



- (51) **International Patent Classification:**  
G06F 9/455 (2006.01) G06F 9/44 (2006.01)  
G06F 13/14 (2006.01)
- (21) **International Application Number:**  
PCT/US2013/021555
- (22) **International Filing Date:**  
15 January 2013 (15.01.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (71) **Applicant:** HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P. [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).
- (72) **Inventors:** WALL, Gary; Hewlett-Packard Company, 1925 West John Carpenter Freeway, Suites 400/475, Irving, Texas 75063 (US). MEYER, Chris; Hewlett-Packard Company, 1925 West John Carpenter Freeway, Suites 400/475, Irving, Texas 75063 (US).
- (74) **Agents:** PAGAR, Preetam et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528-9599 (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, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, 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, 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, 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, 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))

(54) **Title:** SERVER-PLATFORM SIMULATION SERVICE

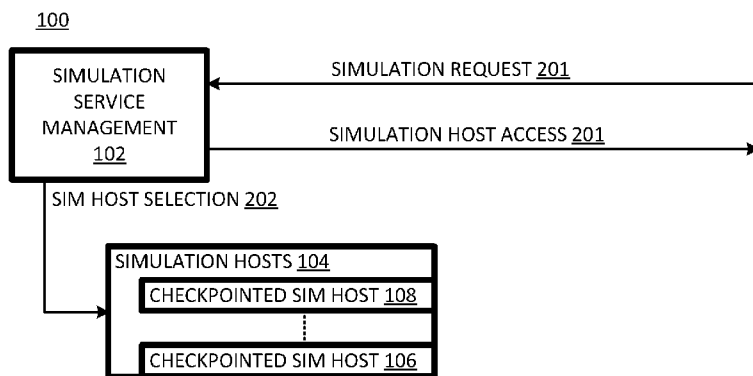


FIG. 1

(57) **Abstract:** A server-platform simulation service process involves receiving requests for server-platform simulation service. Simulation scenes including checkpoint simulation scenes corresponding to respective requests are identified. Respective computer hosts for executing said server-platform simulator based on the identified scenes are configured. Users are provided access to the computer hosts configured with the identified scenes.



## SERVER-PLATFORM SIMULATION SERVICE

### [01] BACKGROUND

[02] Software (including firmware) can be complex and many  
5 teams of developers may be involved in its development. Successful  
execution on one configuration of a hardware platform does not  
guarantee success on other configurations. Therefore, software is  
often tested using a variety of hardware configurations.

Furthermore, platform firmware may undergo a series of revisions,  
10 so software to run on the firmware may be tested on different  
firmware versions and the different firmware versions may be  
tested on a variety of platforms. However, providing access to a  
variety of hardware platforms with different configurations and  
firmware versions can be complex and expensive.

### 15 [03] BRIEF DESCRIPTION OF THE DRAWINGS

[04] The following figures represent examples and not the  
invention itself.

[05] FIGURE 1 is a schematic diagram of a platform-server-  
simulation service system in accordance with an example.

20 [06] FIGURE 2 is a flow chart of a platform-server-simulation  
service process implementable on the system of FIG. 1 in  
accordance with an example.

[07] FIGURE 3 is a schematic diagram of another platform-server-  
simulation service system in accordance with an example.

[08] FIGURE 4 is a flow chart of a platform-server-simulation service process implementable on the system of FIG. 3 in accordance with an example.

[09] DETAILED DESCRIPTION

5 [10] Platform simulations can be used in place of various real platform hardware and firmware to relieve the cost and inconvenience of maintaining and managing actual hardware systems for test purposes. Furthermore, platform hardware can be simulated before it is realized even in prototype form; thus,  
10 firmware can be validated even before the actual hardware is available. On the other hand, server-platform simulations can be difficult, error-prone, and time-consuming to set up. Furthermore, the simulations may run slowly; for example, a firmware boot that would take minutes in a hardware system can take hours in a  
15 simulation.

[11] To address some of these problems, server-platform simulations can be provided as a service. A developer or team of developers can request a particular platform configuration (e.g., specifying a hardware configuration and a firmware version).  
20 Responsibility for setting up the simulations is centralized at the service provider so that each development team does not have to set up their own simulations. To save boot time, a particular platform configuration can be available in multiple platform “scenes”. Each scene can correspond to a boot stage of a version of  
25 firmware in the context of particular platform hardware, as well as a particular operating system version and other software versions. In some examples, the service can provide centralized data mining of test results to characterize and to suggest improvements in firmware, simulation scenes, and in platform simulations.

[12] Accordingly, a platform-server-simulation system 100, shown in FIG. 1, includes simulation service management 102 and simulation hosts 104. Simulation hosts 104 include checkpointed simulation hosts 106 and 108. Herein, a simulation is

5 “checkpointed” if it is saved or “frozen” in a non-initial state, e.g., where the firmware is in a partially or fully booted state.

[13] Platform-server-simulation system 100 can implement a process 200, flow charted in FIG. 2. At 201, simulation service management receives requests for simulators. At 202, simulation

10 service management identifies simulation scenes, including checkpointed simulation scenes, corresponding to the requests. At 203, simulation service management configures hosts for instances of the simulation scenes. In some cases the configuring can precede the requests; in other cases, the identifying precedes

15 the configuring. At 204, simulation service management provides the requesting users access to the hosts configured with simulation scenes.

[14] At a minimum, the time otherwise required to reach the checkpointed state is saved every time the checkpointed simulation

20 is used. This can amount to about an hour per use and tens of hours per week of developer and development time. Process 200 can be implemented on systems other than system 100, and that system 100 can implement processes other than process 200.

[15] In practice, simulation service management can receive many simulation requests from many different requestors. For example, an operating-system qualification team might request multiple instances of a platform scene with fully booted firmware so that various compatibility tests can be run in parallel on multiple instances of a single operating system, or so that different operating systems can be tested in parallel. A separate team working on part of system firmware might want to start with the firmware partially booted to a point where the system firmware is loaded into volatile memory. Once a checkpoint scene is developed for such requestors, it can be stored in the platform scene repository and allocated to hosts as needed. In some cases, platform scenes for which there is high demand may be assigned to hosts in anticipation of requests.

[16] In some cases, a request may call for a platform scene that has not been prepared. In that case, a scene may be built by simulation service management rather than the requestor. In other words, the scene is constructed by those whose main job it is to construct scenes as opposed to being constructed by those whose main job it is to create or test firmware/software. Having scene construction performed by those for whom such preparation is a primary focus can minimize errors in scene construction. Also, the specialized platform scene constructors can benefit from data mining simulation/test results for different users; this can result in improved simulation scenes.

[17] In some cases, new scenes can be prepared before the first requests for them are received. For example, whenever a new firmware version is made available, it can be assumed there will be demand for it. Initial (unbooted) and checkpoint scenes can be developed and stored in the repository in anticipation of the

requests. Requests from those interested in operating system or application capability may call for platform simulations with fully booted firmware. Those interested in checking system firmware, may want boot firmware running but not have the system firmware or management firmware already booted. This can allow debugging and other trouble-shooting operations as the system or management firmware loads.

**[18]** Accordingly, a simulation system 300, shown in FIG. 3, includes simulation service management 302, simulation hosts 304, a platform scene repository 306, a hardware simulation constructor 308, a simulation scene constructor 310, and a data miner 312. Each of these elements includes programmed hardware. Hardware simulation constructor provides simulations of server hardware components. Simulation hosts 304 include both pre-configured simulation hosts 314 (on which a simulation scene is pre-installed) and available simulation hosts 316 (to which a simulation scene may be assigned).

**[19]** Typically, if a request is made for a simulation that can be met by an unassigned suitably pre-configured host, the requestor will be given access to such a host. However, if no such host is available, e.g., either because there is no such host or all such hosts are in use, an available host can be suitably configured in response to the request.

**[20]** Platform scene repository 306 stores simulation scenes 318. A simulation scene is an initial simulator state corresponding to a (checkpointed or initial) server platform state. A server platform state includes models, configurations, and states of active hardware components (e.g., processors, memories, network interface devices, motherboards, power supplies, and fans), firmware versions, and

firmware checkpoints. Simulations scenes can include initial scenes 320 and checkpoint scenes 322. An initial scene corresponds to an initial state of a server, e.g., as it is powered on or reset. A checkpoint scene corresponds to a non-initial server state,  
5 e.g., after firmware has begun to boot.

**[21]** Checkpoints can be “natural” or “frozen”. A “natural” checkpoint is a post-boot server state that would normally be maintained pending further inputs, such as a user or program command to launch an operating system or application. A “frozen”  
10 checkpoint corresponds to a server state that would normally change automatically but that can be frozen, e.g., using a debugger or similarly capable utility.

**[22]** If a desired scene is not available in repository 306, it can be developed. Whenever a new firmware version is made available, an  
15 initial scene for simulation can be made with little effort. The initial scene can be run so that checkpoints can be saved as checkpointed scenes. To this end, simulation scene constructor 310 can develop scenes based on inputs including hardware configuration 330, firmware version 332, and firmware checkpoint 334. The input  
20 hardware configuration is used to select a suitable hardware simulator from platform simulation constructor 308.

[23] Simulation scene constructor 310" provides for both manual and automatic operation. For instance, firmware developer can tweak a scene manually, e.g., by changing register values within the simulation based on experimentation, etc. A developer can then  
5 take these tweaks and make them available to a wider audience for consumption prior to submitting a firmware change to a source code base. Allowing manual checkpoints along with automated checkpoints allows for flexibility for how scenes are generated, and can save time for developers wishing to use custom non-standard  
10 scenes for their development (e.g. using firmware fixes before the fixes are fully submitted to a code base).

[24] Simulation system 300 provides for implementation of a process 400, flow charted in FIG. 4. At 401, a user sends and simulation-service management receives (concurrently and/or at  
15 different times) simulation requests for a server-platform simulator. At 402, simulation service management interprets the requests so that they can be matched with simulation scenes. At 403, simulation-service management selects matching simulation scenes.

[25] At 404, simulation service management provides a host  
20 computer systems configured with respective selected scenes. This can involve selecting a host pre-configured with a selected scene. If there is no such pre-configured host, an instance of a selected scene can be copied from the scene repository and assigned to an available host. There can be further refinements to the host  
25 selection process if some hosts are more capable or more compatible with selected scenes than others. At 405, simulation-service management provides requesting users with access to the suitably configured hosts.



[26] At 406, users run the simulations to which they have been provided access. Simulation/test results are then available to the user for the user's purposes. The simulation/test results are also available to simulation service management. Accordingly, at 407,  
5 the simulation service management may automatically mine simulation/test results for individual simulations, across simulations for a scene instance, across scene instances, across scenes corresponding to different firmware versions, and across firmware versions.

10 [27] In some cases, analysis of the mined data may supplement evaluations of firmware being tested; the supplementary evaluations can be provided to other users interested in testing the same firmware. In some cases analysis of mined data can identify problems with simulations scenes, the platform simulation, or the  
15 simulated hardware. Where problems are identified in a scene or platform simulation, the scene or simulation can be tweaked at 408. The tweaked simulation and/or scene can then be used for future simulation runs, e.g., in future iterations of process 400.

[28] The foregoing discussion regarding feedback and tweaking  
20 illustrates how simulation service management can serve as a central gathering point for information, data mining, and expertise in a way that would be infeasible if each developer or development team had to manage its own simulation setup. Thus, system 300 and process 400 provide for better server-platform simulations and  
25 more effective use of developer time and expertise.

[29] Herein, a “system” is a set of interacting non-transitory tangible elements, wherein the elements can be, by way of example and not of limitation, mechanical components, electrical elements, atoms, physical encodings of instructions, and process actions.

5 Herein, “process” refers to a sequence of actions resulting in or involving a physical transformation.

[30] Herein, “computer” refers to a hardware machine for manipulating physically encoded data in accordance with physically encoded instructions. A “server” is a computer that performs  
10 services for other computers. A “server platform” is a design including hardware elements and firmware elements that servers may share. Depending on context, reference to a computer or server may or may not include software installed on the computer. Herein, unless other apparent from context, a functionally defined  
15 component, e.g., a constructor or miner, of a computer is a combination of hardware and software executing on that hardware to provide the defined functionality.

[31] Herein, “platform firmware” is software encoded, at least under initial conditions, in non-volatile storage media. Platform  
20 firmware can have various components. For example, platform firmware can include: boot firmware that loads into volatile memory for the purpose of initialization and loading other firmware, but which becomes inactive once the firmware is fully booted; system firmware that remains in volatile memory after booting is complete  
25 and serves as an interface between hardware and an operating system; and management firmware that provides an interface for management that bypasses an operating system, e.g., via a lights-out module.

[32] Herein, firmware is “fully-booted” if it has been booted to a state to which no more firmware is being loaded from non-volatile memory to volatile memory. Herein, firmware is “partially booted” if it has been booted to a state in which more firmware is to be  
5 loaded from non-volatile memory to volatile memory without external intervention. Herein, “tweaking” means “modifying”.

[33] In this specification, related art is discussed for expository purposes. Related art labeled “prior art”, if any, is admitted prior art. Related art not labeled “prior art” is not admitted prior art.  
10 The illustrated and other described embodiments, as well as modifications thereto and variations thereupon are within the scope of the following claims.

[34] What Is Claimed Is:

## CLAIMS

1. A server-platform simulation service process comprising:
  - receiving requests for server-platform simulation service;
  - identifying simulation scenes including checkpoint simulation scenes corresponding to respective requests;
  - configuring respective computer hosts to execute the identified scenes; and
  - providing users access to the computer hosts configured to execute the identified scenes.
2. A server-platform simulation service process as recited in Claim 1 wherein the configuring precedes the identifying for at least one of the requests.
3. A server-platform simulation service process as recited in Claim 1 wherein the configuring follows the identifying for at least one of the requests.
4. A server-platform simulation service process as recited in Claim 1 further comprising:
  - users use the simulators on the hosts to which they have been provided access; and
  - automatically mining simulation results.

5. A server-platform simulation service process as recited in Claim 4 further comprising tweaking a simulation scene based automatically mined simulation results.
6. A server-platform simulation service process as recited in Claim 5 further comprising tweaking a platform simulation based on the automatically mined simulation results, wherein the tweaking of a simulation scene is based in part of a tweaked platform simulation.
7. A server-platform simulation service process as recited in Claim 1 wherein at least one of the checkpoint simulation scenes corresponds to a server-platform state in which firmware is fully booted.
8. A server-platform simulation service process as recited in Claim 1 at least one of the checkpoint simulation scenes corresponds to a server-platform state in which firmware is partially booted.
9. A server-platform simulation service system comprising:
  - simulation hosts including simulation hosts configured with checkpointed server-platform simulation scenes; and
  - a simulation service manager to provide users requesting server-platform simulators access to the simulation hosts configured with the server-platform simulation scenes.
10. A server-platform simulation service system as recited in Claim 9 further comprising a platform scene repository to store server-platform scenes including initial scenes and checkpoint scenes.
11. A server-platform simulation service system as recited in Claim 9 wherein at least one of the checkpoint scenes corresponds to a server-platform state in which firmware is fully booted.

12. A server-platform simulation service system as recited in Claim 9 wherein at least one of the checkpoint scenes corresponds to a server-platform state in which firmware is partially booted.
13. A server-platform simulation service system as recited in Claim 9 further comprising a simulation scene constructor to construct a simulation scene based on specifications for hardware configuration, a firmware version, and a firmware checkpoint.
14. A server-platform simulation service system as recited in Claim 13 further comprising a data miner to mine simulation and test results to yield data mining results, said simulation scene constructor being further to tweak simulation scenes based on the data mining results.
15. A server-platform simulation service system as recited in Claim 14 further comprising a platform simulation constructor to tweak platform simulations based on the data mining results.
16. A server-platform simulation service system comprising plural host computers configured with plural instances of a version of firmware, the instances including versions checkpointed at different stages of a boot sequence; and  
simulation service management to select a host in response to a simulation service requested based at least in part on the stage at which an instance of the version on that host is checkpointed.
17. A server-platform simulation service system as recited in Claim 16 further comprising a simulation scene repository for storing differently checkpointed instances of the firmware, the simulation service management being further to install a checkpointed instance of the software on a host in response to a request for a simulation service.

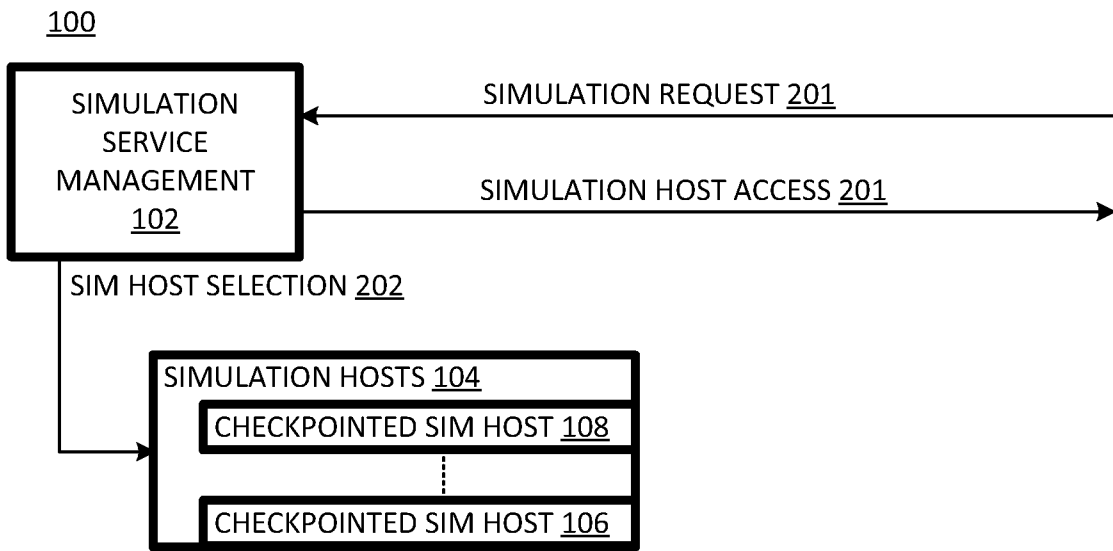


FIG. 1

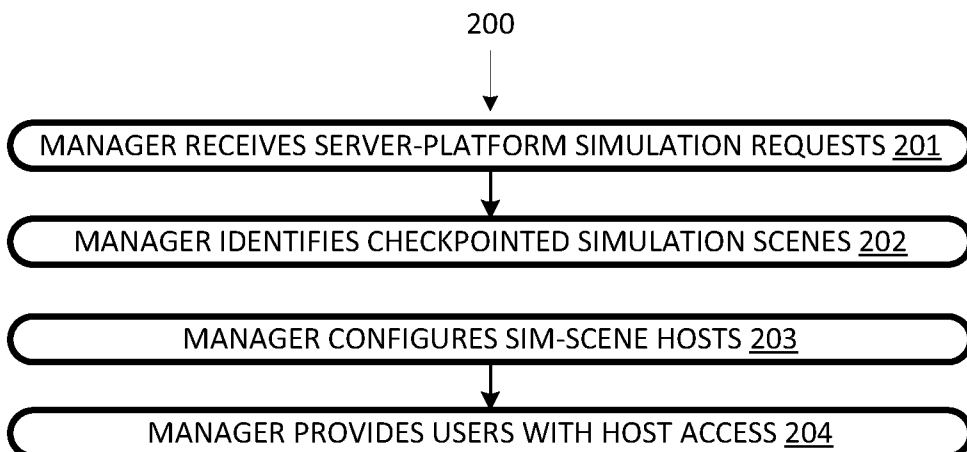


FIG. 2

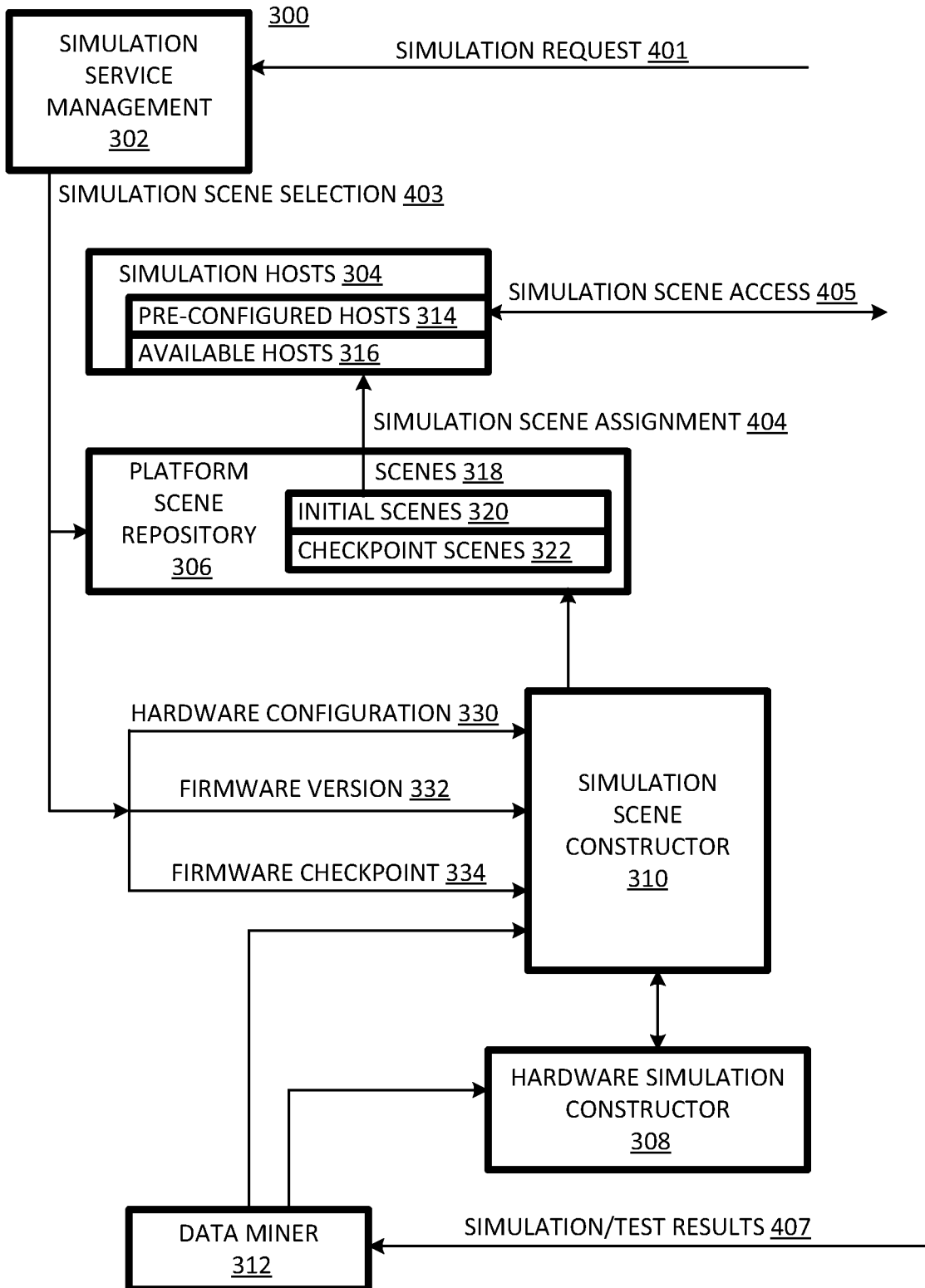


FIG. 3



3/3

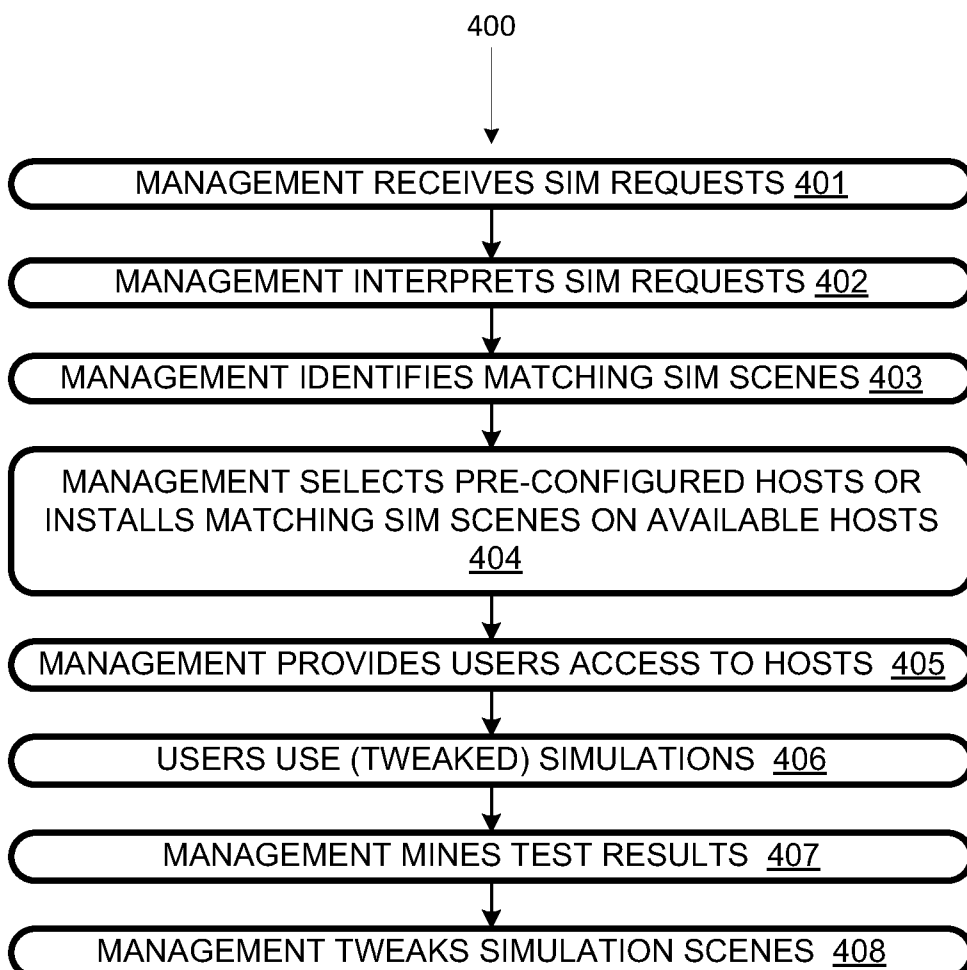


FIG. 4

**A. CLASSIFICATION OF SUBJECT MATTER****G06F 9/455(2006.01)i, G06F 13/14(2006.01)i, 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 9/455; G06F 15/16; A63F 9/24; H04L 12/28; G10L 21/00; H04W 64/00; G06F 13/14; G06F 9/44

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) &amp; Keywords: server, platform, simulation, service, scenes, request, user, computer, host, execute tweak, system, access

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2011-116309 A1 (DIGIMARC CORPORATION et al.) 22 September 2011 See abstract, pages 5-108, claims 1-92, and figures 1-22	1-17
A	WO 2008-109149 A1 (TRION WORLD NETWORK, INC. et al.) 12 September 2008 See abstract, paragraphs [0036]-[0090], claims 1-66, and figures 1-23	1-17
A	KR 10-2009-0051088 A (INTEL CORPORATION) 20 May 2009 See abstract, paragraphs [0017]-[0068], claims 1-30, and figures 1-8	1-17
A	KR 10-2007-0089119 A (IRELESS VALLEY COMMUNICATIONS, INC.) 30 August 2007 See abstract, pages 4-11, claims 1-131, and figures 1-20	1-17



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 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

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


Date of the actual completion of the international search

02 September 2013 (02.09.2013)

Date of mailing of the international search report

**04 September 2013 (04.09.2013)**

Name and mailing address of the ISA/KR

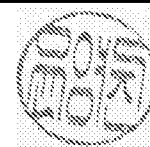

 Korean Intellectual Property Office  
 189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City,  
 302-701, Republic of Korea

Facsimile No. +82-42-472-7140

Authorized officer

YOON Young Jin

Telephone No. +82-42-481-8533



## INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.  
**PCT/US2013/021555**

Patent document cited in search report	Publication date	Patent family member(s)	Publication date		
WO 2011-116309 A1	22/09/2011	CA 2792336 A1	22/09/2011		
		CN102782733 A	14/11/2012		
		CN102893327 A	23/01/2013		
		EP 2559030 A1	20/02/2013		
		JP 2013-527947A	04/07/2013		
		KR 10-2013-0027081 A	14/03/2013		
		KR20130027081A	14/03/2013		
		US 2011-0159921 A1	30/06/2011		
		US 2011-0161076 A1	30/06/2011		
		US 2011-0165917 A1	07/07/2011		
		WO 2011-082332 A1	07/07/2011		
		WO 2008-109149 A1	12/09/2008	AU 2008-223321 A1	12/09/2008
				AU 2008-223396 A1	12/09/2008
CA 2679838 A1	12/09/2008				
CA 2679839 A1	12/09/2008				
CN101678236 A	24/03/2010				
CN101678236 B	31/10/2012				
CN101678237 A	24/03/2010				
CN101678237 B	23/05/2012				
EP 2131932 A1	16/12/2009				
EP 2131932 A4	01/05/2013				
EP 2142267 A1	13/01/2010				
EP 2142267 A4	05/12/2012				
JP 2010-525422 T	22/07/2010				
JP 2010-525422A	22/07/2010				
JP 2010-525423 T	22/07/2010				
JP 2010-525423A	22/07/2010				
KR 10-2010-0014941 A	11/02/2010				
KR 10-2010-0014942 A	11/02/2010				
KR20100014941A	11/02/2010				
KR20100014942A	11/02/2010				
RU2009136683 A	20/04/2011				
RU2009136684 A	20/04/2011				
TW200844766 A	16/11/2008				
US 2008-0220873 A1	11/09/2008				
US 2008-287192 A1	20/11/2008				
US 2008-287193 A1	20/11/2008				
US 2008-287194 A1	20/11/2008				
US 2008-287195 A1	20/11/2008				
US 2009-0275414 A1	05/11/2009				
WO 2008-109132 A1	12/09/2008				
KR 10-2009-0051088 A	20/05/2009			CN101512972 A	19/08/2009
				EP 2067306 A1	10/06/2009
		JP 05-070286B2	07/11/2012		
		JP 2010-500671A	07/01/2010		
		US 2011-0125872 A1	26/05/2011		
		US 8316107 B2	20/11/2012		

**INTERNATIONAL SEARCH REPORT**

Information on patent family members

International application No.  
**PCT/US2013/021555**

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		WO 2008-031273 A1	20/03/2008
KR 10-2007-0089119 A	30/08/2007	WO 2006-012554 A2	02/02/2006
		WO 2006-012554 A3	09/04/2009