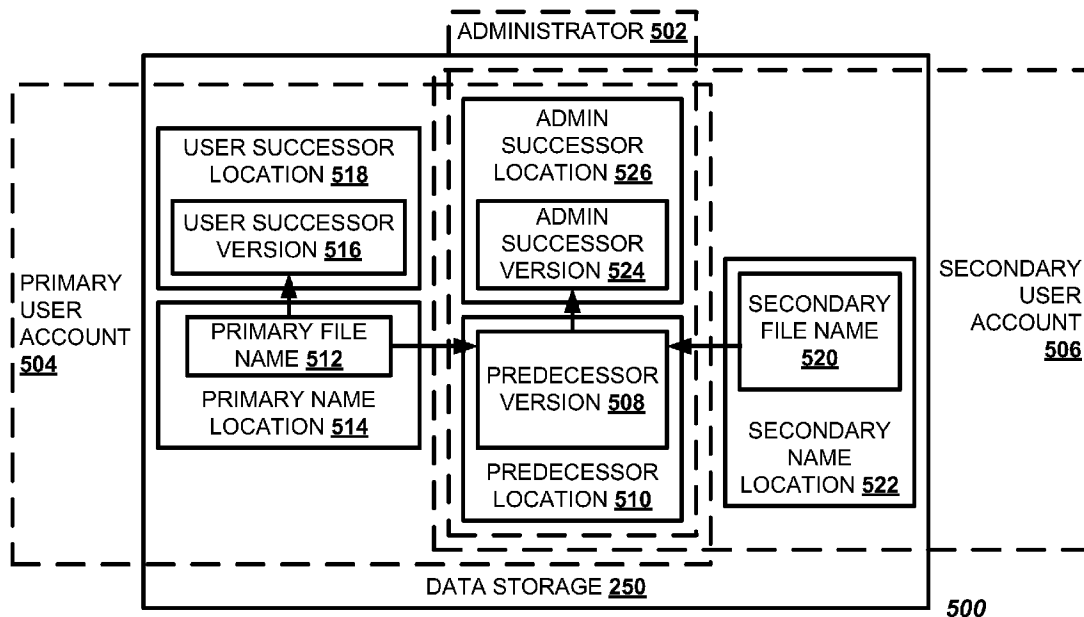




(86) Date de dépôt PCT/PCT Filing Date: 2013/01/07
(87) Date publication PCT/PCT Publication Date: 2013/07/18
(45) Date de délivrance/Issue Date: 2020/10/13
(85) Entrée phase nationale/National Entry: 2014/07/04
(86) N° demande PCT/PCT Application No.: US 2013/020519
(87) N° publication PCT/PCT Publication No.: 2013/106276
(30) Priorité/Priority: 2012/01/15 (US13/350,810)

(51) Cl.Int./Int.Cl. *G06F 8/65* (2018.01),
G06F 8/61 (2018.01)
(72) Inventeurs/Inventors:
HAUBOLD, JEREMY, US;
PACCIARINI, CLAUDIO, US
(73) Propriétaire/Owner:
MICROSOFT TECHNOLOGY LICENSING, LLC, US
(74) Agent: SMART & BIGGAR LLP

(54) Titre : MOTEUR D'INSTALLATION ET FORMAT DE PROGICIEL POUR INSTALLATIONS FIABLES POUVANT
ETRE MISES EN OEUVRE EN PARALLELE
(54) Title: INSTALLATION ENGINE AND PACKAGE FORMAT FOR PARALLELIZABLE, RELIABLE INSTALLATIONS



(57) **Abrégé/Abstract:**

In one embodiment, an application updater may cleanly update a computer application without causing a fault or a reboot of either the system or the process. A processor may execute a predecessor version of a computer application. The processor may install a successor version of the computer application on a user account level. The processor may execute the successor version of the computer application.

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

(19) World Intellectual Property
Organization
International Bureau



(10) International Publication Number
WO 2013/106276 A1

(43) International Publication Date
18 July 2013 (18.07.2013)

(51) International Patent Classification:

G06F 9/44 (2006.01) *G06F 15/16* (2006.01)
G06F 21/62 (2013.01)

(21) International Application Number:

PCT/US2013/020519

(22) International Filing Date:

7 January 2013 (07.01.2013)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

13/350,810 15 January 2012 (15.01.2012) US

(71) Applicant: **MICROSOFT CORPORATION** [US/US];
One Microsoft Way, Redmond, Washington 98052-6399
(US).

(72) Inventors: **HAUBOLD, Jeremy**; c/o Microsoft Corpora-
tion, LCA - International Patents, One Microsoft Way,
Redmond, Washington 98052-6399 (US). **PACCIARINI,**
Claudio; c/o Microsoft Corporation, LCA - International
Patents, One Microsoft Way, Redmond, Washington
98052-6399 (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 applicant's entitlement to apply for and be granted a
patent (Rule 4.17(ii))
- as to the applicant's entitlement to claim the priority of the
earlier application (Rule 4.17(iii))

Published:

- with international search report (Art. 21(3))

(54) Title: INSTALLATION ENGINE AND PACKAGE FORMAT FOR PARALLELIZABLE, RELIABLE INSTALLATIONS

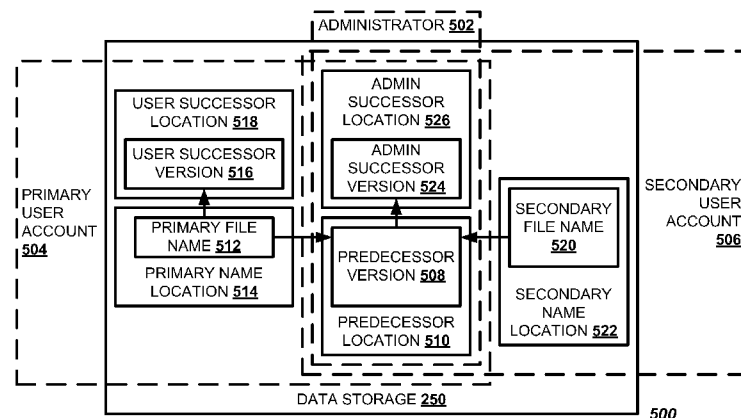


Figure 5

(57) Abstract: In one embodiment, an application updater may cleanly update a computer application without causing a fault or a re-boot of either the system or the process. A processor may execute a predecessor version of a computer application. The processor may install a successor version of the computer application on a user account level. The processor may execute the successor version of the computer application.

WO 2013/106276 A1

INSTALLATION ENGINE AND PACKAGE FORMAT FOR PARALLELIZABLE, RELIABLE INSTALLATIONS

BACKGROUND

[0001] A software developer may continually upgrade a computer program on a user device.

- 5 The user device may contact a server administered by the software developer to see if upgrades are available. If an upgrade is available, the user device may download the new sections of software and add those sections to the existing program currently installed on the user device.

SUMMARY

- 10 [0002] This Summary is provided to introduce a selection of concepts in a simplified form that is further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

- 15 [0003] Embodiments discussed below relate to cleanly updating a computer application without causing a fault or a reboot of either the system or the process. A processor may execute a predecessor version of a computer application. The processor may install a successor version of the computer application on a user account level. The processor may execute the successor version of the computer application.

- 20 [0003a] According to one aspect of the present invention, there is provided a machine-implemented method, comprising: executing a predecessor version of a computer application in a primary user account of a computer device; installing a successor version of the computer application for the primary user account; and executing the successor version of the computer application in the primary user account while a secondary user account of the computer device maintains a hard link to the predecessor version.

- 25 [0003b] According to another aspect of the present invention, there is provided a tangible machine-readable medium having a set of instructions detailing a method stored thereon that when executed by one or more processors cause the one or more processors to perform the

method, the method comprising: executing a predecessor version of a computer application referenced by a primary file name of a primary user account of a computer device; downloading a complete successor version using an updater module disregarding the predecessor version; installing the complete successor version of the computer application
5 referenced by the primary file name in a clean successor data storage location, while a secondary user account of the computer device maintains a hard link to the predecessor version using a secondary file name.

[0003c] According to still another aspect of the present invention, there is provided an application updater, comprising: a communications interface that downloads a complete
10 successor version of a computer application; a data storage that stores a predecessor version of the computer application in a predecessor data storage location and the complete successor version in a clean successor data storage location; and a processor that executes the predecessor version and installs the complete successor version of the computer application in a primary user account while a secondary user account maintains a hard link to the
15 predecessor version.

[0003d] According to yet another aspect of the present invention, there is provided an application updater on a computing device, comprising: a data storage on the computing device, the data storage configured to store multiple application profiles for multiple users of a computer application on the computing device to reference the computer application with a
20 primary file name for a primary user account of the computing device and with a secondary file name for a secondary user account of the same computing device as the primary user account, and to store a predecessor version of the computer application stored in a predecessor location in the data storage and a complete successor version stored in a clean successor data storage location; and a processing core having at least one processor configured to execute the
25 predecessor version and configured to execute an installer module to install the complete successor version of the computer application in the primary user account referenced by the primary file name without interfering with performance of the predecessor version and not install the complete successor version of the computer application for the secondary user account on the computing device, such that the secondary user account has the predecessor

version referenced by the secondary file name when the primary user account has the complete successor version referenced by the primary file name.

[0003e] According to a further aspect of the present invention, there is provided a computing device, having a memory to store a series of instructions that are executed by at least one

5 processor to implement an updater module, the computing device configured to store multiple application profiles for a computer application on the computing device for multiple users of the computing device, with a primary user account of the computing device referencing the computer application with a primary file name and a secondary user account of the computing device referencing the computer application with a secondary file name; execute a predecessor
10 version of the computer application referenced by the primary file name in the primary user account; install a successor version of the computer application for the primary user account in a clean successor data storage location without interfering with the performance of the predecessor version and not install a successor version of the computer application for the secondary user account on the computing device; and execute the successor version of the
15 computer application in the primary user account referenced by the primary file name while the secondary user account references the predecessor version with the secondary file name.

[0003f] According to yet a further aspect of the present invention, there is provided a machine-implemented method, comprising: executing a predecessor version of a computer application referenced by a primary file name of a primary user account of a computer device;

20 downloading a complete successor version using an updater module while disregarding the predecessor version; installing the complete successor version of the computer application referenced by the primary file name in a clean successor data storage location on the computer device without interfering with performance of the predecessor version, and not installing the complete successor version of the computer application for a secondary user account on the
25 computer device, the primary user account and the secondary user account being on a same computer device; and executing the successor version of the computer application in the primary user account of the computer device referenced by the primary file name while the secondary user account of the computer device references the predecessor version with a secondary file name.

DRAWINGS

[0004] In order to describe the manner in which the above-recited and other advantages and features can be obtained, a more particular description is set forth and will be rendered by reference to specific embodiments thereof which are illustrated in the appended drawings.

- 5 Understanding that these drawings depict only typical embodiments and are not therefore to be considered to be limiting of its scope, implementations will be described and explained with additional specificity and detail through the use of the accompanying drawings.

[0005] FIG. 1 illustrates, in a block diagram, one embodiment of a computer network.

[0006] FIG. 2 illustrates, in a block diagram, one embodiment of a computing device.

- 10 [0007] FIG. 3 illustrates, in a data flow diagram, one embodiment of an update service.

[0008] FIG. 4 illustrates, in a block diagram, one embodiment of an update for a computer application.

[0009] FIG. 5 illustrates, in a block diagram, one embodiment of a data storage layout.

- 15 [0010] FIG. 6 illustrates, in a flowchart, one embodiment of a method for upgrading a computer application.

DETAILED DESCRIPTION

[0011] Embodiments are discussed in detail below. While specific implementations are discussed, it should be understood that this is done for illustration purposes only. A person skilled in the relevant art will recognize that other components and configurations may be used without parting from the scope of the subject matter of this disclosure.

The implementations may be a machine-implemented method, a tangible machine-readable medium having a set of instructions detailing a method stored thereon for at least one processor, or an application updater for a computing device.

[0012] A computer device may maintain a user account for each user of a computer device. The user account may protect certain data files from access by other users of the computer device. An application updater may update a computer application on a per-user basis in addition to a traditional per-machine basis. The per-user basis may update the computer application for a user account, rather than the other user accounts on the machine. A user account may elevate a user privilege of the user account to install on a machine level.

[0013] Each time the application updater runs, the application updater may extract the update in a new directory and data storage location, thus eliminating any file-in-use problems causing a reboot of the machine. Further, different versions of the computer application may be installed side-by-side without interfering with other versions of the computer application previously installed on the machine, allowing the computer application to update silently, without causing any issues to the smooth performance of the computer application. The application updater may install the same version or a newer version of the computer application as part of an install, reinstall, or repair of the computer application without interfering with the current version. If the available version is an older version, the application updater may execute a null operation to prevent downgrading the computer application. The application updater may install multiple computer applications in parallel.

[0014] Thus, in one embodiment, an application updater may cleanly update a computer application without causing a fault or a reboot of either the system or the process. A processor may execute a predecessor version of a computer application. The processor may install a successor version of the computer application on a user account level. The processor may execute the successor version of the computer application.

[0015] FIG. 1 illustrates, in a block diagram, one embodiment of a computer network 100. A user device 110 may be a desktop computer, a laptop computer, a tablet computer, a

handheld computer, or other computing device. The user device 110 may run a computer application 112. The computer application 112 may be any software program or firmware program executed by the user device 110. The computer application 112 may have a version indicating the iteration of the computer application 112. The computer application
5 112 may have an updater module 114 that connects to an application server 120 via a data network connection 130. The data network connection 130 may be a local area network connection, an internet connection, a mobile network connection, or other network connection. The updater module 114 may query the application server 120 to see if a new version of the computer application 112 is available. A predecessor version is the version
10 of the computer application 112 currently being executed on the user device. A successor version is the new version of the computer application 112. The updater module 114 may download the successor version to replace the predecessor version.

[0016] FIG. 2 illustrates a block diagram of an exemplary computing device 200 which may act as an application updater. The computing device 200 may combine one or more
15 of hardware, software, firmware, and system-on-a-chip technology to implement an application updater. The computing device 200 may include a bus 210, a processor 220, a memory 230, a read only memory (ROM) 240, a storage device 250, an input device 260, an output device 270, and a communication interface 280. The bus 210 may permit communication among the components of the computing device 200.

[0017] The processor 220 may include at least one conventional processor or
20 microprocessor that interprets and executes a set of instructions. The memory 230 may be a random access memory (RAM) or another type of dynamic storage device that stores information and instructions for execution by the processor 220. The memory 230 may also store temporary variables or other intermediate information used during execution of
25 instructions by the processor 220. The ROM 240 may include a conventional ROM device or another type of static storage device that stores static information and instructions for the processor 220. The storage device 250 may include any type of tangible machine-readable medium, such as, for example, magnetic or optical recording media and its corresponding drive. A tangible machine-readable medium is a physical medium storing
30 machine-readable code or instructions, as opposed to a transitory medium or signal. The storage device 250 may store a set of instructions detailing a method that when executed by one or more processors cause the one or more processors to perform the method.

[0018] The input device 260 may include one or more conventional mechanisms that permit a user to input information to the computing device 200, such as a keyboard, a

mouse, a voice recognition device, a microphone, a headset, etc. The output device 270 may include one or more conventional mechanisms that output information to the user, including a display, a printer, one or more speakers, a headset, or a medium, such as a memory, or a magnetic or optical disk and a corresponding disk drive. The

5 communication interface 280 may include any transceiver-like mechanism that enables computing device 200 to communicate with other devices or networks. The communication interface 280 may include a network interface or a transceiver interface. The communication interface 280 may be a wireless, wired, or optical interface.

[0019] The computing device 200 may perform such functions in response to processor
10 220 executing sequences of instructions contained in a computer-readable medium, such as, for example, the memory 230, a magnetic disk, or an optical disk. Such instructions may be read into the memory 230 from another computer-readable medium, such as the storage device 250, or from a separate device via the communication interface 280.

[0020] FIG. 3 illustrates, in a data flow diagram, one embodiment of an update service
15 300 for a computer application 112. Upon initiation, the current version 302 of the computer application 112 may direct an updater module 114 to request any updates from an application server 120. The application server 120 may send an update package to the updater module 114. The update package may have an update manifest 304 and a current binary 306 of the computer application 112. The update manifest 304 is a list of the files
20 that get installed. The update manifest 304 may describe a file path, a file size, any registry keys, and any error checking mechanisms for each file. The update manifest 304 may be in an extensible markup language (XML) format. The current binary 306 is an executable file or associated file, such as a digital link library, for the computer application 112. The updater module 114 may use the update package to create the installer module
25 308 for installation.

[0021] FIG. 4 illustrates, in a block diagram, one embodiment of an update 400 for a computer application. An update package 402 sent from an application server 120 may have an installer code 404, a set of neutral resources 406, a set of localizable resources 408, and a payload 410. The installer code 404 instantiates the installer module 308 on the
30 user device 110. The set of neutral resources 406 is a set of general resources used during installation, such as icons, non-localizable strings, and non-localizable images to be shown during installation. The set of localizable resources 408 is a set of language specific resources used during installation, such as messages, prompts, and sounds. The payload 410 may be the update manifest 304 and the current binary 306.

[0022] FIG. 5 illustrates, in a block diagram, one embodiment of a data storage layout 500. A user device 100 may support multiple user accounts for multiple users. A user account may be password protected from other users. The data storage device 250 may store multiple application profiles for multiple users. A first user may update a computer application 112 while a second user may forgo the update. An administrator 502 may update the computer application 112 for each user account. A primary user account 504 is the user account that is executing the update of the computer application. A secondary user account 506 is a user account that is not actively executing an update of the computer application. The secondary user account 506 may be active or dormant.

[0023] The data storage 250 may store a predecessor version 508 of a computer application 112 in a predecessor location 510. A primary user account 504 may store a primary file name 512 in a primary name data storage location 514 referencing the predecessor version 508. A file name is a hard link to a data file. The primary file name 512 may be a hard link to the predecessor version 508. The predecessor version 508 may be immutable unless changed by the administrator account 502. The primary user account 504 may install a user successor version 516 of the computer application 112 on a user account level. A user account level installation installs a user successor version 516 for the primary user account 504 but forgoes any update to a secondary user account 506. The primary user account 504 may place the user successor version 516 in a clean user successor data storage location 518. A clean data storage location does not currently have a version of the computer application 112 stored at that location. A data storage location may be considered clean even if a version of the computer application 112 was present at the data storage location and then conventionally erased, such as through deregistering. The user successor version 516 may be a complete successor version of the computer application 112. A complete successor version is a successor version that has not been compared with the predecessor version and not had any overlapping sections removed. The primary file name 512 may be overwritten to be a hard link to the user successor version 516.

[0024] A secondary user account 506 may store a secondary file name 520 of the computer application 112 at a secondary name data storage location 522. The secondary file name 520 may be a hard link to the predecessor version 508. The secondary file name 520 may maintain the hard link to the predecessor version 508 even as the primary file name 512 becomes a hard link to the user successor version 516.

[0025] An administrator account 502 may replace the predecessor version 508 with an administrator successor version 524 of the computer application 112 on a machine level. The administrator account 502 may place the administrator successor version 524 in a clean administrator successor data storage location 524. The operating system may
5 elevate a user status for a user account to an administrative privilege to make updates at a machine level. Updates on a machine level install a successor version to each user account on the user device 110. The administrator account 502 may reset the primary file name 512 and the secondary file name 520 to be hard links to the administrator successor version 524.

10 [0026] FIG. 6 illustrates, in a flowchart, one embodiment of a method 600 for upgrading a computer application 112. The application updater may execute a predecessor version 508 of a computer application 112 (Block 602). The application updater may execute an updater module 114 invisibly to a user (Block 604). The application updater may download a complete successor version 516 using the updater module 114 disregarding
15 the predecessor version 508 (Block 606). If the installation is on a per machine basis (Block 608), the application updater may elevate the user status to an administrative privilege (Block 610). The application updater may install the complete successor version 524 of the computer application 112 on a machine level (Block 612). If the installation is on a per user basis (Block 608), the application updater may install a complete successor
20 version 516 of the computer application on a user account level (Block 614). The application updater may maintain execution of the predecessor version 508 while installation of the complete successor version 516 occurs (Block 616). The application updater may place the complete successor version 516 in a clean successor data storage location 514 (Block 618). The application updater may execute the predecessor version
25 508 and the complete successor version 516 of the computer application 112 simultaneously (Block 620). The application updater may execute a switch from the predecessor version 508 to the complete successor version 516 (Block 622). The application updater may maintain a presentation of a user interface of the computer application 112 during the switch (Block 624). The application updater may schedule the
30 predecessor version 508 for an uninstall operation (Block 626). If the uninstall operation is scheduled during a system reboot (Block 628), the application updater may uninstall predecessor version 508 during the system reboot (Block 630). Otherwise, the application updater may uninstall the predecessor version 508 while executing the complete successor version 516 (Block 632).

[0027] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms for
5 implementing the claims.

[0028] Embodiments within the scope of the present invention may also include non-transitory computer-readable storage media for carrying or having computer-executable instructions or data structures stored thereon. Such non-transitory computer-readable storage media may be any available media that can be accessed by a general purpose or
10 special purpose computer. By way of example, and not limitation, such non-transitory computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to carry or store desired program code means in the form of computer-executable instructions or data structures. Combinations of the above should
15 also be included within the scope of the non-transitory computer-readable storage media.

[0029] Embodiments may also be practiced in distributed computing environments where tasks are performed by local and remote processing devices that are linked (either by hardwired links, wireless links, or by a combination thereof) through a communications network.

[0030] Computer-executable instructions include, for example, instructions and data which cause a general purpose computer, special purpose computer, or special purpose processing device to perform a certain function or group of functions. Computer-executable instructions also include program modules that are executed by computers in stand-alone or network environments. Generally, program modules include routines,
20 programs, objects, components, and data structures, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of the program code means for executing steps of the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represents examples of corresponding
25 acts for implementing the functions described in such steps.

[0031] Although the above description may contain specific details, they should not be construed as limiting the claims in any way. Other configurations of the described embodiments are part of the scope of the disclosure. For example, the principles of the disclosure may be applied to each individual user where each user may individually

deploy such a system. This enables each user to utilize the benefits of the disclosure even if any one of a large number of possible applications do not use the functionality described herein. Multiple instances of electronic devices each may process the content in various possible ways. Implementations are not necessarily in one system used by all end users.

- 5 Accordingly, the appended claims and their legal equivalents should only define the invention, rather than any specific examples given.

CLAIMS:

1. A machine-implemented method, comprising:

executing a predecessor version of a computer application in a primary user account of a computer device;
- 5 installing a successor version of the computer application for the primary user account; and

executing the successor version of the computer application in the primary user account while a secondary user account of the computer device maintains a hard link to the predecessor version.
- 10 2. The method of claim 1, further comprising:

downloading a complete successor version using an updater module.
3. The method of claim 2, further comprising:

executing the updater module invisibly to a user.
4. The method of claim 1, further comprising:

15 placing the successor version in a clean successor data storage location.
5. The method of claim 1, further comprising:

maintaining execution of the predecessor version while installation of the successor version occurs.
6. The method of claim 1, further comprising:

20 executing a switch from the predecessor version to the successor version.

7. The method of claim 6, further comprising:

maintaining a presentation of a user interface of the computer application during the switch.

8. The method of claim 1, further comprising:

5 subsequent to installing the successor version of the computer application for the primary user account, for at least a period of time, executing the predecessor version and the successor version simultaneously.

9. The method of claim 1, further comprising:

uninstalling the predecessor version while executing the successor version.

10 10. A tangible machine-readable medium having a set of instructions detailing a method stored thereon that when executed by one or more processors cause the one or more processors to perform the method, the method comprising:

executing a predecessor version of a computer application referenced by a primary file name of a primary user account of a computer device;

15 downloading a complete successor version using an updater module disregarding the predecessor version;

installing the complete successor version of the computer application referenced by the primary file name in a clean successor data storage location, while a secondary user account of the computer device maintains a hard link to the predecessor version using a
20 secondary file name.

11. The tangible machine-readable medium of claim 10, wherein the method further comprises:

installing the complete successor version of the computer application on a machine level.

12. The tangible machine-readable medium of claim 11, wherein the method further comprises:

elevating a user status to an administrative privilege.

13. The tangible machine-readable medium of claim 10, wherein the method further
5 comprises:

installing the complete successor version of the computer application on a user account level.

14. The tangible machine-readable medium of claim 10, wherein the method further comprises:

10 executing the updater module invisibly to a user.

15. The tangible machine-readable medium of claim 10, wherein the method further comprises:

resetting the primary file name to be a hard link to an administrator successor version.

15 16. The tangible machine-readable medium of claim 15, wherein the method further comprises:

maintaining a presentation of a user interface during resetting the primary file name to be the hard link to the administrator successor version.

17. The tangible machine-readable medium of claim 10, wherein the method further
20 comprises:

subsequent to installing the successor version of the computer application for the primary user account, for at least a period of time, executing the predecessor version and the complete successor version simultaneously.

18. The tangible machine-readable medium of claim 10, wherein the method further comprises:

scheduling the predecessor version for an uninstall operation during a system reboot.

19. An application updater, comprising:

5 a communications interface that downloads a complete successor version of a computer application;

a data storage that stores a predecessor version of the computer application in a predecessor data storage location and the complete successor version in a clean successor data storage location; and

10 a processor that executes the predecessor version and installs the complete successor version of the computer application in a primary user account while a secondary user account maintains a hard link to the predecessor version.

20. The application updater of claim 19, wherein the processor maintains execution of the predecessor version while installation of the complete successor version occurs.

15 21. An application updater on a computing device, comprising:

a data storage on the computing device, the data storage configured to store multiple application profiles for multiple users of a computer application on the computing device to reference the computer application with a primary file name for a primary user account of the computing device and with a secondary file name for a secondary user account
20 of the same computing device as the primary user account, and to store a predecessor version of the computer application stored in a predecessor location in the data storage and a complete successor version stored in a clean successor data storage location; and

a processing core having at least one processor configured to execute the predecessor version and configured to execute an installer module to install the complete
25 successor version of the computer application in the primary user account referenced by the

primary file name without interfering with performance of the predecessor version and not install the complete successor version of the computer application for the secondary user account on the computing device, such that the secondary user account has the predecessor version referenced by the secondary file name when the primary user account has the

5 complete successor version referenced by the primary file name.

22. The application updater of claim 21, wherein the installer module is configured to install the complete successor version of the computer application on a machine level.

23. The application updater of claim 22, wherein the primary user account is configured to elevate a user status to an administrative privilege to install on the machine level.

10 24. The application updater of claim 21, wherein the installer module is configured to install the complete successor version of the computer application on a user account level.

25. The application updater of claim 21, wherein the processing core is further configured to execute an updater module to download the complete successor version invisibly to a user.

15 26. The application updater of claim 21, wherein the installer module is configured to reset the primary file name to be a hard link to an administrator successor version.

27. The application updater of claim 26, further comprising:

a display configured to maintain a presentation of a user interface for the computer application during the switch.

20 28. The application updater of claim 21, wherein the processing core is further configured to execute a switch from the predecessor version to the complete successor version.

29. The application updater of claim 21, wherein the processing core is further configured to schedule the predecessor version for an uninstall operation during a system reboot.

30. A computing device, having a memory to store a series of instructions that are executed by at least one processor to implement an updater module, the computing device configured to

store multiple application profiles for a computer application on the computing device for multiple users of the computing device, with a primary user account of the computing device referencing the computer application with a primary file name and a secondary user account of the computing device referencing the computer application with a secondary file name;

execute a predecessor version of the computer application referenced by the primary file name in the primary user account;

install a successor version of the computer application for the primary user account in a clean successor data storage location without interfering with the performance of the predecessor version and not install a successor version of the computer application for the secondary user account on the computing device; and

execute the successor version of the computer application in the primary user account referenced by the primary file name while the secondary user account references the predecessor version with the secondary file name.

31. The computing device of claim 30, further comprising:

download a complete successor version using an updater module.

32. The computing device of claim 31, further comprising:

execute the updater module invisibly to a user.

33. The computing device of claim 30, further comprising:

maintain execution of the predecessor version while installation of the successor version occurs.

34. The computing device of claim 30, further comprising:

execute a switch from the predecessor version to the successor version.

35. The computing device of claim 34, further comprising:

maintain a presentation of a user interface of the computer application during the
5 switch.

36. The computing device of claim 30, further comprising:

subsequent to installing the successor version of the computer application for the
primary user account, for at least a period of time, execute the predecessor version and the
successor version simultaneously.

10 37. The computing device of claim 30, further comprising:

uninstall the predecessor version while executing the successor version.

38. A machine-implemented method, comprising:

executing a predecessor version of a computer application referenced by a primary
file name of a primary user account of a computer device;

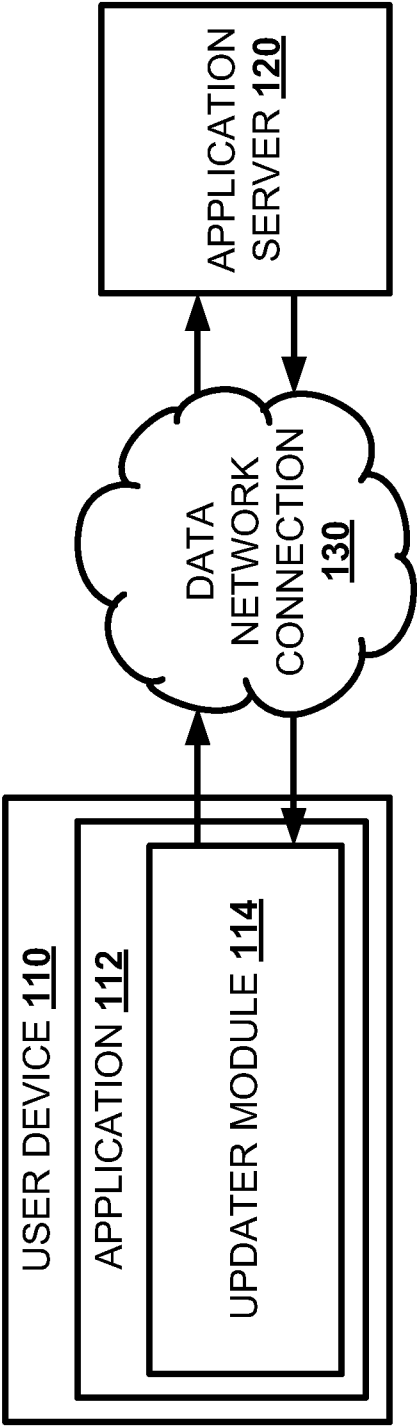
15 downloading a complete successor version using an updater module while
disregarding the predecessor version;

installing the complete successor version of the computer application referenced by
the primary file name in a clean successor data storage location on the computer device
without interfering with performance of the predecessor version, and not installing the
20 complete successor version of the computer application for a secondary user account on the
computer device, the primary user account and the secondary user account being on a same
computer device; and

executing the successor version of the computer application in the primary user account of the computer device referenced by the primary file name while the secondary user account of the computer device references the predecessor version with a secondary file name.

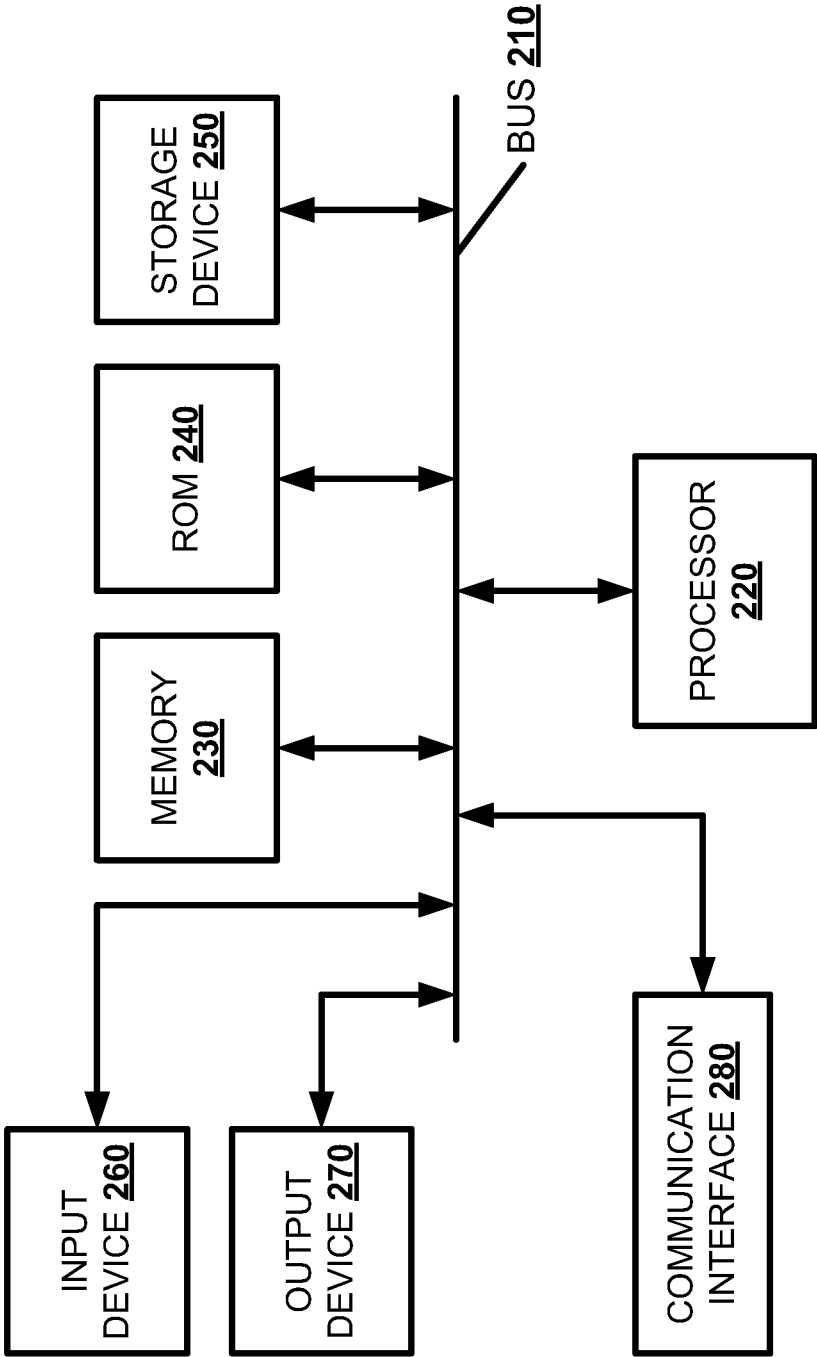
39. The method of claim 38, wherein the method further comprises:

5 resetting the primary file name to be a hard link to an administrator successor version.



100

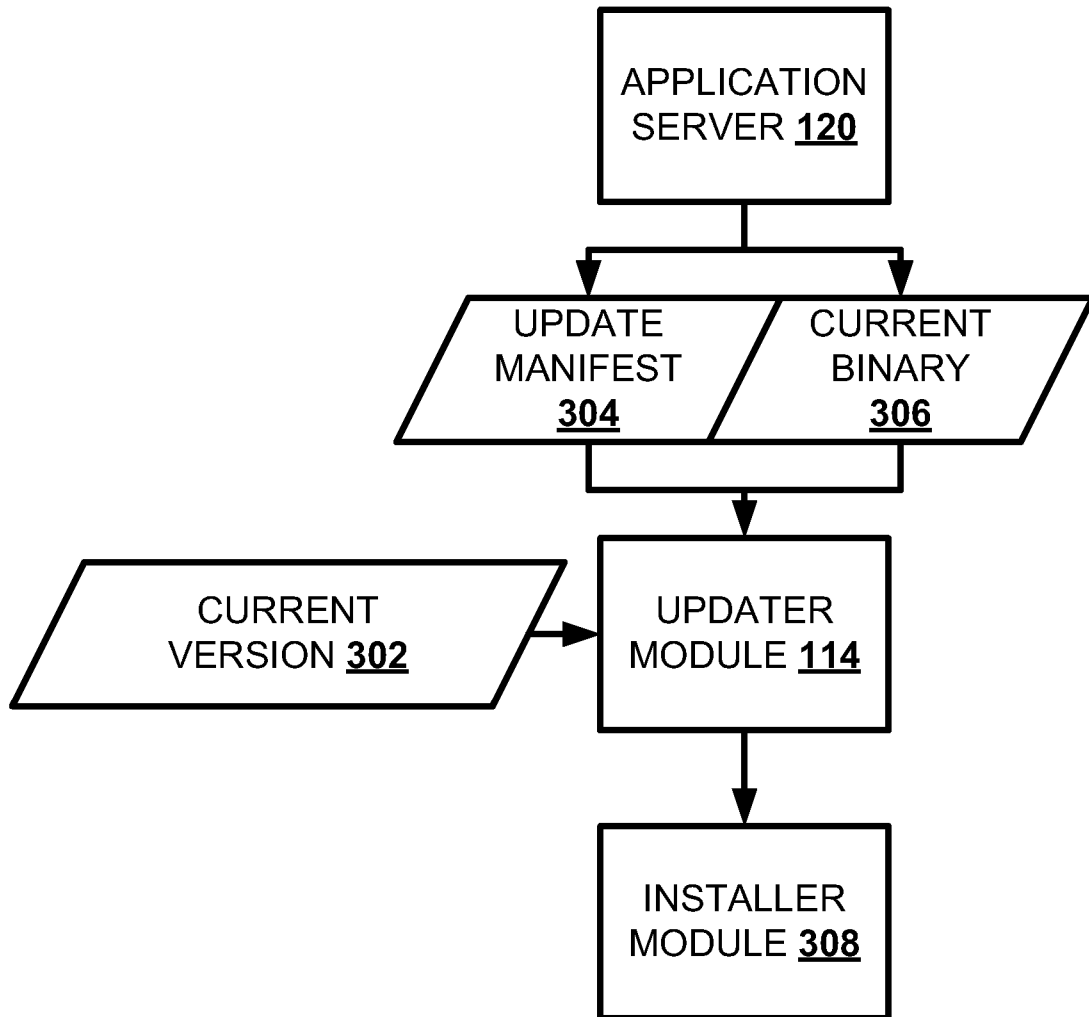
Figure 1

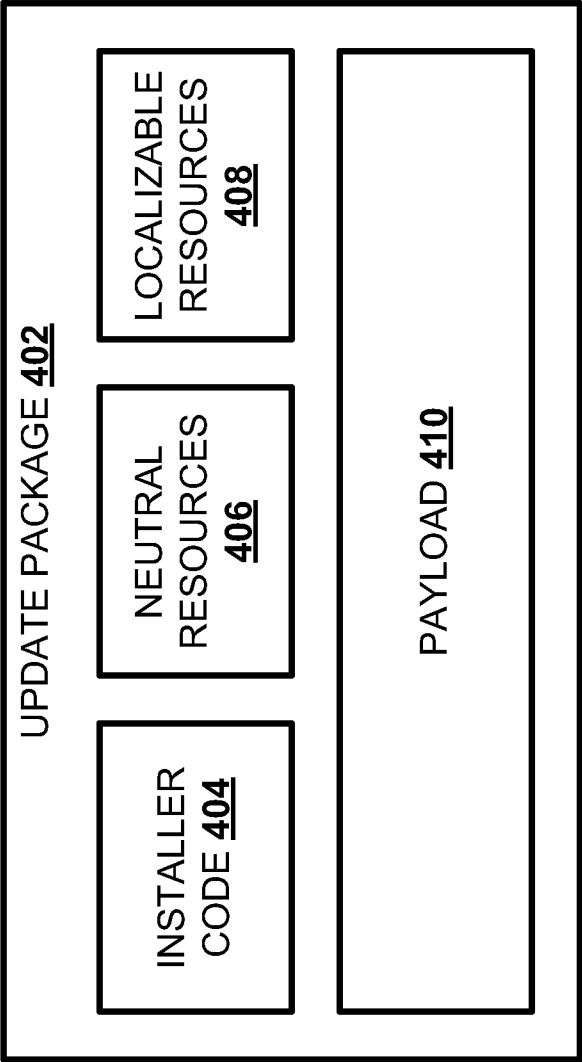


200

Figure 2

3/6

300**Figure 3**



400

Figure 4

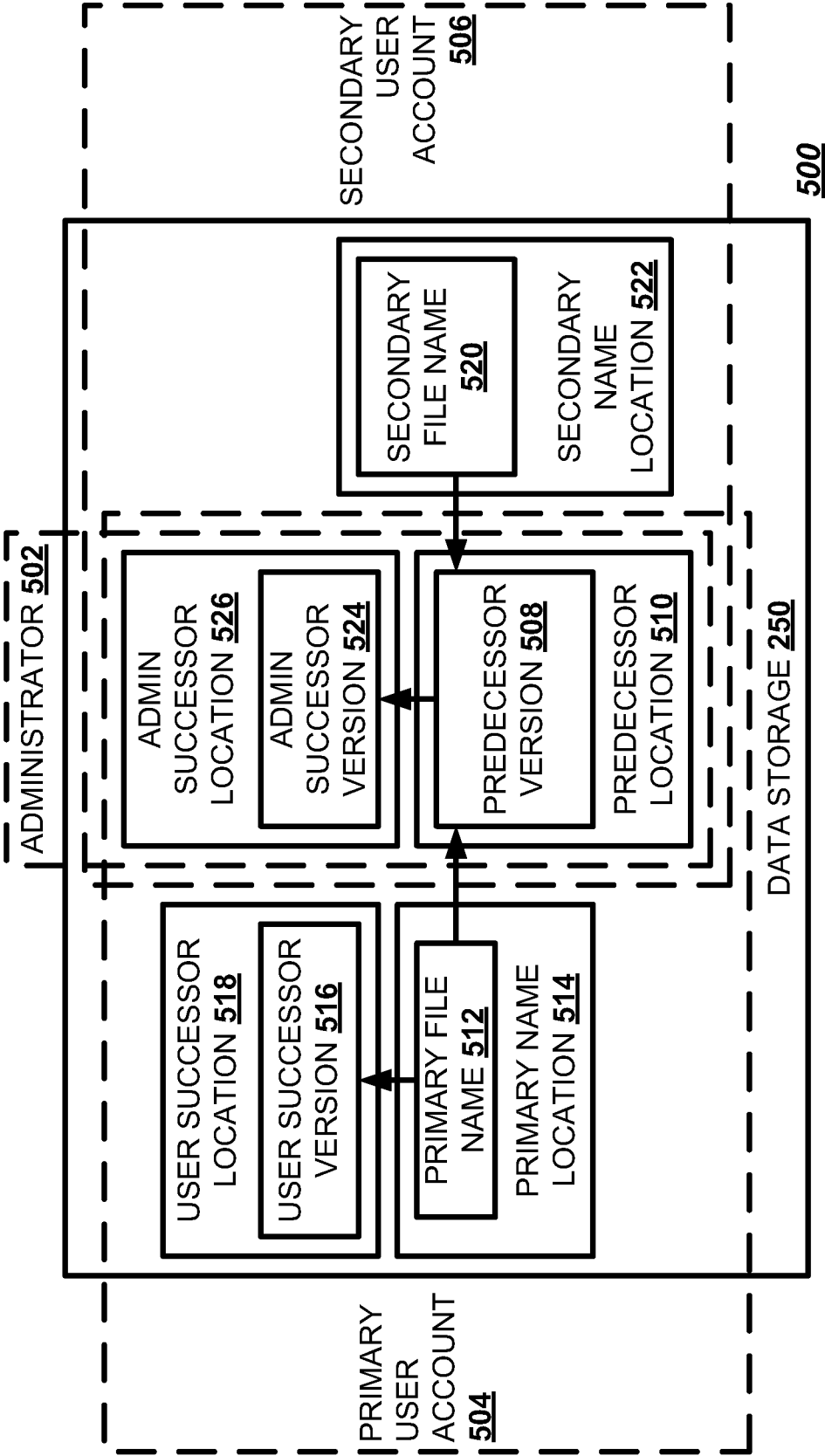
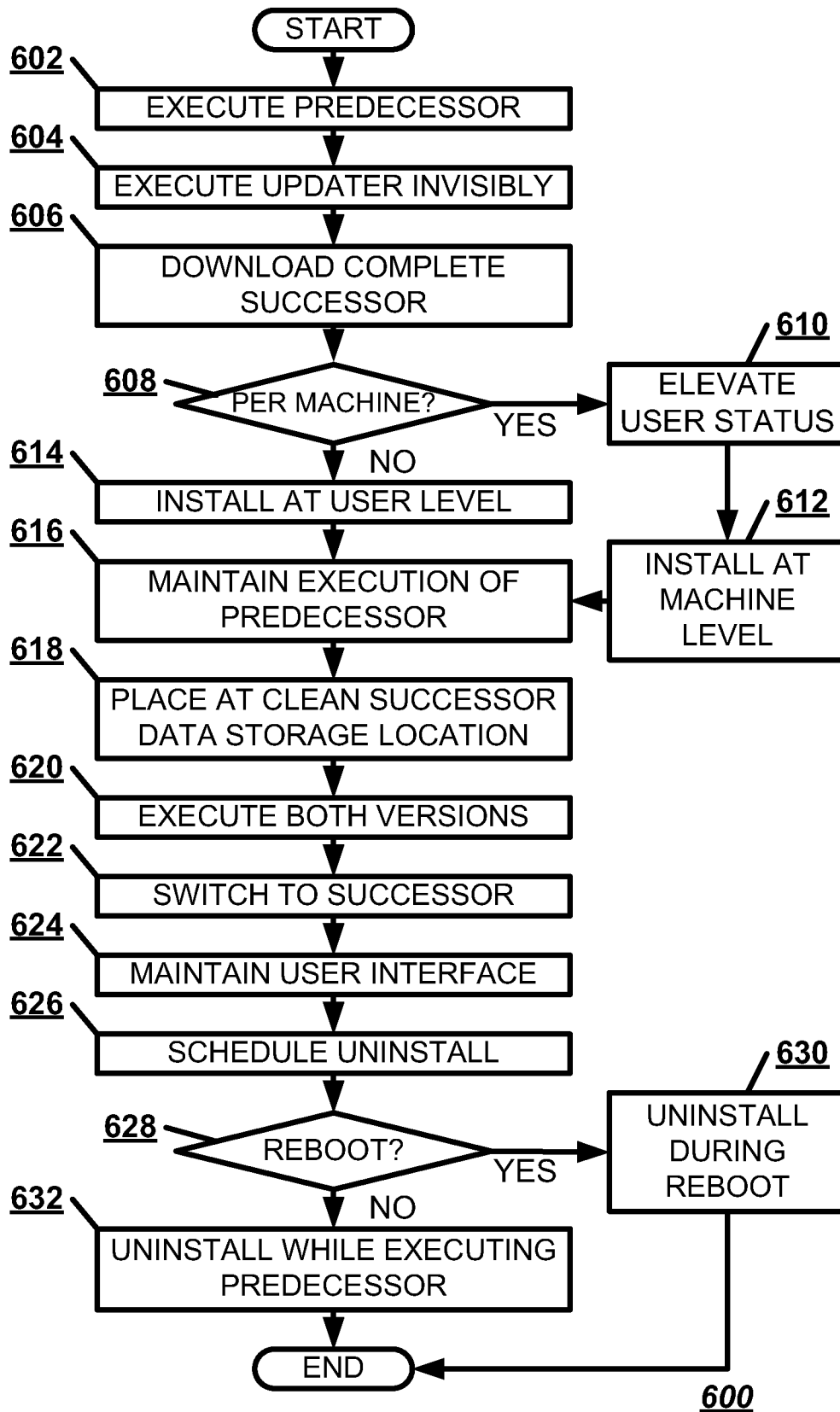


Figure 5

6/6

**Figure 6**

