



(51) International Patent Classification:

H04L 41/084 (2022.01) G06F 8/70 (2018.01)  
H04L 41/082 (2022.01)

(21) International Application Number:

PCT/IN2024/051233

(22) International Filing Date:

15 July 2024 (15.07.2024)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

202321048394 19 July 2023 (19.07.2023) IN

(71) Applicant: **JIO PLATFORMS LIMITED** [IN/IN]; Office-101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad, Gujarat 380006 (IN).

(72) Inventors: **SHETTY, Mukta**; Flat No 302, Mukund Park, Sheetal Nagar, Mira Road (East), Thane, Maharashtra 401107 (IN). **JHA, Alok B**; B1-1701, G21 Avenue, Sector 83, Vatika INXT, Gurugram, Haryana 122004 (IN). **KUMAR, Sanjeev**; House No. 8, V.P.O. - Kalawar, Tehsil Jagadhri, Distt - Yamuna Nagar - Haryana, India 133103 (IN). **KALAVADIYA, Milankumar**; 4-68, Braman Street, Hadiyana, Jamanagar, Gujarat 361011 (IN). **DUNDI, Ki-**

ran; House No. 17-30, Indraprastha Shanthigram, Eluru, West Godavari, Andhra Pradesh 534007 (IN). **BHATNAGAR, Aayush**; Tower-7, 15B, Beverly Park, Sector 14, Koper Khairne, Navi Mumbai, Maharashtra 400709 (IN). **SAGAR, Gautam**; Reliance Corporate Park, Thane-Belapur Road, Ghansoli, Navi Mumbai, Maharashtra 400701 (IN). **MALHOTRA, Deeksha**; Reliance Corporate Park, Thane-Belapur Road, Ghansoli, Navi Mumbai, Maharashtra 400701 (IN). **HINGU, Ketan**; Reliance Corporate Park, Thane-Belapur Road, Ghansoli, Navi Mumbai, Maharashtra 400701 (IN).

(74) Agent: **DJ, Solomon et al.**; De Penning & De Penning, 120 Velachery Main Road, Guindy, Chennai 600032 (IN).

(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,

(54) Title: SYSTEM AND METHOD FOR CONFIGURING AN APPLICATION

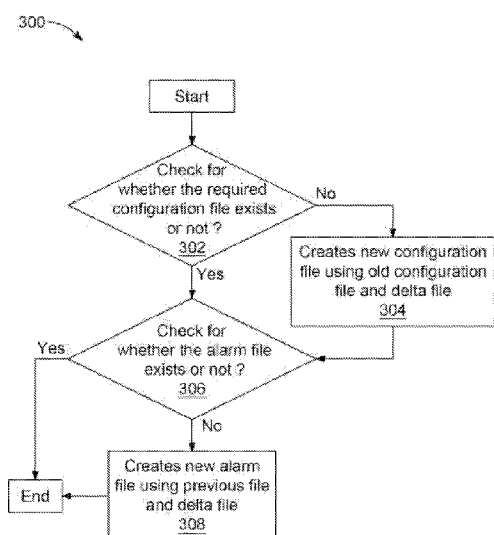


FIG. 3

(57) Abstract: The present disclosure describes a method(600) and system(108) for configuring an application. The method(600) comprises (i) identifying, by a merging engine(212), an existence of a required configuration file corresponding to a new release of a network function of the application, (ii) creating, by the merging engine(212), if the required configuration file does not exist, a new configuration file by merging an old configuration file and a delta file available in a configuration repository(220), (iii) identifying, by an alarm file module(218), the existence of a required alarm file, (iv) creating, by the alarm file module(218), if the required alarm file does not exist, a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository(220), and (v) configuring the application, by a processing engine(208), with newly created configuration and alarm files.

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))*
- *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*
- *in black and white; the international application as filed contained color or greyscale and is available for download from PATENTSCOPE*

## SYSTEM AND METHOD FOR CONFIGURING AN APPLICATION

### RESERVATION OF RIGHTS

[001] A portion of the disclosure of this patent document contains material  
5 which is subject to intellectual property rights such as, but are not limited to,  
copyright, design, trademark, integrated circuit (IC) layout design, and/or trade  
dress protection, belonging to Jio Platforms Limited (JPL) or its affiliates  
(hereinafter referred as owner). The owner has no objection to the facsimile  
10 reproduction by anyone of the patent document or the patent disclosure, as it  
appears in the Patent and Trademark Office patent files or records, but otherwise  
reserves all rights whatsoever. All rights to such intellectual property are fully  
reserved by the owner.

### TECHNICAL FIELD

15 [002] The present disclosure relates to a field of a wireless network, and  
specifically to a system and a method for configuring an application.

### BACKGROUND

[003] The following description of related art is intended to provide  
20 background information pertaining to the field of the disclosure. This section may  
include certain aspects of the art that may be related to various features of the  
present disclosure. However, it should be appreciated that this section be used only  
to enhance the understanding of the reader with respect to the present disclosure,  
and not as admissions of prior art.

25 [004] In the realm of 5th Generation (5G) networks, various network  
functions such as policy control function (PCF), binding support function (BSF),  
network repository function (NRF), and charging function (CHF) rely on  
configuration files to manage operational parameters and to raise alarms. These  
configuration files need to be modified at every version upgrade to include new  
30 parameters and alarms. Traditionally, this process of updating configuration files  
and alarms is performed manually. Manual reconfiguration involves adding,

modifying, and sometimes removing parameters, which is a time-consuming and error-prone process. Due to the complexity and volume of configurations involved, manual updates increase the likelihood of errors during the upgrade phase. These errors can severely impact the application's performance and metrics, potentially causing the application to fail during booting and resulting in longer downtimes.

[005] For example, consider a scenario where a network function is being upgraded from version 1.0 to version 2.0. The upgrade necessitates the addition of new configuration parameters (e.g., transactions per second) and alarms (e.g., host unavailable, database (DB) down, threshold values, etc.) specific to the new version. Manually updating these configurations requires identifying the changes, modifying the existing configuration files, and ensuring that all new parameters are correctly deployed. This manual process is prone to human error, such as typographical mistakes, omission of critical parameters, or incorrect parameter values, leading to potential application failures or degraded performance.

[006] There is, therefore, a need in the art to improve the state of automatically upgrading configuration files of the application by overcoming the deficiencies of the prior arts. There is a need for an automated solution for merging new parameters and alarms with those from previous releases using delta files. Delta files contain only the changes, such as additions, modifications, or removals needed for the new version.

[007] There is, therefore, a need in the art to improve the state of automatically upgrading configuration files by overcoming the deficiencies of the prior arts.

## OBJECTS OF THE PRESENT DISCLOSURE

[008] It is an object of the present disclosure to provide a system and a method for configuring an application.

[009] It is an object of the present disclosure to provide a system and a

method to merge configurations and upgrade network functions automatically.

[0010] It is an object of the present disclosure to automate the merging of new parameters and alarms with old parameters and alarms in an old release through additionally provided delta files.

- 5 [0011] It is an object of the present disclosure to remove and modify existing parameters and alarms of the network functions.

[0012] It is an object of the present disclosure to ensure accurate configuration upgrade of the network functions.

- 10 [0013] It is an object of the present disclosure to reduce the chances of errors during an upgrade phase and ensure that the application's performance and metrics are not affected.

- 15 [0014] It is an object of the present disclosure to promote smooth process flows and consistency in reducing cycle time, resources, and improve productivity by performing automation of merging of the new parameters with the old parameters.

[0015] It is an object of the present disclosure to achieve time-saving, improved efficiency, reduced risk of outages, minimal errors, faster deployment, and consistency.

## SUMMARY

- 20 [0016] In an exemplary embodiment, a method for configuring an application is described. The method comprises identifying, by a merging engine, an existence of a required configuration file corresponding to a new release of a network function of the application. The method comprises creating, by the merging engine, if the required configuration file does not exist, a new configuration file by  
25 merging an old configuration file and a configuration delta file available in a configuration repository. The method comprises identifying, by an alarm file

module, the existence of a required alarm file corresponding to the new release of the network function of the application. The method comprises creating, by the alarm file module, if the required alarm file does not exist, a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository. The method comprises configuring the application, by a processing engine, with the newly created configuration and alarm files.

**[0017]** In an embodiment, the merging engine reads environment variables and populates them in the configuration file.

**[0018]** In an embodiment, the configuration delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the configuration file.

**[0019]** In an embodiment, the alarm delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the alarm file.

**[0020]** In an embodiment, the merging engine is configured to run before the application is booted.

**[0021]** In another exemplary embodiment, a system for configuring an application is described. The system comprises a merging engine configured to identify the existence of a required configuration file corresponding to a new release of a network function of the application and create a new configuration file by merging an old configuration file and a configuration delta file available in a configuration repository if the required configuration file does not exist. The system comprises an alarm file module configured to identify the existence of a required alarm file corresponding to the new release of the network function of the application and create a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository if the required alarm file does not exist. The system comprises a processing engine, operatively connected to the configuration tool, merging engine, and the alarm file module, to configure the

application with the newly created configuration and alarm files.

**[0022]** In an embodiment, the merging engine is further configured to read environment variables and populate them in the configuration file.

**[0023]** In an embodiment, the configuration delta file includes at least one  
5 or more parameters from a group comprising of new, modified, or removed parameters associated with the configuration file.

**[0024]** In an embodiment, the alarm delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the alarm file.

10 **[0025]** In an embodiment, the merging engine is configured to run before the application is booted.

**[0026]** In an embodiment, the present invention discloses a user equipment (UE) communicatively coupled with a network. The coupling comprises steps of receiving, by the network, a connection request from the UE, sending, by the  
15 network, an acknowledgment of the connection request to the UE and transmitting a plurality of signals in response to the connection request. Configuring an application in the network is performed by a system that comprises a merging engine configured to identify the existence of a required configuration file corresponding to a new release of a network function of the application and create  
20 a new configuration file by merging an old configuration file and a configuration delta file available in a configuration repository if the required configuration file does not exist. The system comprises an alarm file module configured to identify the existence of a required alarm file corresponding to the new release of the network function of the application and create a new alarm file by merging an old  
25 alarm file and an alarm delta file available in the configuration repository if the required alarm file does not exist. The system comprises a processing engine, operatively connected to the configuration tool, merging engine, and the alarm file module, to configure the application with the newly created configuration and alarm

files.

[0027] The foregoing general description of the illustrative embodiments and the following detailed description thereof are merely exemplary aspects of the teachings of this disclosure, and are not restrictive.

## 5 BRIEF DESCRIPTION OF THE DRAWINGS

[0028] In the figures, similar components and/or features may have the same reference label. Further, various components of the same type may be distinguished by following the reference label with a second label that distinguishes among the similar components. If only the first reference label is used in the  
10 specification, the description is applicable to any one of the similar components having the same first reference label irrespective of the second reference label.

[0029] The diagrams are for illustration only, which thus is not a limitation of the present disclosure, and wherein:

[0030] FIG. 1 illustrates an exemplary network architecture in which or with  
15 which embodiments of the present disclosure may be implemented.

[0031] FIG. 2 illustrates an exemplary block diagram of a proposed system (108), in accordance with an embodiment of the present disclosure.

[0032] FIG. 3 illustrates an exemplary flow chart implementing a method for merging configurations and upgrading network functions, in accordance with  
20 an embodiment of the present disclosure.

[0033] FIG. 4 illustrates an exemplary architecture of the proposed system (108), in accordance with an embodiment of the present disclosure.

[0034] FIG. 5 illustrates an exemplary computer system in which or with which embodiments of the present disclosure may be implemented.

25 [0035] FIG. 6 illustrates an exemplary flowchart depicting a method for

configuring an application in a network, in accordance with an embodiment of the present disclosure.

## **LIST OF REFERENCE NUMERALS**

- 100 - Network architecture
- 5    102-1, 102-2...102-N - Users
- 104-1, 104-2...104-N - User equipments
- 106 - Network
- 108 - System
- 112 - Centralized server
- 10   202 - Processor(s)
- 204 - Memory
- 206 - Interface(s)
- 208 - Processing engine
- 210 - Database
- 15   212 - Merging engine
- 214 - Other engines
- 216 - Configuration tool
- 218 - Alarm file module
- 220 - Configuration repository
- 20   300 - Flow diagram

400 - System architecture

402 - Configuration folder

404 - Network function

406 - Configuration tool

5 500 - Computer System

510 - External Storage Device

520 - Bus

530 - Main Memory

540 - Read-Only Memory

10 550 - Mass Storage Device

560 - Communication Port(s)

570 - Processor

600 - Flow diagram

## DETAILED DESCRIPTION

15 **[0036]** The following is a detailed description of embodiments of the disclosure depicted in the accompanying drawings. The embodiments are in such detail as to clearly communicate the disclosure. However, the amount of detail offered is not intended to limit the anticipated variations of embodiments; on the contrary, the intention is to cover all modifications, equivalents, and alternatives  
20 falling within the spirit and scope of the present disclosure as defined by the appended claims.

[0037] In the following description, for explanation, various specific details are outlined in order to provide a thorough understanding of embodiments of the present disclosure. It will be apparent, however, that embodiments of the present disclosure may be practiced without these specific details. Several features  
5 described hereafter can each be used independently of one another or with any combination of other features. An individual feature may not address all of the problems discussed above or might address only some of the problems discussed above. Some of the problems discussed above might not be fully addressed by any of the features described herein.

10 [0038] The ensuing description provides exemplary embodiments only and is not intended to limit the scope, applicability, or configuration of the disclosure. Rather, the ensuing description of the exemplary embodiments will provide those skilled in the art with an enabling description for implementing an exemplary embodiment. It should be understood that various changes may be made in the  
15 function and arrangement of elements without departing from the spirit and scope of the disclosure as set forth.

[0039] Specific details are given in the following description to provide a thorough understanding of the embodiments. However, it will be understood by one of ordinary skill in the art that the embodiments may be practiced without these  
20 specific details. For example, circuits, systems, networks, processes, and other components may be shown as components in block diagram form in order not to obscure the embodiments in unnecessary detail. In other instances, well-known circuits, processes, algorithms, structures, and techniques may be shown without unnecessary detail to avoid obscuring the embodiments.

25 [0040] Specific details are given in the following description to provide a thorough understanding of the embodiments. However, it will be understood by one of ordinary skill in the art that the embodiments may be practiced without these specific details. For example, circuits, systems, networks, processes, and other components may be shown as components in block diagram form in order not to

obscure the embodiments in unnecessary detail. In other instances, well-known circuits, processes, algorithms, structures, and techniques may be shown without unnecessary detail in order to avoid obscuring the embodiments.

**[0041]** Also, it is noted that individual embodiments may be described as a process that is depicted as a flowchart, a flow diagram, a data flow diagram, a structure diagram, or a block diagram. Although a flowchart may describe the operations as a sequential process, many of the operations can be performed in parallel or concurrently. In addition, the order of the operations may be re-arranged. A process is terminated when its operations are completed but could have additional steps not included in a figure. A process may correspond to a method, a function, a procedure, a subroutine, a subprogram, etc. When a process corresponds to a function, its termination can correspond to a return of the function to the calling function or the main function.

**[0042]** The word “exemplary” and/or “demonstrative” is used herein to mean serving as an example, instance, or illustration. For the avoidance of doubt, the subject matter disclosed herein is not limited by such examples. In addition, any aspect or design described herein as “exemplary” and/or “demonstrative” is not necessarily to be construed as preferred or advantageous over other aspects or designs, nor is it meant to preclude equivalent exemplary structures and techniques known to those of ordinary skill in the art. Furthermore, to the extent that the terms “includes,” “has,” “contains,” and other similar words are used in either the detailed description or the claims, such terms are intended to be inclusive like the term “comprising” as an open transition word without precluding any additional or other elements.

**[0043]** Reference throughout this specification to “one embodiment” or “an embodiment” or “an instance” or “one instance” means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present disclosure. Thus, the appearances of the phrases “in one embodiment” or “in an embodiment” in various places throughout

this specification are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

**[0044]** The terminology used herein is to describe particular embodiments only and is not intended to be limiting the disclosure. As used herein, the singular forms “a”, “an”, and “the” are intended to include the plural forms as well, unless the context 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. As used herein, the term “and/or” includes any combinations of one or more of the associated listed items.

**[0045]** The present disclosure describes an automated solution for merging new parameters (e.g., transactions per second) and alarms (e.g., host unavailable, database (DB) down, threshold values, etc.) with those from previous releases using delta files. The delta files contain only the changes, such as additions, modifications, or removals needed for the new version. A configuration tool is utilized to automate the merging process, significantly reducing the chances of errors during upgrades. In an embodiment, delta files are provided with each application release when any new configuration parameter(s) and/or alarm(s) are added or deleted in that release.

**[0046]** The automation of the configuration merging process during the upgradation of network functions offers several benefits. Firstly, it is time-saving and efficient. Automating the configuration merging through delta files saves time by eliminating the need for manual reconfiguration. Network agility is required for network-related organizations, and automation during upgrades becomes essential. For instance, a process that previously required several hours of manual configuration upgradation can now be completed in seconds, allowing network

administrators to focus on other critical tasks.

[0047] Secondly, the configuration tool minimizes errors. The tool reduces the likelihood of errors during the upgrade phase, ensures that all parameters are accurately updated. Manual configuration is susceptible to human error, which can lead to major network downtimes. Automation ensures that the application's performance and metrics are not affected.

[0048] Lastly, the automation facilitates faster deployment and consistency. Upgrading network functions using the configuration tool's features can be performed rapidly and validated with minimal human intervention. This reduces deployment time from a manual half-day process to a task that requires minimal human intervention. Automation of merging new parameters also promotes smooth process flows and consistency in reducing cycle time, resources, and improving productivity.

[0049] In Docker and Kubernetes (K8s) environments, where it is necessary to take some configuration from environment variables, the configuration tool can read environment variables and populate them in the configuration file. This ensures that when the application starts, it only needs to read the configuration file for all configuration parameters, eliminating the need to read environment variables. This approach further streamlines the configuration process and enhances the efficiency and reliability of network function upgrades.

[0050] The various embodiments of the present disclosure will be explained in detail with reference to FIGs. 1 to 6.

[0051] FIG. 1 illustrates an exemplary network architecture in which or with which a system (108) for upgrading the network functions in a wireless network is implemented, in accordance with embodiments of the present disclosure.

[0052] Referring to FIG. 1, the network architecture (100) includes one or more computing devices or user equipments (104-1, 104-2...104-N) associated

with one or more users (102-1, 102-2...102-N) in an environment. A person of ordinary skill in the art will understand that one or more users (102-1, 102-2...102-N) may be individually referred to as the user (102) and collectively referred to as the users (102). Similarly, a person of ordinary skill in the art will understand that one or more user equipments (104-1, 104-2...104-N) may be individually referred to as the user equipment (104) and collectively referred to as the user equipment (104). A person of ordinary skill in the art will appreciate that the terms “computing device(s)” and “user equipment” may be used interchangeably throughout the disclosure. Although three user equipments (104) are depicted in FIG. 1, however any number of the user equipments (104) may be included without departing from the scope of the ongoing description.

**[0053]** In an embodiment, the user equipment (104) includes smart devices operating in a smart environment, for example, an Internet of Things (IoT) system. In such an embodiment, the user equipment (104) may include, but is not limited to, smart phones, smart watches, smart sensors (e.g., mechanical, thermal, electrical, magnetic, etc.), networked appliances, networked peripheral devices, networked lighting system, communication devices, networked vehicle accessories, networked vehicular devices, smart accessories, tablets, smart television (TV), computers, smart security system, smart home system, other devices for monitoring or interacting with or for the users (102) and/or entities, or any combination thereof. A person of ordinary skill in the art will appreciate that the user equipment (104) may include, but is not limited to, intelligent, multi-sensing, network-connected devices, that can integrate seamlessly with each other and/or with a central server or a cloud-computing system or any other device that is network-connected.

**[0054]** In an embodiment, the user equipment (104) includes, but is not limited to, a handheld wireless communication device (e.g., a mobile phone, a smart phone, a phablet device, and so on), a wearable computer device (e.g., a head-mounted display computer device, a head-mounted camera device, a wristwatch computer device, and so on), a Global Positioning System (GPS) device, a laptop computer, a tablet computer, or another type of portable computer, a media playing

device, a portable gaming system, and/or any other type of computer device with wireless communication capabilities, and the like. In an embodiment, the user equipment (104) includes, but is not limited to, any electrical, electronic, electro-mechanical, or an equipment, or a combination of one or more of the above devices such as virtual reality (VR) devices, augmented reality (AR) devices, laptop, a general-purpose computer, desktop, personal digital assistant, tablet computer, mainframe computer, or any other computing device, wherein the user equipment (104) may include one or more in-built or externally coupled accessories including, but not limited to, a visual aid device such as a camera, an audio aid, a microphone, a keyboard, and input devices for receiving input from the user (102), or the entity (110) such as touch pad, touch enabled screen, electronic pen, and the like. A person of ordinary skill in the art will appreciate that the user equipment (104) may not be restricted to the mentioned devices and various other devices may be used.

**[0055]** Referring to FIG. 1, the user equipment (104) communicates with a system (108) through a network (106). In an embodiment, the network (106) includes at least one of a Fifth Generation (5G) network, 6G network, or the like. The network (106) enables the user equipment (104) to communicate with other devices in the network architecture (100) and/or with the system (108). The network (106) includes a wireless card or some other transceiver connection to facilitate this communication. In another embodiment, the network (106) is implemented as, or include any of a variety of different communication technologies such as a wide area network (WAN), a local area network (LAN), a wireless network, a mobile network, a Virtual Private Network (VPN), the Internet, the Public Switched Telephone Network (PSTN), or the like.

**[0056]** In another exemplary embodiment, the centralized server (112) includes or comprises, by way of example but not limitation, one or more of a stand-alone server, a server blade, a server rack, a bank of servers, a server farm, hardware supporting a part of a cloud service or system, a home server, hardware running a virtualized server, one or more processors executing code to function as a server, one or more machines performing server-side functionality as described herein, at

least a portion of any of the above, some combination thereof.

[0057] In an embodiment, the network (106) is further configured with the centralized server (112) including a database, where all output is stored as part of the operational records. It can be retrieved whenever there is a need to reference  
5 this output in the future.

[0058] In an embodiment, the computing device (102) associated with one or more users (110) may transmit the at least one captured data packet over a point-to-point or point-to-multipoint communication channel or network (106) to the system (102).

10 [0059] In an embodiment, the computing device (102) may involve collection, analysis, and sharing of data received from the system (108) via the network (106).

[0060] In an embodiment, the system (108) may determine that the database or the application is not able to process the data successfully, and the system may  
15 pause the loading at the database or the application. In an example, the system may pause sending the data to the database or the application. Once the system determines that the loading on the application or the database is reduced, the system may start sending the data to the database or the application.

[0061] Although FIG. 1 shows exemplary components of the network  
20 architecture (100), in other embodiments, the network architecture (100) may include fewer components, different components, differently arranged components, or additional functional components than depicted in FIG. 1. Additionally, or alternatively, one or more components of the network architecture (100) may perform functions described as being performed by one or more other components  
25 of the network architecture (100).

[0062] FIG. 2 illustrates an exemplary block diagram (200) of the system (108), in accordance with an embodiment of the present disclosure.

**[0063]** In an aspect, the system (108) may include one or more processor(s) (202). The one or more processor(s) (202) may be implemented as one or more microprocessors, microcomputers, microcontrollers, edge or fog microcontrollers, digital signal processors, central processing units, logic circuitries, and/or any devices that process data based on operational instructions. Among other capabilities, one or more processor(s) (202) may be configured to fetch and execute computer-readable instructions stored in a memory (204) of the system (108). The memory (204) may be configured to store one or more computer-readable instructions or routines in a non-transitory computer-readable storage medium, which may be fetched and executed to create or share data packets over a network service. The memory (204) may comprise any non-transitory storage device including, for example, volatile memory such as Random-Access Memory (RAM), or non-volatile memory such as Erasable Programmable Read-Only Memory (EPROM), flash memory, and the like.

**[0064]** In an embodiment, the system (108) may include an interface(s) (206). The interface(s) (206) may include a variety of interfaces, for example, interfaces for data input and output devices, referred to as I/O devices, storage devices, and the like. The interface(s) (206) may facilitate communication of the system (108). The interface(s) (206) may also provide a communication pathway for one or more components of the system (108). Examples of such components include, but are not limited to, processing unit/engine(s) (208) and a database (210).

**[0065]** The processing unit/engine(s) (208) may be implemented as a combination of hardware and programming (for example, programmable instructions) to implement one or more functionalities of the processing engine(s) (208). In examples described herein, such combinations of hardware and programming may be implemented in several different ways. For example, the programming for the processing engine(s) (208) may be processor-executable instructions stored on a non-transitory machine-readable storage medium and the hardware for the processing engine(s) (208) may comprise a processing resource (for example, one or more processors), to execute such instructions. In the present

examples, the machine-readable storage medium may store instructions that, when executed by the processing resource, implement the processing engine(s) (208). In such examples, the system (108) may comprise the machine-readable storage medium storing the instructions and the processing resource to execute the instructions, or the machine-readable storage medium may be separate but accessible to the system (108) and the processing resource. In other examples, the processing engine(s) (208) may be implemented by an electronic circuitry.

**[0066]** In an embodiment, the processing engine (208) may include one or more engine(s) selected from a merging engine (212) and other engine(s) (214).

10 **[0067]** The merging engine (212) is configured to identify the existence of a configuration file. If the required configuration file does not exist, the merging engine (212) creates a new configuration file by merging an old configuration file and a configuration delta file present in the configuration repository/folder (220). For example, if an old configuration file contains parameters for version 1.0 and a delta file contains updates for version 1.1, the merging engine (212) merges these to create a new configuration file for version 1.1.

**[0068]** The alarm file module (218) is configured to identify the existence of an alarm file. If the alarm file does not exist, the alarm file module (218) creates a new alarm file by merging an old alarm file and an alarm delta file present in the configuration repository/folder (220). For instance, if an old alarm file specifies alarms for certain thresholds and the alarm delta file contains new or modified alarm thresholds, the alarm file module (218) merges these to create an updated alarm file.

**[0069]** The configuration repository/folder (220) is a designated storage location where old configuration files, delta files, old alarm files, and alarm delta files are kept. This folder serves as a repository for all necessary files required for the configuration and alarm merging process. The merging engine (212) and the alarm file module (218) access this repository/folder to fetch the relevant files and perform the merging operations.

[0070] In an embodiment, the merging engine (212) is further configured to read environment variables and populate them in the configuration file. In an embodiment, the environment variables may include information related to a directory/path where a run-time environment is installed. In an embodiment, the environment variables may be accessed by the network functions/applications running in a docker/K8/virtualized environment.

[0071] This ensures that dynamic parameters that may change based on the environment in which the application is running are correctly included in the configuration file. The delta file captures new, modified, or removed parameters associated with the configuration file, while the alarm delta file captures new, modified, or removed parameters associated with the alarm file. The merging engine (212) is configured to run before the application is booted, ensuring that all configurations are up-to-date and accurate before the application starts.

[0072] The merging engine (212), thus, is implemented to automate the configuration merging process during the upgradation of network functions. During this process, the merging engine (212) takes delta files, which contain new, modified, or removed parameters and alarms, and merges them with the existing parameters and alarms from an old release stored in the configuration files. This ensures that the new release is configured correctly without the need for manual intervention. For example, if a network function is being upgraded from version 1.0 to version 2.0, the delta file will include only the changes necessary for the new version. The merging engine (212) will automatically incorporate these changes into the existing configuration, creating a new configuration file that is ready for deployment.

[0073] The other engines (214) in the processing engine (208) may perform additional tasks to support the operation and management of the network functions. These could include engines for monitoring performance, managing resources, handling security protocols, and ensuring compliance with regulatory standards. Each of these engines works in conjunction with the merging engine (212) to

provide a comprehensive solution for network function upgrades and management.

**[0074]** The database (210) is a component of the system (108) that stores data either provided by the user, generated by the functionalities of the processors (202) or processing engines (208), or resulting from the system (102) operations.

5 This database (210) may include historical configuration files, delta files, alarm logs, performance metrics, and other relevant data. For example, it can store previous configuration versions and their corresponding delta files, allowing the system (102) to track changes over time and ensure consistency across updates. Additionally, the database (210) can store alarm data, which helps in monitoring  
10 and troubleshooting network functions by providing a record of past issues and their resolutions.

**[0075]** By leveraging the database (210), the system (102) can ensure that all necessary data is available for the configuration merging process, facilitating a seamless upgrade and reducing the likelihood of errors. The combination of the  
15 merging engine (212), other engines (214), and the database (210) creates a robust framework for managing the complexities of network function upgrades in a highly automated and efficient manner.

**[0076]** Although FIG. 2 shows an exemplary block diagram (200) of the system (108), in other embodiments, the system (108) may include fewer  
20 components, different components, differently arranged components, or additional functional components than depicted in FIG. 2. Additionally, or alternatively, one or more components of the system (108) may perform functions described as being performed by one or more other components of the system (108).

**[0077]** FIG. 3 illustrates an exemplary flow chart implementing a method  
25 (300) for merging configurations and upgrading network functions, in accordance with an embodiment of the present disclosure.

**[0078]** Referring to FIG. 3, the method (300) includes providing a configuration tool in an application and running the configuration tool before the

application is booted. Inside the configuration tool, at step 302, the method (300) starts by determining whether a required configuration file corresponding to a new release of a network function of the application exists or not. If the required configuration file exists, the method (300) includes checking for the existence of an alarm file (at step 306). If the required configuration file does not exist, the method creates a new configuration file using an old configuration file and a configuration delta file present in a configuration folder (at step 304). For example, if the current version of the application (e.g., version 1.0) is released with one or more new network functions in the version 2.0, the method (300) determines that the required configuration file corresponding to the new release of the one or more network functions of the application does not exist. As a result, in an embodiment, the method (300) creates a new configuration file using an old configuration file (e.g., the configuration file of version 1.0) and a configuration delta file present in a configuration folder (in step 304). After creating the new configuration file, the method (300) then checks for the alarm file (in step 306).

**[0079]** If the alarm file exists (in step 306), the method (300) proceeds to boot the application. If the alarm file does not exist, the method (300) creates a new alarm file using the previous alarm file and the alarm delta file present in the configuration folder (in step 308). After creating the new alarm file, the method (300) proceeds to boot the application. This process ensures that all necessary configurations and alarms are updated correctly before the application starts, reducing the likelihood of errors and ensuring optimal performance.

**[0080]** FIG. 4 illustrates an exemplary architecture (400) of the system (108), in accordance with an embodiment of the present disclosure.

**[0081]** Referring to FIG. 4, a network function (404) may be implemented by the system (108), for example, the configuration tool (406) that simplifies the configuration upgrade procedures. Using the configuration tool (406), a need for filling the configuration file from a scratch may be avoided. Instead, a delta file that captures newly added parameters may be provided, and only the values of the newly

added parameters may be provided in a delta sheet unless it is being removed. The old configuration file and the delta file may be stored in a configuration repository/folder (402). On the first run of an upgraded release (e.g., version 2.0), a new configuration file may be prepared using the old configuration file (e.g.,  
5 version 1.0) and the delta file. By preparing the new configuration file using the old configuration file and the delta file, the chances of errors may be reduced during the upgrade phase.

**[0082]** FIG. 5 illustrates an exemplary computer system (500) in which or with which embodiments of the present disclosure may be implemented. As shown  
10 in FIG. 5, the computer system (500) may include an external storage device (510), a bus (520), a main memory (530), a read only memory (540), a mass storage device (550), a communication port (560), and a processor (570). A person skilled in the art will appreciate that the computer system (500) may include more than one processor (570) and communication ports (560). Processor (570) may include  
15 various modules associated with embodiments of the present disclosure.

**[0083]** In an embodiment, the communication port (560) may be any of an RS-232 port for use with a modem-based dialup connection, a 10/100 Ethernet port, a Gigabit or 10 Gigabit port using copper or fiber, a serial port, a parallel port, or other existing or future ports. The communication port (560) may be chosen  
20 depending on a network, such a Local Area Network (LAN), Wide Area Network (WAN), or any network to which the computer system (500) connects.

**[0084]** In an embodiment, the main memory (530) may be Random Access Memory (RAM), or any other dynamic storage device commonly known in the art. Read-only memory (540) may be any static storage device(s) e.g., but not limited to, a Programmable Read Only Memory (PROM) chips for storing static  
25 information e.g., start-up or Basic Input/Output System (BIOS) instructions for the processor (570).

**[0085]** In an embodiment, the mass storage (550) may be any current or future mass storage solution, which may be used to store information and/or

instructions. Exemplary mass storage solutions include, but are not limited to, Parallel Advanced Technology Attachment (PATA) or Serial Advanced Technology Attachment (SATA) hard disk drives or solid-state drives (internal or external, e.g., having Universal Serial Bus (USB) and/or Firewire interfaces), one  
5 or more optical discs, Redundant Array of Independent Disks (RAID) storage, e.g., an array of disks (e.g., SATA arrays).

**[0086]** In an embodiment, the bus (520) communicatively couples the processor(s) (570) with the other memory, storage and communication blocks. The bus (520) may be, e.g., a Peripheral Component Interconnect (PCI)/PCI Extended  
10 (PCI-X) bus, Small Computer System Interface (SCSI), Universal Serial Bus (USB) or the like, for connecting expansion cards, drives and other subsystems as well as other buses, such a front side bus (FSB), which connects the processor (570) to the computer system (500).

**[0087]** Optionally, operator and administrative interfaces, e.g., a display,  
15 keyboard, joystick, and a cursor control device, may also be coupled to the bus (520) to support direct operator interaction with the computer system (500). Other operator and administrative interfaces may be provided through network connections connected through the communication port (560). Components described above are meant only to exemplify various possibilities. In no way should  
20 the aforementioned exemplary computer system (500) limit the scope of the present disclosure.

**[0088]** FIG. 6 illustrates an exemplary flowchart depicting a method (600) for configuring an application in the network (106), in accordance with an embodiment of the present disclosure. In an embodiment, the application may  
25 include, but is not limited to, a video streaming application, a voice-over long-term evolution (VoLTE) application, a network slicing management application, etc. In an embodiment, the network (106) includes at least one of a Fifth Generation (5G) network, 6G network, or the like.

**[0089]** At step (602), the merging engine (212) identifies an existence of a

required configuration file corresponding to a new release of a network function of the application. In an embodiment, before the application is booted by the system (200), the merging engine (212) may communicate with the configuration tool (216) to check for the availability of the required/new configuration file  
5 corresponding to the new release of the network function of the application (e.g., version 2.0). In an embodiment, the required/new configuration file may be available/stored in the configuration repository/folder (220) or in some other repository/folder not shown in Fig. 2. In another embodiment, the merging engine (212) may directly check (i.e., without communicating with the configuration tool  
10 (216)) for the availability of the required configuration file corresponding to the new release of the network function of the application. In an embodiment, the required configuration file may include one or more new/updated parameters (e.g., transactions per second, etc.). In another embodiment, the required configuration file may include one or more parameters to be deleted corresponding to the new  
15 release of the network function of the application. In an embodiment, the network function may include, but is not limited to, policy control function (PCF), binding support function (BSF), network repository function (NRF), charging function (CHF), etc. These network functions rely on the configuration files to manage operational parameters. In an embodiment, these configuration files need to be  
20 modified at every version upgrade of the network function or the application to include new parameters. The configuration file which includes the one or more new parameters is called as the required/new configuration file.

**[0090]** At step (604), if the required configuration file does not exist, the merging engine (212) creates a new configuration file by merging an old  
25 configuration file and a delta file available in a configuration repository (220). For example, if the current version of the application (e.g., version 1.0) is released with one or more new network functions in the version 2.0, the method determines that the required configuration file corresponding to the new release of the one or more network functions of the application does not exist. As a result, in an embodiment,  
30 the method creates a new configuration file using an old configuration file (e.g., the

configuration file of version 1.0) and a configuration delta file present in a configuration folder (304).

**[0091]** At step (606), the alarm file module (218) identifies existence of a required alarm file corresponding to the new release of the network function of the application. In an embodiment, before the application is booted by the system (200), the alarm file module (218) checks for the availability of the required/new alarm file corresponding to the new release of the network function of the application (e.g., version 2.0). In an embodiment, the required/new alarm file may be available/stored in the configuration repository/folder (220) or in some other repository/folder not shown in Fig. 2. In an embodiment, the required alarm file may include one or more new/updated parameters/settings (e.g., host unavailable, database (DB) down, threshold values, etc.). In an embodiment, the network functions also rely on the alarm files to raise alarms when specific one or more conditions are met. For example, the alarm file allows generating alerts for various conditions of the network functions, such as performance degradation, failures, threshold breaches, etc. In an embodiment, the alarm files also need to be modified, similar to the configuration files, at every version upgrade of the network function or the application to include new parameters/settings.

**[0092]** At step (608), if the alarm file does not exist before the application is booted by the system (200), the alarm file module (218) creates a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository (220). In an embodiment, for example, if the old alarm file specifies alarms for certain thresholds and the alarm delta file contains new or modified alarm thresholds, the alarm file module (218) merges the old alarm file and the alarm delta file to create the new/updated alarm file.

**[0093]** At step (610), the processing engine (208) configures the application with the newly created configuration and alarm files. In an embodiment, once the new configuration file and new alarm file are created in steps 304 and 308 respectively, the processing engine (208) of the system (200) boots/loads the

application with the newly created configuration file and alarm file. In an embodiment, the newly created or updated configuration file and alarm file may be stored in the configuration repository/folder (220) so that the application can use them while booting. In an embodiment, the newly created or updated configuration  
5 file may include new configuration parameters (e.g., transactions per second, etc.) and the newly created or updated alarm file may include new alarms (e.g., host unavailable, database (DB) down, threshold values, etc.) specific to the new version of the network functions of the application.

**[0094]** In an embodiment, the method (600) allows the merging engine  
10 (212) to read environment variables and populate them in the new configuration file. For example, once the old configuration file is merged with the delta file available in a configuration repository (220) to create the new configuration file, the merging engine (212) may also write/update the environment variables (e.g., directory/path where the run-time environment is stored) in the new configuration  
15 file itself. This ensures that the application only reads from the merged/consolidated configuration file when starting/booting in the docker/K8/virtualized environment.

**[0095]** In an embodiment, the method (600) further provides the configuration delta file that includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the  
20 configuration file. In an embodiment, the parameters of the configuration delta file may be added, modified, or removed by the user.

**[0096]** In an embodiment, the method (600) further provides the alarm delta file that includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the alarm file. In an embodiment,  
25 the parameters of the alarm delta file may be added, modified, or removed by the user.

**[0097]** In an embodiment, the method (600) further allows the configured merging engine (212) to run before the application is booted to ensure that the application is booted/loaded with the newly created or updated configuration file to

reflect the upgraded network functions of the application.

[0098] In an exemplary embodiment, the present disclosure discloses a system (108) for configuring an application in the network (106). The system (108) that comprises a merging engine (212) configured to: identify the existence of a  
5 required configuration file corresponding to a new release of a network function of the application and create a new configuration file by merging an old configuration file and a configuration delta file available in a configuration repository if the required configuration file does not exist. The system (108) comprises an alarm file  
10 module (218) configured to: identify the existence of a required alarm file corresponding to the new release of the network function of the application and create a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository if the required alarm file does not exist. The system (108) further comprises a processing engine (208), operatively connected to the configuration tool (216), merging engine (212), and the alarm file module (218), to  
15 configure the application with the newly created configuration and alarm files.

[0099] In an exemplary embodiment, the present disclosure discloses a user equipment (UE) (104) communicatively coupled with a network (106). The coupling comprises steps of receiving, by the network (106), a connection request from the UE (104), sending, by the network (106), an acknowledgment of the  
20 connection request to the UE (104) and transmitting a plurality of signals in response to the connection request. Configuring an application in the network is performed by a system (108) that comprises a merging engine (212) configured to: identify the existence of a required configuration file corresponding to a new release of a network function of the application and create a new configuration file by  
25 merging an old configuration file and a configuration delta file available in a configuration repository if the required configuration file does not exist. The system (108) comprises an alarm file module (218) configured to: identify the existence of a required alarm file corresponding to the new release of the network function of the application and create a new alarm file by merging an old alarm file and an  
30 alarm delta file available in the configuration repository if the required alarm file

does not exist. The system (108) further comprises a processing engine (208), operatively connected to the configuration tool (216), merging engine (212), and the alarm file module (218), to configure the application with the newly created configuration and alarm files.

- 5    **[00100]**       The present disclosure provides technical advancement related to configuring/upgrading the application in the network. This advancement addresses the limitations of existing solutions that do not use the delta files in the automated configuration merging process during the upgradation of the application. The disclosure provides automating the configuration merging through the delta files  
10   for upgrading the network functions of the application, which offers significant improvements in reducing the risk of outages, minimizing errors, and faster deployment with minimal human intervention.

- [00101]**       While the foregoing describes various embodiments of the present disclosure, other and further embodiments of the present disclosure may be devised  
15   without departing from the basic scope thereof. The scope of the present disclosure is determined by the claims that follow. The present disclosure is not limited to the described embodiments, versions or examples, which are included to enable a person having ordinary skill in the art to make and use the present disclosure when combined with information and knowledge available to the person having ordinary  
20   skill in the art.

#### **ADVANTAGES OF THE PRESENT DISCLOSURE**

- [00102]**       The present disclosure provides a system and a method to merge configurations and upgrade network functions automatically.

- [00103]**       The present disclosure automates the merging of new parameters and  
25   alarms with old parameters and alarms in an old release through additionally provided delta files.

- [00104]**       The present disclosure removes and modifies existing parameters

and alarms of the network functions.

**[00105]** The present disclosure ensures accurate configuration upgrade of the network functions.

5 **[00106]** The present disclosure reduces the chances of errors during an upgrade phase and ensures that the application's performance and metrics are not affected. Thereby, preventing network downtime.

**[00107]** The present disclosure performs upgradation of network functions in seconds and reduces deployment of the system from a manual half-day process to a task that requires minimal human intervention.

10 **[00108]** The present disclosure promotes smooth process flows and consistency in reducing cycle time, resources, and improves productivity by performing automation of merging of the new parameters with the old parameters.

**[00109]** The present disclosure achieves time-saving, improved efficiency, reduced risk of outages, minimal errors, faster deployment, and consistency.

## CLAIMS:

1. A method (600) for configuring an application in a network (106), the method (600) comprising:
  - identifying (602), by a merging engine (212), an existence of a required configuration file corresponding to a new release of a network function of the application;
  - creating (604), by the merging engine (212), if the required configuration file does not exist, a new configuration file by merging an old configuration file and a configuration delta file available in a configuration repository (220);
  - identifying (606), by an alarm file module (218), the existence of a required alarm file corresponding to the new release of the network function of the application;
  - creating (608), by the alarm file module (218), if the required alarm file does not exist, a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository (220);
  - configuring (610) the application, by a processing engine (208), with the new created configuration and alarm files.
2. The method (600) as claimed in claim 1, wherein the merging engine (212) reads environment variables and populates them in the new configuration file.
3. The method (600) as claimed in claim 1, wherein the configuration delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the configuration file.
4. The method (600) as claimed in claim 1, wherein the alarm delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the alarm file.

5. The method (600) as claimed in claim 1, wherein the merging engine (212) is configured to run before the application is booted.

6. A system (108) for configuring an application in a network (106), the system  
5 (108) comprising:

a merging engine (212) configured to:

identify an existence of a required configuration file corresponding  
to a new release of a network function of the application, and

create a new configuration file by merging an old configuration file  
10 and a configuration delta file available in a configuration repository if the  
required configuration file does not exist;

an alarm file module (218) configured to:

identify the existence of a required alarm file corresponding  
to the new release of the network function of the application;

15 create a new alarm file by merging an old alarm file and an  
alarm delta file available in the configuration repository if the  
required alarm file does not exist; and

a processing engine (208), operatively connected to a configuration  
tool (216), the merging engine (212), and the alarm file module (218), to  
20 configure the application with the new created configuration and alarm files.

7. The system (108) as claimed in claim 6, wherein the merging engine (212)  
is further configured to read environment variables and populate them in the new  
configuration file.

25

8. The system (108) as claimed in claim 6, wherein the configuration delta file  
includes at least one or more parameters from a group comprising of new, modified,  
or removed parameters associated with the configuration file.

9. The system (108) as claimed in claim 6, wherein the alarm delta file includes at least one or more parameters from a group comprising of new, modified, or removed parameters associated with the alarm file.

5 10. The system (108) as claimed in claim 6, wherein the merging engine (212) is configured to run before the application is booted.

11. A user equipment (UE) (104) communicatively coupled with a network (106), the coupling comprises steps of:

10 receiving, by the network (106), a connection request from the UE (104);  
sending, by the network (106), an acknowledgment of the connection request to the UE (104); and  
transmitting a plurality of signals in response to the connection request, wherein configuring an application in the network (106) is  
15 performed by a system (108) as claimed in claim 6.

12. A computer program product comprising a non-transitory computer-readable medium comprising instructions that, when executed by one or more processors, cause the one or more processors to perform a method (600) for  
20 configuring an application in a network (106), the method (600) comprising:

identifying (602), by a merging engine (212), an existence of a required configuration file corresponding to a new release of a network function of the application;

25 creating (604), by the merging engine (212), if the required configuration file does not exist, a new configuration file by merging an old configuration file and a configuration delta file available in a configuration repository (220);

30 identifying (606), by an alarm file module (218), the existence of a required alarm file corresponding to the new release of the network function of the application;

creating (608), by the alarm file module (218), if the required alarm file does not exist, a new alarm file by merging an old alarm file and an alarm delta file available in the configuration repository (220); and

5           configuring (610) the application, by a processing engine (208), with the new created configuration and alarm files.

1/6

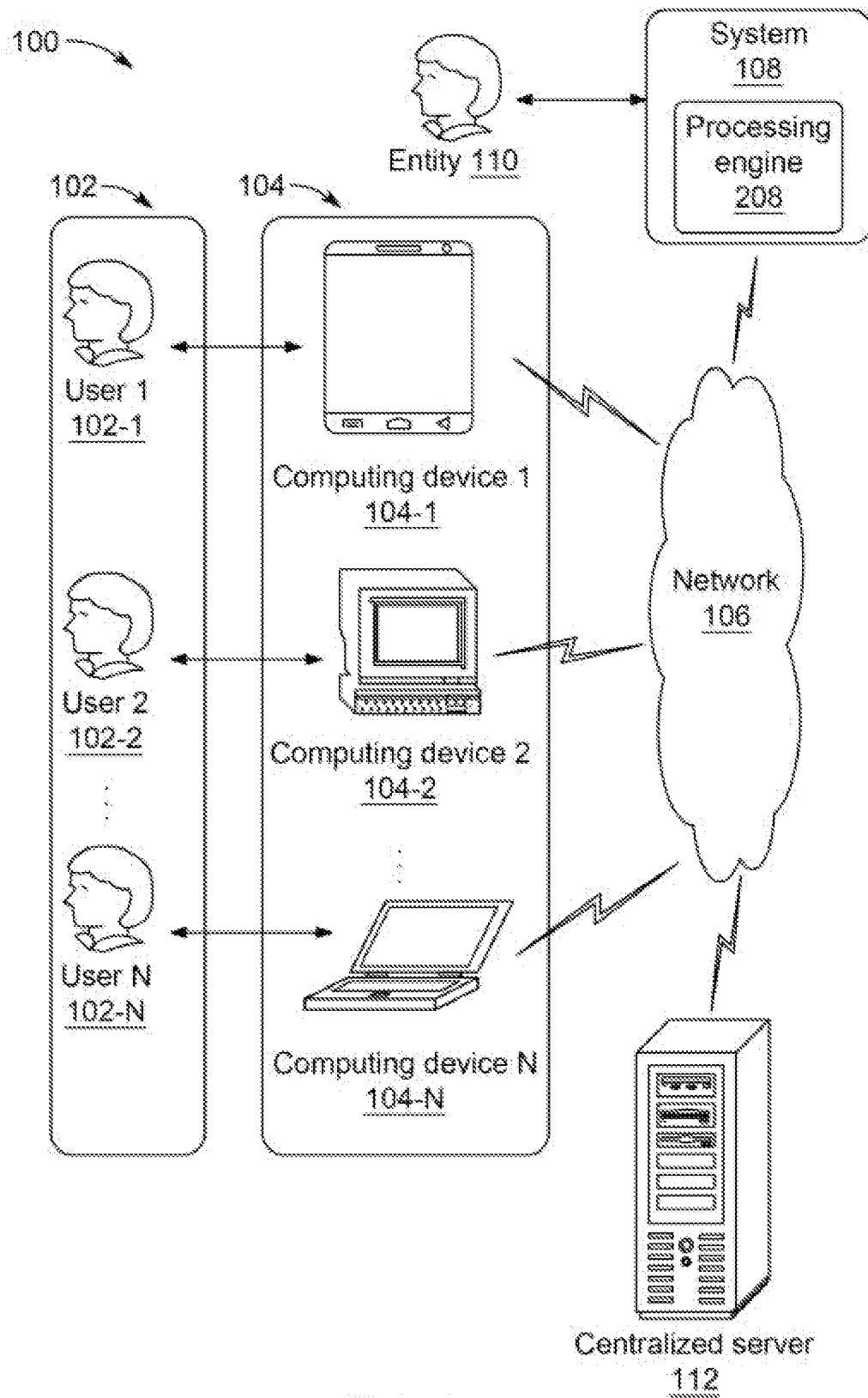


FIG. 1

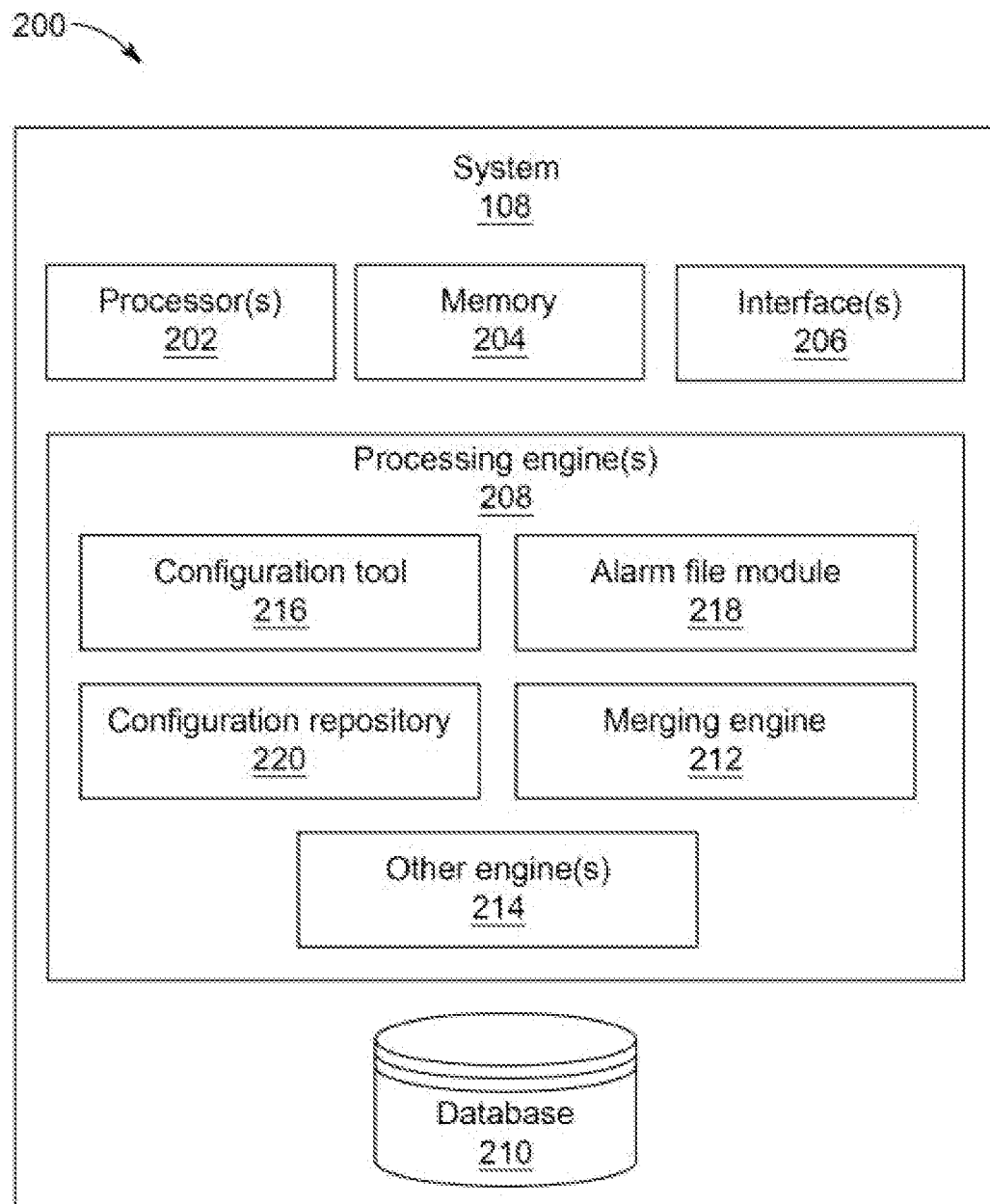


FIG. 2

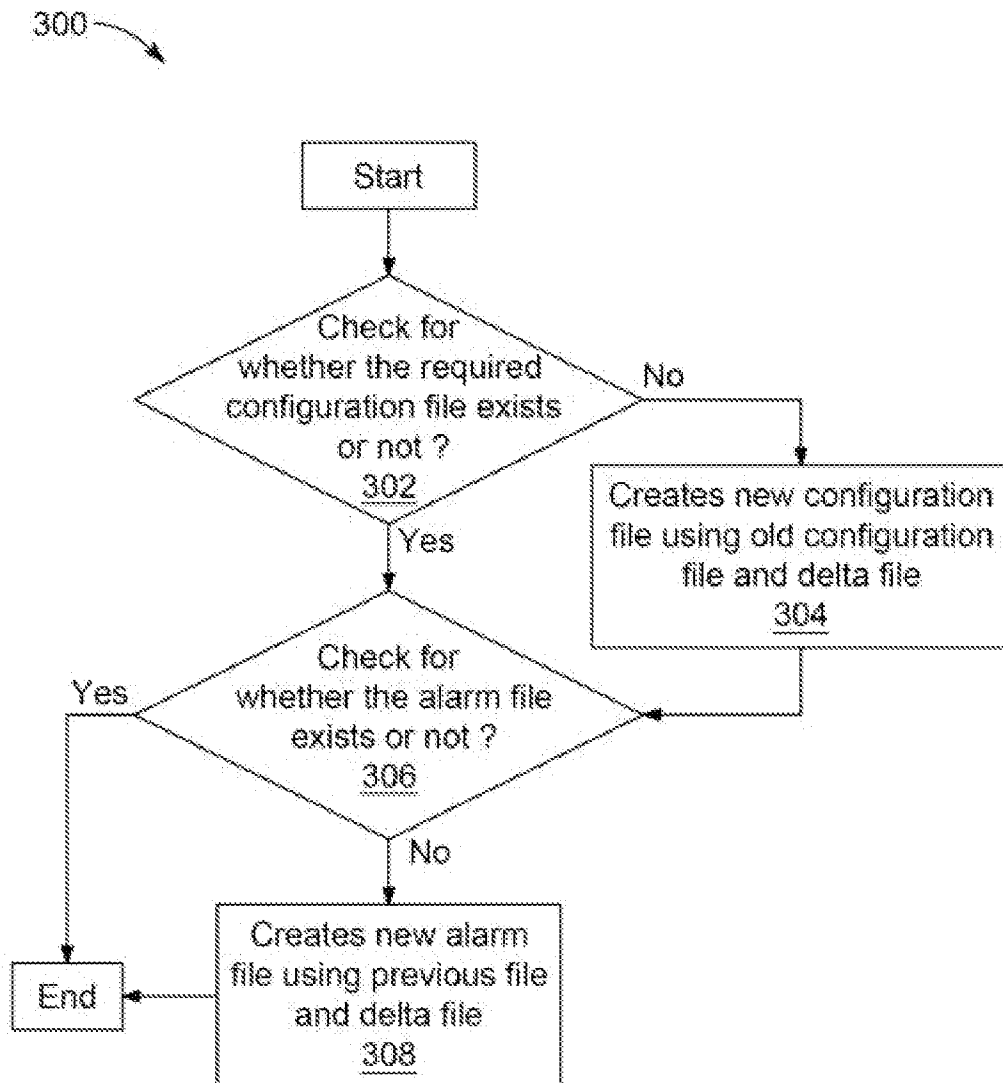


FIG. 3

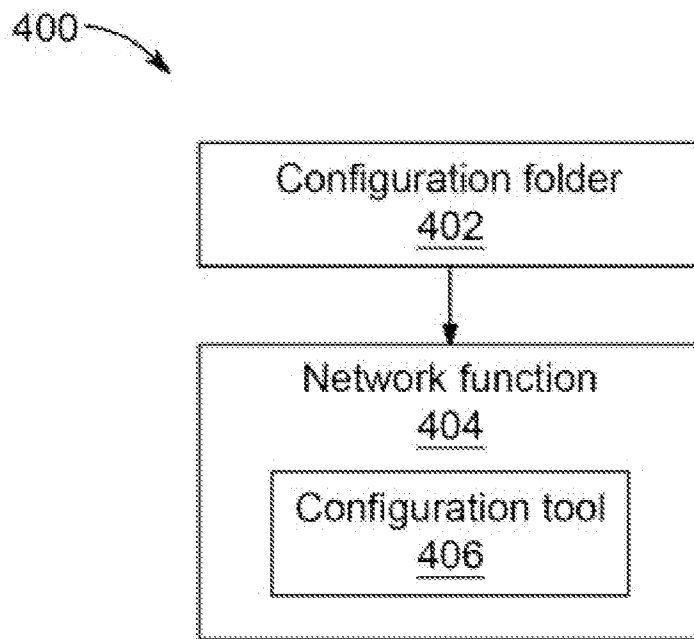


FIG. 4

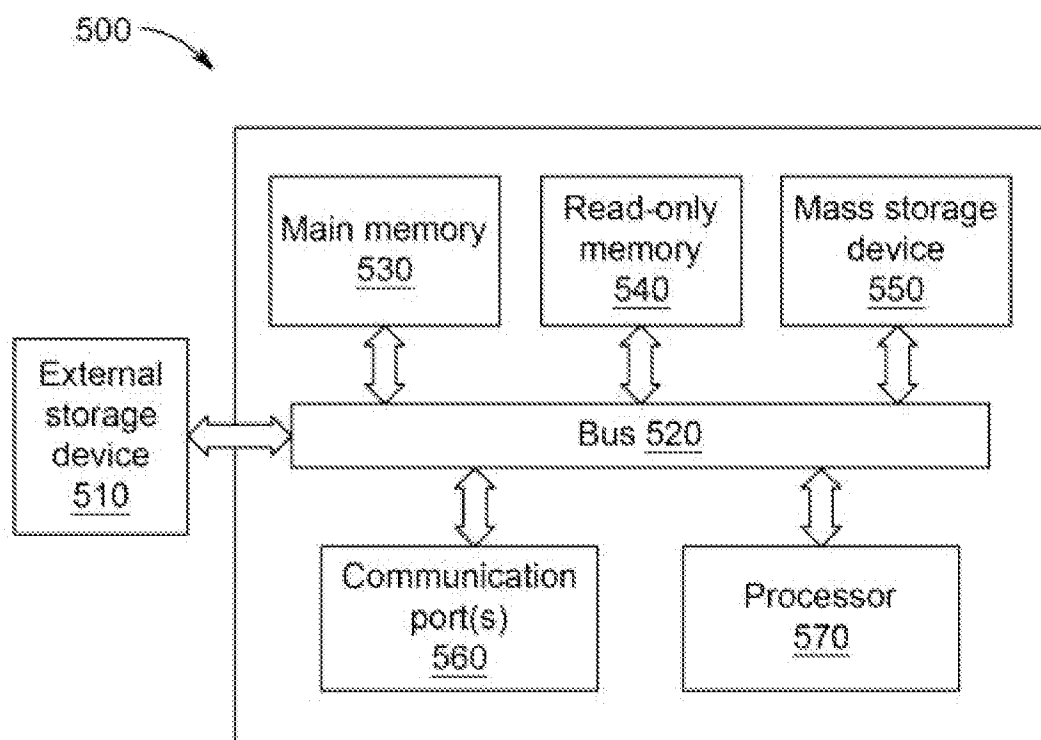


FIG. 5

6/6

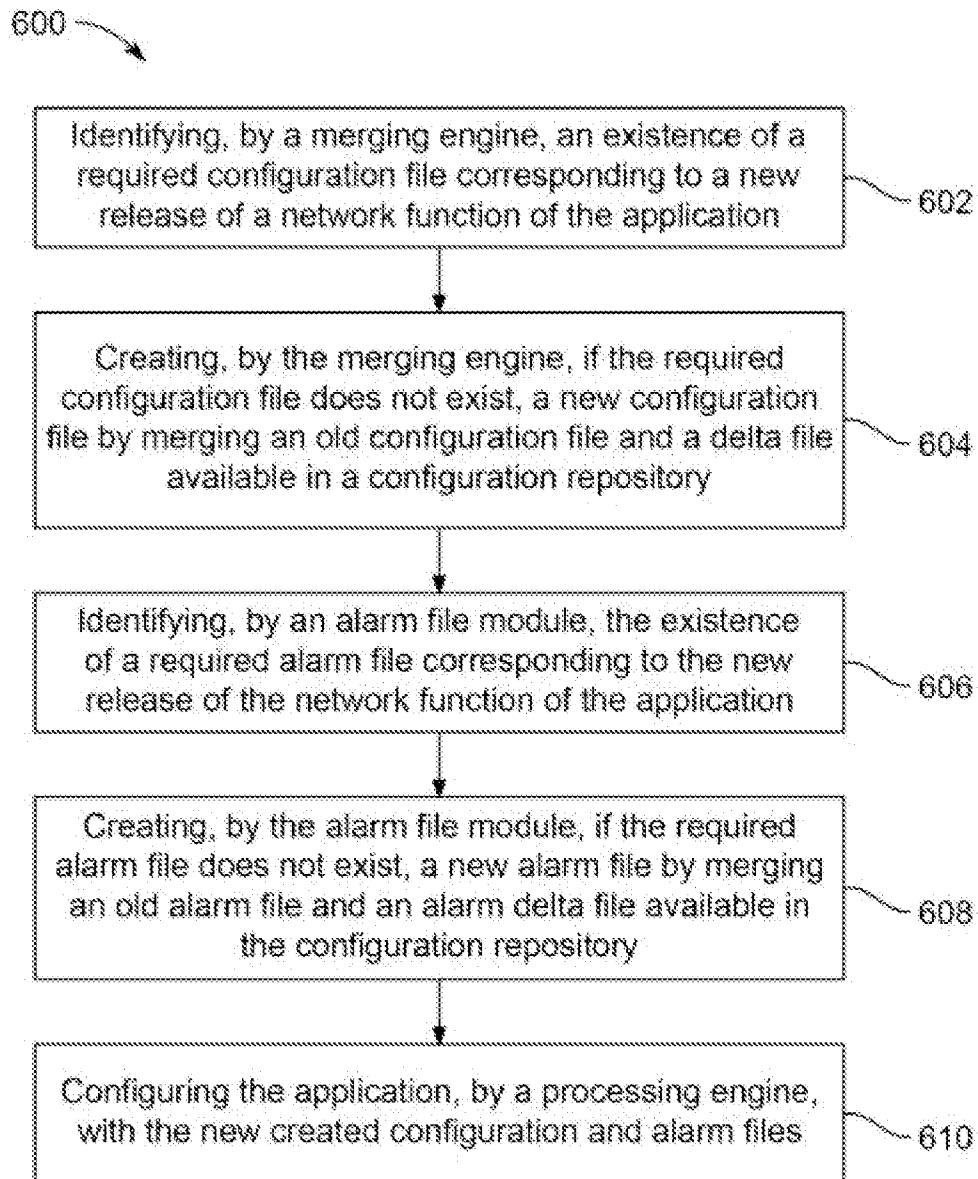


FIG. 6

## INTERNATIONAL SEARCH REPORT

International application No.

PCT/IN2024/051233

## A. CLASSIFICATION OF SUBJECT MATTER

H04L41/084, H04L41/082, G06F8/70 Version=2024.01

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, G06F

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

Electronic database consulted during the international search (name of database and, where practicable, search terms used)

PatSeer, IPO Internal Database, Google Patents

Keywords - merge, configuration file, network function, alarm file, delta

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages              | Relevant to claim No. |
|-----------|-------------------------------------------------------------------------------------------------|-----------------------|
| Y         | US20210119871A1 (CITRIX SYSTEMS INC) 22-APR-2021<br>(22-04-2021) Para. [0037-0041], [0133-0137] | 1-12                  |
| Y         | US20120096063A1 (PATTINGTON IP LLC) 19-APR-2012<br>(19-04-2012) Para. [0054-0059], [0071-0076]  | 1-12                  |



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

"I" 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

"&amp;" document member of the same patent family

Date of the actual completion of the international search

11-11-2024

Date of mailing of the international search report

11-11-2024

Name and mailing address of the ISA/

Indian Patent Office

Plot No.32, Sector 14, Dwarka, New Delhi-110075

Facsimile No.

Authorized officer

Saurabh Saxena

Telephone No. +91-1125300200

INTERNATIONAL SEARCH REPORT  
Information on patent family members

International application No.  
PCT/IN2024/051233

| Citation          | Pub.Date   | Family           | Pub.Date   |
|-------------------|------------|------------------|------------|
| US 20210119871 A1 | 22-04-2021 | WO 2021080963 A1 | 29-04-2021 |
| US 20120096063 A1 | 19-04-2012 | WO 2011003261 A1 | 13-01-2011 |
|                   |            | CN 101610176 A   | 23-12-2009 |
|                   |            | EP 2445140 A1    | 25-04-2012 |
|                   |            | AU 2009349406 A1 | 23-02-2012 |
|                   |            | DK 2445140 T3    | 06-01-2014 |