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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 August 2020 (20.08.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/164950 A1

(51) International Patent Classification:
G06F 9/54 (2006.01)

(21) International Application Number:
PCT/EP2020/052660

(22) International Filing Date:
04 February 2020 (04.02.2020)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
19157085.2 14 February 2019 (14.02.2019) EP

(71) Applicant: **SIGNIFY HOLDING B.V.** [NL/NL]; High Tech Campus 48, 5656 AE Eindhoven (NL).

(72) Inventor: **SLEGERS, Walter, Jeroen**; c/o High Tech Campus 7, 5656 AE Eindhoven (NL).

(74) Agent: **VAN DE LAARSCHOT, Huon Urbald Ogier Norbert** et al.; High Tech Campus 7, 5656 AE Eindhoven (NL).

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

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

Declarations under Rule 4.17:
— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

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

(54) **Title:** DISTRIBUTING EVENTS INDICATIVE OF DEVICE STATE CHANGES

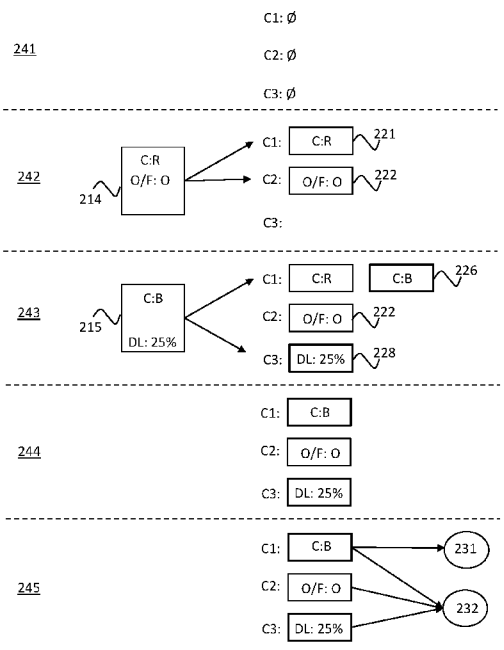


Fig. 5

(57) **Abstract:** A method comprises receiving an event subscription request from a further system, receiving a first signal (214) comprising an attribute and a first value of the attribute, creating one or more events (221,222) to be distributed to the further system based on the first signal, and storing the events in a queue to await distribution to the further system. The first value of the attribute at least partly represents a first state of the device. The method further comprises receiving a second signal (215) comprising the attribute and a second value of the attribute. The second value of the attribute at least partly represents a second state of the device, different from the first state. The method also comprises determining whether the queue comprises an event (221) indicative of a state of the device different from the second state with respect to the attribute and removing or amending this event.

WO 2020/164950 A1

Distributing events indicative of device state changes

FIELD OF THE INVENTION

The invention relates to a system for distributing events to a further system over a communication network, said further system being subscribed to receive said events.

The invention further relates to a method of distributing events to a further system over a communication network, said further system being subscribed to receive said events.

The invention also relates to a computer program product enabling a computer system to perform such a method.

BACKGROUND OF THE INVENTION

Certain connected lighting systems such as the Philips Hue system allow a user to control his local lighting devices via various clients, e.g. Amazon Echo, Google Home, and Apple TV. This control may be realized entirely locally or may be realized via the cloud. In order to detect changes to the state of the lighting devices, the clients can frequently ask (i.e. poll) the connected lighting system for the current state of the lighting devices and detect changes themselves. This frequent polling is not very efficient however.

In Internet systems, it is known to replace polling with event distribution. The distribution of events can be implemented using the publish-subscribe paradigm, which is used by Oracle's HTTP Publish-Subscribe Server and NCHAN, amongst others. With the publish-subscribe paradigm, messages are not sent directly to specific receivers, but are categorized into classes, e.g. channels, to which receivers can subscribe. A queue is required if subscribers want to receive messages that were published before they were connected and/or while they were temporarily not connected.

Distribution of events to subscribers requires the use of an event queue. An example of a system that uses an event queue is US 2016/0147579. US 2016/0147579 discloses a system for handling events in an industrial control system which generates events responsive to a predefined signal or combination of signals occurring and transfers the event to an event queue for subsequent execution. If a new event is generated and the queue is full, the oldest or the newest event in the queue may be dropped to make room for the new event.

A drawback of the event queuing disclosed in US 2016/0147579 is that it is not suitable for distributing events indicative of changes to a state of a (lighting) device. If a client is temporarily disconnected, e.g. due to an ISP outage or a reboot of the client, it wants to know the current state of the (lighting) devices of the connected system upon reconnecting. However, if the queue becomes full due to the temporary disconnection, certain events will be dropped and one or more events indicative of the current state of one or more of the devices may be among the dropped events.

SUMMARY OF THE INVENTION

It is a first object of the invention to provide a system, which is able to communicate a current state of a device to a client while limiting the amount of time that the client does not know the current state of the device.

It is a second object of the invention to provide a method, which can be used to communicate a current state of a device to a client while limiting the amount of time that the client does not know the current state of the device.

In a first aspect of the invention, a system for distributing events to a further system over a communication network, said further system being subscribed to receive said events, comprises at least one memory unit, at least one receiver, at least one transmitter, and at least one processor configured to receive an event subscription request from said further system, said event subscription request requesting said system to distribute events to said further system, said events being indicative of changes to a state of a device, use said at least one receiver to receive a first signal comprising an attribute and a first value of said attribute, said first value of said attribute at least partly representing a first state of said device, create, based on said received event subscription request and said received first signal, one or more events to be distributed to said further system, and store said one or more events in a queue in said at least one memory unit to await distribution to said further system over said communication network using said at least one transmitter.

Said at least one processor is further configured to use said at least one receiver to receive a second signal comprising said attribute and a second value of said attribute, said second value of said attribute at least partly representing a second state of said device, said second state being different from said first state and determine whether said queue comprises an event indicative of a state of said device different from said second state of said device with respect to said attribute and remove or amend said event.

By communicating a current state of a device to a client by distributing events indicative of changes to a state of the device, the communication is achieved more efficiently. By removing or amending events indicative of outdated states, the risk that that events indicative of the current state of one or more of the devices are dropped due to the event queue being full is limited. Some events occur very rarely (e.g. a light in the attic being switched off) and in these cases, it might take clients days to get back in sync. Even if the system is configured not to remove older events from the queue, rare events, like other events, may be dropped if the queue is full. In this case, the user will observe an incorrect state, which is annoying if he is at work or on holiday and sees an inconsistent state for lights or motion sensors in his house.

As additional advantage, certain undesirable behavior is avoided. For example, a user might walk through a hallway with a motion sensor multiple times while at the same time an ISP outage occurs, e.g. for 2 minutes, resulting in motion sensor events being queued. On reconnecting to the cloud home automation when the ISP outage has ended, the actions triggered by the sensor events might be executed in quick succession, which is not what a user would expect.

Said system may be a lighting control device, e.g. a bridge such as the Philips Hue bridge. Said device may be a lighting device. Said attribute may represent a color setting, a dim level setting or an on/off setting of a state of said lighting device, for example. Said system may a lighting system which comprises one or more lighting devices or may be part of such a lighting system.

Said at least one processor may be configured to include said attribute and said first value of said attribute in one of said one or more events upon creating said one or more events and determine that said event is indicative of a state of said device different from said second state with respect to said attribute if said event comprises said attribute and a different value than said second value for said attribute. Said event may be distributed in the form of a message. Said attribute and value may be included in a message that is then published on a publication-subscription server such as NCHAN.

Said at least one processor may be configured to include a second attribute and a value of said second attribute in said one of said one or more events. Such an event is also referred to as an aggregate event. Often, a change in the state of a device means a change in the values of multiple attributes, e.g. color and dim level (light output level) in a lamp. By including values of such attributes in a single event, clients interested in receiving events indicative of changes in these attributes may be able to obtain them in one message, thereby

optimizing efficiency. Attributes relating to the same device or relating to different devices may be aggregated. For example, a client interested in on/off events for all lights can get one message (on resubscription or when they all change simultaneously) instead of N messages for N lights.

Said at least one processor may be configured to remove said attribute and said different value from said event determined to be indicative of a state of said device different from said second state with respect to said attribute and create a further event comprising said attribute and said second value.

Said at least one processor may be configured to replace said different value with said second value in said event determined to be indicative of a state of said device different from said second state with respect to said attribute.

Said at least one processor may be configured to remove said event determined to be indicative of a state of said device different from said second state with respect to said attribute and create a further event comprising said attribute and said second value.

Said at least one processor may be configured to determine whether said queue comprises an event indicative of a state of said device different from said second state with respect to said attribute and remove or amend said event upon receiving said second signal.

Said at least one processor may be configured to determine all events stored in said queue indicative of a state of said device different from said second state of said device and remove or amend said events. This reduces the risk that events indicative of the current state of one or more of the devices are dropped due to the event queue being full more than if only a subset of all events is determined and removed or amended.

Said further system may be subscribed to receive changes relating to said attribute specifically. By allowing clients to subscribe to events relating to specific attributes of a specific device rather than to all attributes of this specific device, the bandwidth that is used for the event distribution may be reduced.

Said queue may be specific to said further system. A publication-subscription server makes it easy for multiple clients to subscribe to the same channel, which normally results in the channel queue being used for these multiple subscribers. However, it is also possible to create a channel per subscriber. This is especially beneficial if aggregate events are supported.

In a second aspect of the invention, a method of distributing events to a further system over a communication network, said further system being subscribed to receive said

events, comprises receiving an event subscription request from said further system, said event subscription request requesting said system to distribute events to said further system, said events being indicative of changes to a state of a device, receiving a first signal comprising an attribute and a first value of said attribute, said first value of said attribute at least partly representing a first state of said device, creating, based on said received event subscription request and said received first signal, one or more events to be distributed to said further system, and storing said one or more events in a queue in a memory to await distribution to said further system over said communication network.

The method further comprises receiving a second signal comprising said attribute and a second value of said attribute, said second value of said attribute at least partly representing a second state of said device, said second state being different from said first state and determining whether said queue comprises an event indicative of a state of said device different from said second state of said device with respect to said attribute and remove or amend said event. Said method may be performed by software running on a programmable device. This software may be provided as a computer program product.

Moreover, a computer program for carrying out the methods described herein, as well as a non-transitory computer readable storage-medium storing the computer program are provided. A computer program may, for example, be downloaded by or uploaded to an existing device or be stored upon manufacturing of these systems.

A non-transitory computer-readable storage medium stores at least one software code portion, the software code portion, when executed or processed by a computer, being configured to perform executable operations for distributing events to a further system over a communication network, said further system being subscribed to receive said events.

The executable operations comprise receiving an event subscription request from said further system, said event subscription request requesting said system to distribute events to said further system, said events being indicative of changes to a state of a device, receiving a first signal comprising an attribute and a first value of said attribute, said first value of said attribute at least partly representing a first state of said device, creating, based on said received event subscription request and said received first signal, one or more events to be distributed to said further system, and storing said one or more events in a queue in a memory to await distribution to said further system over said communication.

The executable operations further comprise receiving a second signal comprising said attribute and a second value of said attribute, said second value of said attribute at least partly representing a second state of said device, said second state being

different from said first state and determining whether said queue comprises an event indicative of a state of said device different from said second state of said device with respect to said attribute and remove or amend said event.

As will be appreciated by one skilled in the art, aspects of the present invention may be embodied as a device, a method or a computer program product. Accordingly, aspects of the present invention may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit", "module" or "system." Functions described in this disclosure may be implemented as an algorithm executed by a processor/microprocessor of a computer. Furthermore, aspects of the present invention may take the form of a computer program product embodied in one or more computer readable medium(s) having computer readable program code embodied, e.g., stored, thereon.

Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples of a computer readable storage medium may include, but are not limited to, the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of the present invention, a computer readable storage medium may be any tangible medium that can contain, or store, a program for use by or in connection with an instruction execution system, apparatus, or device.

A computer readable signal medium may include a propagated data signal with computer readable program code embodied therein, for example, in baseband or as part of a carrier wave. Such a propagated signal may take any of a variety of forms, including, but not limited to, electro-magnetic, optical, or any suitable combination thereof. A computer readable signal medium may be any computer readable medium that is not a computer readable storage medium and that can communicate, propagate, or transport a program for use by or in connection with an instruction execution system, apparatus, or device.

Program code embodied on a computer readable medium may be transmitted using any appropriate medium, including but not limited to wireless, wireline, optical fiber, cable, RF, etc., or any suitable combination of the foregoing. Computer program code for carrying out operations for aspects of the present invention may be written in any combination of one or more programming languages, including an object oriented programming language such as Java(TM), Smalltalk, C++ or the like and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer, or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

Aspects of the present invention are described below with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems), and computer program products according to embodiments of the present invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a processor, in particular a microprocessor or a central processing unit (CPU), of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer, other programmable data processing apparatus, or other devices create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

These computer program instructions may also be stored in a computer readable medium that can direct a computer, other programmable data processing apparatus, or other devices to function in a particular manner, such that the instructions stored in the computer readable medium produce an article of manufacture including instructions which implement the function/act specified in the flowchart and/or block diagram block or blocks.

The computer program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to

produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

The flowchart and block diagrams in the figures illustrate the architecture, functionality, and operation of possible implementations of devices, methods and computer program products according to various embodiments of the present invention. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of code, which comprises one or more executable instructions for implementing the specified logical function(s). It should also be noted that, in some alternative implementations, the functions noted in the blocks may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustrations, and combinations of blocks in the block diagrams and/or flowchart illustrations, can be implemented by special purpose hardware-based systems that perform the specified functions or acts, or combinations of special purpose hardware and computer instructions.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other aspects of the invention are apparent from and will be further elucidated, by way of example, with reference to the drawings, in which:

Fig. 1 is a block diagram of an embodiment of the system;

Fig. 2 is a flow diagram of a first embodiment of the method;

Fig. 3 is a flow diagram of a second embodiment of the method;

Fig. 4 depicts an example of the operation of a first implementation of the method of Fig.3;

Fig. 5 depicts an example of the operation of a second implementation of the method of Fig. 3;

Fig. 6 depicts an example of the operation of a third implementation of the method of Fig. 3;

Fig. 7 depicts an example of the operation of a fourth implementation of the method of Fig. 3;

Fig. 8 is a flow diagram of a second embodiment of the method;

Fig. 9 depicts an example of the operation of a first implementation of the method of Fig. 8;

Fig. 10 depicts an example of the operation of a second implementation of the method of Fig.8;

Fig. 11 depicts an example of the operation of a third implementation of the method of Fig. 8;

Fig. 12 shows an example of a software architecture for the embodiment of the system of Fig. 1; and

Fig. 13 is a block diagram of an exemplary data processing system for performing the method of the invention.

Corresponding elements in the drawings are denoted by the same reference numeral.

DETAILED DESCRIPTION OF THE EMBODIMENTS

Fig. 1 shows an embodiment of the system for distributing events to a further system over a communication network: a bridge **1**. The bridge **1** distributes events to Internet servers **25** and **26** via an intermediary Internet server **23** and to a local user device **19**. Internet servers **25** and **26** and local user device **19** run automation clients which are subscribed to receive these events. The Internet servers **25** and **26** provide cloud services, e.g. the Google Home cloud service and the Alexa cloud service. The cloud services typically communicate with a further local user device (not shown) in the same spatial area as the bridge **1**, e.g. a Google Home device or Amazon Echo device. The local user device **19** may be an Apple TV device, for example.

The Internet servers **25** and **26** and the intermediary Internet server **23** are connected to the Internet (backbone) **21**. A lighting system **11** comprises the bridge **1**, lighting devices **13** and **14** and a presence sensor **15**. The bridge **1** is connected to the wireless LAN access point **18**, e.g. via Ethernet. The bridge **1** communicates with the lighting devices **13** and **14** and presence sensor **15**, e.g. using Zigbee technology. The lighting devices **13** and **14** may be Philips Hue lights, for example. The presence sensor **15** may be the Hue motion sensor, for example. The local user device **19** is also connected to the wireless LAN access point **18**, e.g. via Wi-Fi (IEEE 802.11).

The bridge **1** comprises a receiver **3**, a transmitter **4,** a processor **5**, and a memory **7**. The processor **5** is configured to receive an event subscription request from one or more of the Internet servers **25** and **26** and/or the local user device **19**. The event subscription request requests the bridge **1** to distribute events to the requestor. The events are indicative of

changes to a state of one or more of the connected devices, i.e. lighting devices **13** and **14** and the presence sensor **15**.

The processor **5** is further configured to use the receiver **3** to receive a first signal comprising an attribute and a first value of the attribute. This first value of the attribute at least partly represents a first state of one of the connected devices. The first signal may be received from the connected device whose status is represented in the first signal or from another device (not shown) controlling this connected device, e.g. a mobile phone or tablet.

The processor **5** is further configured to create, based on the received event subscription request and the received first signal, one or more events to be distributed to the subscribed system(s) and store the one or more events in a queue in the memory **7** to await distribution to the subscribed system(s) over the communication network using the transmitter **4**.

The processor **5** is also configured to use the receiver **3** to receive a second signal comprising the attribute and a second value of the attribute. The second value of the attribute at least partly represents a second state of the device whose status was represented in the first signal. The second state is different from the first state. The processor **5** is also configured to determine whether the queue comprises an event indicative of a state of this device which is different from the second state of this device with respect to the attribute and remove or amend this event.

In the embodiment of the bridge **1** shown in **Fig. 1**, the bridge **1** comprises one processor **5**. In an alternative embodiment, the bridge **1** comprises multiple processors. The processor **5** of the bridge **1** may be a general-purpose processor, e.g. ARM-based, or an application-specific processor. The processor **5** of the bridge **1** may run a Unix-based operating system for example. The memory **7** may comprise one or more memory units. The memory **7** may comprise one or more hard disks and/or solid-state memory, for example. The memory **7** may be used to store a table of connected lights, for example.

The receiver **3** and the transmitter **4** may use one or more wired or wireless communication technologies such as Ethernet to communicate with the wireless LAN access point **18**, for example. In an alternative embodiment, multiple receivers and/or multiple transmitters are used instead of a single receiver and a single transmitter. In the embodiment shown in **Fig. 1**, a separate receiver and a separate transmitter are used. In an alternative embodiment, the receiver **3** and the transmitter **4** are combined into a transceiver. The bridge **1** may comprise other components typical for a network device such as a power connector.

The invention may be implemented using a computer program running on one or more processors.

In the embodiment of **Fig. 1**, the system for distributing events to a further system over a communication network is a bridge. In an alternative embodiment, the system may be a different type of system, e.g. a lighting device or accessory or an Internet server. For example, the invention may be used in the Internet servers **25** and **26**. In the embodiment of **Fig. 1**, the system for distributing events to a further system over a communication network consists of only one device. In an alternative embodiment, the system comprises multiple devices.

A first embodiment of the method of distributing events to a further system over a communication network is shown in **Fig. 2**. The further system is subscribed to receive the events. A step **101** comprises receiving an event subscription request from the further system. The event subscription request requests the system to distribute events to the further system. The events are indicative of changes to a state of a device. A step **103** comprises receiving a first signal comprising an attribute and a first value of the attribute. The first value of the attribute at least partly represents a first state of the device.

A step **105** comprises creating, based on the received event subscription request and the received first signal, one or more events to be distributed to the further system. A step **107** comprises storing the one or more events in a queue in a memory to await distribution to the further system over the communication network.

A step **109** comprises receiving a second signal comprising the attribute and a second value of the attribute. The second value of the attribute at least partly represents a second state of the device, the second state being different from the first state. Step **111** is performed upon receiving the signal in step **109**. Step **111** comprises determining whether the queue comprises an event indicative of a state of the device different from the second state of the device with respect to the attribute. This event is removed event in step **113** or amended in step **115**.

A second embodiment of the method of distributing events to a further system over a communication network is shown in **Fig. 3**. Step **101** comprises receiving an event subscription request from the further system. The event subscription request requests the system to distribute events to the further system. The events are indicative of changes to a state of a device. A step **121** comprises receiving a signal comprising an attribute and a value of the attribute. The value of the attribute at least partly represents a current state of the device.

Step **105** comprises creating, based on the received event subscription request and the received signal, one or more events to be distributed to the further system. Step **107** comprises storing the one or more events in a queue in a memory to await distribution to the further system over the communication network. In the embodiment of **Fig. 3**, the attribute and the value of the attribute are included in one of the one or more events in step **105**. The attribute may identify the device, e.g. an attribute "/lights/4/color" identifies light 4 as the device whose state is (partly) described in the value corresponding to this attribute.

If the received signal comprises a second attribute and a value of the second attribute, this second attribute and this value of the second attribute may be included in the same event as the first attribute and the value of the first attribute, thereby creating an aggregate event. This is useful when both attributes are published on the same channel. All attributes included in the received signal may be included in the same event, e.g. if they are all published on the same channel, so that only one event needs to be created. Alternatively, multiple events may be created if the received signal comprises multiple attributes, e.g. one event per attribute.

Step **122** comprises looking in the queue for an event indicative of a state of the device different from the state of the device represented by the received signal with respect to the attribute(s). In the embodiment of **Fig. 3**, an event is considered to be indicative of a state of the device different from the state represented by the received signal if the event comprises an attribute included in the received signal and a different value for this attribute than the value included in the received signal.

In the embodiment of **Fig. 3**, step **122** further comprises looking in the queue for an event indicative of a state of the device identical to the state of the device represented by the received signal with respect to at least one of the attribute(s). This is done to be able to remove duplicate states, because new events are created and stored in steps **105** and **107** before step **122** is performed.

If it is determined in step **123** that the queue comprises no such event, then step **121** is repeated to receive the next signal at a later time. If it is determined in step **123** that the event found in step **122** comprises all attributes included in the signal received in step **121**, e.g. if the found event is not an aggregate event, then the found event is removed in step **113**.

If is determined in step **123** that the event found in step **122** is an aggregate event, i.e. an event comprising multiple attributes and corresponding values, and the found event comprises one or more attributes not included in the signal received in step **121**, then

the one or more attributes included in the signal are removed from the found event in step **125**, thereby amending the found event. The values of these attributes are removed from the found event as well.

Step **122** is repeated after steps **113** and **115** and comprises looking in the queue for another event indicative of a state of the device different from the state of the device represented by the last received signal with respect to the attribute(s). Steps **122**, **123**, **113** and **125** are repeated until all events comprising at least one of the attribute(s) received in the signal have been found in the queue.

Figs. 4-7 depict examples of the operation of different implementations of the method of **Fig. 3**. In these examples, events for 3 attributes (on-off, dim level, color) are published/distributed. Interested clients can subscribe to individual attributes and receive published events relating to these attributes. If a connection is temporarily lost, events are transmitted to a subscriber upon reconnection. This requires the use of a queue.

Fig. 4 depicts an example of the operation of a first implementation of the method of **Fig. 3**. In the example of **Fig. 4**, three channels used to distribute events: channels C1, C2 and C3. Each channel has a queue. Events relating to the attribute "color" are published on channel C1, events relating to the attribute "on/off" are published on channel C2 and events relating to the attribute "dim level" are published on channel C3. At moment **201**, all queues are empty.

At moment **202**, a signal **211** comprising all 3 attributes of light 4 is received. The signal **211** comprises a color attribute with the value "Red", an on/off attribute with the value "On" and a dim level attribute with the value "25%". The signal **211** may comprise ("/lights/4": {"color":"red", "on-ff":"on", "dimlevel":"25") or ("/lights/4/color:red", "/lights/4/on-off:on", "/lights/4/dimlevel:25"), for example. Three events **221-223** are created based on this signal, one for each attribute. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **203**, a signal **212** comprising all 3 attributes is received. The signal **212** comprises a color attribute with the value "Blue", an on/off attribute with the value "On" and a dim level attribute with the value "25%". Again, three events **226-228** are created based on this signal, one for each attribute.

After the signal **212** has been received at moment **203**, step **122** of **Fig. 3** is performed. For the color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and event **221** is therefore removed. For the on-off

attribute, an event **222** is found in the queue of channel C2 which represents a duplicate state of light 4 and is therefore removed. For the dim level attribute, an event **223** is found in the queue of channel C3 which represents a duplicate state of light 4 and is therefore removed.

At moment **204**, the outdated event and duplicate events have been removed and only events **226-228** are left. At moment **205**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **226-228** are transmitted to the subscribers **231** and **232**.

Fig. 5 depicts an example of the operation of a second implementation of the method of **Fig. 3**. In the example of **Fig. 5**, signals are received that do not comprise all attributes of a device. In the example of **Fig. 5**, again three channels used to distribute events: channels C1, C2 and C3. Each channel has a queue. At moment **241**, all queues are empty.

At moment **242**, a signal **214** comprising 2 attributes of light 4 is received. The signal **214** comprises a color attribute with the value "Red" and an on/off attribute with the value "On". Two events **221** and **222** are created based on this signal, one for each attribute. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **243**, a signal **215** comprising 2 attributes of light 4 is received. The signal **215** comprises a color attribute with the value "Blue" and a dim level attribute with the value "25%". Two events **226** and **228** are created based on this signal, one for each attribute.

After the signal **215** has been received at moment **243**, step **122** of **Fig. 3** is performed. For the color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and event **221** is therefore removed. For the dim level attribute, no outdated or duplicate event is found is found in the queue of channel C3.

At moment **244**, the outdated event has been removed and events **226, 222** and **228** are left. At moment **245**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **226, 222** and **228** are transmitted to the subscribers **231** and **232**.

Fig. 6 depicts an example of the operation of a third implementation of the method of **Fig. 3**. In the example of **Fig. 6**, like in the example of **Fig. 4**, signals are received that comprise all attributes of a device, but different than in the example of **Fig. 4**, aggregate events are used. In the example of **Fig. 6**, two channels are used to distribute events: channels C1 and C2. Channel C2 is used to distribute aggregate events. Each channel has a queue. At moment **251**, all queues are empty.

At moment **252**, a signal **211** comprising all 3 attributes of light 4 is received. The signal **211** comprises a color attribute with the value "Red", an on/off attribute with the value "On" and a dim level attribute with the value "25%". Two events **221** and **222** are

created based on this signal. Event **221** is created for the color attribute and event **222** is created for the aggregate of all three attributes. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **253**, a signal **212** comprising all 3 attributes is received. The signal **212** comprises a color attribute with the value "Blue", an on/off attribute with the value "On" and a dim level attribute with the value "25%". Two events **226** and **228** are created based on this signal. Event **226** is created for the color attribute and event **236** is created for the aggregate of all three attributes.

After the signal **212** has been received at moment **253**, step **122** of **Fig. 3** is performed. For the color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and event **221** is therefore removed. Furthermore, for the color attribute, an event **216** is found in the queue of channel C2 which represents an outdated state of light 4. Since event **236** comprises all attributes of event **216**, event **216** is removed.

At moment **254**, the outdated events have been removed and events **226** and **236** are left. At moment **255**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **226** and **236** are transmitted to the subscribers **231** and **232**. In the example of **Fig. 6**, different channels may be created for different subscribers or channels may be used by multiple subscribers. For example, if a subscriber **233** subscribes to events relating to all three attributes of light 4, then it may receive events on channel C2 (shown in **Fig. 6**) or a third channel C3 may be created (not shown)

Fig. 7 depicts an example of the operation of a fourth implementation of the method of **Fig. 3**. In the example of **Fig. 7**, like in the example of **Fig. 5**, signals are received that do not comprise all attributes of a device, but different than in the example of **Fig. 5**, aggregate events are used. In the example of **Fig. 5**, two channels are used to distribute events: channels C1 and C2. Channel C2 is used to distribute aggregate events. Each channel has a queue. At moment **261**, all queues are empty.

At moment **262**, a signal **214** comprising 2 attributes of light 4 is received. The signal **214** comprises a color attribute with the value "Red" and an on/off attribute with the value "On". Two events **221** and **217** are created based on this signal. Event **221** is created for the color attribute and event **217** is created for the aggregate of both received attributes. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **263**, a signal **215** comprising 2 attributes of light 4 is received. The signal **215** comprises a color attribute with the value "Blue" and a dim level attribute with the value "25%". Two events **226** and **237** are created based on this signal. Event **226** is created for the color attribute and event **237** is created for the aggregate of both received attributes.

After the signal **215** has been received at moment **263**, step **122** of **Fig. 3** is performed. For the received color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and event **221** is therefore removed. Furthermore, for the color attribute, an event **217** is found in the queue of channel C2 which represents an outdated state of light 4. Since the event **217** also comprises an attribute (attribute on/off) that is not included in the received signal **215**, event **217** is not removed, but amended. The attribute color and the corresponding value "Red" are removed from the event **217**, thereby resulting in an amended event **247**.

At moment **264**, the outdated events have been removed or amended and events **226, 247** and **237** are left. At moment **265**, the bridge **1** reconnects to the intermediary Internet server **23** and the event **226** is transmitted to subscriber **231** and the events **247** and **237** are transmitted to the subscriber **232**.

A third embodiment of the method of distributing events to a further system over a communication network is shown in **Fig. 8**. Step **101** comprises receiving an event subscription request from the further system. The event subscription request requests the system to distribute events to the further system. The events are indicative of changes to a state of a device. Step **121** comprises receiving a signal comprising one or more attributes and corresponding values. The values of these one or more attributes at least partly represent a current state of a device.

Step **130** is performed after step **121**. Step **130** comprises looking in a queue for an event comprising at least one of the attribute(s) received in the signal. Step **131** is performed if it is determined in step **123** that an event indicative of a state of the device different from the state of the device represented by the received signal with respect to the attribute(s) has been found. In the embodiment of **Fig. 8**, an event is considered to be indicative of a state of the device different from the state represented by the last received signal if the event comprises an attribute included in the last received signal and a different value for this attribute than the value included in the last received signal.

Step **131** comprising replacing the one or more values of the one or more attributes that are both included in the received signal and the found event. The one or more values included in the found event that correspond to these one or more attributes are

replaced with the corresponding one or more values included in the received signal, thereby amending the event. Step **130** is repeated after step **131** and comprises looking in the queue for another event comprising at least one of the attribute(s) received in the signal.

Step **130** is also repeated if it is determined in step **123** that an event comprising at least one of the attribute(s) received in the signal exists in the queue, but the event is not indicative of a state of the device different from the state of the device represented by the received signal with respect to the attribute(s).

Steps **130**,**123**, and **131** are repeated until all events comprising at least one of the attribute(s) received in the signal have been found in the queue. If it is determined in step **123** that the queue comprises no such event, then it is checked in step **133** whether all values included in the received signal were found in at least one event in step **130** or included in at least one event in step **131**. If so, step **121** is repeated after step **133**. If not, steps **105** and **107** of **Fig. 3** are performed next.

Step **105** comprises creating, based on the received event subscription request and the received signal, one or more events to be distributed to the further system. The attributes not found in any event in step **130** and not yet included in any event in step **131**, plus the corresponding values, are included in these one or more events. Step **107** comprises storing the one or more events in the queue. Step **121** is repeated after step **107** to receive the next signal at a later time.

As an alternative or addition to steps **105** and **107**, one or more of the attributes not found in any event in step **130** and not yet included in any event in step **131**, plus the corresponding values, may be added to an existing event (not shown in **Fig. 8**).

Figs. 9-11 depict examples of the operation of different implementations of the method of **Fig. 8**. In these examples, events for 3 attributes (on-off, dim level, color) are published/distributed. Interested clients can subscribe to individual attributes and receive published events relating to these attributes. If a connection is temporarily lost, events are transmitted to a subscriber upon reconnection. This requires the use of a queue.

Fig. 9 depicts an example of the operation of a first implementation of the method of **Fig. 8**. In the example of **Fig. 9**, three channels are used to distribute events: channels C1, C2 and C3. Each channel has a queue. Events relating to the attribute “color” are published on channel C1, events relating to the attribute “on/off” are published on channel C2 and events relating to the attribute “dim level” are published on channel C3. At moment **271**, all queues are empty.

At moment **272**, a signal **211** comprising all 3 attributes of light 4 is received. The signal **211** comprises a color attribute with the value "Red", an on/off attribute with the value "On" and a dim level attribute with the value "25%". Three events **221-223** are created based on this signal, one for each attribute. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **273**, a signal **212** comprising all 3 attributes is received. The signal **212** comprises a color attribute with the value "Blue", an on/off attribute with the value "On" and a dim level attribute with the value "25%". After the signal **212** has been received, step **130** of **Fig. 8** is performed. For the color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and the value "Red" is therefore replaced with the value "Blue" in the event **221**, resulting in an amended event **276**.

For the on-off attribute, an event **222** is found in the queue of channel C1 which comprises the same value for the on-off attribute as received in signal **212** (i.e. "on") and therefore no new event is created for this attribute. For the dim level attribute, an event **222** is found in the queue of channel C2 which comprises the same value for the dim level attribute as received in signal **212** (i.e. "25%") and therefore no new event is created for this attribute.

At moment **274**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **276, 222** and **223** are transmitted to the subscribers **231** and **232**.

Fig. 10 depicts an example of the operation of a second implementation of the method of **Fig. 8**. In the example of **Fig. 10**, signals are received that do not comprise all attributes of a device and aggregate events are used. In the example of **Fig. 10**, two channels are used to distribute events: channels C1 and C2. Channel C2 is used to distribute aggregate events. Each channel has a queue. At moment **281**, all queues are empty.

At moment **282**, a signal **214** comprising 2 attributes of light 4 is received. The signal **214** comprises a color attribute with the value "Red" and an on/off attribute with the value "On". Two events **221** and **218** are created based on this signal. Event **221** is created for the color attribute and event **218** is created for the aggregate of both received attributes. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **283**, a signal **215** comprising 2 attributes of light 4 is received. The signal **215** comprises a color attribute with the value "Blue" and a dim level attribute with the value "25%". For the color attribute, an event **221** is found in the queue of channel C1 which

represents an outdated state of light 4 and the value "Red" is therefore replaced with the value "Blue" in the event **221**, resulting in an amended event **276**.

Furthermore, for the color attribute, an event **218** is found in the queue of channel C2 which represents an outdated state of light 4 and the value "Red" is therefore replaced with the value "Blue" in the event **218**, resulting in an amended event **238**. For the dim level attribute, no event is found which comprises the dim level attribute and the dim level attribute and the corresponding value from signal **215** ("25%") are therefore added to amended event **238**.

At moment **284**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **276** and **238** are transmitted to the subscribers **231** and **232**. In the example of **Fig. 10**, different channels may be created for different subscribers or channels may be used by multiple subscribers. For example, if a subscriber **233** subscribes to events relating to all three attributes of light 4, then it may receive events on channel C2 (shown in **Fig. 10**) or a third channel C3 may be created (not shown)

Fig. 11 depicts an example of the operation of a third implementation of the method of **Fig. 8**. In the example of **Fig. 11**, signals are received that do not comprise all attributes of a device and aggregate events are used, like in the example of **Fig. 10**. In the example of **Fig. 11**, two channels are used to distribute events: channels C1 and C2. Channel C2 is used to distribute aggregate events. Each channel has a queue. At moment **291**, all queues are empty.

At moment **292**, a signal **214** comprising 2 attributes of light 4 is received. The signal **214** comprises a color attribute with the value "Red" and an on/off attribute with the value "On". Two events **221** and **218** are created based on this signal. Event **221** is created for the color attribute and event **218** is created for the aggregate of both received attributes. These events would normally be distributed to all current subscribers, but due to a temporary disconnection, they cannot be distributed to subscribers **231** and **232**.

At moment **293**, a signal **215** comprising 2 attributes of light 4 is received. The signal **215** comprises a color attribute with the value "Blue" and a dim level attribute with the value "25%". For the color attribute, an event **221** is found in the queue of channel C1 which represents an outdated state of light 4 and the value "Red" is therefore replaced with the value "Blue" in the event **221**, resulting in an amended event **276**.

Furthermore, for the color attribute, an event **218** is also found in the queue of channel C2 which represents an outdated state of light 4 and the value "Red" is therefore replaced with the value "Blue" in the event **218**, resulting in an amended event **219**. For the

dim level attribute, no event is found which comprises the dim level attribute and a new event **239** comprising the dim level attribute and the corresponding value from signal **215** ("25%") is therefore created.

At moment **294**, the bridge **1** reconnects to the intermediary Internet server **23** and the events **276, 219** and **239** are transmitted to the subscribers **231** and **232**.

Fig. 12 shows an example of a software architecture for the embodiment of the system of **Fig. 1**. In this example, at least four software components run on the bridge **1**: a bridge application **41**, an eventing daemon **44**, a Nginx web server **46** and a web-socket client daemon **49**. The bridge application **41** controls and monitors lights, sensors, and switches and receives signals comprising one or more attributes and their corresponding values, e.g. from the devices being monitored or from a device that transmits commands destined for the devices being controlled. In the example of **Fig. 12**, the bridge application **41** receives signals from the lighting devices **13** and **14** and from the presence sensor **16** using the Zigbee protocol. The bridge application **41** comprises a Google Remote Procedure Call (GRPC) server **42** which provides the received attributes and corresponding values to the eventing daemon **44**.

The eventing daemon **44** creates events from the received attributes and corresponding values as NCHAN formatted messages. If the eventing daemon **44** receives values of the same attribute at too high a frequency, it may decide to skip creating events from some of its values, thereby reducing used network bandwidth. The eventing daemon **44** may create a separate event for each received value or it may be able to combine multiple values in a single event, thereby creating aggregate events. In the latter case, the eventing daemon **44** determines which attributes may be included in the same event based on stored channel information.

An attribute and its value may be included in multiple events. For example, a first subscriber may be subscribed to receive only events relating to the color of a lamp, while a second subscriber may be subscribed to receive all events relating to the state of this lamp. The eventing daemon **44** determines on which one or more channels the created NCHAN formatted messages should be published based on the stored channel information and publishes the NCHAN formatted messages on these one or more determined channels on Nginx web server **46**. In the software architecture of **Fig. 12**, no event is created for a received attribute if no subscriber is subscribed to receive events related to this attribute.

Nginx web server **46** comprises a modified NCHAN module **47**. NCHAN is a buffered publication-subscription server and implements a standard SSE (Server-Sent

Events) interface for clients. The NCHAN module **47** comprises a queue per channel and registers per message which subscribers have received the message. In the software architecture of **Fig. 12**, an event to be distributed to a further system is considered to be created if the event has been created for publication on a channel to which the further system is subscribed.

Each queue may be associated with a maximum number of messages and a maximum time during which a message is stored in the queue. This maximum time may be configured to be longer than the maximum duration of a disconnection to prevent that events are dropped as a result of the disconnection. Alternatively, the NCHAN module **47** may be adapted not to drop messages that could not be delivered to all subscribers that were subscribed at the time of publication of the message. The NCHAN module has been adapted to remove all but the latest event/value per attribute from the queue.

The web-socket client daemon **49** uses the SSE interface to communicate with subscriber endpoints on the Nginx web server **46** and uses web-socket to communicate with the intermediary Internet server **23**. The web-socket client daemon **49** packs the NCHAN messages, i.e. the events, that it obtains from the Nginx web server **46** into a format to be used on a logical event channel on a web-socket connection to the intermediary Internet server **23**, e.g. running a Hue cloud service. This allows messages on multiple NCHAN channels to be multiplexed onto a single web-socket connection. A web-socket channel may be created for each NCHAN channel or for each combination of NCHAN channel and subscriber, and a mapping between web-socket channels and NCHAN channels may be stored in the memory of the bridge **1** and used by the web-socket client daemon **49**. A web-socket channel may be created when the NCHAN channel is created or when a subscriber transmits its subscription request, for example.

In the software architecture of **Fig. 12**, the web-socket client daemon **49** only maintains SSE connections with subscriber endpoints on the Nginx web server **46** as long as it is able to maintain its web-socket connection with the intermediary Internet server **23**. In an alternative embodiment, the web-socket client daemon **49** always maintains SSE connections with subscriber endpoints on the Nginx web server **46** and therefore needs to use one or more queues. In this alternative embodiment, it is the web-socket client daemon **49** has been adapted to remove all but the latest event/value per attribute from its queue(s).

In the software architecture of **Fig. 12**, the web-socket client daemon **49** also receives subscription requests from the Internet servers **25** and **26** via the intermediary Internet server **23**. These subscription requests identify one or more attributes. The web-

socket client daemon **49** then determines on which one or more NCHAN channels events relating to the specified attributes are published based on the channel information that is also used by the eventing daemon **44** to create and publish the NCHAN messages. The channel information is stored in the memory of the bridge **1**. Optionally, the web-socket client daemon **49** is able to create new NCHAN channels and include them in the above-mentioned channel information.

In the software architecture of **Fig. 12**, an automation client is able to subscribe to receive events related to specified attributes. Alternatively, an automation client may only be able to subscribe to receive events related to all attributes of specified devices.

In the example of **Fig. 12**, the Internet servers **25** and **26** receive the events to which they are subscribed from the intermediary Internet server **23**. The local user device **19** may also be subscribed to receive events from the bridge **1**. In this case, the local user device **19** may also open an SSE connection to one or more subscriber endpoints on the Nginx web server **46**. This is not shown in **Fig. 12**.

In the software architecture of **Fig. 12**, the queue that fills up when a connection is temporarily lost is located in the NCHAN module **47**. In an alternative embodiment, this queue is located in another software component, e.g. in the web-socket client daemon **49**.

In the example of **Fig. 12**, the software architecture comprises four separate software components. In an alternative embodiment, one or more of the software components may be combined into a single software component. For example, the eventing daemon **44** may be a module of the Nginx web server **46**. In the example of **Fig. 12**, an NCHAN module **47** is used as publication-subscription server. In an alternative embodiment, a message broker that implements the MQTT (machine-to-machine/Internet of Things) protocol may be used as publication-subscription server.

Fig. 13 depicts a block diagram illustrating an exemplary data processing system that may perform the method as described with reference to **Figs. 2,3** and **8**.

As shown in **Fig. 13,** the data processing system **300** may include at least one processor **302** coupled to memory elements **304** through a system bus **306**. As such, the data processing system may store program code within memory elements **304**. Further, the processor **302** may execute the program code accessed from the memory elements **304** via a system bus **306**. In one aspect, the data processing system may be implemented as a computer that is suitable for storing and/or executing program code. It should be appreciated, however, that the data processing system **300** may be implemented in the form of any system

including a processor and a memory that is capable of performing the functions described within this specification.

The memory elements **304** may include one or more physical memory devices such as, for example, local memory **308** and one or more bulk storage devices **310**. The local memory may refer to random access memory or other non-persistent memory device(s) generally used during actual execution of the program code. A bulk storage device may be implemented as a hard drive or other persistent data storage device. The processing system **300** may also include one or more cache memories (not shown) that provide temporary storage of at least some program code in order to reduce the quantity of times program code must be retrieved from the bulk storage device **310** during execution. The processing system **300** may also be able to use memory elements of another processing system, e.g. if the processing system **300** is part of a cloud-computing platform.

Input/output (I/O) devices depicted as an input device **312** and an output device **314** optionally can be coupled to the data processing system. Examples of input devices may include, but are not limited to, a keyboard, a pointing device such as a mouse, a microphone (e.g. for voice and/or speech recognition), or the like. Examples of output devices may include, but are not limited to, a monitor or a display, speakers, or the like. Input and/or output devices may be coupled to the data processing system either directly or through intervening I/O controllers.

In an embodiment, the input and the output devices may be implemented as a combined input/output device (illustrated in **Fig. 13** with a dashed line surrounding the input device **312** and the output device **314**). An example of such a combined device is a touch sensitive display, also sometimes referred to as a "touch screen display" or simply "touch screen". In such an embodiment, input to the device may be provided by a movement of a physical object, such as e.g. a stylus or a finger of a user, on or near the touch screen display.

A network adapter **316** may also be coupled to the data processing system to enable it to become coupled to other systems, computer systems, remote network devices, and/or remote storage devices through intervening private or public networks. The network adapter may comprise a data receiver for receiving data that is transmitted by said systems, devices and/or networks to the data processing system **300**, and a data transmitter for transmitting data from the data processing system **300** to said systems, devices and/or networks. Modems, cable modems, and Ethernet cards are examples of different types of network adapter that may be used with the data processing system **300**.

As pictured in **Fig. 13**, the memory elements **304** may store an application **318**. In various embodiments, the application **318** may be stored in the local memory **308, t**he one or more bulk storage devices **310**, or separate from the local memory and the bulk storage devices. It should be appreciated that the data processing system **300** may further execute an operating system (not shown in **Fig. 13**) that can facilitate execution of the application **318**. The application **318**, being implemented in the form of executable program code, can be executed by the data processing system **300**, e.g., by the processor **302**. Responsive to executing the application, the data processing system **300** may be configured to perform one or more operations or method steps described herein.

Various embodiments of the invention may be implemented as a program product for use with a computer system, where the program(s) of the program product define functions of the embodiments (including the methods described herein). In one embodiment, the program(s) can be contained on a variety of non-transitory computer-readable storage media, where, as used herein, the expression “non-transitory computer readable storage media” comprises all computer-readable media, with the sole exception being a transitory, propagating signal. In another embodiment, the program(s) can be contained on a variety of transitory computer-readable storage media. Illustrative computer-readable storage media include, but are not limited to: (i) non-writable storage media (e.g., read-only memory devices within a computer such as CD-ROM disks readable by a CD-ROM drive, ROM chips or any type of solid-state non-volatile semiconductor memory) on which information is permanently stored; and (ii) writable storage media (e.g., flash memory, floppy disks within a diskette drive or hard-disk drive or any type of solid-state random-access semiconductor memory) on which alterable information is stored. The computer program may be run on the processor **302** described herein.

The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the invention. As used herein, the singular forms "a," "an," and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in the claims below are intended to include any structure,

material, or act for performing the function in combination with other claimed elements as specifically claimed. The description of embodiments of the present invention has been presented for purposes of illustration, but is not intended to be exhaustive or limited to the implementations in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the present invention. The embodiments were chosen and described in order to best explain the principles and some practical applications of the present invention, and to enable others of ordinary skill in the art to understand the present invention for various embodiments with various modifications as are suited to the particular use contemplated.

CLAIMS:

1. A system (1) for distributing events to a further system (19,25,26) over a communication network, said further system (19,25,26) being subscribed to receive said events, said system (1) comprising:

at least one memory unit (7);

at least one receiver (3);

at least one transmitter (4); and

at least one processor (5) configured to:

- receive an event subscription request from said further system (19,25,26), said event subscription request requesting said system (1) to distribute events to said further system (19,25,26), said events being indicative of changes to a state of a lighting device (13-14),

- use said at least one receiver (3) to receive a first signal comprising an attribute and a first value of said attribute, said first value of said attribute at least partly representing a color setting, a dim level setting or an on/off setting of a first state of said lighting device (13-14),

- create, based on said received event subscription request and said received first signal, one or more events to be distributed to said further system (19,25,26),

- store said one or more events in a queue in said at least one memory unit (7) to await distribution to said further system (19,25,26) over said communication network using said at least one transmitter (4),

wherein said at least one processor (5) is further configured to:

- use said at least one receiver (3) to receive a second signal comprising said attribute and a second value of said attribute, said second value of said attribute at least partly representing a color setting, a dim level setting or an on/off setting of a second state of said lighting device (13-14), said second state being different from said first state,

- determine whether said queue comprises at least one event indicative of a state of said lighting device (13-14) different from said second state of said lighting device (13-14) with respect to said attribute and remove or amend said at least one event if it has been determined to be indicative of an outdated state of said lighting device (13-14).

2. A system (1) as claimed in claim 1, wherein said at least one processor (5) is configured to include said attribute and said first value of said attribute in one of said one or more events upon creating said one or more events and determine that said event is indicative of a state of said lighting device (13-14) different from said second state with respect to said attribute if said event comprises said attribute and a different value than said second value for said attribute.

3. A system (1) as claimed in claim 2, wherein said at least one processor (5) is configured to include a second attribute and a value of said second attribute in said one of said one or more events.

4. A system (1) as claimed in claim 2 or 3, wherein said at least one processor (5) is configured to remove said attribute and said different value from said event determined to be indicative of a state of said lighting device (13-14) different from said second state with respect to said attribute and create a further event comprising said attribute and said second value.

5. A system (1) as claimed in claim 2 or 3, wherein said at least one processor (5) is configured to replace said different value with said second value in said event determined to be indicative of a state of said lighting device (13-14) different from said second state with respect to said attribute.

6. A system (1) as claimed in claim 2 or 3, wherein said at least one processor (5) is configured to remove said event determined to be indicative of a state of said lighting device (13-14) different from said second state with respect to said attribute and create a further event comprising said attribute and said second value.

7. A system (1) as claimed in claim 1 or 2, wherein said at least one processor (5) is configured to determine whether said queue comprises an event indicative of a state of said lighting device (13-14) different from said second state with respect to said attribute and remove or amend said event upon receiving said second signal.

8. A system (1) as claimed in claim 1 or 2, wherein said at least one processor (5) is configured to determine all events stored in said queue indicative of a state of said lighting device (13-14) different from said second state of said lighting device (13-14) and remove or amend said events.

9. A system (1) as claimed in claim 1 or 2, wherein said further system (19,25,26) is subscribed to receive changes relating to said attribute specifically.

10. A system (1) as claimed in claim 1 or 2, wherein said queue is specific to said further system (19,25,26).

11. A method of distributing events to a further system over a communication network, said further system being subscribed to receive said events, said method comprising:

- receiving (101) an event subscription request from said further system, said event subscription request requesting said system to distribute events to said further system, said events being indicative of changes to a state of a lighting device,
- receiving (103,121) a first signal comprising an attribute and a first value of said attribute, said first value of said attribute at least partly representing a color setting, a dim level setting or an on/off setting of a first state of said lighting device,
- creating (105), based on said received event subscription request and said received first signal, one or more events to be distributed to said further system,
- storing (107) said one or more events in a queue in a memory to await distribution to said further system over said communication network,

wherein said method comprises:

- receiving (109,121) a second signal comprising said attribute and a second value of said attribute, said second value of said attribute at least partly representing a color setting, a dim level setting or an on/off setting of a second state of said lighting device, said second state being different from said first state,
- determining (111) whether said queue comprises an event indicative of a state of said lighting device different from said second state of said lighting device with respect to said attribute and remove (113) or amend (115) said event.

12. A computer program or suite of computer programs comprising at least one software code portion or a computer program product storing at least one software code portion, the software code portion, when run on a computer system, being configured for enabling the method of claim 11 to be performed.

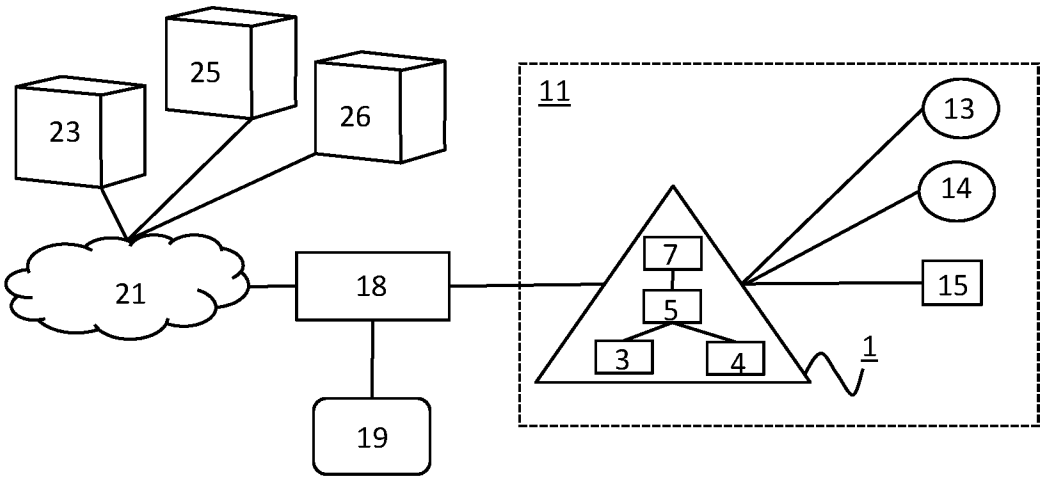


Fig. 1

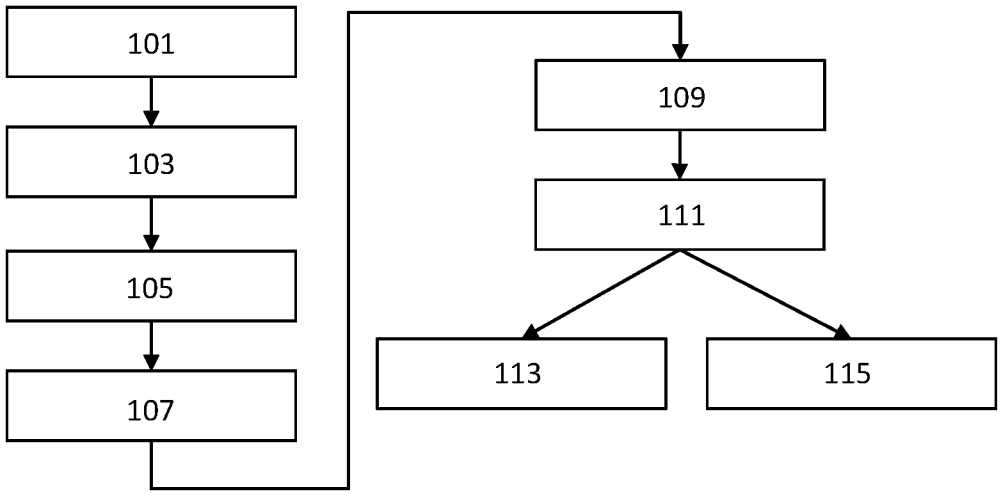


Fig. 2

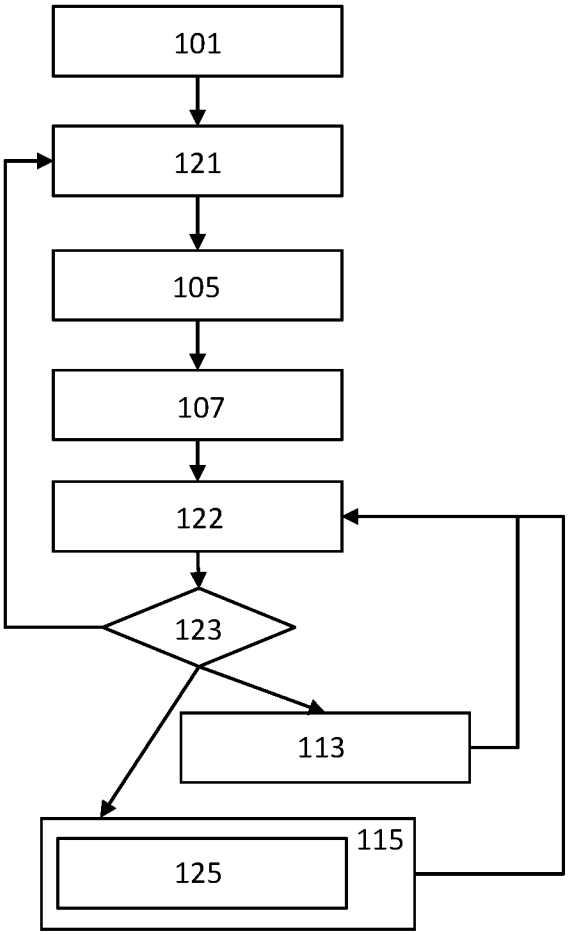


Fig. 3

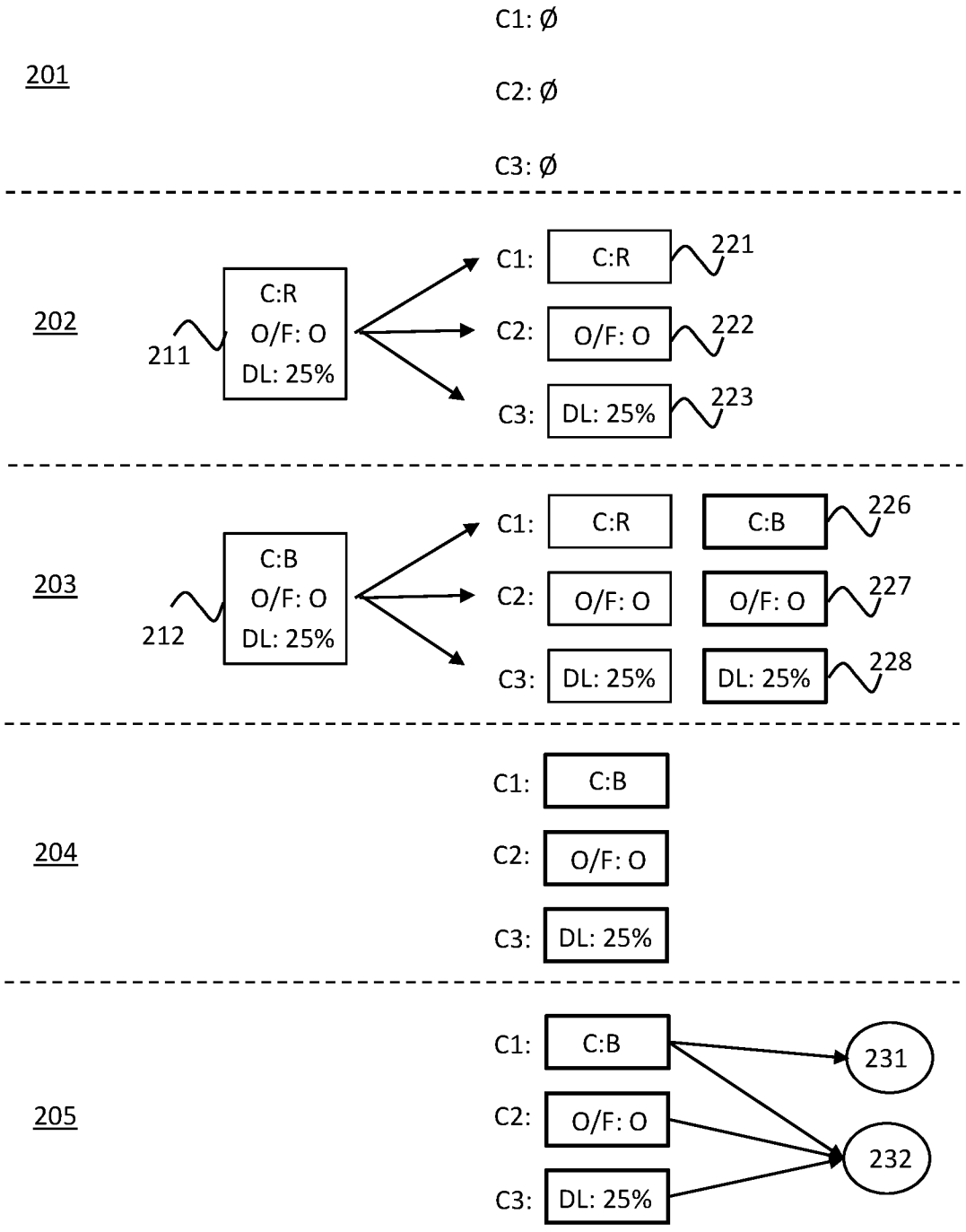

201
C1: Ø
C2: Ø
C3: Ø
202
211
C:R
O/F: O
DL: 25%
C1: C:R
221
C2: O/F: O
222
C3: DL: 25%
223
203
212
C:B
O/F: O
DL: 25%
C1: C:R
C:B
226
C2: O/F: O
O/F: O
227
C3: DL: 25%
DL: 25%
228
204
C1: C:B
C2: O/F: O
C3: DL: 25%
205
C1: C:B
C2: O/F: O
C3: DL: 25%
231
232

Fig. 4

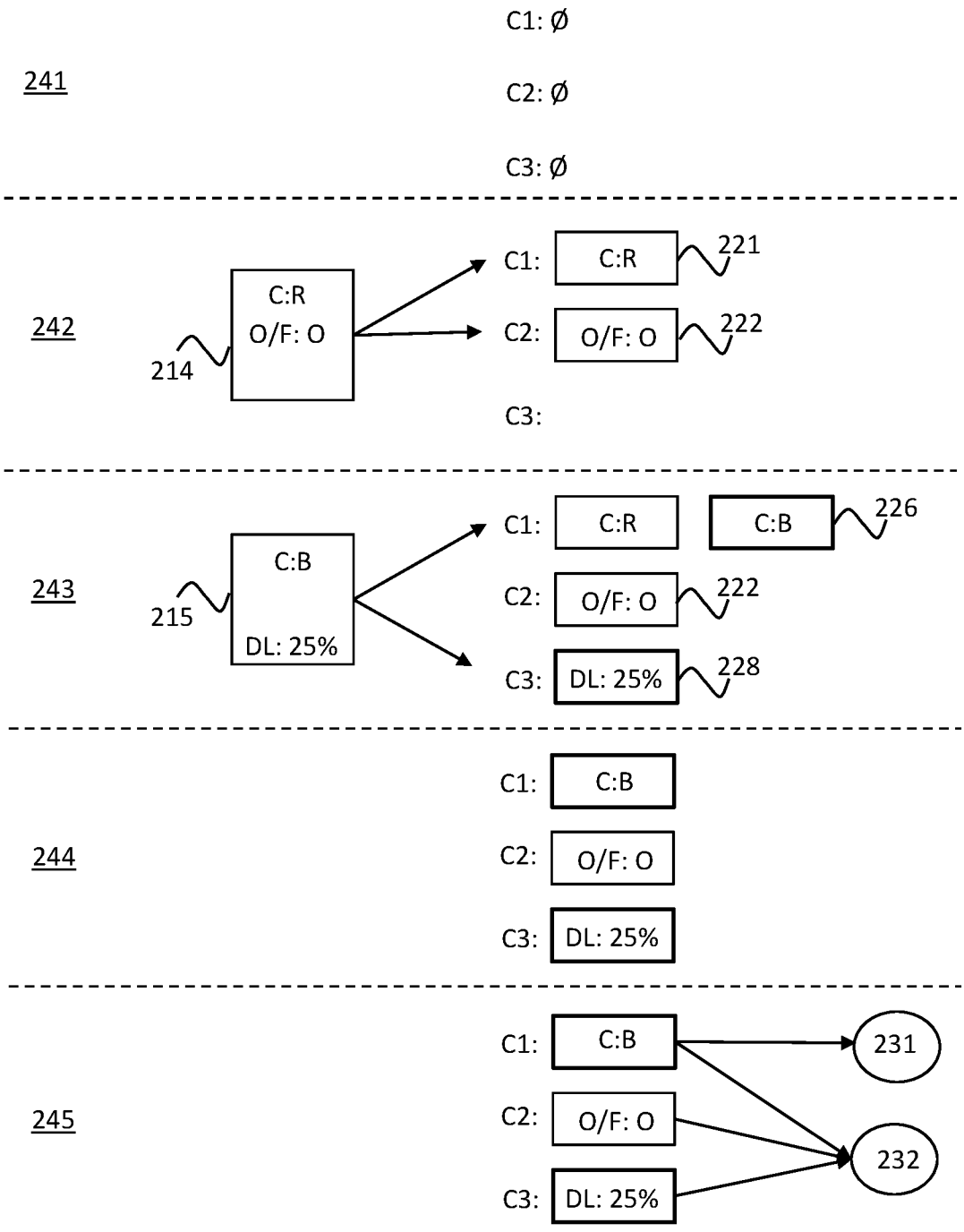

241
C1: Ø
C2: Ø
C3: Ø
242
C:R
O/F: O
214
C1:
C:R
221
C2:
O/F: O
222
C3:
243
C:B
DL: 25%
215
C1:
C:R
C:B
226
C2:
O/F: O
222
C3:
DL: 25%
228
244
C1:
C:B
C2:
O/F: O
C3:
DL: 25%
245
C1:
C:B
231
C2:
O/F: O
232
C3:
DL: 25%

Fig. 5

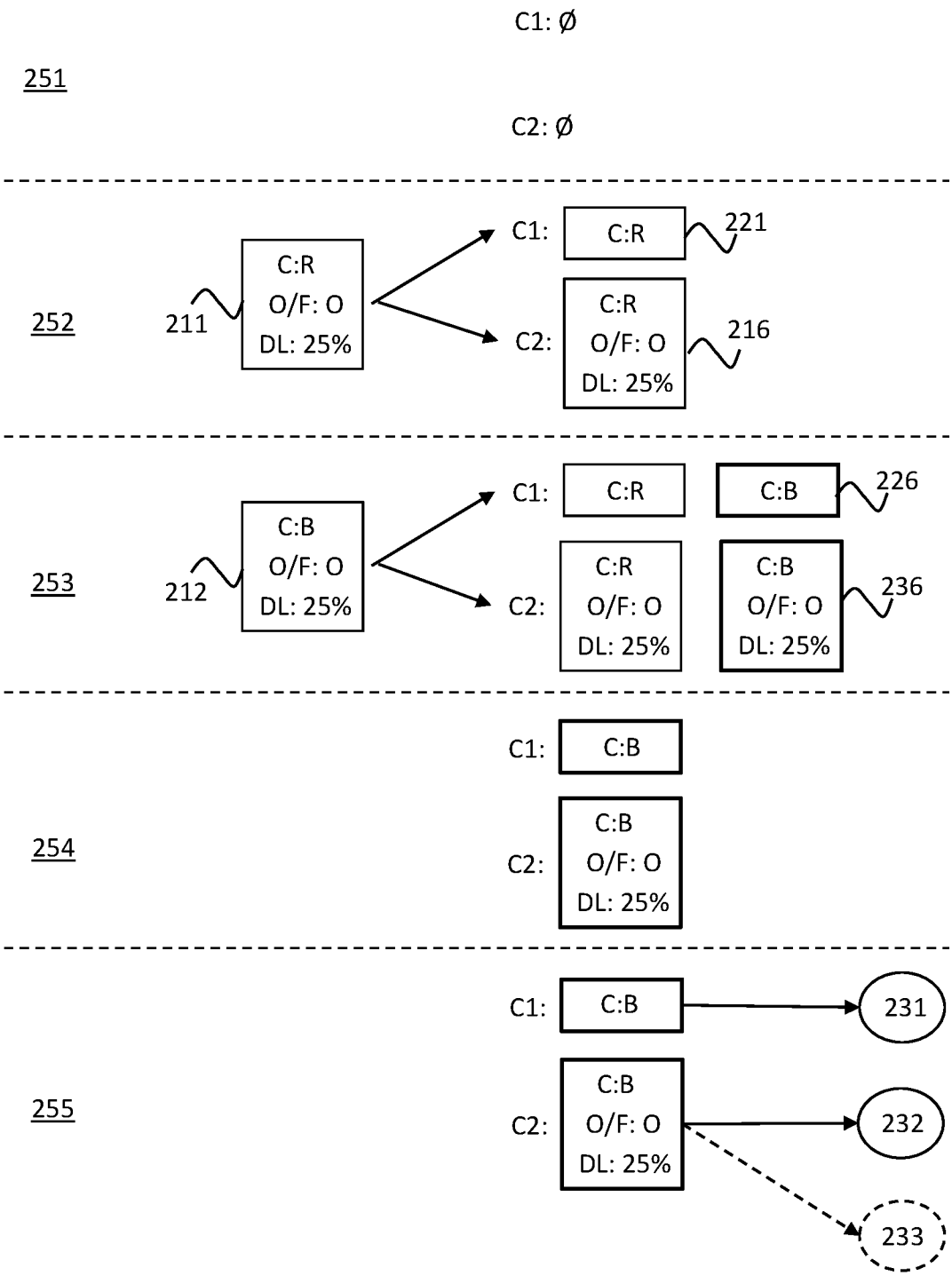

C1: Ø
251
C2: Ø
252
211
C:R
O/F: O
DL: 25%
C1:
C:R
221
C2:
C:R
O/F: O
DL: 25%
216
253
212
C:B
O/F: O
DL: 25%
C1:
C:R
C:B
226
C2:
C:R
O/F: O
DL: 25%
C:B
O/F: O
DL: 25%
236
254
C1:
C:B
C2:
C:B
O/F: O
DL: 25%
255
C1:
C:B
231
C2:
C:B
O/F: O
DL: 25%
232
233

Fig. 6

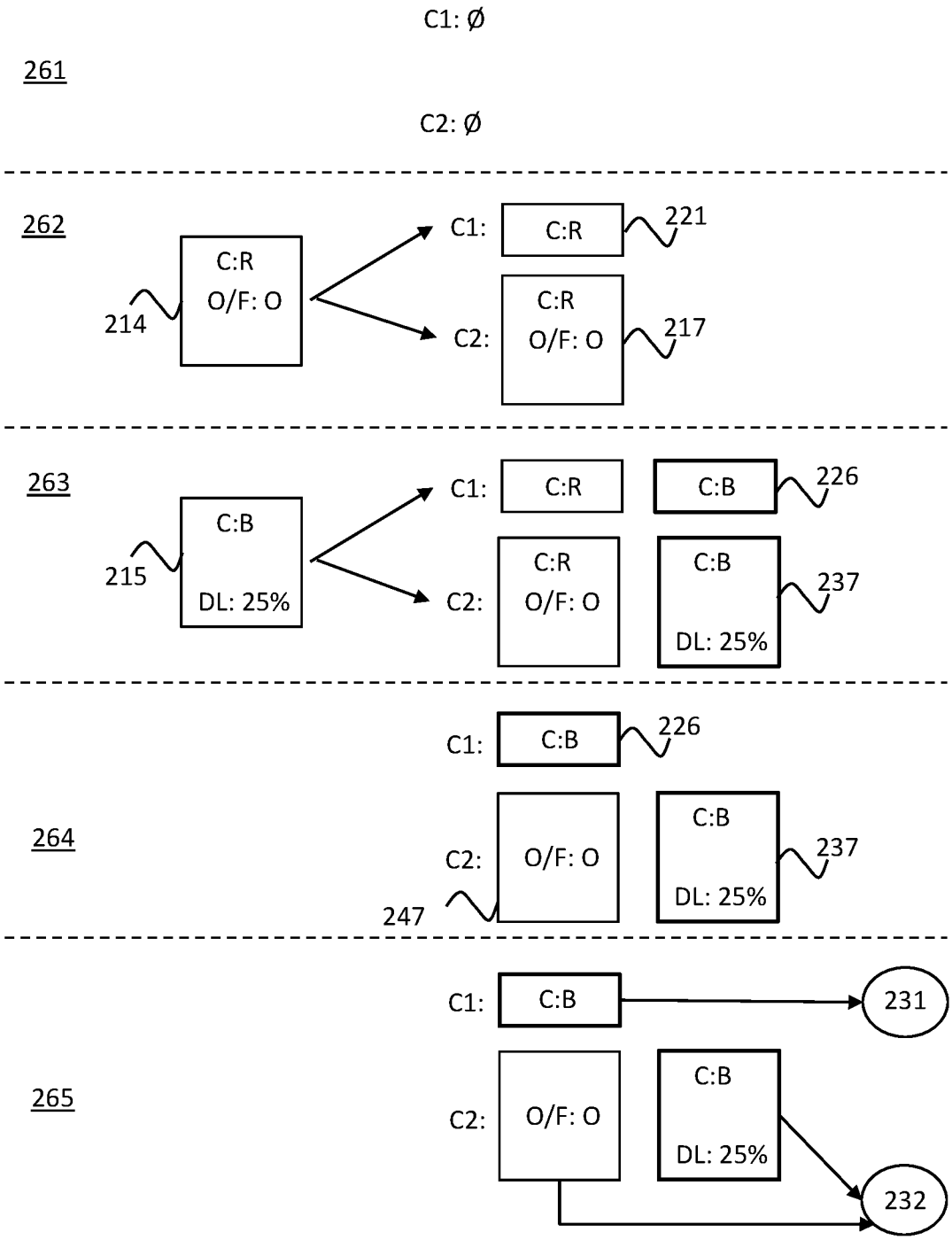

C1: Ø
261
C2: Ø
262
C:R
O/F: O
214
C1:
C:R
221
C2:
C:R
O/F: O
217
263
C:B
DL: 25%
215
C1:
C:R
C:B
226
C2:
C:R
O/F: O
C:B
DL: 25%
237
264
C1:
C:B
226
C2:
O/F: O
C:B
DL: 25%
237
247
265
C1:
C:B
231
C2:
O/F: O
C:B
DL: 25%
232

Fig. 7

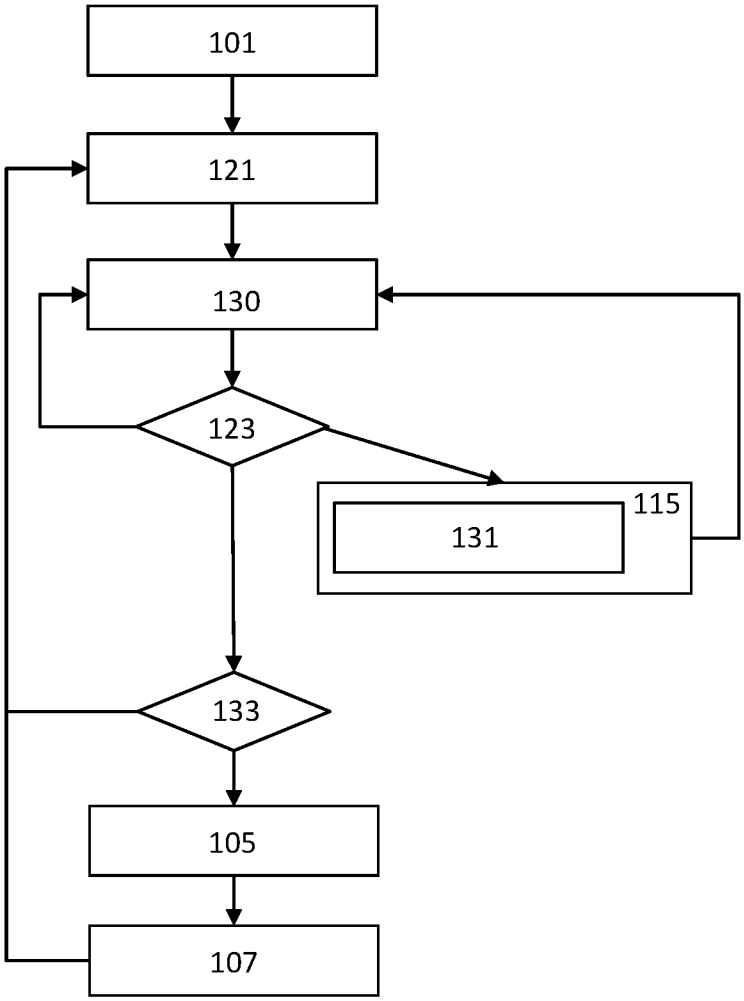


Fig. 8

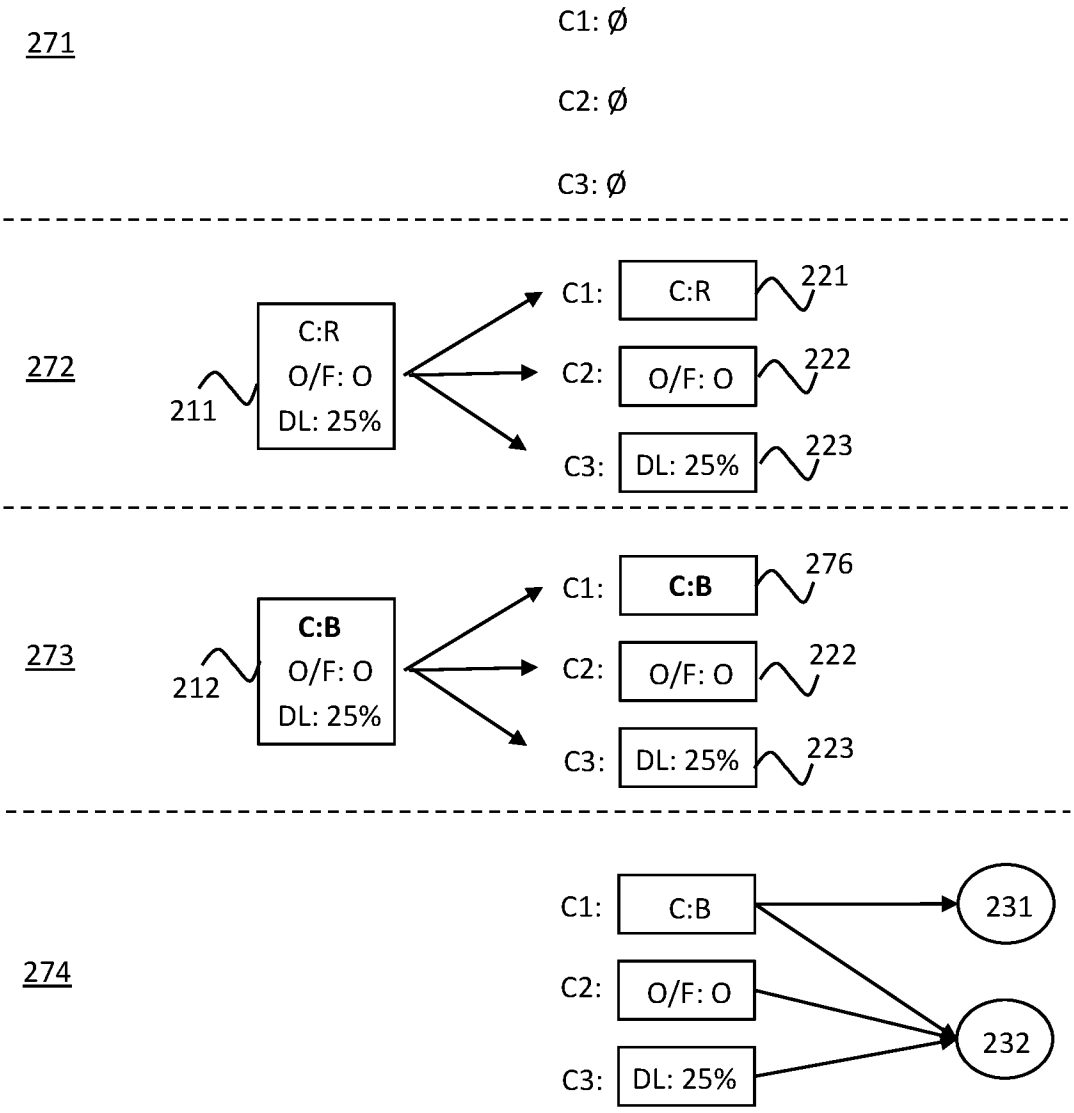

271
C1: Ø
C2: Ø
C3: Ø
272
211
C:R
O/F: O
DL: 25%
C1:
C:R
221
C2:
O/F: O
222
C3:
DL: 25%
223
273
212
C:B
O/F: O
DL: 25%
C1:
C:B
276
C2:
O/F: O
222
C3:
DL: 25%
223
274
C1:
C:B
C2:
O/F: O
C3:
DL: 25%
231
232

Fig. 9

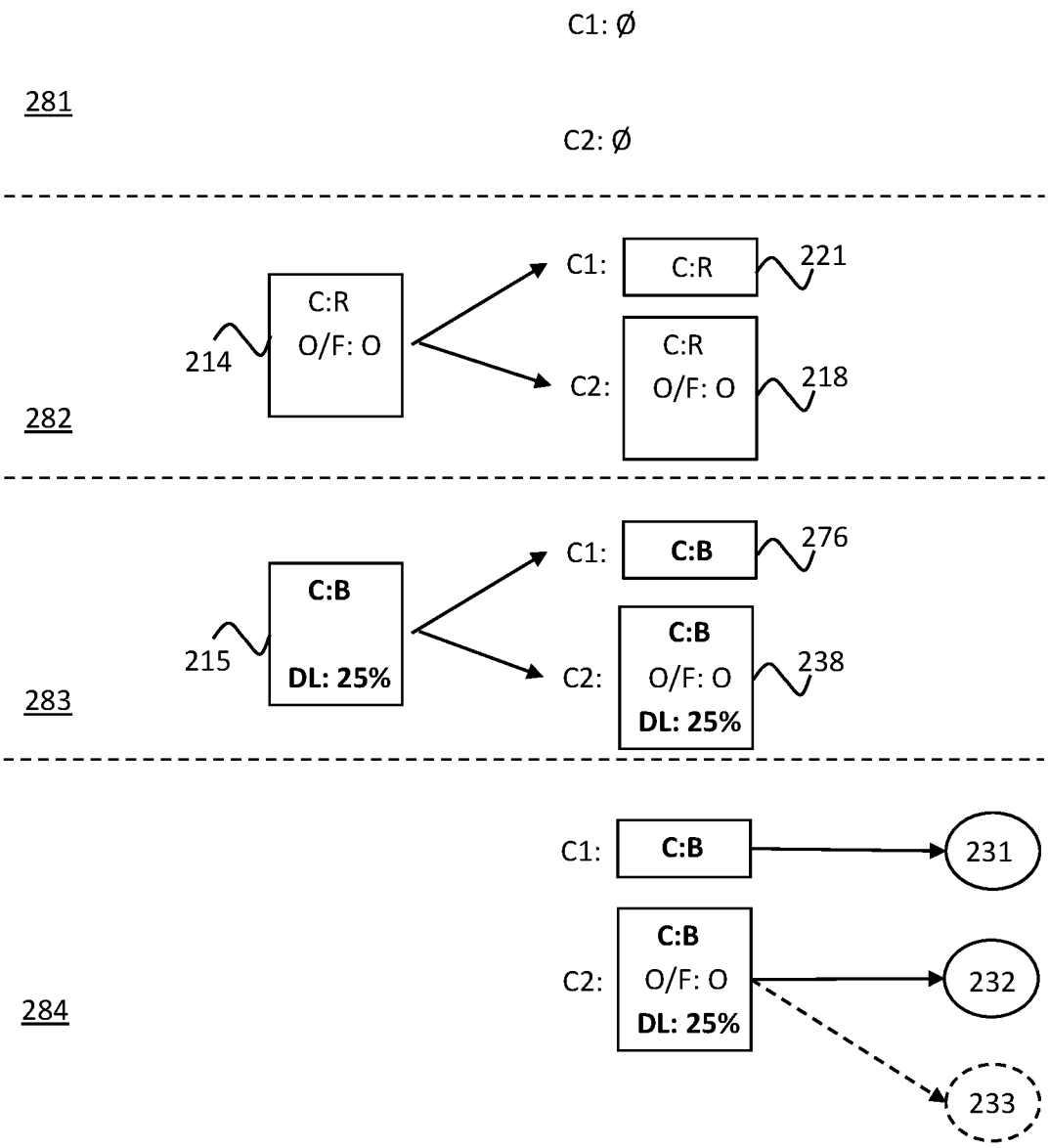

C1: Ø
281
C2: Ø
C:R
O/F: O
214
C1:
C:R
221
C2:
C:R
O/F: O
218
282
C:B
DL: 25%
215
C1:
C:B
276
C2:
C:B
O/F: O
DL: 25%
238
283
C1:
C:B
231
C2:
C:B
O/F: O
DL: 25%
232
233
284

Fig. 10

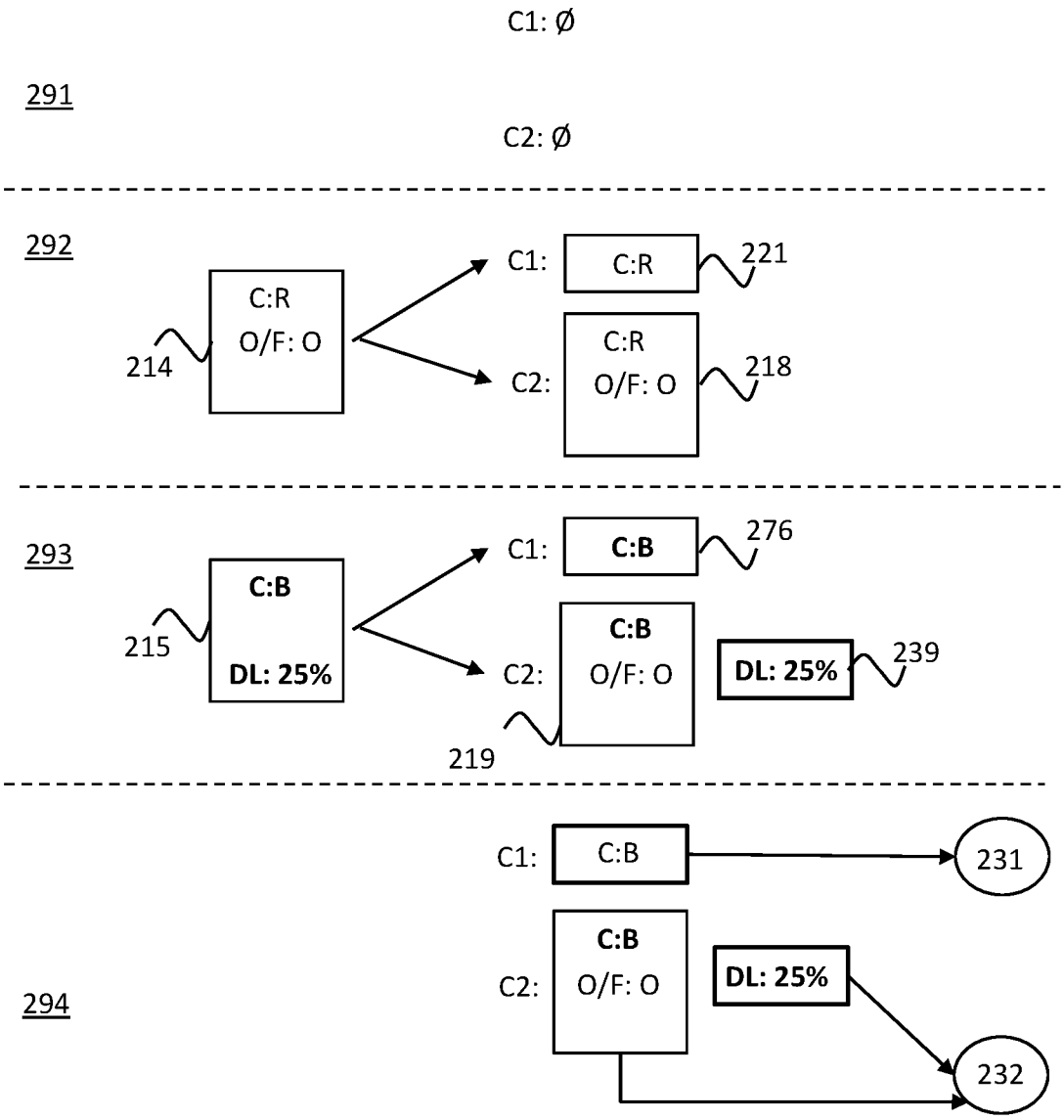

C1: Ø
291
C2: Ø
292
C:R
O/F: O
214
C1:
C:R
221
C2:
C:R
O/F: O
218
293
C:B
DL: 25%
215
C1:
C:B
276
C2:
C:B
O/F: O
219
DL: 25%
239
294
C1:
C:B
231
C2:
C:B
O/F: O
DL: 25%
232

Fig. 11

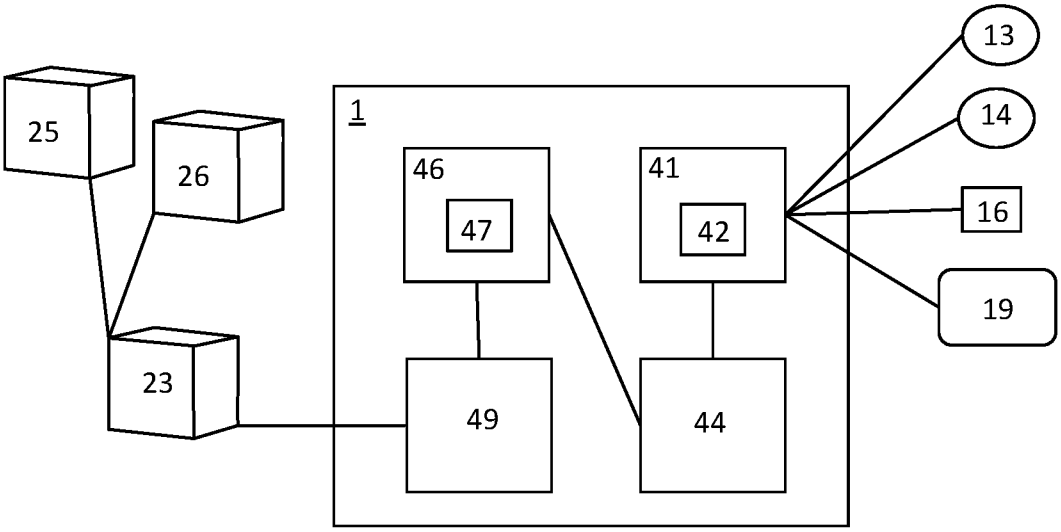

25
26
23
1
46
47
41
42
49
44
13
14
16
19

Fig. 12

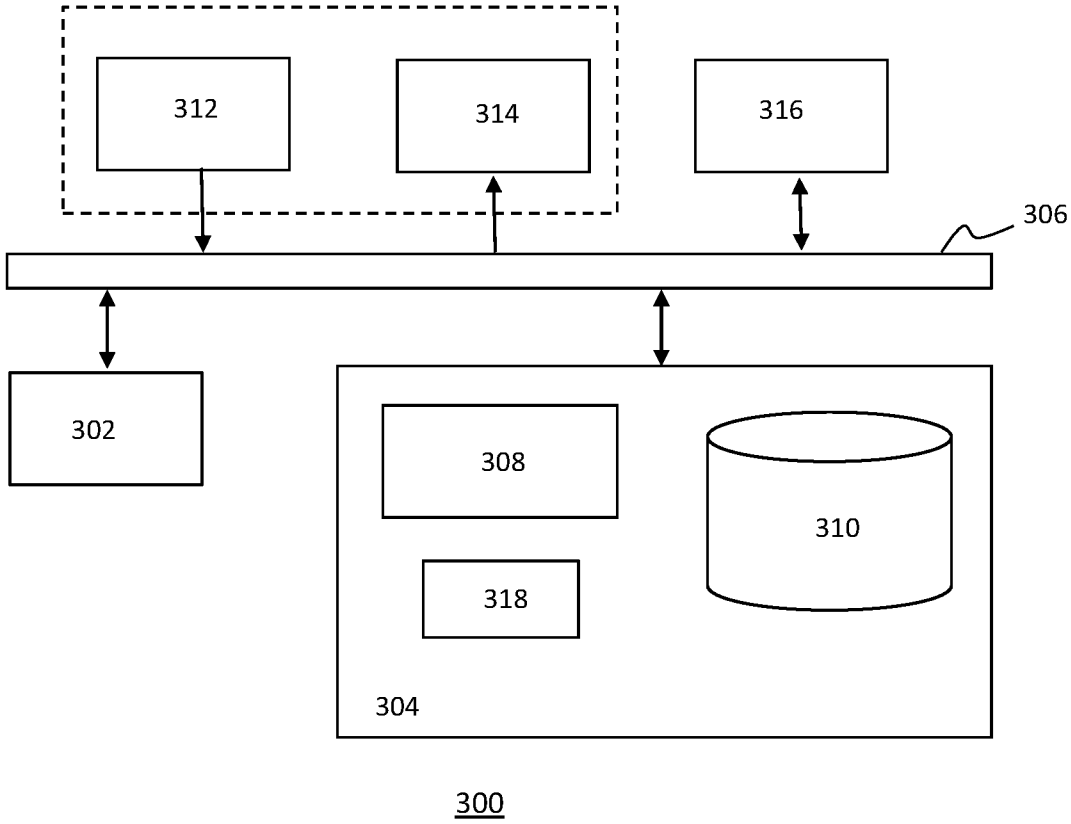

312
314
316
306
302
308
310
318
304
300

Fig. 13

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2020/052660

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/54
ADD.

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CHEN ET AL: "Data-centric middleware for context-aware pervasive computing", PERVASIVE AND MOBILE COMPUTING, ELSEVIER, NL, vol. 4, no. 2, 22 October 2007 (2007-10-22), pages 216-253, XP022527976, ISSN: 1574-1192, DOI: 10.1016/J.PMCJ.2007.10.001 section 2; page 219 section 6; pages 229-235; figures 7a, 7b section 7.1; page 243	1-12

-/--

[X] Further documents are listed in the continuation of Box C.

[ ] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
1 April 2020	15/04/2020

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Nourestani, S

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Ohad Oman: "What is IoT? Understanding IoT Protocols,Clients, and Servers", Friendly Technologies documents online, 1 November 2017 (2017-11-01), pages 1-24, XP055524801, Retrieved from the Internet: URL:https://walt-tech.com.au/documents/Walt%20Tech%20-%20FTL%20-%20What%20is%20IoT%20(in%20terms%20of%20Device%20Management).pdf [retrieved on 2018-11-19] page 5 -----	1,11,12