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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 September 2016 (29.09.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/153481 A1

(51) International Patent Classification:
G06F 9/44 (2006.01)

(21) International Application Number:
PCT/US2015/022111

(22) International Filing Date:
24 March 2015 (24.03.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventor: **HAMILL, Hugh**; Ballybrit Business Park, Galway, 0000 (IE).

(74) Agents: **SORENSEN, C. Blake** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, 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, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, 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 the identity of the inventor (Rule 4.17(i))*

— *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))*

WO 2016/153481 A1

(54) Title: CONTEXT-BASED FEATURE MANAGEMENT

(57) Abstract: Examples disclosed herein comprise feature management instructions to create a feature map of a plurality of available features for an application, wherein each of the plurality of available features is associated with at least one context state, identify a current context state associated with the application at a feature option point, determine whether at least one of the plurality of available features is associated with the current context state, and in response to determining that at least one of the plurality of available features is associated with the current context state, execute the at least one of the plurality of available features.

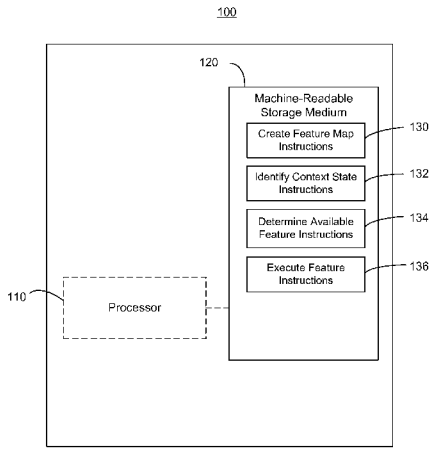

FIG. 1

CONTEXT-BASED FEATURE MANAGEMENT

BACKGROUND

[0001] Feature management in an application allows for different functionality to be enabled and disabled in an application. This allows for the application, for example, to support different data architectures, to perform tests, and/or disable broken and/or incomplete features without necessarily needing to modify the application's user interface.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] In the accompanying drawings, like numerals refer to like components or blocks. The following detailed description references the drawings, wherein:

[0003] FIG. 1 is a block diagram of an example feature management device consistent with disclosed implementations;

[0004] FIG. 2 is a flowchart of an embodiment of a method for feature management consistent with disclosed implementations; and

[0005] FIG. 3 is a block diagram of a system for feature management consistent with disclosed implementations.

DETAILED DESCRIPTION

[0006] As described above, feature management enables an application developer to control which functional features are performed at different points in the application's execution. These feature option points may allow for different behaviors based on a current context for the application, the application's environment, and/or a user of the application.

[0007] In the description that follows, reference is made to the term, "machine-readable storage medium." As used herein, the term "machine-readable storage medium" refers to any electronic, magnetic, optical, or other physical storage device that stores executable instructions or other data (e.g., a hard disk drive, random access memory, flash memory, etc.).

[0008] Referring now to the drawings, FIG. 1 is a block diagram of an example feature management device 100 consistent with disclosed implementations. Feature management device 100 may comprise a processor 110 and a non-transitory machine-readable storage medium 120. Feature management device 100 may comprise a computing device such as a server computer, a desktop computer, a laptop computer, a handheld computing device, a mobile phone, or the like.

[0009] Processor 110 may comprise a central processing unit (CPU), a semiconductor-based microprocessor, or any other hardware device suitable for retrieval and execution of instructions stored in machine-readable storage medium 120. In particular, processor 110 may fetch, decode, and execute a plurality of create feature map instructions 130, identify context state instructions 132, determine available feature instructions 134 and execute feature instructions 136 to implement the functionality described in detail below.

[0010] Executable instructions such as create feature map instructions 130, identify context state instructions 132, determine available feature instructions 134 and execute feature instructions 136 may be stored in any portion and/or component of machine-readable storage medium 120. The machine-readable storage medium 120 may comprise both volatile and/or nonvolatile memory and data storage components. Volatile components are those that do not retain data values upon loss of power. Nonvolatile components are those that retain data upon a loss of power.

[0011] The machine-readable storage medium 120 may comprise, for example, random access memory (RAM), read-only memory (ROM), hard disk drives, solid-state drives, USB flash drives, memory cards accessed via a memory card reader, floppy disks accessed via an associated floppy disk drive, optical discs accessed

via an optical disc drive, magnetic tapes accessed via an appropriate tape drive, and/or other memory components, and/or a combination of any two and/or more of these memory components. In addition, the RAM may comprise, for example, static random access memory (SRAM), dynamic random access memory (DRAM), and/or magnetic random access memory (MRAM) and other such devices. The ROM may comprise, for example, a programmable read-only memory (PROM), an erasable programmable read-only memory (EPROM), an electrically erasable programmable read-only memory (EEPROM), and/or other like memory device.

[0012] Create feature map instructions 130 may create a feature map of a plurality of available features for an application, wherein each of the plurality of available features is associated with at least one context state. An application may comprise a plurality of feature option points where one of several subroutines and/or algorithms may be available to perform a functionality. For example, an application may retrieve data from databases comprising different access credentials, schemas, and/or architectures. For another example, the application may use different algorithms to calculate a mathematical value, such as generating a checksum. The different available features may also comprise incomplete, disabled, and/or malfunctioning subroutines and/or algorithms in the application.

[0013] The application features may each comprise a pre-compiled software class. For example, each application feature available at a feature option point may be provided in software code that is compiled at the same time as a main execution path of the application and/or that is linked into the application prior to and/or at execution time. In some embodiments, application features may be implemented in interpreted code that does not need to be compiled prior to execution.

[0014] In some embodiments, the features may comprise instantiated objects of a software class and/or object. In object-oriented programming, for example, a class may comprise an extensible program-code-template for creating objects. The class may provide initial values for state (member variables) and implementations

of behavior (e.g., functions and/or methods). When an object is created for the class, the resulting object may be referred to as an instance of the class.

[0015] Identify context state instructions 132 may identify a current context state associated with the application at a feature option point. For example, context states may comprise a user identifier, a session identifier, a user group, an environment variable, and/or an application mode.

[0016] User identifier and/or user group contexts may allow the application to execute a feature based on the identity of a user of the application and/or of a group associated with the user. For example, a user may comprise a developer or quality-assurance user. Such a context may be associated with features that provide additional debugging information while executing. For another example, a guest user may receive reduced functionality and/or lower priority for resources. A user who is a member of an internal employee group may be granted access to corporate resources by the feature that a non-employee group user may not be permitted to access by the feature(s).

[0017] Session identifier contexts may comprise information about a current execution of the application. For example, the session identifier may be associated with a particular thread in a multi-threaded application and/or a particular network/communication session for the application. The session identifier may allow the application, for example, to use a higher degree of encryption for communications associated with a public network and a lower degree and/or no encryption for communications associated with a corporate network.

[0018] Environment variable contexts may comprise information such as application version, operating system, hardware and/or other application/software availability, date, time, and/or location information. For example, a beta version of the application may enable new functionality by adding additional features to a feature option point compared to a previous versions. For another example, a software availability context may enable additional features in a user interface of the application if an add-on to the application is installed.

[0019] An application mode context may comprise settings and/or configurations associated with the execution of the application. For example, different features may be associated with different display resolutions for a user interface for the application. For another example, different features may be used when an application has an Internet connection than when the application is executing in an offline mode.

[0020] In some embodiments, the context types may be compared and/or ordered. For example, an offline version application mode context may take priority over a session identifier context that selects a feature associated with a degree of encryption for network communications. For another example, a beta version environment variable context may be combined with a user identifier and/or user group context that grants only some users/groups access to new features associated with the beta version of the application.

[0021] Determine available feature instructions 134 may determine whether at least one of the plurality of available features is associated with the current context state. The application may compare the context state to the feature map for the current feature option point. For example, the feature option point may comprise the application needing to connect to a database to retrieve some data. Environment variable context states may identify each of a plurality of available databases that may be able to provide the data and may be enabled or disabled based on various context factors. Such factors may comprise, for example, a load on the database, latency in communication with the database, availability of login credentials for the database, compatibility with the type of database (e.g., Oracle, MySQL, PostgreSQL, etc.).

[0022] In some embodiments, determining whether an available feature is associated with the current context state may comprise requesting a copy of an instantiated reference object of the pre-compiled software class associated with the current context state for the feature option point. For example, the application may create a copy of an already instantiated object that comprises variables for logging

into a MySQL database based on an environment variable context indicating that the MySQL database is a preferred and available data source.

[0023] Execute feature instructions 136 may, in response to determining that at least one of the plurality of available features is associated with the current context state, execute the at least one of the plurality of available features. For example, the application may execute a method associated with the software class associated with an instantiated object of that class, such as by calling a "connect" method in a database connection class object.

[0024] FIG. 2 is a flowchart of an embodiment of a method 200 for feature management consistent with disclosed implementations. Although execution of method 200 is described below with reference to the components of feature management device 100, other suitable components for execution of method 200 may be used.

[0025] Method 200 may start in block 205 and proceed to block 210 where device 100 may create a feature map for an application comprising a plurality of available features for each of a plurality of feature option points wherein each of the plurality of available features is associated with at least one context state. In some embodiments, creating the feature map may comprise designating at least one of the plurality of available features as a default feature for the first feature option point. For example, creating the feature map may comprise instantiating a reference object of a pre-compiled software class for each of the plurality of available features.

[0026] In some embodiments, create feature map instructions 130 may create a feature map of a plurality of available features for an application, wherein each of the plurality of available features is associated with at least one context state. An application may comprise a plurality of feature option points where one of several subroutines and/or algorithms may be available to perform a functionality. For example, an application may retrieve data from databases comprising different access credentials, schemas, and/or architectures. For another example, the application may use different algorithms to calculate a mathematical value, such as

generating a checksum. The different available features may also comprise incomplete, disabled, and/or malfunctioning subroutines and/or algorithms in the application.

[0027] The application features may each comprise a pre-compiled software class. For example, each application feature available at a feature option point may be provided in software code that is compiled at the same time as a main execution path of the application and/or that is linked into the application prior to and/or at execution time. In some embodiments, application features may be implemented in interpreted code that does not need to be compiled prior to execution.

[0028] In some embodiments, the features may comprise instantiated objects of a software class and/or object. In object-oriented programming, for example, a class may comprise an extensible program-code-template for creating objects. The class may provide initial values for state (member variables) and implementations of behavior (e.g., functions and/or methods). When an object is created for the class, the resulting object may be referred to as an instance of the class.

[0029] In some embodiments, creating the feature map may comprise associating at least one second available feature of the plurality of available features with the current context state. For example, after an application has been running for a period of time, a new feature may be assigned to a context state at a feature option point. Such a new feature may comprise, for example, a memory clean up function.

[0030] Method 200 may proceed to block 215 where device 100 may identify a current context state at a feature option point. For example, identify context state instructions 132 may identify a current context state associated with the application at a feature option point. For example, context states may comprise a user identifier, a session identifier, a user group, an environment variable, and/or an application mode.

[0031] In some embodiments, the context types may be compared and/or ordered. For example, an offline version application mode context may take

priority over a session identifier context that selects a feature associated with a degree of encryption for network communications. For another example, a beta version environment variable context may be combined with a user identifier and/or user group context that grants only some users/groups access to new features associated with the beta version of the application.

[0032] Method 200 may proceed to block 220 where device 100 may determine whether the current context state is associated with a first available feature of the plurality of available features for the first feature option point. For example, determine available feature instructions 134 may determine whether at least one of the plurality of available features is associated with the current context state. The application may compare the context state to the feature map for the current feature option point. For example, the feature option point may comprise the application needing to connect to a database to retrieve some data. Environment variable context states may identify each of a plurality of available databases that may be able to provide the data and may be enabled or disabled based on various context factors. Such factors may comprise, for example, a load on the database, latency in communication with the database, availability of login credentials for the database, compatibility with the type of database (e.g., Oracle, MySQL, PostgreSQL, etc.).

[0033] In some embodiments, determining whether an available feature is associated with the current context state may comprise requesting a copy of an instantiated reference object of the pre-compiled software class associated with the current context state for the feature option point. For example, the application may create a copy of an already instantiated object that comprises variables for logging into a MySQL database based on an environment variable context indicating that the MySQL database is a preferred and available data source.

[0034] In response to determining that the current context state is associated with the first available feature of the plurality of available features for the first feature option point, method 200 may proceed to block 225 where device 100 may execute the first available feature of the plurality of available features for the first

feature option point. For example, execute feature instructions 136 may, in response to determining that at least one of the plurality of available features is associated with the current context state, execute the at least one of the plurality of available features. For example, the application may execute a method associated with the software class associated with an instantiated object of that class, such as by calling a "connect" method in a database connection class object.

[0035] In response to determining that the current context state is not associated with the first available feature of the plurality of available features, method 200 may proceed to block 230 where device 100 may execute the default feature for the first feature option point. For example, execute feature instructions 136 may receive a copy of a reference object of a default feature and execute a method of the associated software class. The default feature may, for example, be a log message and/or a debugging message to inform a developer that an unknown context state was encountered. In some embodiments, the default feature may comprise a feature associated with at least one of the other context states for which features have been designated.

[0036] Method 200 may then end at block 240.

[0037] FIG. 3 is a block diagram of a system 300 for feature management consistent with disclosed implementations. System 300 may comprise a computing device 310 comprising a feature management engine 320 and an application engine 330. In some embodiments, feature management engine 320 and application engine 330 may be associated with multiple computing devices. Computing device 310 may comprise, for example, a general and/or special purpose computer, server, mainframe, desktop, laptop, tablet, smart phone, game console, and/or any other system capable of providing computing capability consistent with providing the implementations described herein. Feature management engine 320 and application engine 330 may each comprise, for example, instructions stored a machine readable medium executable by a processor, logic circuitry, and/or other implementations of hardware and/or software.

[0038] In some embodiments, feature management engine 320 may create a feature map 325 for an application comprising a plurality of available features for each of a plurality of feature option points. Each of the plurality of available features may be associated with at least one context state associated with application engine 330. Creating the feature map may comprise, for example, instantiating a reference object of a pre-compiled software class for each of the each of the plurality of available features, and designating at least one of the plurality of available features as a default feature for each of the plurality of feature option points.

[0039] In some embodiments, application engine 330 may determine whether the application has reached a first feature option point of the plurality of feature option points. In response to determining that the application has reached the first feature option point of the plurality of feature option points, application engine 330 may identify a current context state 335. For example, current context state 335 may comprise a user identifier, a session identifier, a user group, an environment variable, and/or an application mode.

[0040] Application engine 330 may determine whether current context state 335 is associated with a first available feature of the plurality of available features for the first feature option point. In some embodiments, determining whether the current context state is associated with the first available feature of the plurality of available features for the first feature option point may comprise requesting a copy of the reference object of the pre-compiled software class for the first available feature from feature map 325 of feature management engine 320. In response to determining that the current context state is associated with the first available feature of the plurality of available features for the first feature option point, application engine 330 may execute the first available feature of the plurality of available features for the first feature option point. In response to determining that the current context state is not associated with the first available feature of the plurality of available features, application engine 330 may execute the default feature for the first feature option point.

[0041] The disclosed examples may include systems, devices, computer-readable storage media, and methods for feature management. For purposes of explanation, certain examples are described with reference to the components illustrated in FIGs. 1-3. The functionality of the illustrated components may overlap, however, and may be present in a fewer or greater number of elements and components. Further, all or part of the functionality of illustrated elements may co-exist or be distributed among several geographically dispersed locations. Moreover, the disclosed examples may be implemented in various environments and are not limited to the illustrated examples.

[0042] Moreover, as used in the specification and the appended claims, the singular forms “a,” “an,” and “the” are intended to include the plural forms as well, unless the context indicates otherwise. Additionally, although the terms first, second, etc. may be used herein to describe various elements, these elements should not be limited by these terms. Instead, these terms are only used to distinguish one element from another.

[0043] Further, the sequence of operations described in connection with the Figures are examples and are not intended to be limiting. Additional or fewer operations or combinations of operations may be used or may vary without departing from the scope of the disclosed examples. Thus, the present disclosure merely sets forth possible examples of implementations, and many variations and modifications may be made to the described examples. All such modifications and variations are intended to be included within the scope of this disclosure and protected by the following claims.

CLAIMS

We claim:

1. A non-transitory machine-readable storage medium including instructions for feature management which, when executed by a processor, cause the processor to:

create a feature map of a plurality of available features for an application, wherein each of the plurality of available features is associated with at least one context state;

identify a current context state associated with the application at a feature option point;

determine whether at least one of the plurality of available features is associated with the current context state; and

in response to determining that at least one of the plurality of available features is associated with the current context state, execute the at least one of the plurality of available features.

2. The non-transitory machine-readable medium of claim 1, wherein each of the plurality of available features comprises a pre-compiled software class.

3. The non-transitory machine-readable medium of claim 2, wherein the instructions to create the feature map cause the processor to instantiate a reference object of the pre-compiled software class for each of the plurality of available features.

4. The non-transitory machine-readable medium of claim 4, wherein to determine whether at least one of the plurality of available features is associated with the current context state, the instructions cause the processor to request a copy of the instantiated reference object of the pre-compiled software class associated with the current context state for the feature option point.

5. The non-transitory machine-readable medium of claim 1, wherein the instructions, in response to determining that at least one of the plurality of available features is not associated with the current context, cause the processor to execute a default feature of the plurality of available features.

6. The non-transitory machine-readable medium of claim 1, wherein the current context state comprises at least one of the following: a user identifier, a session identifier, a user group, an environment variable, and an application mode.

7. The non-transitory machine-readable medium of claim 1, wherein the instructions further cause the processor to modify the feature map to associate a different available feature of the plurality of available features with the at least one context state.

8. The non-transitory machine-readable medium of claim 1, wherein the instructions further cause the processor to identify the current context state associated with the application at each of a plurality of feature option points.

9. A computer-implemented method for feature management comprising:

creating a feature map for an application comprising a plurality of available features for each of a plurality of feature option points wherein each of the plurality of available features is associated with at least one context state;

identifying a current context state at a feature option point;

determining whether the current context state is associated with a first available feature of the plurality of available features for the first feature option point; and

in response to determining that the current context state is associated with the first available feature of the plurality of available features for the first feature option point, executing the first available feature of the plurality of available features for the first feature option point.

10. The computer-implemented method of claim 9, wherein creating the feature map comprises associating at least one second available feature of the plurality of available features with the current context state.

11. The computer-implemented method of claim 9, wherein creating the feature map comprises designating at least one of the plurality of available features as a default feature for the first feature option point.

12. The computer-implemented method of claim 11, in response to determining that the current context state is not associated with the first available feature of the plurality of available features, executing the default feature for the first feature option point.

13. The computer-implemented method of claim 9, wherein creating the feature map comprises instantiating a reference object of a pre-compiled software class for each of the each of the plurality of available features.

14. The computer-implemented method of claim 9, wherein the current context state comprises at least one of the following: a user identifier, a session identifier, a user group, an environment variable, and an application mode.

15. A system for feature management, comprising:

a feature management engine to:

create a feature map for an application comprising a plurality of available features for each of a plurality of feature option points wherein each of the plurality of available features is associated with at least one context state, wherein creating the feature map comprises instantiating a reference object of a pre-compiled software class for each of the each of the plurality of available features, and

designate at least one of the plurality of available features as a default feature; and

an application engine to:

determine whether the application has reached a first feature option point, and

in response to determining that the application has reached the first feature option point of the plurality of feature option points:

identify a current context state,

determine whether the current context state is associated with a first available feature of the plurality of available features for the first feature option point,

in response to determining that the current context state is associated with the first available feature of the plurality of available features for the first feature option point, executing the first available feature of the plurality of available features for the first feature option point, and

in response to determining that the current context state is not associated with the first available feature of the plurality of available features, executing the default feature for the first feature option point.

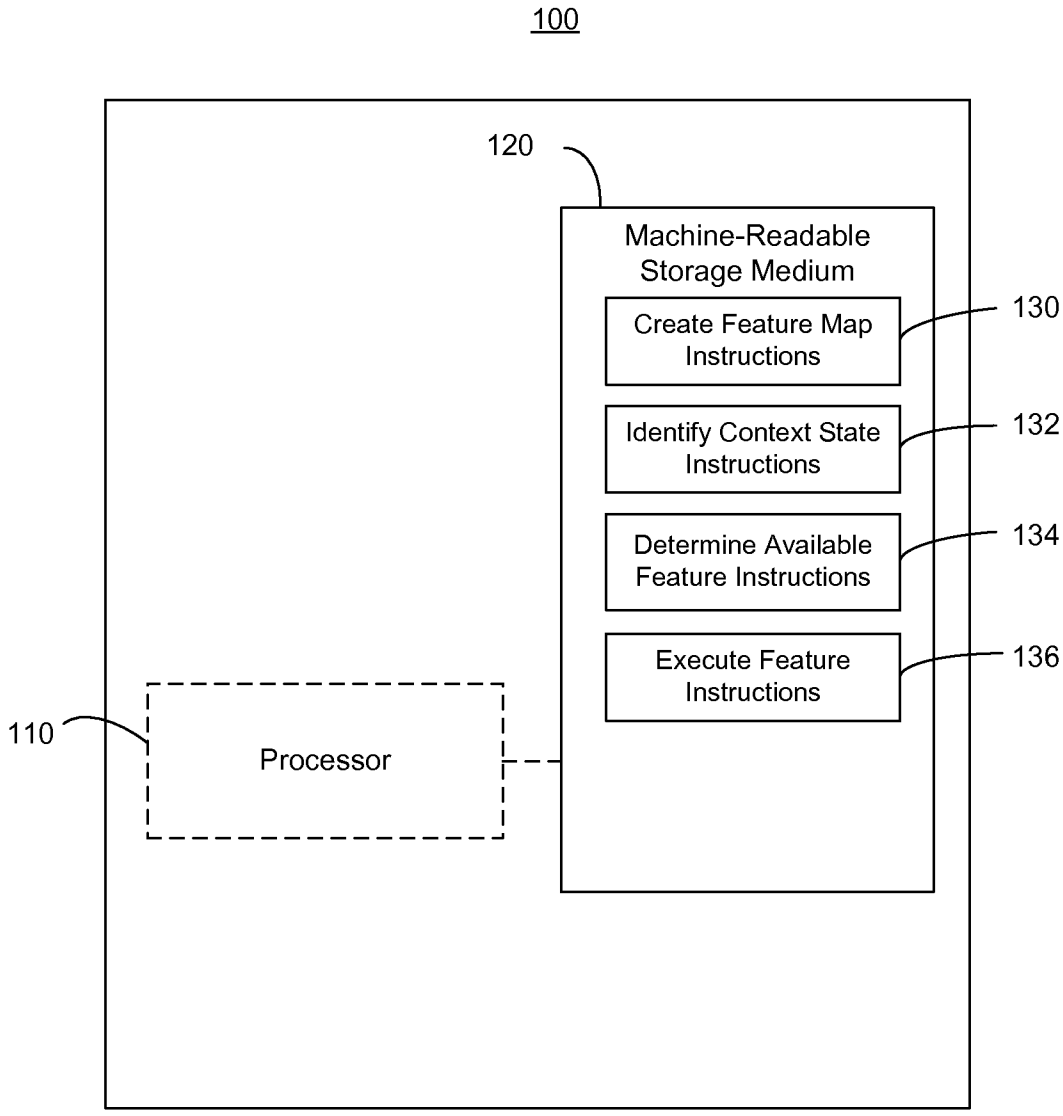
100
120
Machine-Readable Storage Medium
Create Feature Map Instructions
130
Identify Context State Instructions
132
Determine Available Feature Instructions
134
Execute Feature Instructions
136
110
Processor

FIG. 1

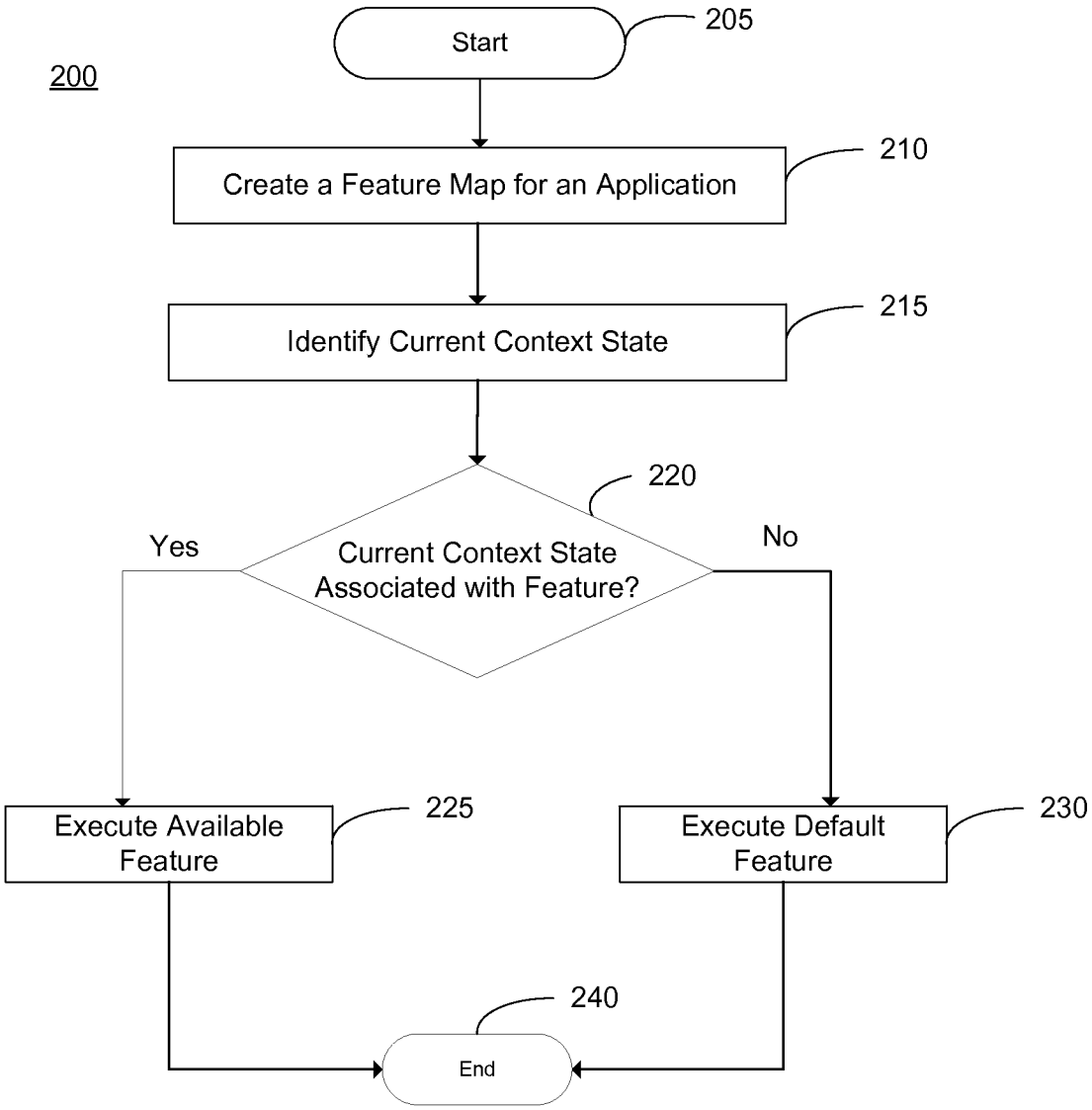
200
Start
205
Create a Feature Map for an Application
210
Identify Current Context State
215
Current Context State Associated with Feature?
220
Yes
No
Execute Available Feature
225
Execute Default Feature
230
End
240

FIG. 2

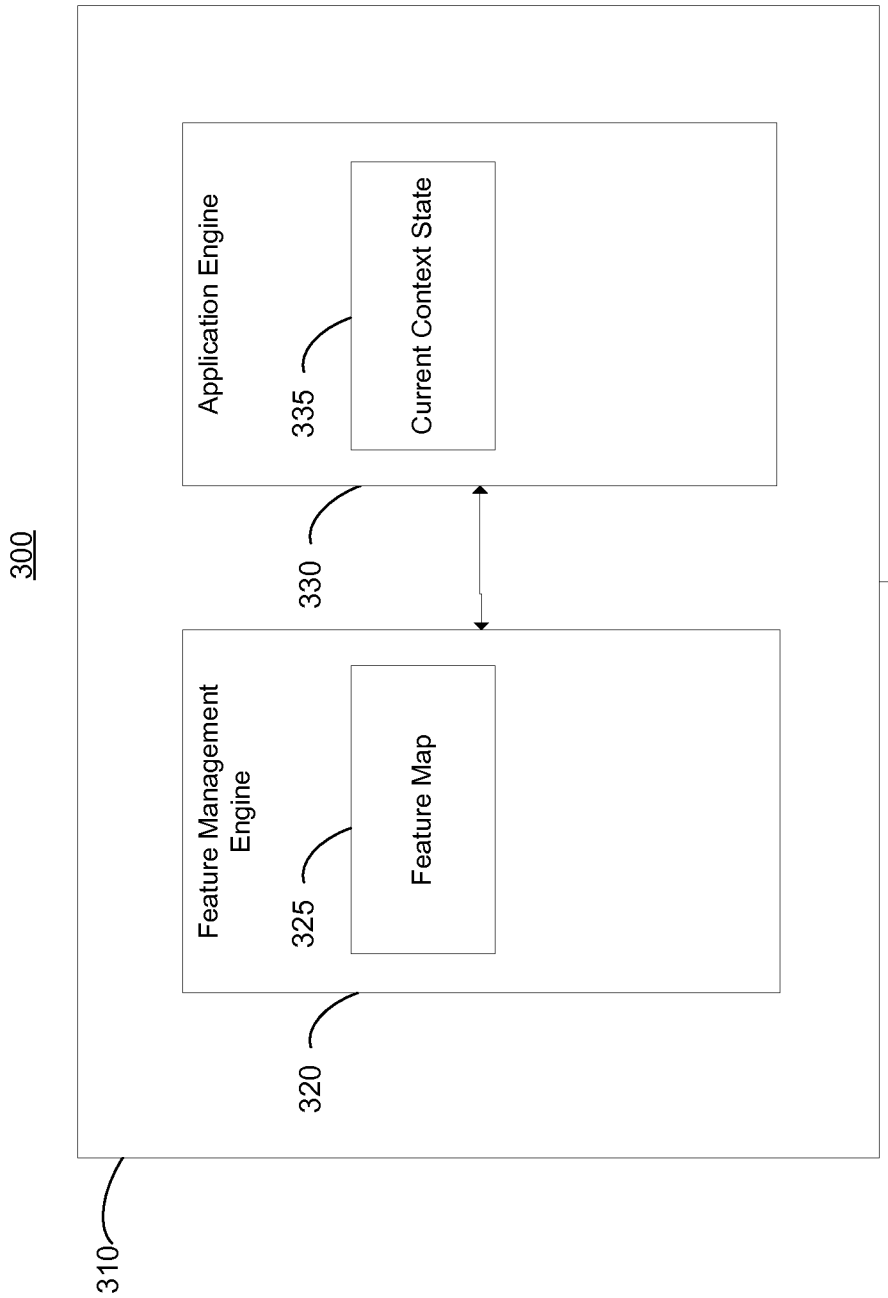
300
Application Engine
335
Current Context State
330
Feature Management Engine
325
Feature Map
320
310

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/022111

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/44(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 9/44; G06F 9/54; G06F 9/00; G06F 9/40

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: feature management, application, instruction, object, software class, context state, session identifier

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2003-0023662 A1 (ALAN TSU-I YAUNG) 30 January 2003 See abstract, paragraphs [0007]-[0017] and claim 1.	1-15
A	US 2012-0311531 A1 (BERTHOLD MARTIN LEBERT) 06 December 2012 See abstract, paragraphs [0002]-[0008] and claim 1.	1-15
A	US 6964051 B1 (MURUGAPPAN PALANIAPPAN) 08 November 2005 See abstract, column 1, line 65 - column 3, line 7, claim 1 and figure 2.	1-15
A	US 2006-0059458 A1 (Michael J Plummer) 16 March 2006 See abstract, claim 1 and figure 1.	1-15
A	US 05943496 A (LI; SHIH-GONG et al.) 24 August 1999 See abstract, claim 1 and figure 5.	1-15

☐ 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
30 November 2015 (30.11.2015)	**30 November 2015 (30.11.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
KWON, Hyun Su
Telephone No. +82-42-481-8686

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2015/022111

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003-0023662 A1	30/01/2003	US 7228547 B2	05/06/2007
US 2012-0311531 A1	06/12/2012	US 8650537 B2	11/02/2014
US 6964051 B1	08/11/2005	None	
US 2006-0059458 A1	16/03/2006	AU 2003-267757 A1	25/05/2004
		AU 2003-267757 A8	25/05/2004
		CN 1961287 A	09/05/2007
		EP 1586030 A2	19/10/2005
		GB 0225097 D0	11/12/2002
		JP 2006-518491 A	10/08/2006
		JP 2006-518491 T	10/08/2006
		KR 10-2005-0056270 A	14/06/2005
		WO 2004-040442 A2	13/05/2004
		WO 2004-040442 A3	27/04/2006
US 05943496 A	24/08/1999	WO 98-53398 A1	26/11/1998