uncompiled code (e.g., ASCII), interpreted code, linked code, executable code (e.g., a binary), etc.). In some examples, the computer readable instructions 682 stored in the software distribution platform 705 are in a first format when transmitted to the example processor platform(s) 710. In some examples, the first format is an executable binary in which particular types of the processor platform(s) 710 can execute. However, in some examples, the first format is uncompiled code that requires one or more preparation tasks to transform the first format to a second format to enable execution on the example processor platform(s) 710. For instance, the receiving processor platform(s) 710 may need to compile the computer readable instructions 682 in the first format to generate executable code in a second format that is capable of being executed on the processor platform(s) 710. In still other examples, the first format is interpreted code that, upon reaching the processor platform(s) 710, is interpreted by an interpreter to facilitate execution of instructions.

[0106] FIG. 8 illustrates an example information centric network (ICN) 800, according to an embodiment. ICNs operate differently than traditional host-based (e.g., address-based) communication networks. ICN is an umbrella term for a networking paradigm in which information and/or functions themselves are named and used in a request instead of a host's name (e.g., machines that provide information). In a host-based networking paradigm, such as used in the Internet protocol (IP), a device locates a host and requests content from the host. The network understands how to route (e.g., direct) packets based on the address specified in

the packet. In contrast, ICN does not include a request for a particular machine and does not use addresses. Instead, to get content, a device 805 (e.g., subscriber) requests named content from the network 800 itself. The content request may be called an interest and transmitted via an interest packet 830. As the interest packet traverses network devices (e.g., network elements, routers, switches, hubs, etc.)—such as network elements 810, 815, and 820—a record of the interest is kept, for example, in a pending interest table (PIT) at each network element. Thus, network element 810 maintains an entry in its PIT 835 for the interest packet 830, network element 815 maintains the entry in its PIT, and network element 820 maintains the entry in its PIT.

5
10 **[0107]** When a device, such as publisher 840, that has content matching the name in the interest packet 830 is encountered, that device 840 may send a data packet 845 in response to the interest packet 830. Typically, the data packet 845 is tracked back through the network 800 to the source (e.g., device 805) by following the traces of the interest packet 830 left in the network elements' PITs. Thus, the PIT 835 at each network element establishes a trail back to the subscriber 805 for the data packet 845 to follow.

15 **[0108]** Matching the named data in an ICN may follow several strategies. Generally, the data is named hierarchically, such as with a universal resource identifier (URI). For example, a video may be named www.somedomain.com or videos or v8675309. Here, the hierarchy may be seen as the publisher, "www.somedomain.com," a sub-category, "videos," and the canonical identification "v8675309." As an interest 830 traverses the ICN, ICN network elements will generally attempt to match the name to a greatest degree. Thus, if an ICN element has a cached item or route for both "www.somedomain.com or videos" and "www.somedomain.com or videos or v8675309," the ICN element will match the later for an interest packet 830 specifying "www.somedomain.com or videos or v8675309." In an example, an expression may be used in matching by the ICN device. For example, the interest packet may specify "www.somedomain.com or videos or v8675*" where '*' is a wildcard. Thus, any cached item or route that includes the data other than the wildcard will be matched.

20
25 **[0109]** Item matching involves matching the interest 830 to data cached in the ICN element. Thus, for example, if the data 845 named in the interest 830 is cached in network element 815, then the network element 815 will return the data 845 to the subscriber 805 via the network element 810. However, if the data 845 is not cached at network element 815, the network element 815 routes the interest 830 on (e.g., to network element 820). To facilitate routing, the network elements may use a forwarding information base 825 (FIB) to match named data to an

interface (e.g., physical port) for the route. Thus, the FIB 825 operates much like a routing table on a traditional network device.

[0110] In an example, additional meta-data may be attached to the interest packet 830, the cached data, or the route (e.g., in the FIB 825), to provide an additional level of matching. For example, the data name may be specified as “www.somedomain.com or videos or v8675309,” but also include a version number—or timestamp, time range, endorsement, etc. In this example, the interest packet 830 may specify the desired name, the version number, or the version range. The matching may then locate routes or cached data matching the name and perform the additional comparison of meta-data or the like to arrive at an ultimate decision as to whether data or a route matches the interest packet 830 for respectively responding to the interest packet 830 with the data packet 845 or forwarding the interest packet 830.

[0111] ICN has advantages over host-based networking because the data segments are individually named. This enables aggressive caching throughout the network 800 as a network element may provide a data packet 830 in response to an interest 830 as easily as an original author 840. Accordingly, it is less likely that the same segment of the network 800 will transmit duplicates of the same data requested by different devices.

[0112] Fine grained encryption is another feature of many ICN networks. A typical data packet 845 includes a name for the data that matches the name in the interest packet 830. Further, the data packet 845 includes the requested data and may include additional information to filter similarly named data (e.g., by creation time, expiration time, version, etc.). To address malicious entities providing false information under the same name, the data packet 845 may also encrypt its contents with a publisher key or provide a cryptographic hash of the data and the name. Thus, knowing the key (e.g., from a certificate of an expected publisher 840) enables the recipient to ascertain whether the data is from that publisher 840. This technique also facilitates the aggressive caching of the data packets 845 throughout the network because each data packet 845 is self-contained and secure. In contrast, many host-based networks rely on encrypting a connection between two hosts to secure communications. This may increase latencies while connections are being established and prevents data caching by hiding the data from the network elements.

[0113] Example ICN networks include content centric networking (CCN), as specified in the Internet Engineering Task Force (IETF) draft specifications for CCNx 0.x and CCN 1.x, and named data networking (NDN), as specified in the NDN technical report DND-0001. It is understood that an ICN network may be implemented on top of or in combination with other

types of networks, such as IP networks. For instance, an ICN network may use IP networks for backhaul.

[0114] As discussed above, Named Function Networking (NFN) is an extension to existing ICN networks that enables an ICN to deliver not only data that was already published but also on-demand results of computations (i.e., “functions”). This has the advantage of optimizing the location of the computation in a dynamic manner and a further advantage of decentralizing the supply of computation and allowing a named function to be provided from anywhere suitable and allowing computations to be proximal to wherever data happens to be generated, replicated, or cached.

[0115] FIG. 9 is a diagram illustrating an operating environment 900, according to an embodiment. The operating environment 900 includes compute nodes 902A, 902B, ..., 902N (collectively referred to as 902). A client device 904 is connected to the compute nodes 902 via a network 906. Client device 904 may be a user equipment (UE), and may be any device used directly by an end-user to communicate with the network 906. The client device 904 may be a mobile telephone, a smartphone, a laptop computer equipped with a mobile broadband adapter, a desktop computer, a kiosk, a navigation unit in a vehicle, or any other device that can communicate over the network 906 an interest packet 908 that includes a named function to be executed. The interest packet 908 includes the function and actual data or a reference to data (e.g., a data location) that the function is to act upon. The function included in the interest packet may specify a name of a computational operation or may specify a block of code (e.g., a lambda expression) to be evaluated, where the block of code may include references to other functions. The reference to data may be a reference used in named data networking (NDN). The compute nodes 902 are connected to the network 906 and provide support for NFN functionality. The compute nodes 902 may be associated with one another. A compute node 902 may be an edge device, a fog device, a datacenter, a mobile network, or the like. In an embodiment, the compute nodes 902 are part of one or more blockchain networks.

[0116] One or more function providers (FPs) 910A, 910B, ..., 910N (collectively referred to as FPs 910) are connected via an NFN network 912. The NFN network 912 may be a portion of the network 906. The FPs 910 are repositories for functions that are available via NFN networking. Compute nodes 902 may act as FPs 910. The compute node 902 or FPs 910 may execute the function on behalf of the client device 904. Alternatively, the FPs 910 may respond to an interest packet for a function with the names of functions that match the query in the interest packet. For instance, the interest packet 908 may express an interest in a function name of “function1*” and the response data packet may include a list of “function1_v1” and

“function1_v2”. Additionally or alternatively, the response data packet may include a respective compute node 902 that hosts each of the respective functions in the list of functions.

[0117] In operation, a network administrator device 914 is used to configure the compute nodes 902. Various administrative activities may be used, such as registering a compute node
5 902 with one or more blockchains, associating the compute node 902 with other compute nodes 902, installing or updating functions on a compute node 902, registering the compute node 902 with one of the function providers 910, configuring security and access settings for one or more compute nodes 902, and the like.

[0118] When a function F is called by a client device 904 (e.g., user equipment), the
10 compute node 902 transmits an interest packet 908 to the NFN network 912 to obtain the function. In an embodiment, the function providers store bytecode, bit streams, or other executable instructions, and this content is cached in the NFN network 912, similar to how data is cached in an ICN or an NDN.

[0119] The compute node 902 that receives the initial function call from the client device
15 904 constructs the interest packet 908 to send to the NFN network 912, receives the response from the NFN network 912, and then executes the function with the indicated data from the interest packet 908. The data may be fetched from an NDN 916 or be provided directly in the interest packet 908 from the client device 904 as a literal value, or some combination of references and literal values.

[0120] The compute node 902 coordinates the function’s execution among a number N of
20 compute nodes 902. The coordinating compute node 902 initiates execution of the function F at a number N-1 of other compute nodes 902 (with the first receiving/initiating compute node 902 being the Nth compute node). In an embodiment, the coordinating compute node 902 relays the interest packet 908 received from the client device 904 to N-1 other compute nodes
25 902. This may be performed using broadcasting or multicasting. Once the other N-1 compute nodes 902 receive the interest packet, they may act in a similar manner by requesting the function from the NFN network 912 and executing on the data referenced or provided in the interest packet 908. If the coordinating compute node 902 obtains a copy of the function from an FPs 910, then the function may be cached at the coordinating compute node 902 and other
30 compute nodes 902 may be able to retrieve the function from the coordinating compute node’s cache.

[0121] The number N may be set by the client device 904 in the initial interest packet 908. The interest packet 908 may include a minimum required number of compute nodes 902 that are to execute the function F.

[0122] Multiple compute nodes 902 are used to execute instances of the function and evaluate the results with peer compute nodes 902 using the shared consensus protocol. This approach provides a higher level of confidence that the instances of the function (e.g., *fn1* executing on *node1* and *fn2* executing on *node2*) are interchangeable if the evaluation of the results produces consensus.

[0123] Further, a function call in an interest packet 908 by a client device 904 may include constraints on the execution of the function. For instance, a latency constraint may be specified by the client device 904 that indicates a maximum latency within which the consensus must happen. For example, in a vehicle-to-everything (V2X) scenario, a pedestrian identification image detection operation may be constrained to 50ms whereas a use case to select a safe speed on a roadway may be constrained to 2 seconds latency.

[0124] Also, a function call in an interest packet 908 by a client device 904 may include function range restrictions, such as a maximum deviation acceptable across the results of the function instances to achieve consensus. For instance, the range may be provided as a percentage such that to achieve consensus, the results can differ up to a maximum of 10%. These function constraints or function execution parameters may be used to determine the number N of compute nodes 902 to use, the type of compute node 902 to use, and the like.

[0125] One example of a shared consensus protocol includes each compute node 902 transmitting its results of the function's execution to all other compute nodes 902 that are participating in the consensus evaluation. The compute nodes 902 are able to compare results of their own execution of the function with the results of other compute nodes' execution of the function. If there is a consensus (e.g., greater than 50% of the compute nodes have a result that is within a maximum acceptable deviation), then the function's results are considered to be accurate and the agreed-upon result is transmitted back to the client device 904 as a valid result. Additionally, the consensus result is stored in a blockchain that the N compute nodes 902 participate in.

[0126] The winning block can be used as the reference value or expected value of the function's execution given the particular input data. The consensus value is catalogued for use by attestation verifiers or other utilities that are meant to satisfy the specified KPIs for NFN and FaaS functions. Other ingredients may be captured in the blockchain. For instance, captured results may include telemetry of performance, resource utilization, resource access requests, code path exceptions, and so forth.

[0127] Using the mechanisms described here, a different version of a function may be analyzed using the NFN and the coordinated compute nodes 902. The results may be stored in

a blockchain based on a consensus result. The blockchain blocks can record how different vendor implementations may differ in precision, accuracy, latency, etc. Such information may be used to make a better selection of the function for a particular user workload or execution plan.

5 [0128] FIG. 10 is a diagram illustrating edge nodes participating in function consensus blockchain, according to an embodiment. Edge nodes 1002A, 1002B, ..., 1002N (collectively referred to as 1002) may be compute nodes 902 in network 906. The edge nodes 1002 may be temporarily associated with one another when a function request is received. The edge nodes 1002 participate in a distributed named function consensus mechanism, where each edge node
10 1002 is peer in a blockchain network that has access to a blockchain. Based on the execution results of each edge node 1002, a consensus may be reached and if so, then a new transaction is stored in a block in the blockchain to record the consensus-based result.

[0129] The edge nodes 1002 use distributed logic to create a consensus about a request from a device, user, or service. Hence, as shown in FIG. 10, after a request for a named function
15 arrives at a particular edge node, depending on the level of reliability required on the response, the edge function consensus (EFC) logic 1004A, 1004B, ..., 1004N, residing on a particular edge node 1002 may decide to multicast the interest packet to other peers in the function consensus blockchain. The EFC logic 1004 refers, in part, to blockchain logic that implements a distributed consensus algorithm such as Ethereum, Dfinity, Sovrin, Bitcoin, or other
20 distributed consensus algorithms.

[0130] An edge node 1002 may implement an edge architecture that allows implementation of any edge workload, function, or operation on any edge-aware platform. This mechanism augments existing consensus algorithms by adding a function execution task to the existing computation prescribed by the consensus algorithm.

25 [0131] For example, in an embodiment, the EFC logic may perform forking and executing a process containing the edge function F1. The results of the F1 may be compared with an expected result that may include expected telemetry and so forth. The results may be digested by a hash function where the digest becomes part of the digest value that is stored in the transaction block in the blockchain.

30 [0132] In another embodiment, the EFC logic may include a dedicated execution hosting environment that isolates the execution of a function such as F1 from other blockchain processing logic. For example, this may be a sandboxed Linux process, a container, virtual machine, TEE, enclave, FPGA design, secure co-processor/graphics processor, or ASIC. The

blockchain processing logic controls the invocation, monitoring, and collection of results of F1.

[0133] The function results, function identification, function vendor, code, expected result for a give set of test data, the location of the test data, type of implementation (e.g., bit-stream, bytecode, etc.), safety and security aspects, and performance metadata may be stored in a registry accessible by edge nodes 1002. The registry may be used to store known good functions and may be updated with consensus results of executions of functions. Other NFN or edge nodes 1002 may consult the registry to select which instance of a function to include in an edge SLA, workflow, or execution plan.

[0134] FIG. 11 is a block diagram illustrating edge function consensus (EFC) logic 1100 to implement function consensus blockchains, according to an embodiment. Logic may be implemented, arranged, or performed by a device, such as compute node 902, edge node 1002, compute node 600, or Edge computing node 650.

[0135] The EFC logic 1100 may include a number of logic components, such as modules, intellectual property (IP) blocks or cores, circuits, programmable circuitry, or mechanisms. Such logic may be hardware or firmware communicatively coupled to one or more processors in order to carry out the operations described herein. Logic may be hardware modules (e.g., respective IP blocks), and as such may be considered tangible entities capable of performing specified operations and may be configured or arranged in a certain manner. In an example, circuits may be arranged (e.g., internally or with respect to external entities such as other circuits) in a specified manner as an IP block, IP core, system-on-chip (SoC), or the like.

[0136] The EFC logic 1100 includes interface logic 1102 and execution logic 1110. The interface logic 1102 includes an execution interface 1104, named function provider interface 1106, and a local configuration interface 1108. The execution logic 1110 includes consensus logic 1112 and multicast logic 1114.

[0137] The execution interface 1104 is used to specify the level of consensus required in the execution of a named function. The execution interface 1104 receives parameters including 1) the name of the function, 2) the payload (data or location of data to be operated on), 3) a required amount of reliability, 4) a required amount of maximum variability, 5) security parameters, and 6) security requests that are a part of a call chain or execution pipeline.

[0138] The name of the function may be provided in an interest packet using the naming convention discussed above in ICN. The payload may be a named data source or may be literal values passed in the interest packet. The required amount of reliability may be an integer X , such that at least X number of peers (or a certain percentage of peers) must agree on the result

in order for that result to be considered as having met a consensus. The required amount of maximum variability may be an integer Y , such that results among the peers may have at most a variability of $Y\%$ in order to be considered within an acceptable range to achieve a consensus result. Security parameters may be used to select a certain flavor of a named function (e.g.,

5 version of named function), request all types of flavors to be executed, etc. The security requests may flow through a call chain or execution pipeline so that each node executing a function in the call chain or execution pipeline is expected to conform to the security parameters.

[0139] The named function provider interface 1106 is used for configuring the providers of named functions. The named function provider interface 1106 may receive parameters
10 including 1) the function name, 2) the identification of the function provider, and 3) known metadata for that provider and named function. The named function provider interface 1106 may be implemented by other components of the architecture (such as a named function discovery service).

[0140] The identification of the function provider may be an internet address (e.g., an IP
15 address, URI, or other unique identifier to locate the function). The known metadata may include the functions performance metrics (e.g., latency), accuracy, reliability, or other aspects of the function or function provider.

[0141] The local configuration interface 1108 is used to configure various local flavors of functions that are implemented on a compute node or edge node. The local configuration
20 interface 1108 may receive parameters including 1) the function name, 2) an identifier of the particular function flavor, 3) a bit-stream implementing the flavor, 4) the known metadata for that provider and named function.

[0142] The identifier of the particular function flavor may be a named function using the syntax of ICN (e.g., an IP address, URI, or other unique identifier). Alternatively, the identifier
25 of the function flavor may be an arbitrary unique value assigned by the edge network. The bit-stream may be instantiated in an FPGA on execution. Instead of a bit-stream, the parameter may be a location of the bit-stream, bytecode, or other executable instructions to execute the function flavor. The known metadata may include the functions performance metrics (e.g., latency), accuracy, reliability, or other aspects of the function or function provider.

30 [0143] Using the execution interface 1104, the execution logic 1110 is invoked. The execution logic 1110 selects a certain number of edge nodes M to execute the function. The number of edge nodes M may be determined by service level objectives, such as the required amount of reliability, latency requirements, or other parameters received by the execution interface 1104. Service level objectives may include aspects related to trustworthiness,

security, reliability, availability (beyond latency), resiliency, etc. Note that it may happen that M and the execution requirements (e.g., latency requirement) are not compatible. In that case, an error would be returned to the requestor (e.g., client device). Alternatively, the execution logic 1110 may accept a latency as a first constraint and the accuracy (degree of agreement) as a second constraint, and then if it cannot meet the latency without sacrificing accuracy beyond that which is acceptable, then it may return an error. It is understood that the first and second constraints discussed here are examples and that any number of constraints may be used in any order.

[0144] Using the multicast logic 1114, the execution logic 1110 sends the function to M other edge nodes (or $M-1$ in the case where the local node will execute the function and count as one of the M required nodes). The consensus logic 1112 is used to track results of the function's execution at other edge nodes and determine when a consensus is reached. If there is a consensus based on the parameters (e.g., a common result according to the percentage of accepted variation), then the consensus function result is added to the blockchain using standard blockchain flows and the consensus function result is sent back to the requestor. If there is no consensus, the consensus logic 1112 may attempt to achieve a consensus by enlisting additional edge nodes to execute the function or may return the result to the requestor with the associated variation seen in all the results (notifying that consensus did not happen). The operation of enlisting additional edge nodes to gain a consensus may depend on time available in the latency SLA on the request, the number of available edge nodes, the percentage of agreeing edge nodes in the initial execution, or other factors. For instance, if there is consensus of 49% of the edge nodes, then there may be a few nodes that are producing results that are only slightly out of agreement. By enlisting additional nodes, the percentage of agreement may increase and the function may have a consensus result. Alternatively, if the highest bloc of edge nodes in agreement is only 17%, then it may be too burdensome to have many more edge nodes enlisted to attempt to gain consensus.

[0145] The consensus may be used in establishing trust of a function, a flavor of a function, or function provider. Once the function (or flavor) is executed and a consensus is reached regarding the results, the entry in the blockchain acts as a record of the function's execution. Other function providers, edge nodes, clients, edge platforms, orchestrators, or the like may refer to the blockchain to determine a trust level of a function, function flavor, or function provider. In some implementations, a limited trust may be established when there are fewer than a threshold number of entries in a blockchain for a function, function flavor, or function provider. As more entries are added to a blockchain, the trust level is increased. The trust level

of a function, function flavor, or function provider may be based on the number of entries in the blockchain. Note that if there is no consensus, then a blockchain entry is not added. As such, a baseline of trust (e.g., a single entry) is already a solid indicator of validity and trustworthiness.

5 [0146] FIG. 12 is a diagram illustrating a consensus-based named function networking system 1200, according to an embodiment. The consensus-based named function networking system 1200 includes edge compute node 1202A, edge compute node 1202B, edge compute node 1202C, ..., edge compute node 1202N (collectively referred to as edge compute nodes 1202). The edge compute nodes 1202 are used to execute some or all of a workflow 1204. The
10 workflow 1204 may be divided up into two or more workloads (e.g., workload 1206A, workload 1206B, ..., workload 1206N), with each workload 1206 involving a single function or a pipeline of functions. In the case of a function pipeline, the workload 1206 is further deconstructed into two or more functions (e.g., function 1208A, function 1208B, ..., function 1208N). It is understood that a workflow 1204 may include one workload 1206. Further it is
15 understood that a workload 1206 may include a single function 1208.

[0147] Edge nodes 1202 are used to execute the respective function 1208 where multiple edge compute nodes 1202 may be enlisted to execute an instance of the same function to determine a consensus result. Edge compute nodes 1202 may be associated with one or more blockchains (e.g., blockchain 1210A, blockchain 1210B, ..., blockchain 1210N) and use the
20 consensus algorithms described here to identify a consensus result and store the consensus result in a transaction in the respective blockchain 1210.

[0148] FIG. 13 is a flowchart illustrating a method 1300 for coordinating named function execution in a named function network, according to an embodiment. The method 1300 may be performed by a device, such as a network appliance computing device, a compute node 902,
25 an edge node 1002, a compute node 600, or an Edge computing node 650.

[0149] At 1302, an interest packet received from a user device is accessed, the interest packet including a function name of a function and a data payload.

[0150] In an embodiment, the interest packet includes a latency-based service level objective. In another embodiment, the interest packet includes a deviation-based service level
30 objective. In a further embodiment, the method 1300 includes determining a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.

[0151] At 1304, the interest packet is broadcasted to a plurality of compute nodes, where the plurality of compute nodes are configured to execute a respective instance of the function.

[0152] In an embodiment, wherein the plurality of compute nodes are members of a blockchain network. In a further embodiment, the network appliance computing device is a member of the blockchain network.

5 [0153] In an embodiment, the interest packet is broadcasted to the plurality of compute nodes using named functioned networking (NFN) to multicast the interest packet to the plurality of compute nodes.

[0154] At 1306, a plurality of responses is received from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function.

10 [0155] At 1308, the plurality of responses is analyzed using a consensus protocol to identify a consensus result. In an embodiment, the consensus protocol is a majority protocol.

[0156] At 1310, the consensus result is transmitted to the user device. In an embodiment, the consensus result indicates a percentage of compute nodes that formed the consensus result. In a related embodiment, the consensus result indicates that a consensus was not reached. In an embodiment, the method 1300 includes adding the consensus result to a block in a blockchain managed by the blockchain network.

15 [0157] Embodiments may be implemented in one or a combination of hardware, firmware, and software. Embodiments may also be implemented as instructions stored on a machine-readable storage device, which may be read and executed by at least one processor to perform the operations described herein. A machine-readable storage device may include any non-transitory mechanism for storing information in a form readable by a machine (e.g., a computer). For example, a machine-readable storage device may include read-only memory (ROM), random-access memory (RAM), magnetic disk storage media, optical storage media, flash-memory devices, and other storage devices and media.

20 [0158] Examples, as described herein, may include, or may operate on, logic or a number of components, such as modules, intellectual property (IP) blocks or cores, or mechanisms. Such logic or components may be hardware, software, or firmware communicatively coupled to one or more processors in order to carry out the operations described herein. Logic or components may be hardware modules (e.g., IP block), and as such may be considered tangible entities capable of performing specified operations and may be configured or arranged in a certain manner. In an example, circuits may be arranged (e.g., internally or with respect to external entities such as other circuits) in a specified manner as an IP block, IP core, system-on-chip (SoC), or the like.

[0159] In an example, the whole or part of one or more computer systems (e.g., a standalone, client or server computer system) or one or more hardware processors may be configured by firmware or software (e.g., instructions, an application portion, or an application) as a module that operates to perform specified operations. In an example, the software may reside on a machine-readable medium. In an example, the software, when executed by the underlying hardware of the module, causes the hardware to perform the specified operations. Accordingly, the term hardware module is understood to encompass a tangible entity, be that an entity that is physically constructed, specifically configured (e.g., hardwired), or temporarily (e.g., transitorily) configured (e.g., programmed) to operate in a specified manner or to perform part or all of any operation described herein.

[0160] Considering examples in which modules are temporarily configured, each of the modules need not be instantiated at any one moment in time. For example, where the modules comprise a general-purpose hardware processor configured using software; the general-purpose hardware processor may be configured as respective different modules at different times. Software may accordingly configure a hardware processor, for example, to constitute a particular module at one instance of time and to constitute a different module at a different instance of time. Modules may also be software or firmware modules, which operate to perform the methodologies described herein.

[0161] An IP block (also referred to as an IP core) is a reusable unit of logic, cell, or integrated circuit. An IP block may be used as a part of a field programmable gate array (FPGA), application-specific integrated circuit (ASIC), programmable logic device (PLD), system on a chip (SoC), or the like. It may be configured for a particular purpose, such as digital signal processing or image processing. Example IP cores include central processing unit (CPU) cores, integrated graphics, security, input/output (I/O) control, system agent, graphics processing unit (GPU), artificial intelligence, neural processors, image processing unit, communication interfaces, memory controller, peripheral device control, platform controller hub, or the like.

Additional Notes & Examples:

[0162] Example 1 is a network appliance computing device, comprising: a processor; and memory to store instructions, which when executed by the processor, cause the network appliance computing device to: access an interest packet received from a user device, the interest packet including a function name of a function and a data payload; broadcast the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are

configured to execute a respective instance of the function; receive a plurality of responses from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function; analyze the plurality of responses using a consensus protocol to identify a consensus result; and transmit the consensus result to the user device.

[0163] In Example 2, the subject matter of Example 1 includes, wherein the plurality of compute nodes are members of a blockchain network.

[0164] In Example 3, the subject matter of Example 2 includes, wherein the network appliance computing device is a member of the blockchain network.

10 [0165] In Example 4, the subject matter of Example 3 includes, wherein the network appliance computing device adds the consensus result to a block in a blockchain managed by the blockchain network.

[0166] In Example 5, the subject matter of Examples 1–4 includes, wherein to broadcast the interest packet to the plurality of compute nodes, the network applicant is configured to use named functioned networking (NFN) to multicast the interest packet to the plurality of compute nodes.

[0167] In Example 6, the subject matter of Examples 1–5 includes, wherein the interest packet includes a latency-based service level objective.

20 [0168] In Example 7, the subject matter of Example 6 includes, wherein the interest packet includes a deviation-based service level objective.

[0169] In Example 8, the subject matter of Examples 6–7 includes, wherein the network appliance computing device is to determine a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.

25 [0170] In Example 9, the subject matter of Examples 1–8 includes, wherein the consensus protocol is a majority protocol.

[0171] In Example 10, the subject matter of Examples 1–9 includes, wherein the consensus result indicates a percentage of compute nodes that formed the consensus result.

[0172] In Example 11, the subject matter of Examples 1–10 includes, wherein the consensus result indicates that a consensus was not reached.

30 [0173] Example 12 is a method performed by a network appliance computing device, comprising: accessing an interest packet received from a user device, the interest packet including a function name of a function and a data payload; broadcasting the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are configured to execute a respective instance of the function; receiving a plurality of responses from the plurality of

compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function; analyzing the plurality of responses using a consensus protocol to identify a consensus result; and transmitting the consensus result to the user device.

5 [0174] In Example 13, the subject matter of Example 12 includes, wherein the plurality of compute nodes are members of a blockchain network.

[0175] In Example 14, the subject matter of Example 13 includes, wherein the network appliance computing device is a member of the blockchain network.

[0176] In Example 15, the subject matter of Example 14 includes, adding the consensus result to a block in a blockchain managed by the blockchain network.

10 [0177] In Example 16, the subject matter of Examples 12–15 includes, wherein broadcasting the interest packet to the plurality of compute nodes comprises using named functioned networking (NFN) to multicast the interest packet to the plurality of compute nodes.

[0178] In Example 17, the subject matter of Examples 12–16 includes, wherein the interest packet includes a latency-based service level objective.

15 [0179] In Example 18, the subject matter of Example 17 includes, wherein the interest packet includes a deviation-based service level objective.

[0180] In Example 19, the subject matter of Examples 17–18 includes, determining a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.

20 [0181] In Example 20, the subject matter of Examples 12–19 includes, wherein the consensus protocol is a majority protocol.

[0182] In Example 21, the subject matter of Examples 12–20 includes, wherein the consensus result indicates a percentage of compute nodes that formed the consensus result.

25 [0183] In Example 22, the subject matter of Examples 12–21 includes, wherein the consensus result indicates that a consensus was not reached.

[0184] Example 23 is at least one machine-readable medium including instructions, which when performed by a network appliance computing device, cause the network appliance computing device to: access an interest packet received from a user device, the interest packet including a function name of a function and a data payload; broadcast the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are configured to execute a respective instance of the function; receive a plurality of responses from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function; analyze the plurality of responses using a consensus protocol to identify a consensus result; and transmit the consensus result to the user device.

30

- [0185] In Example 24, the subject matter of Example 23 includes, wherein the plurality of compute nodes are members of a blockchain network.
- [0186] In Example 25, the subject matter of Example 24 includes, wherein the network appliance computing device is a member of the blockchain network.
- 5 [0187] In Example 26, the subject matter of Example 25 includes, instructions to add the consensus result to a block in a blockchain managed by the blockchain network.
- [0188] In Example 27, the subject matter of Examples 23–26 includes, wherein to broadcast the interest packet to the plurality of compute nodes, the instructions cause the network applicant computing device to use named functioned networking (NFN) to multicast the
- 10 interest packet to the plurality of compute nodes.
- [0189] In Example 28, the subject matter of Examples 23–27 includes, wherein the interest packet includes a latency-based service level objective.
- [0190] In Example 29, the subject matter of Example 28 includes, wherein the interest packet includes a deviation-based service level objective.
- 15 [0191] In Example 30, the subject matter of Examples 28–29 includes, instructions to determine a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.
- [0192] In Example 31, the subject matter of Examples 23–30 includes, wherein the consensus protocol is a majority protocol.
- 20 [0193] In Example 32, the subject matter of Examples 23–31 includes, wherein the consensus result indicates a percentage of compute nodes that formed the consensus result.
- [0194] In Example 33, the subject matter of Examples 23–32 includes, wherein the consensus result indicates that a consensus was not reached.
- [0195] Example 34 is an edge computing node, operable in an edge computing system,
- 25 comprising processing circuitry configured to implement any of the examples of 1–33.
- [0196] Example 35 is an edge computing node, operable as a server in an edge computing system, configured to perform any of the examples of 1–33.
- [0197] Example 36 is an edge computing node, operable as a client in an edge computing system, configured to perform any of the examples of 1–33.
- 30 [0198] Example 37 is an edge computing node, operable in a layer of an edge computing network as an aggregation node, network hub node, gateway node, or core data processing node, configured to perform any of the examples of 1–33.

- [0199] Example 38 is an edge computing network, comprising networking and processing components configured to provide or operate a communications network, to enable an edge computing system to implement any of the examples of 1–33.
- 5 [0200] Example 39 is an access point, comprising networking and processing components configured to provide or operate a communications network, to enable an edge computing system to implement any of the examples of 1–33.
- [0201] Example 40 is a base station, comprising networking and processing components configured to provide or operate a communications network, to enable an edge computing system to implement any of the examples of 1–33.
- 10 [0202] Example 41 is a road-side unit, comprising networking components configured to provide or operate a communications network, to enable an edge computing system to implement any of the examples of 1–33.
- [0203] Example 42 is an on-premise server, operable in a private communications network distinct from a public edge computing network, the server configured to enable an edge
- 15 computing system to implement any of the examples of 1–33.
- [0204] Example 43 is a 3GPP 4G/LTE mobile wireless communications system, comprising networking and processing components configured with the biometric security methods of any of the examples of 1–33.
- [0205] Example 44 is a 5G network mobile wireless communications system, comprising
- 20 networking and processing components configured with the biometric security methods of any of the examples of 1–33.
- [0206] Example 45 is a user equipment device, comprising networking and processing circuitry, configured to connect with an edge computing system configured to implement any of the examples of 1–33.
- 25 [0207] Example 46 is a client computing device, comprising processing circuitry, configured to coordinate compute operations with an edge computing system, the edge computing system configured to implement any of the examples of 1–33.
- [0208] Example 47 is an edge provisioning node, operable in an edge computing system, configured to implement any of the examples of 1–33.
- 30 [0209] Example 48 is a service orchestration node, operable in an edge computing system, configured to implement any of the examples of 1–33.
- [0210] Example 49 is an application orchestration node, operable in an edge computing system, configured to implement any of the examples of 1–33.

- [0211] Example 50 is a multi-tenant management node, operable in an edge computing system, configured to implement any of the examples of 1–33.
- [0212] Example 51 is an edge computing system comprising processing circuitry, the edge computing system configured to operate one or more functions and services to implement any
5 of the examples of 1–33.
- [0213] Example 52 is networking hardware with network functions implemented thereupon, operable within an edge computing system configured with the biometric security methods of any of examples of 1–33.
- [0214] Example 53 is acceleration hardware with acceleration functions implemented
10 thereupon, operable in an edge computing system, the acceleration functions configured to implement any of the examples of 1–33.
- [0215] Example 54 is storage hardware with storage capabilities implemented thereupon, operable in an edge computing system, the storage hardware configured to implement any of the examples of 1–33.
- [0216] Example 55 is computation hardware with compute capabilities implemented
15 thereupon, operable in an edge computing system, the computation hardware configured to implement any of the examples of 1–33.
- [0217] Example 56 is an edge computing system adapted for supporting vehicle-to-vehicle (V2V), vehicle-to-everything (V2X), or vehicle-to-infrastructure (V2I) scenarios, configured to
20 implement any of the examples of 1–33.
- [0218] Example 57 is an edge computing system configured as an edge mesh, provided with a microservice cluster, a microservice cluster with sidecars, or linked microservice clusters with sidecars, configured to implement any of the examples of 1–33.
- [0219] Example 58 is an edge computing system, comprising circuitry configured to
25 implement one or more isolation environments provided among dedicated hardware, virtual machines, containers, virtual machines on containers, configured to implement any of the examples of 1–33.
- [0220] Example 59 is an edge computing server, configured for operation as an enterprise server, roadside server, street cabinet server, or telecommunications server, configured to
30 implement any of the examples of 1–33.
- [0221] Example 60 is an edge computing system configured to implement any of the examples of 1–33 with use cases provided from one or more of: compute offload, data caching, video processing, network function virtualization, radio access network management, augmented reality, virtual reality, autonomous driving, vehicle assistance, vehicle

communications, industrial automation, retail services, manufacturing operations, smart buildings, energy management, internet of things operations, object detection, speech recognition, healthcare applications, gaming applications, or accelerated content processing.

5 [0222] Example 61 is an edge computing system, comprising computing nodes operated by multiple owners at different geographic locations, configured to implement any of the examples of 1–33.

[0223] Example 62 is a cloud computing system, comprising data servers operating respective cloud services, the respective cloud services configured to coordinate with an edge computing system to implement any of the examples of 1–33.

10 [0224] Example 63 is a server, comprising hardware to operate cloudlet, edgelet, or applet services, the services configured to coordinate with an edge computing system to implement any of the examples of 1–33.

[0225] Example 64 is an edge node in an edge computing system, comprising one or more devices with at least one processor and memory to implement any of the examples of 1–33.

15 [0226] Example 65 is an edge node in an edge computing system, the edge node operating one or more services provided from among: a management console service, a telemetry service, a provisioning service, an application or service orchestration service, a virtual machine service, a container service, a function deployment service, or a compute deployment service, or an acceleration management service, the one or more services configured to
20 implement any of the examples of 1–33.

[0227] Example 66 is a set of distributed edge nodes, distributed among a network layer of an edge computing system, the network layer comprising a close edge, local edge, enterprise edge, on-premise edge, near edge, middle, edge, or far edge network layer, configured to implement any of the examples of 1–33.

25 [0228] Example 67 is an apparatus of an edge computing system comprising: one or more processors and one or more computer-readable media comprising instructions that, when executed by the one or more processors, cause the one or more processors to perform any of the examples of 1–33.

[0229] Example 68 is one or more computer-readable storage media comprising instructions
30 to cause an electronic device of an edge computing system, upon execution of the instructions by one or more processors of the electronic device, to perform any of the examples of 1–33.

[0230] Example 69 is a communication signal communicated in an edge computing system, to perform any of the examples of 1–33.

- [0231] Example 70 is a data structure communicated in an edge computing system, the data structure comprising a datagram, packet, frame, segment, protocol data unit (PDU), or message, to perform any of the examples of 1–33.
- 5 [0232] Example 71 is a signal communicated in an edge computing system, the signal encoded with a datagram, packet, frame, segment, protocol data unit (PDU), message, or data to perform any of the examples of 1–33.
- 10 [0233] Example 72 is an electromagnetic signal communicated in an edge computing system, the electromagnetic signal carrying computer-readable instructions, wherein execution of the computer-readable instructions by one or more processors causes the one or more processors to perform any of the examples of 1–33.
- [0234] Example 73 is a computer program used in an edge computing system, the computer program comprising instructions, wherein execution of the program by a processing element in the edge computing system is to cause the processing element to perform any of the examples of 1–33.
- 15 [0235] Example 74 is an apparatus of an edge computing system comprising means to perform any of the examples of 1–33.
- [0236] Example 75 is an apparatus of an edge computing system comprising logic, modules, or circuitry to perform any of the examples of 1–33.
- 20 [0237] Example 76 is at least one machine-readable medium including instructions that, when executed by processing circuitry, cause the processing circuitry to perform operations to implement of any of Examples 1–33.
- [0238] Example 77 is an apparatus comprising means to implement of any of Examples 1–33.
- [0239] Example 78 is a system to implement of any of Examples 1–33.
- 25 [0240] Example 79 is a method to implement of any of Examples 1–33.
- [0241] The above detailed description includes references to the accompanying drawings, which form a part of the detailed description. The drawings show, by way of illustration, specific embodiments that may be practiced. These embodiments are also referred to herein as “examples.” Such examples may include elements in addition to those shown or described.
- 30 However, also contemplated are examples that include the elements shown or described. Moreover, also contemplated are examples using any combination or permutation of those elements shown or described (or one or more aspects thereof), either with respect to a particular example (or one or more aspects thereof), or with respect to other examples (or one or more aspects thereof) shown or described herein.

[0242] Publications, patents, and patent documents referred to in this document are incorporated by reference herein in their entirety, as though individually incorporated by reference. In the event of inconsistent usages between this document and those documents so incorporated by reference, the usage in the incorporated reference(s) are supplementary to that of this document; for irreconcilable inconsistencies, the usage in this document controls.

[0243] In this document, the terms “a” or “an” are used, as is common in patent documents, to include one or more than one, independent of any other instances or usages of “at least one” or “one or more.” In this document, the term “or” is used to refer to a nonexclusive or, such that “A or B” includes “A but not B,” “B but not A,” and “A and B,” unless otherwise indicated. In the appended claims, the terms “including” and “in which” are used as the plain-English equivalents of the respective terms “comprising” and “wherein.” Also, in the following claims, the terms “including” and “comprising” are open-ended, that is, a system, device, article, or process that includes elements in addition to those listed after such a term in a claim are still deemed to fall within the scope of that claim. Moreover, in the following claims, the terms “first,” “second,” and “third,” etc. are used merely as labels, and are not intended to suggest a numerical order for their objects.

[0244] The above description is intended to be illustrative, and not restrictive. For example, the above-described examples (or one or more aspects thereof) may be used in combination with others. Other embodiments may be used, such as by one of ordinary skill in the art upon reviewing the above description. The Abstract is to allow the reader to quickly ascertain the nature of the technical disclosure. It is submitted with the understanding that it will not be used to interpret or limit the scope or meaning of the claims. Also, in the above Detailed Description, various features may be grouped together to streamline the disclosure. However, the claims may not set forth every feature disclosed herein as embodiments may feature a subset of said features. Further, embodiments may include fewer features than those disclosed in a particular example. Thus, the following claims are hereby incorporated into the Detailed Description, with a claim standing on its own as a separate embodiment. The scope of the embodiments disclosed herein is to be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled.

CLAIMS

What is claimed is:

1. A network appliance computing device, comprising:
5 a processor; and
memory to store instructions, which when executed by the processor, cause the network appliance computing device to:
access an interest packet received from a user device, the interest packet including a function name of a function and a data payload;
10 broadcast the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are configured to execute a respective instance of the function;
receive a plurality of responses from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective
15 instances of the function;
analyze the plurality of responses using a consensus protocol to identify a consensus result; and
transmit the consensus result to the user device.
- 20 2. The network appliance computing device of claim 1, wherein the plurality of compute nodes are members of a blockchain network.
3. The network appliance computing device of claim 2, wherein the network appliance computing device is a member of the blockchain network.
- 25 4. The network appliance computing device of claim 3, wherein the network appliance computing device adds the consensus result to a block in a blockchain managed by the blockchain network.
- 30 5. The network appliance computing device of claim 1, wherein to broadcast the interest packet to the plurality of compute nodes, the network applicant is configured to use named functioned networking (NFN) to multicast the interest packet to the plurality of compute nodes.

6. The network appliance computing device of claim 1, wherein the interest packet includes a latency-based service level objective.
7. The network appliance computing device of claim 6, wherein the interest packet
5 includes a deviation-based service level objective.
8. The network appliance computing device of claim 6, wherein the network appliance computing device is to determine a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.
10
9. The network appliance computing device of claim 1, wherein the consensus protocol is a majority protocol.
10. The network appliance computing device of claim 1, wherein the consensus result
15 indicates a percentage of compute nodes that formed the consensus result.
11. The network appliance computing device of claim 1, wherein the consensus result indicates that a consensus was not reached.
- 20 12. A method performed by a network appliance computing device, comprising:
accessing an interest packet received from a user device, the interest packet including a function name of a function and a data payload;
broadcasting the interest packet to a plurality of compute nodes, wherein the plurality of compute nodes are configured to execute a respective instance of the function;
25 receiving a plurality of responses from the plurality of compute nodes, the plurality of responses including respective results of the execution of the respective instances of the function;
analyzing the plurality of responses using a consensus protocol to identify a consensus result; and
30 transmitting the consensus result to the user device.
13. The method of claim 12, wherein the plurality of compute nodes are members of a blockchain network.

14. The method of claim 13, wherein the network appliance computing device is a member of the blockchain network.
15. The method of claim 14, comprising adding the consensus result to a block in a
5 blockchain managed by the blockchain network.
16. The method of claim 12, wherein broadcasting the interest packet to the plurality of compute nodes comprises using named functioned networking (NFN) to multicast the interest packet to the plurality of compute nodes.
10
17. The method of claim 12, wherein the interest packet includes a latency-based service level objective.
18. The method of claim 17, wherein the interest packet includes a deviation-based service
15 level objective.
19. The method of claim 17, comprising determining a number of compute nodes to broadcast the interest packet to, based on the latency-based service level objective.
20. The method of claim 12, wherein the consensus protocol is a majority protocol.
20
21. The method of claim 12, wherein the consensus result indicates a percentage of compute nodes that formed the consensus result.
22. The method of claim 12, wherein the consensus result indicates that a consensus was
25 not reached.
23. At least one machine-readable medium including instructions, which when executed by a machine, cause the machine to perform operations of any of the methods of claims 12-22.
30
24. An apparatus comprising means for performing any of the methods of claims 12-22.

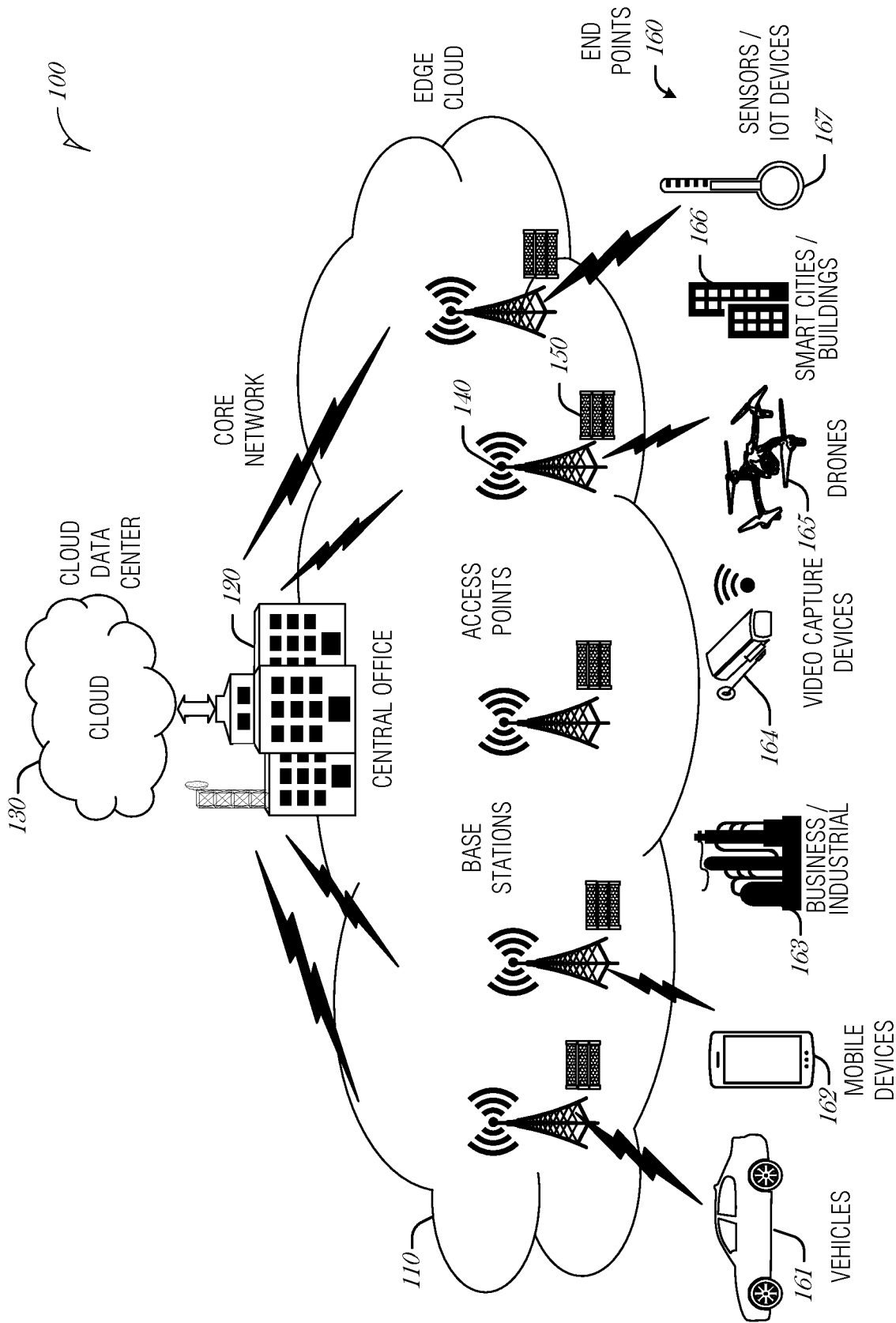


FIG. 1

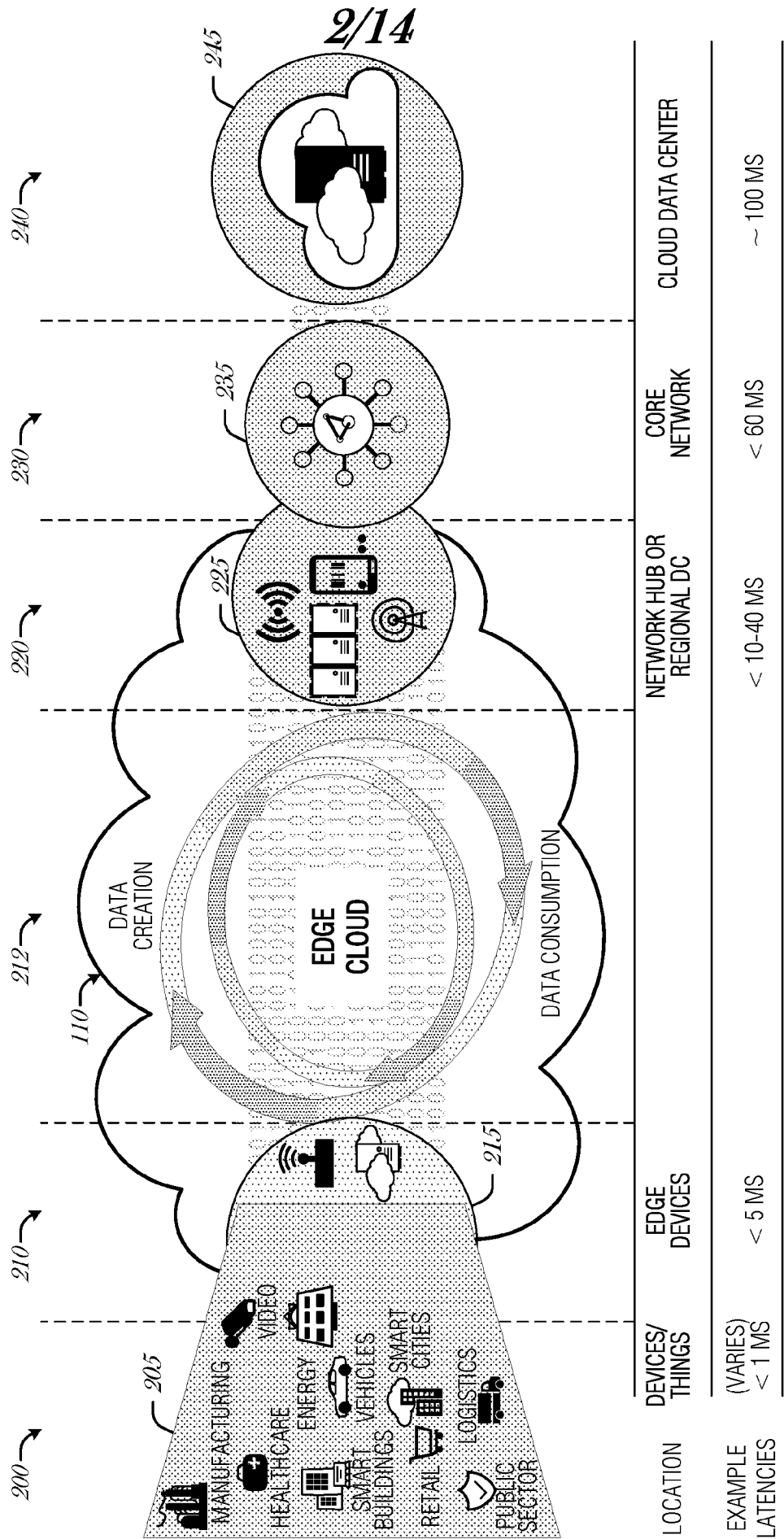


FIG. 2

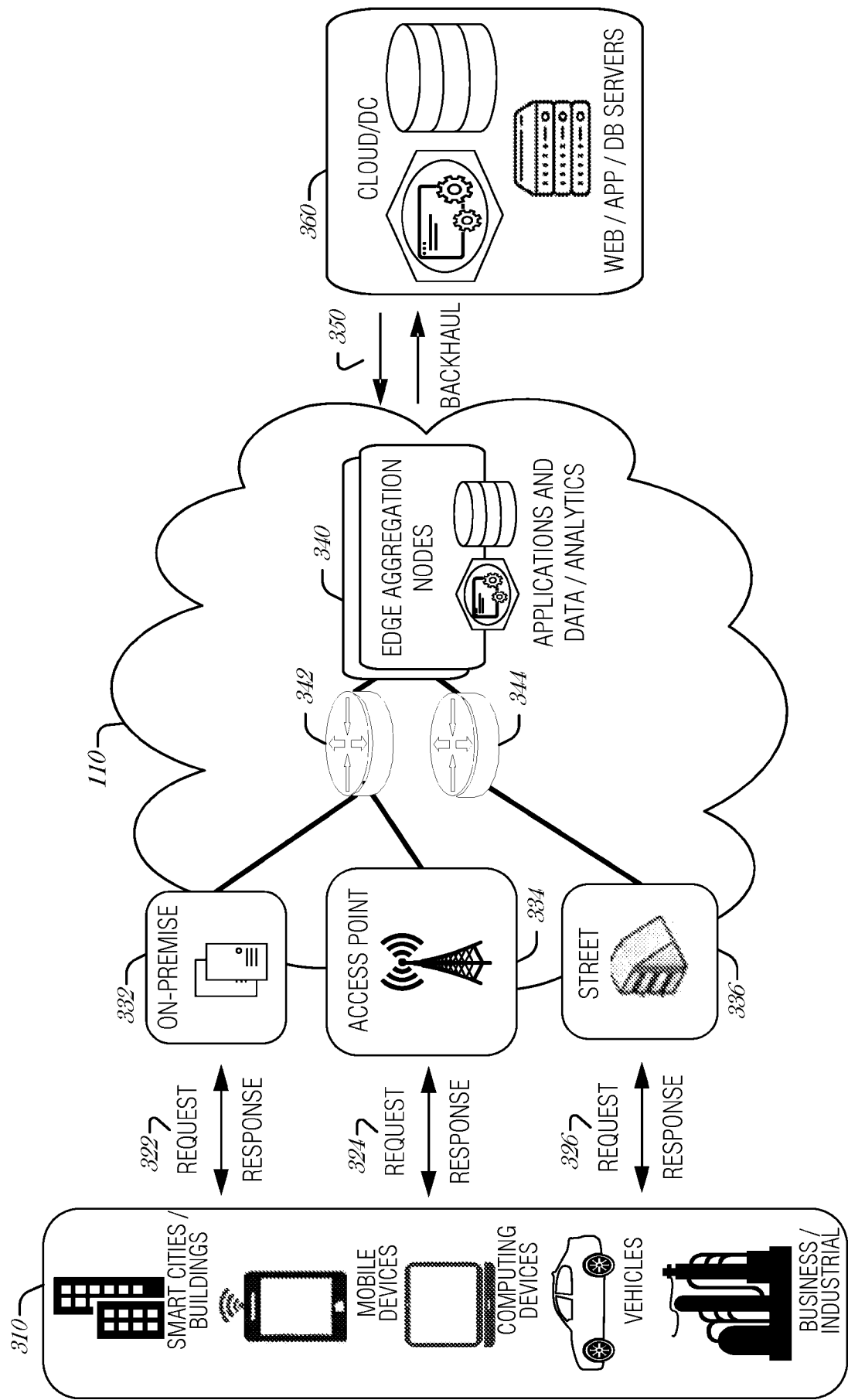


FIG. 3

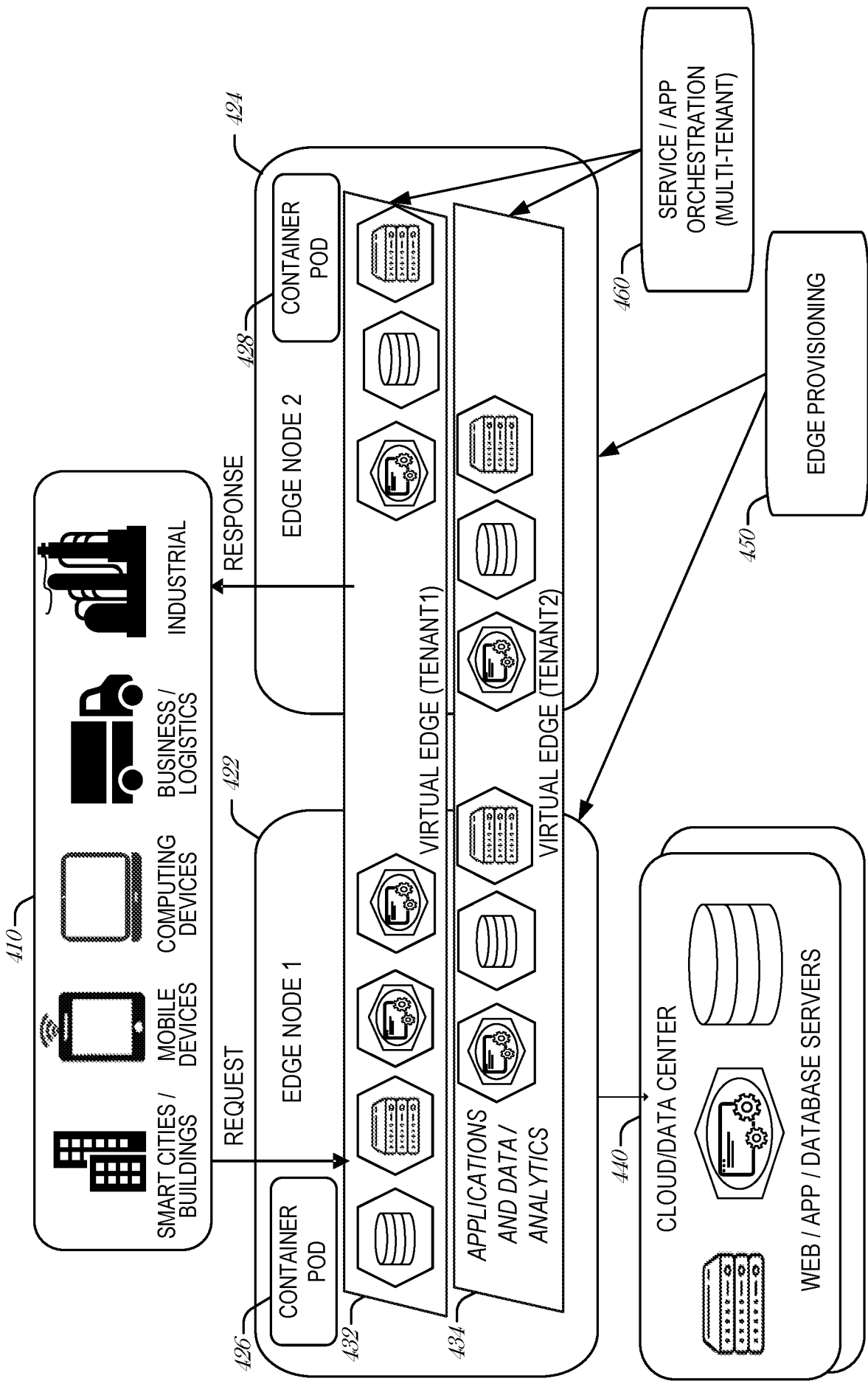
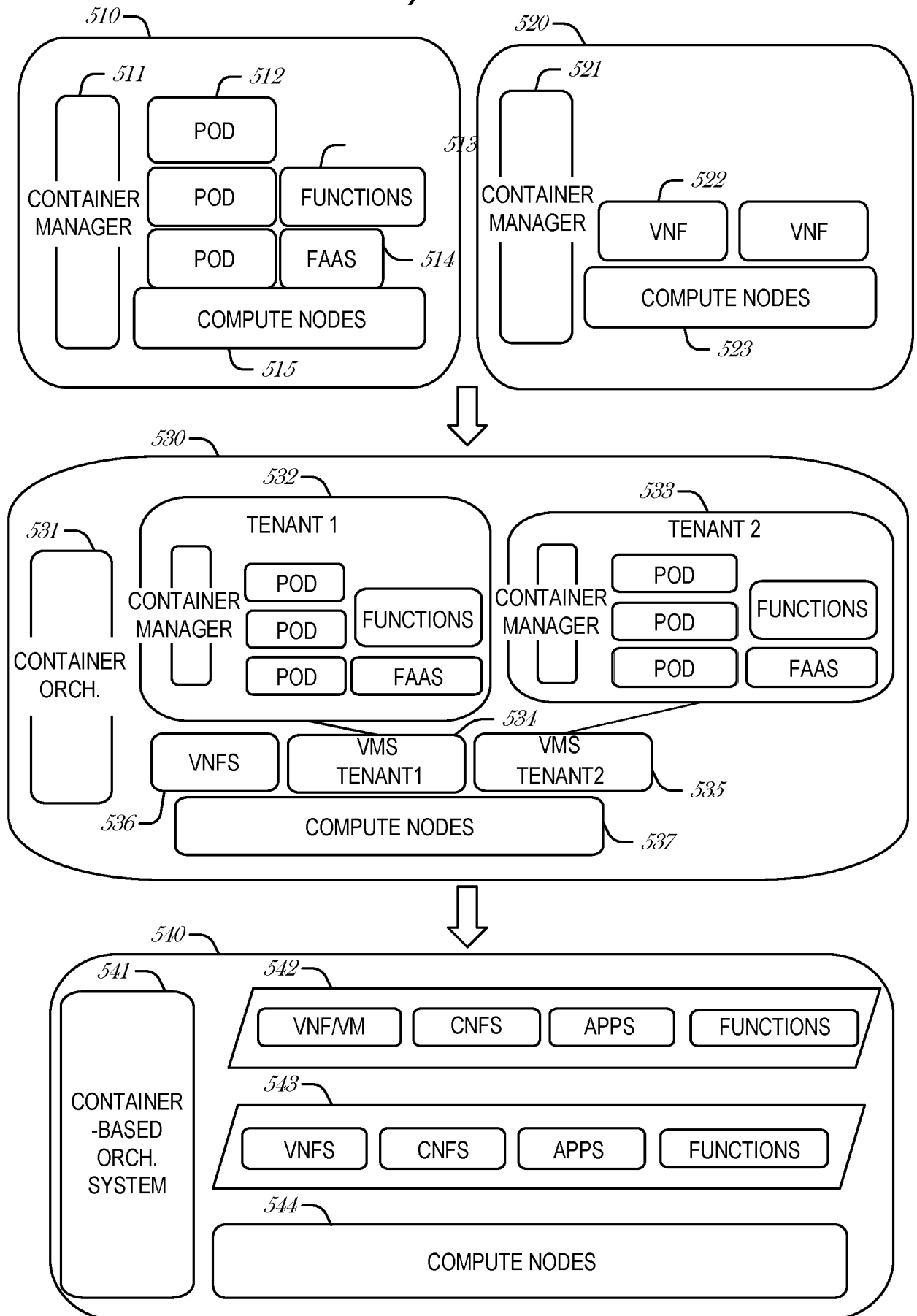


FIG. 4

5/14**FIG. 5**

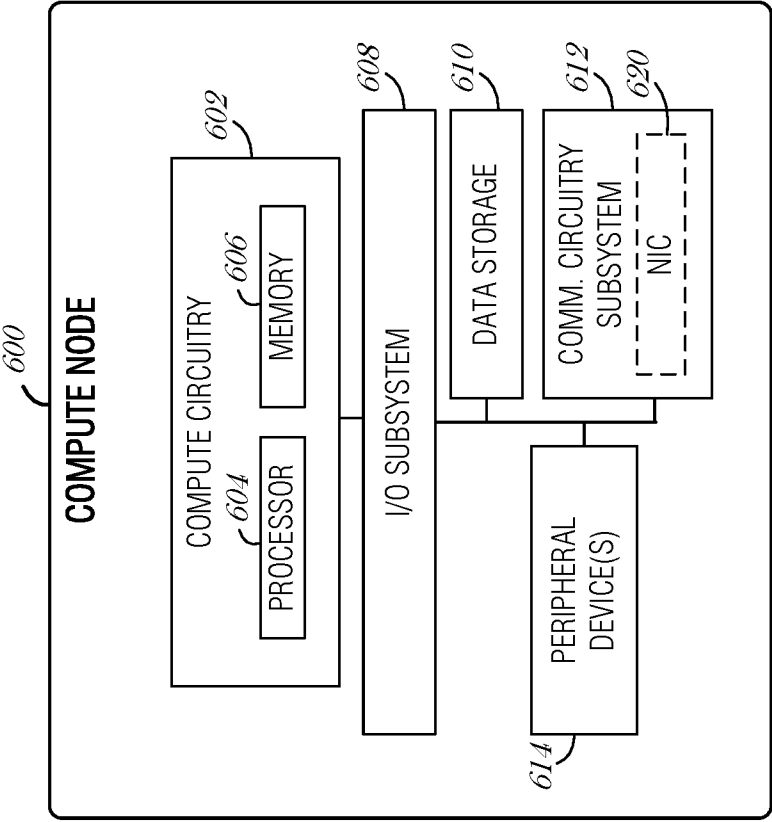


FIG. 6A

7/14

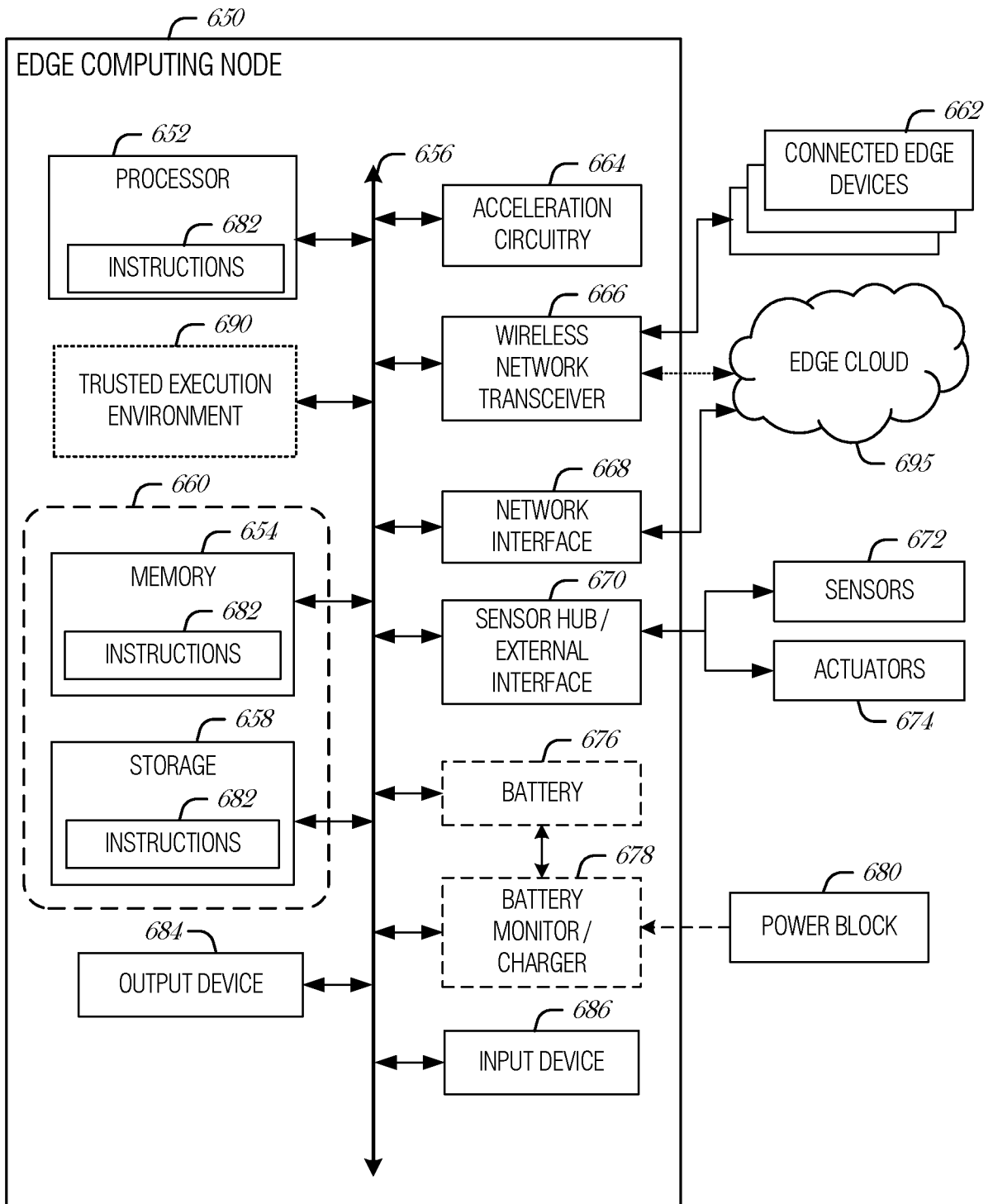
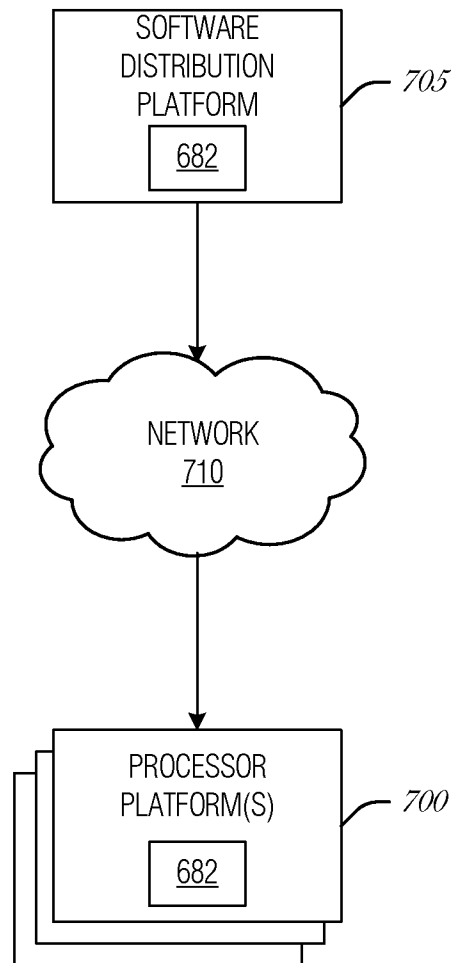


FIG. 6B

8/14**FIG. 7**

9/14

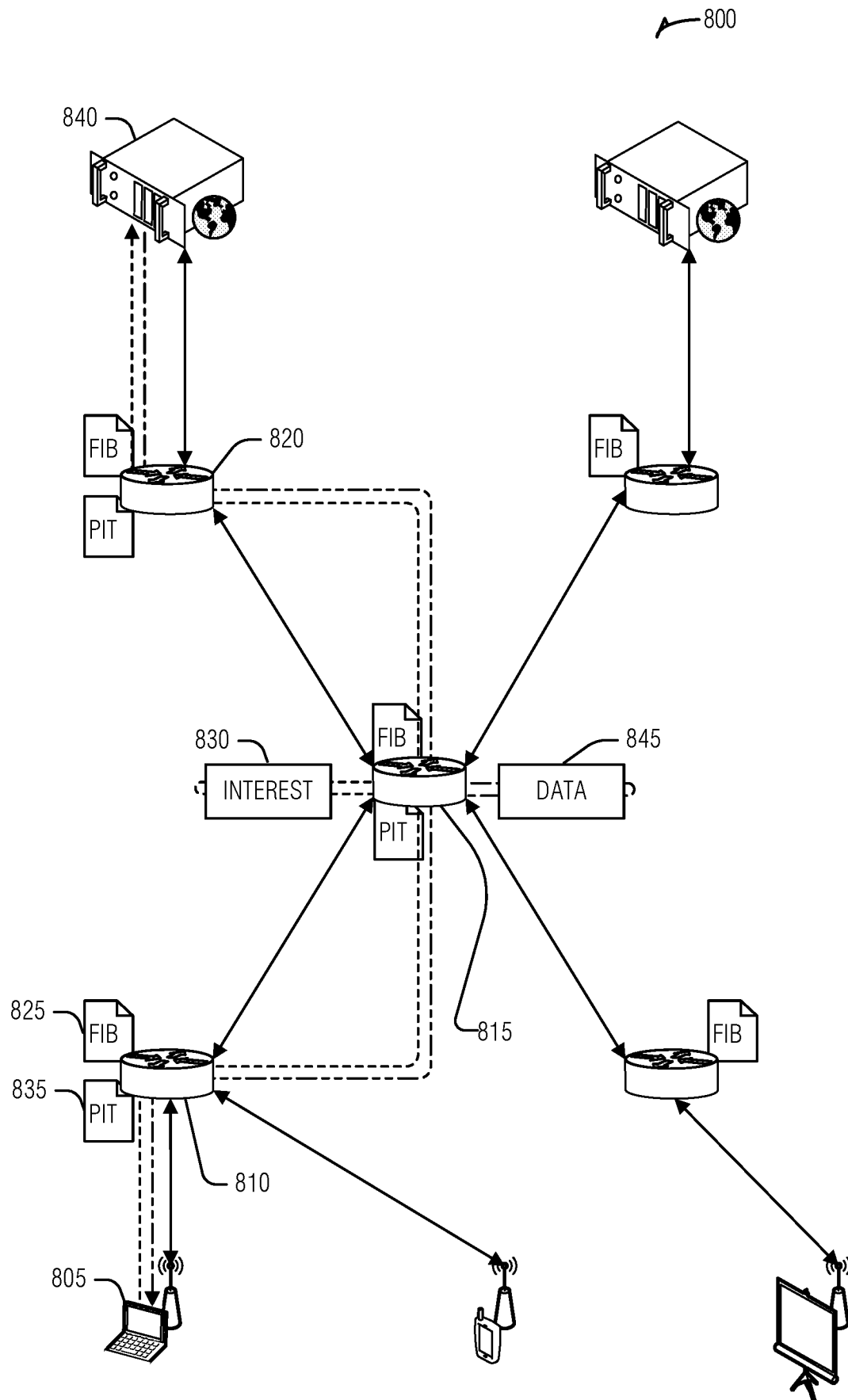


FIG. 8

10/14

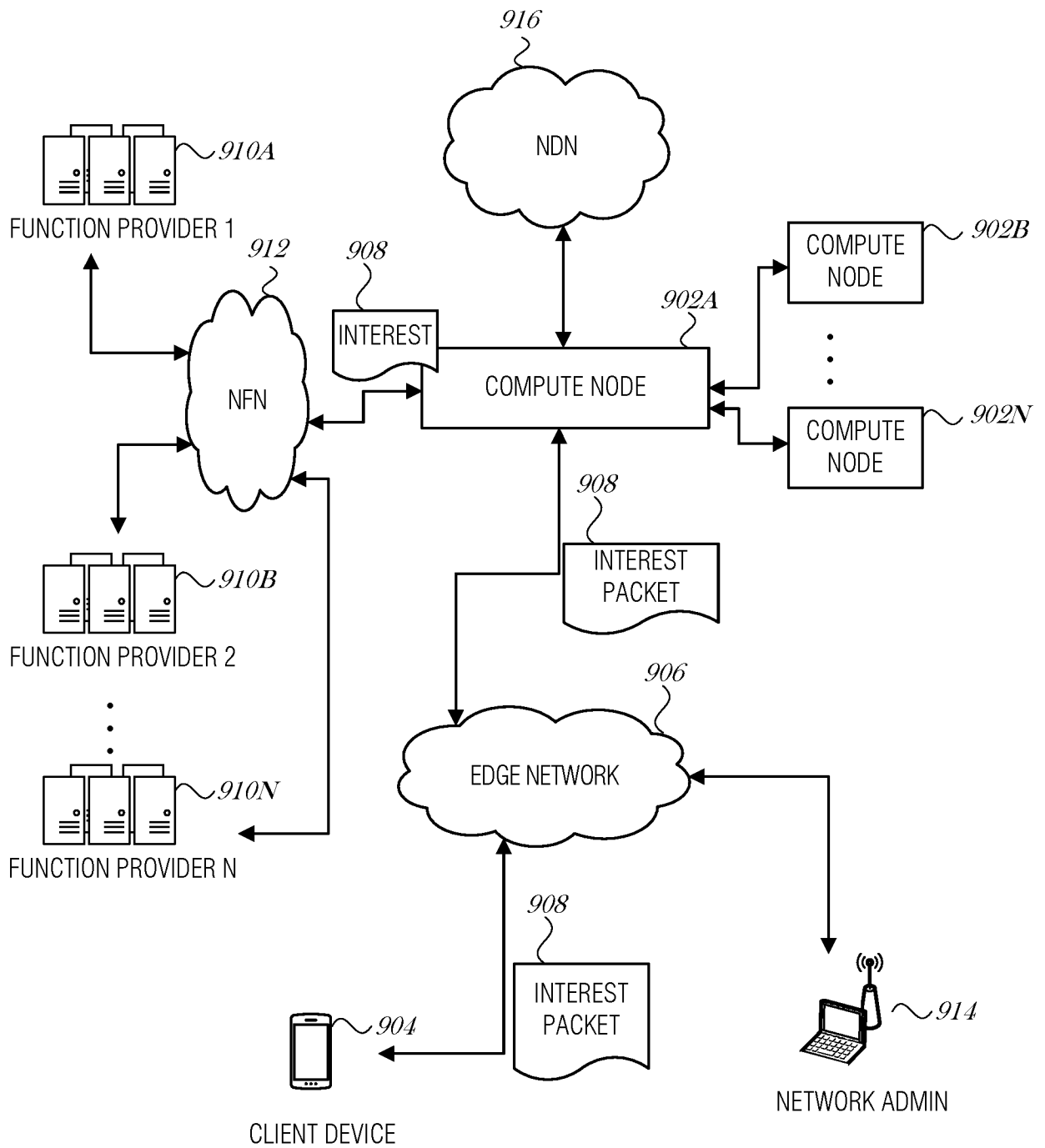
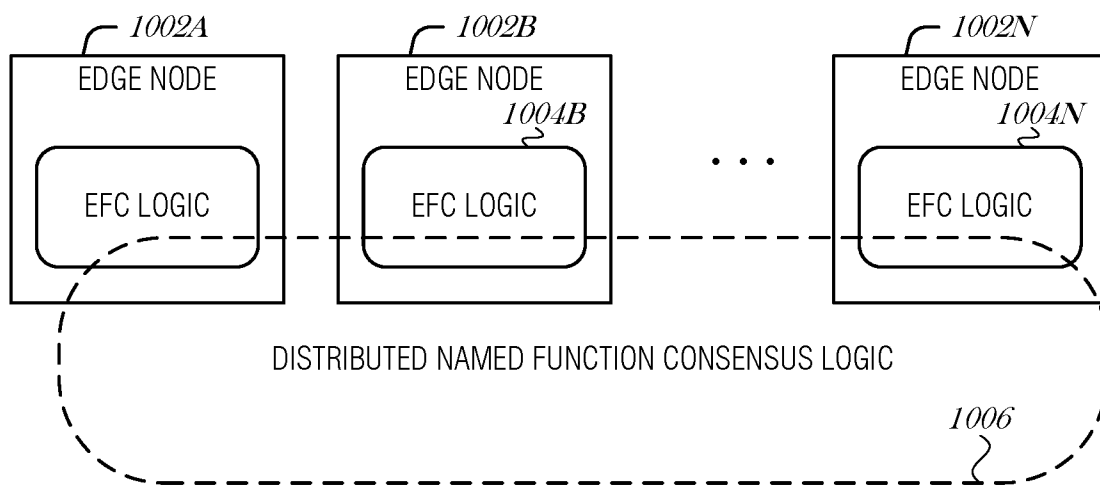


FIG. 9

11/14**FIG. 10**

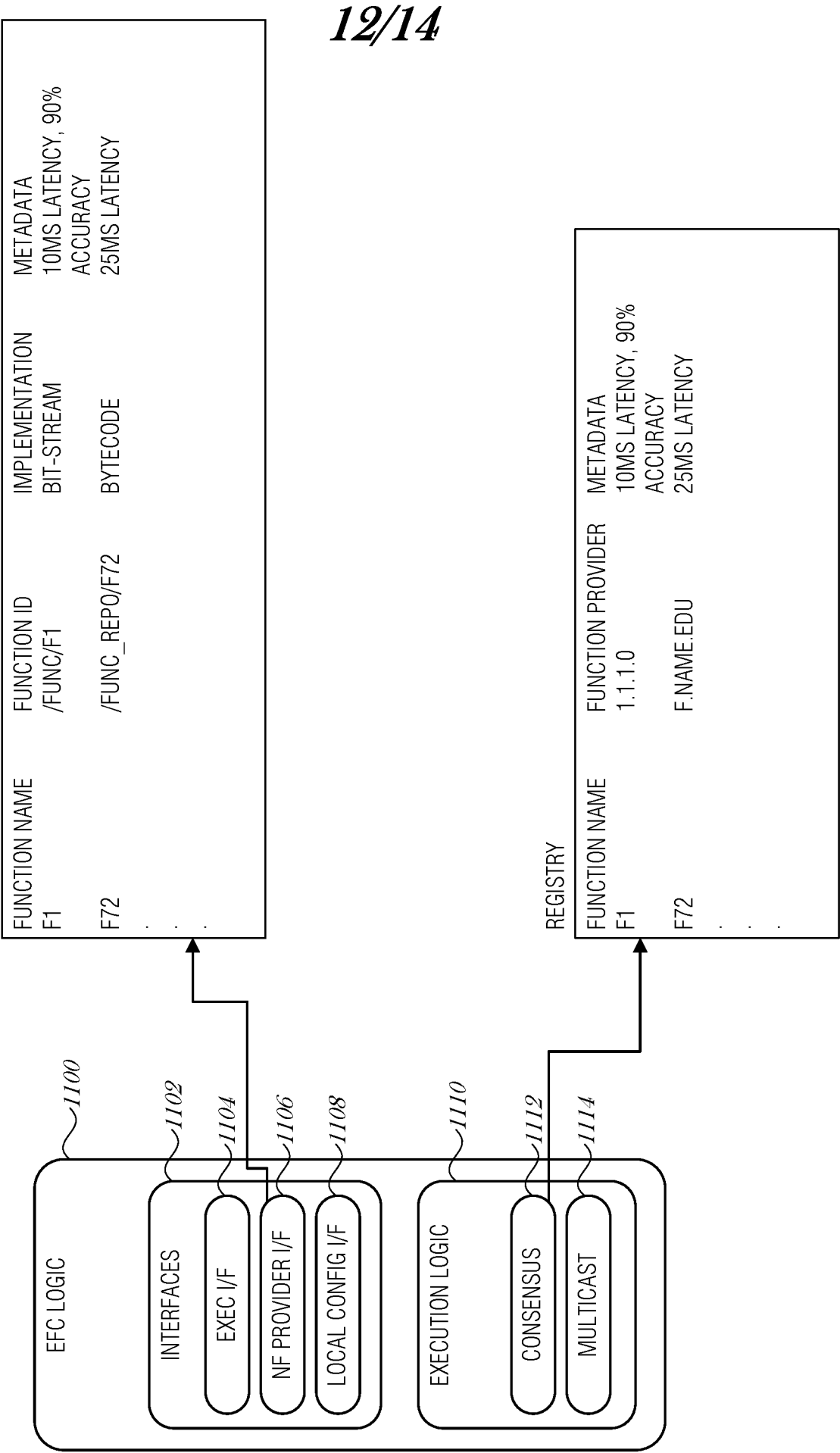


FIG. 11

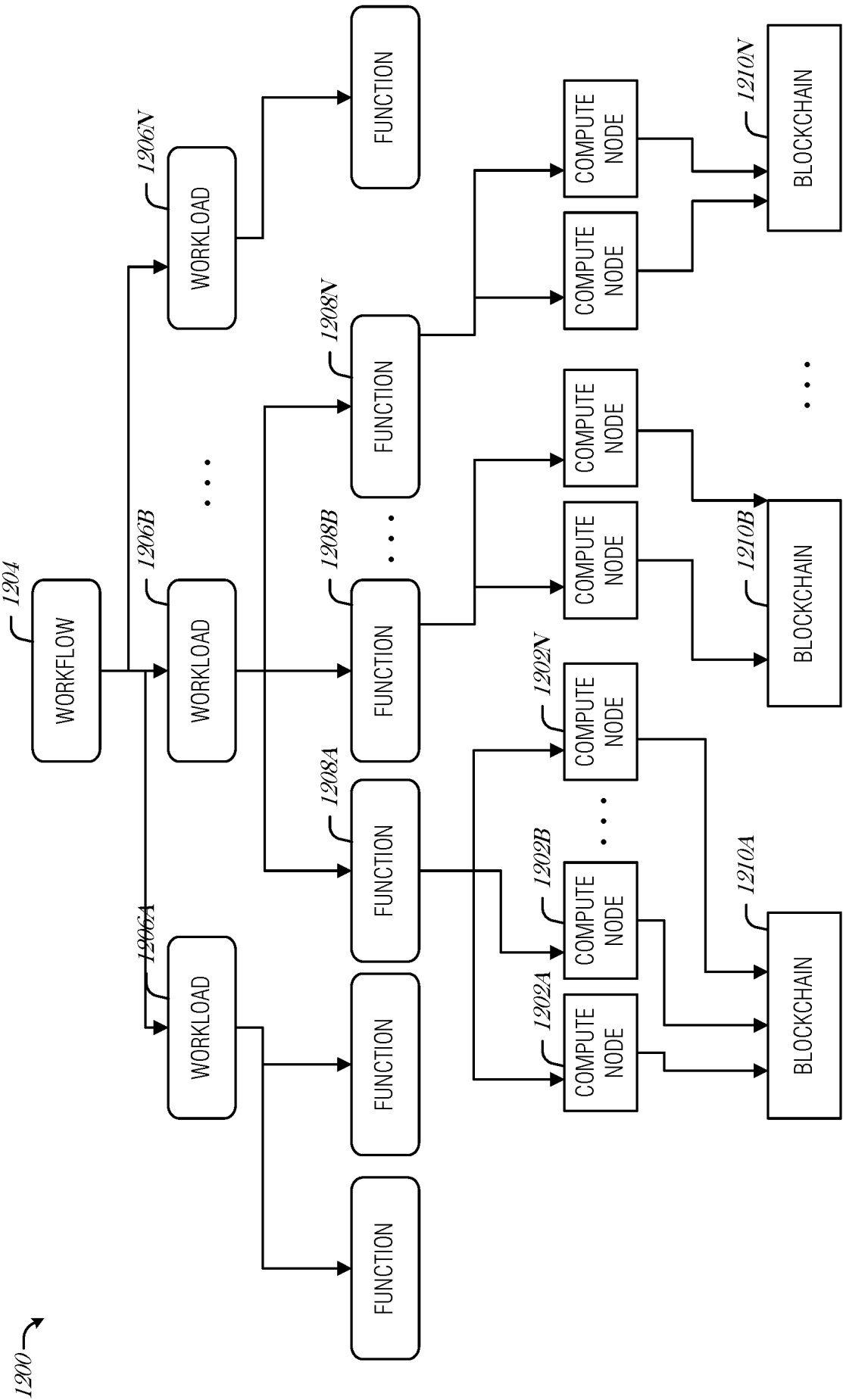
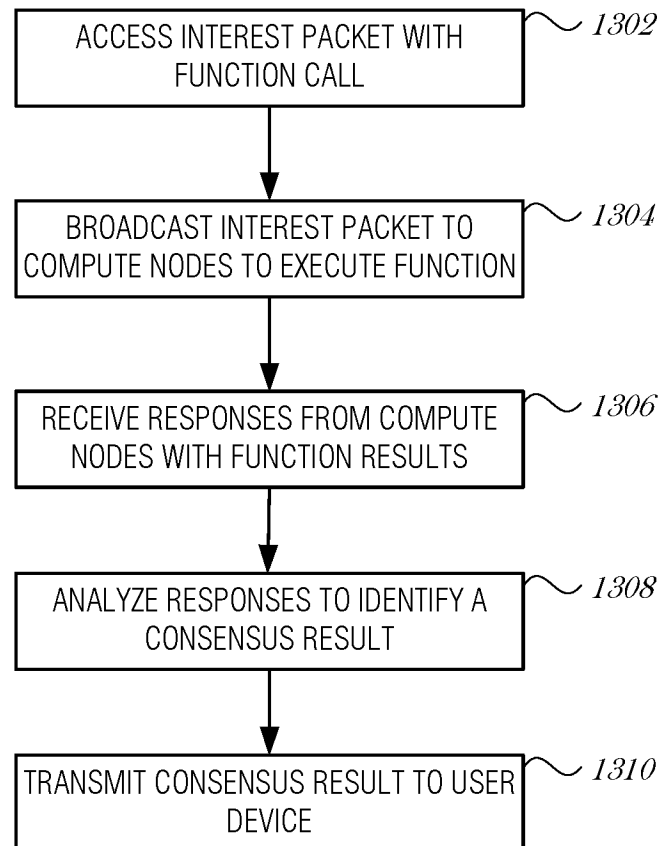


FIG. 12

14/14*1300***FIG. 13**

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US2023/084872

A. CLASSIFICATION OF SUBJECT MATTER H04L 67/12(2022.01)i; H04L 12/18(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) H04L 67/12(2022.01); G06F 16/182(2019.01); G06F 16/27(2019.01); H04L 12/18(2006.01); H04L 29/06(2006.01); H04L 45/745(2022.01); H04L 67/1087(2022.01); H04L 67/63(2022.01) 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: interest packet, function name, data payload, instance, consensus protocol		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2023-0014064 A1 (SUNIL CHERUVU et al.) 19 January 2023 (2023-01-19) paragraphs [0118]-[0127] and claim 1	1-24
A	KR 10-2022-0074971 A (TENCENT TECHNOLOGY (SHENZHEN) COMPANY LIMITED) 03 June 2022 (2022-06-03) paragraphs [0014]-[0015]	1-24
A	US 2021-0021609 A1 (INTEL CORPORATION) 21 January 2021 (2021-01-21) paragraphs [0027]-[0100]	1-24
A	US 2023-0135938 A1 (MARCOS E. CARRANZA et al.) 04 May 2023 (2023-05-04) paragraphs [0022]-[0130]	1-24
PX	US 2023-0319141 A1 (KSHITIJ ARUN DOSHI et al.) 05 October 2023 (2023-10-05) claims 1-20 * The above document is a publication of the earlier application whose priority has been claimed in this international application	1-24
☐ 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 30 April 2024		Date of mailing of the international search report 30 April 2024
Name and mailing address of the ISA/KR Korean Intellectual Property Office 189 Cheongsa-ro, Seo-gu, Daejeon 35208, Republic of Korea Facsimile No. +82-42-481-8578		Authorized officer KIM, Sung Hee Telephone No. +82-42-481-3516

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/US2023/084872

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)		Publication date (day/month/year)
US	2023-0014064	A1	19 January 2023	None		
KR	10-2022-0074971	A	03 June 2022	CN	111368008 A	03 July 2020
				CN	111368008 B	25 August 2020
				EP	4030314 A1	20 July 2022
				JP	2021-189426 A	13 December 2021
				JP	2023-508210 A	01 March 2023
				JP	7072629 B2	20 May 2022
				JP	7325894 B2	15 August 2023
				US	2022-0253432 A1	11 August 2022
				WO	2021-237781 A1	02 December 2021
				WO	2021-238514 A1	02 December 2021
US	2021-0021609	A1	21 January 2021	US	2021-0152563 A1	20 May 2021
US	2023-0135938	A1	04 May 2023	US	2023-0132992 A1	04 May 2023
				US	2023-0133020 A1	04 May 2023
				US	2023-0134683 A1	04 May 2023
				US	2023-0135645 A1	04 May 2023
				US	2023-0136048 A1	04 May 2023
				US	2023-0136612 A1	04 May 2023
				US	2023-0136615 A1	04 May 2023
				US	2023-0137879 A1	04 May 2023
				US	2023-0140252 A1	04 May 2023
US	2023-0319141	A1	05 October 2023	None		