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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 November 2016 (10.11.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/178681 A1

(51) International Patent Classification:
G06F 15/16 (2006.01) *G06F 9/22* (2006.01)

(21) International Application Number:
PCT/US2015/029533

(22) International Filing Date:
6 May 2015 (06.05.2015)

(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) Inventors: **CHANG, Yun-Cheng**; 10F-1, No. 66, Jingmao 2nd Rd, NanGang District, Taipei, 11568 (TW). **KAO, Liang-Chin**; 10F-1, No. 66, Jingmao 2nd Rd, NanGang District, Taipei, 11568 (TW). **HUNG, Shang-Ching**; 10F-1, No. 66, Jingmao 2nd Rd, NanGang District, Taipei, 11568 (TW). **SHENG, Yu Cheng**; 10F-1, No. 66, Jingmao 2nd Rd, NanGang District, Taipei, 11568 (TW).

(74) Agents: **ADEKUNLE, Olaolu O** 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))*

[Continued on next page]

(54) Title: PRE-OPERATING SYSTEM CONTENT TRANSMISSION

(57) Abstract: The present subject matter relates to pre-Operating System (pre-OS) content transmission. In an example, the present subject matter includes receiving a request from a remote device to access contents of a screen of a pre-OS program. The pre-OS program is executable prior to loading an OS. The present subject matter further includes sending a request to the pre-OS program for translated contents. The translated contents are contents of the screen of the pre-OS program that are translated into a web-based language. Further, the present subject matter includes transmitting the translated contents to the remote device using a web protocol.

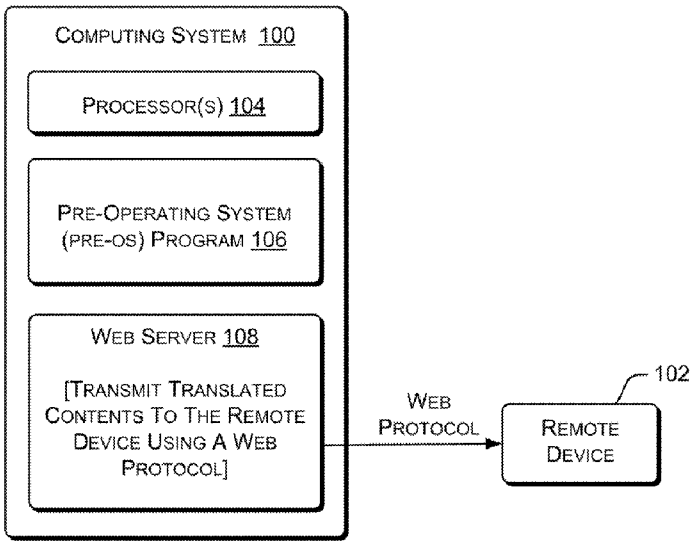

FIG. 1

WO 2016/178681 A1

— *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))*

PRE-OPERATING SYSTEM CONTENT TRANSMISSION

BACKGROUND

[0001] When a computing system is powered ON, a number of programs are executed on the computing system to perform various tasks, before an Operating System (OS) is loaded on the computing system. Examples of such tasks include initializing and testing of components of the computing system, power management, and the like. The programs may be referred to as pre-OS programs as these are executed prior to loading of the OS. A user may provide an input to the computing system, for example, to change settings of the tasks performed by the pre-OS programs.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] The following detailed description references the drawings, wherein:

[0003] FIG. 1 illustrates an interaction between a computing system and a remote device for transmission of contents of a pre-Operating System (pre-OS) program, according to an example of the present subject matter;

[0004] FIG. 2 illustrates a block diagram of a computing system for transmission of contents of a pre-OS program, according to an example of the present subject matter;

[0005] FIG. 3 illustrates a block diagram of a computing system for transmission of contents of a pre-OS program, according to another example of the present subject matter;

[0006] FIG. 4 illustrates a call flow diagram for transmitting contents of a pre-OS program running on a computing system to a remote device, according to an example of the present subject matter;

[0007] FIG. 5 is a flowchart of a method for providing contents of a pre-OS program running on a computing system to a remote device, according to an example of the present subject matter;

[0008] FIG. 6 is a flowchart of a method for providing contents of a pre-OS program running on a computing system to a remote device, according to another example of the present subject matter; and

[0009] FIG. 7 is a block diagram of a network environment implementing a non-transitory computer-readable medium, for providing contents of a pre-OS program running on a computing system to a remote device, according to an example of the present subject matter.

DETAILED DESCRIPTION

[0010] When a computing system is powered ON, pre-Operating System (pre-OS) programs are executed in a pre-OS environment prior to loading an Operating System (OS). A user can view and change settings for the computing system using the pre-OS programs. According to an example, when a computing system is starting up, the user may press a particular key on a keyboard to enter into a Basic Input/Output System (BIOS) setup screen. Once the user has entered into the BIOS setup screen, the user can view and change BIOS settings for the computing system. In an example, the user can change memory settings, configure a new hard drive, change boot order, and reset BIOS password.

[0011] A user can also remotely access the pre-OS program screen of the computing system from a remote device to view and change the settings. For example, the user can remotely access a screen of the pre-OS program using a controller, such as a Baseboard Management Controller (BMC), on the computing system and a remote console on the remote device. The BMC may also include a Video Graphics Array (VGA) Controller. The VGA controller allows for graphical contents of the pre-OS program screen to be redirected to the remote device over a network. Accordingly, the user can remotely perform various tasks, such as reconfiguring hardware settings, power management features, and power supply controls.

[0012] In the absence of a BMC on a computing system, the user may not be able to remotely access a screen of a pre-OS program. Further, there may be a case where the BMC is present on the computing system, but a VGA controller is not present. In such a case, deploying such additional hardware for accessing a pre-OS screen remotely would be costly and inefficient. Moreover, the VGA controller consumes additional power and may not be deployable in low power

or low resource computing systems, such as mobile devices. Therefore, in such a case also, the user may not be able to access the screen of the pre-OS program using a remote device.

[0013] Moreover, generally a remote console has to be installed on a remote device to enable a user to view and change settings of pre-OS programs running on a separate system, such as a computing system. The remote console is usually implemented in a separate hardware component which adds up to the cost of pre-OS content transmission.

[0014] Approaches for transmitting contents of a screen of a pre-Operating System (pre-OS) program are described. In an example, the contents of the screen of the pre-OS program may be redirected to a remote device. In the present example, the contents of the screen of the pre-OS program are translated into a web-based language. Thereafter, the translated contents are transmitted to the remote device using a web protocol. Thus, the contents of the pre-OS program screen can be transmitted from a computing system to the remote device without a controller, such as a BMC, or with a BMC that does not have a VGA controller in the computing system, or without a remote console in the remote device. Consequently, directing the contents of the screen of the pre-OS program from the computing system to the remote device using the web protocol would lead to reduced resource consumption, such as memory consumption and processor consumption, on the computing system. This would further result in reduced cost and increased efficiency in remote management of the computing system.

[0015] In accordance with an example implementation, contents of a screen of a pre-OS program running on a computing system may be redirected to a remote device on receiving a request from the remote device. The pre-OS program is executable prior to loading an OS on the computing system. In an example, the pre-OS program may be one of a Basic Input/Output System (BIOS) program and a Unified Extensible Firmware Interface (UEFI) program. Accordingly, examples of the screen include, but are not limited to, a BIOS setup screen and a BIOS Power-On Self-Test (POST) screen.

[0016] On receiving the request from the remote device to access the

contents of the screen of the pre-OS program, the contents of the screen are translated into a web-based language. For example, the contents of the screen may be translated into one of a HyperText Markup Language (HTML) language, a JavaScript (JS) language, an EXtensible Markup Language (XML) language, and an Asynchronous JavaScript and XML (AJAX) language. In an example, translation into the AJAX language can provide enhanced user experience on the remote device. The translated contents can be transmitted to a web browser on the remote device using a web protocol. In an example, the web protocol is a Hypertext Transfer Protocol (HTTP). Thus, a user of the remote device may view the contents of the screen of the pre-OS program, running on a separate system, on the web browser. The user is also able to remotely change the settings of the pre-OS program using the approaches described herein.

[0017] With the approaches described herein, cost, power consumption, and resources associated with transmission of the contents of the screen of the pre-OS program running on the computing system to the remote device are substantially reduced. Although general concepts related to the claimed subject matter have been described in conjunction with the BIOS program, any other program that is executable in a pre-OS environment would also be within the scope of the present subject matter.

[0018] The various approaches are further described in conjunction with the following figures. It should be noted that the description and figures merely illustrate the principles of the present subject matter. Further, various arrangements may be devised that, although not explicitly described or shown herein, embody the principles of the present subject matter and are included within its scope.

[0019] The above approaches are further described with reference to FIGS. 1 to 7. Further, while aspects of described system and method for pre-OS content transmission can be implemented in any number of different computing systems, environments, and/or implementations, the examples and implementations are described in the context of the following system(s).

[0020] FIG. 1 illustrates an interaction between a computing system 100 and a remote device 102 for transmission of contents of a pre-operating system (pre-OS)

program, according to an example of the present subject matter. The computing system 100 may be a computing device, such as a laptop computer, a desktop computer, a workstation, or a server. Further, the remote device 102 may also be a computing device, such as a laptop computer, a desktop computer, a workstation, a mobile device, or a server. In an example, the remote device 102 may be a touch based computing device. In an example implementation, a user of the remote device 102 may remotely access the computing system 100 to view and change settings for the computing system 100 using pre-OS programs running on the computing system 100. A pre-OS program is a program that is executable prior to loading an OS.

[0021] The computing system 100 includes processor(s) 104. The processor(s) 104 may be implemented as microprocessors, microcomputers, microcontrollers, digital signal processors, central processing units, state machines, logic circuitries, and/or any devices that manipulate signals based on operational instructions. The functions of various elements shown in FIG. 1, including a functional block labeled as "processor(s)", may be provided through the use of dedicated hardware as well as hardware capable of executing machine readable instructions.

[0022] The computing system 100 further includes a pre-OS program 106, executable by the processor 104. In an example, the pre-OS program 106 may be a Basic Input/Output System (BIOS) program. In another example, the pre-OS program 106 may be a Unified Extensible Firmware Interface (UEFI) program. The computing system 100 further includes a web server 108, executable by the processor 104. The web server 108 accepts and processes web protocol requests, such as Hypertext Transfer Protocol (HTTP) requests. In an example, the web server 108 stores, processes, and delivers web pages or web applications to client devices, such as the remote device 102. The communication between the web server 108 and the client devices may take place using HTTP.

[0023] The pre-OS program 106 and the web server 108, amongst other things, may include routines, programs, objects, components, and data structures, which perform particular tasks or implement particular abstract data

types. The pre-OS program 106 and the web server 108 may also be implemented as, signal processor(s), state machine(s), logic circuitries, and/or any other device or component that manipulates signals based on operational instructions. Further, the pre-OS program 106 and the web server 108 can be implemented by hardware, by computer-readable instructions executed by a processing unit, or by a combination thereof.

[0024] In operation, the web server 108 may receive a request from the remote device 102 to access contents of a screen of the pre-OS program 106. Examples of the screen include, but are not limited to, a BIOS setup screen and a BIOS Power-On Self-Test (POST) screen. Further, the contents of the screen may include one of graphical contents, text contents, and a combination of the graphical contents and the text contents. On receiving the request from the remote device 102 to access the contents of the screen of the pre-OS program 106, the web server 108 may send a request to the pre-OS program 106 for translated contents. In an example, the translated contents may be understood as contents of the screen of the pre-OS program 106 that are translated into a web-based language. According to an example, the web-based language is one of a HyperText Markup Language (HTML) language, a JavaScript (JS) language, an EXtensible Markup Language (XML) language, and an Asynchronous JavaScript and XML (AJAX) language. Thereafter, the web server 108 may transmit the translated contents to the remote device 102 using a web protocol, such as HTTP. These and other aspects are further described in conjunction with FIG. 2, FIG. 3, and FIG. 4.

[0025] FIG. 2 illustrates a block diagram of the computing system 100 for transmission of contents of the pre-OS program 106, according to an example of the present subject matter. In addition to the processor(s) 104, the computing system 100 includes interface(s) 202. The interface(s) 202 may include a variety of interfaces, for example, interfaces for peripheral device(s), such as data input output devices, referred to as I/O devices, storage devices, and/or network device. The I/O device(s) may include Universal Serial Bus (USB) ports, Ethernet ports, host bus adaptors, and their corresponding device drivers. The interface(s) 202 may be used for facilitating communication between the

computing system 100 and various other computing devices, such as the remote device 102 (not shown in FIG. 2).

[0026] The computing system 100 further includes a memory 204 coupled to the processor(s) 104. Among other capabilities, the processor(s) 104 may fetch and execute computer-readable instructions stored in the memory 204. The memory 204 may include any computer-readable medium known in the art including, for example, volatile memory, such as static random access memory (SRAM) and dynamic random access memory (DRAM), and/or non-volatile memory, such as read only memory (ROM), erasable programmable ROM, and flash memories.

[0027] As described earlier, the computing system 100 includes the pre-OS program 106 and the web server 108. As shown in the FIG. 2, in an example, the web server 108 is provided in the pre-OS program 106. Accordingly, the web server 108 is executable by the pre-OS program 106. The computing system 100 also includes module(s) 206 and data 208. The module(s) 206 include a content translator 210 and other module(s) 212. The content translator 210 may be capable of communicating with the pre-OS program 106. The other module(s) 212 may include programs or coded instructions that supplement applications or functions performed by the computing system 100. The data 208 includes screen contents 214, translated contents 216, and other data 218. The other data 218 may include data generated and/or saved by the modules 206 for providing various functionalities of the computing system 100.

[0028] In an example implementation, when the computing system 100 is powered ON, the pre-OS program 106 is executed on the computing system 100. Further, when a connection between the computing system 100 and the remote device 102 is established, the web server 108 may receive a request from the remote device 102 to access contents of a screen of the pre-OS program 106 running on the computing system 100. The pre-OS program 106 may be a BIOS program. Further, examples of the screen (also referred to as pre-OS program screen) of the pre-OS program 106 include a BIOS setup screen, a BIOS POST screen, a text mode screen, and a Graphical User Interface (GUI) application screen. The contents of the screen of the pre-OS

program 106 may include one of graphical contents, text contents, and a combination of the graphical contents and the text contents.

[0029] In an example, the request to access the contents of the pre-OS program screen may be received from a user of the remote device 102, for example, a system administrator. In an example, the request may be received when the user wishes to view, change, or update settings of the pre-OS program screen from the remote device 102. To be able to view, change, or update the settings, the user may send a request to the web server 108 using a web browser on the remote device 102. For instance, the user may send the request to the web server 108 using an Application Program Interface (API), such as XMLHttpRequest (XHR), available to the web browser. The API may be used to send one or more HTTP requests to the web server 108 and load response data of the web server 108 on the web browser.

[0030] In a case when the web server 108 is inactive, an error message may be displayed on the remote device 102. The web server 108 can be considered to be inactive if the computing system 100 is powered OFF, a connection between the computing system 100 and the remote device 102 is not established, a network capability for the web server 108 is not ready, or when the web server 108 is itself not ready. In an example, if the computing system 100 is powered OFF or if the network capability is not ready for the web server 108, then an error message “Network error” may be displayed on the web browser of the remote device 102. In another example, if the computing system 100 is powered ON and the network capability is ready, but the web server 108 is not ready, then an error message “HTTP 503 Service Unavailable” may be displayed on the web browser of the remote device 102. In such a case, the user of the remote device 102 may have to refresh the web browser to connect to the web server 108.

[0031] Continuing with the present implementation, on receiving the request from the remote device 102, the web server 108 may send a request to the pre-OS program 106 for the contents of the pre-OS program screen. The pre-OS program 106 may provide the contents of the screen to the content translator 210 for translation. Further, the pre-OS program 106 may store the contents of

the screen as the screen contents 214.

[0032] Thereafter, the content translator 210 may translate the contents of the pre-OS program screen into a web-based language to generate web-based contents. The web-based language may be one of a HTML language, a JS language, an XML language, and an AJAX language.

[0033] In an example, the content translator 210 may also embed a code in the translated content for receiving user input. The code may capture an activity performed on the web browser of the remote device 102, such as an input provided by a user, and may translate the activity into a web-based language.

[0034] In an example, the translated web-based contents may correspond to a web page or a web application. A web page may be a document created using a web-based language that is accessible using a web browser. The web page may include text, graphics, and hyperlinks to other web pages and files. Further, a web application may be a program that is coded in a browser-supported programming language, for example, a JS language.

[0035] In an example, the translated contents include web elements in the web-based language format. For instance, the translated contents may include HTML elements, such as image elements, button elements, static text elements, checkbox elements, and text-input elements. According to an example, a Graphical User Interface (GUI) foundation, such as a Pre-OS BIOS GUI foundation may be used to translate the contents of the pre-OS program screen into the web-based language. The GUI foundation may provide an abstract layer to a user for creating GUI applications and elements, such as form elements, the button elements, and the checkbox elements. In an example, the user may create HTML elements on the abstract layer using the GUI foundation. The GUI applications and the elements may be a part of the content translator 210, and may be used by the content translator 210 for translating the contents of the pre-OS program screen into a web-based language. The GUI foundation also provides an interface for creating the embedded code for capturing user interaction with the web browser on the remote device 102.

[0036] Further, the content translator 210 may provide the translated contents to the pre-OS program 106 for provision to the web server 108. In an example,

the content translator 210 may store the translated contents as the translated contents 216. On receiving the translated contents, the pre-OS program 106 may provide the translated contents to the web server 108.

[0037] Upon receiving the translated contents from the pre-OS program 106, the web server 108 transmits the translated contents to the remote device 102 using a web protocol. The web protocol may be HTTP. In case the translated contents are not ready for being provided to the web server 108, a default web page may be displayed on the remote device 102. In an example, if the translated contents are not ready, a default web page showing a message "Waiting for the pre-OS program screen" may be displayed on the web browser of the remote device 102.

[0038] Thus, the contents of the screen of the pre-OS program 106 running on the computing system 100 are directed to the web browser on the remote device 102. In an example, representation of the contents of the pre-OS program screen on the remote device 102 may be different from representation of the contents of the pre-OS program screen on the computing system 100. Further, the representation of the contents of the pre-OS program screen may be different on different web browsers on the remote device 102. While the representation may be different, various elements of the pre-OS program screen and functionalities of the elements may remain the same.

[0039] Once the contents of the pre-OS program screen are provided to the remote device 102, the user of the remote device 102 may interact with the pre-OS program screen using input devices of the remote device 102, such as a mouse, a touchscreen, a touchpad, and a keyboard. In an example, the web server 108 may receive an input from the remote device 102. The input may correspond to a user event at the remote device 102. For instance, when the user clicks on the mouse, types on the keyboard, or touches a screen of the remote device 102, the user event occurs.

[0040] For example, the translated contents may include the web elements, such as the image elements, the button elements, the static text elements, the checkbox elements, and the text-input elements. Therefore, the user event on the remote device 102 may be a click on the button element on a web page on

the web browser. In an example, the user may click on the button element to change or update the settings of the pre-OS program 106. As described above, the translated contents include an embedded code. The code captures an activity, such as the user event, performed on the web browser of the remote device 102 and translates the activity into a web-based language event, which is then sent by the web browser as an input to the web server 108.

[0041] On receiving the input, the web server 108 may provide the input to the pre-OS program 106 for translation into a computer processable command related to the pre-OS program 106 and execution of the translated command. The pre-OS program 106 may then send a request to the content translator 210 to translate the input into the computer processable format. In an example, the computer processable command corresponds to receipt of the input from one of the mouse of the remote device 102, the keyboard of the remote device 102, the touchpad of the remote device 102, and the touchscreen of the remote device 102. Thereafter, the content translator 210 may provide the translated input to the pre-OS program 106. Once the pre-OS program 106 receives and executes the translated input, the screen of the pre-OS program 106 is updated as changes or modifications made by the user of the remote device 102 are reflected on the pre-OS program 106 running on the computing system 100.

[0042] According to an example implementation, the content translator 210 may also translate contents of the updated pre-OS program screen into a web-based language. The content translator 210 may then provide the translated contents of the updated pre-OS program screen to the pre-OS program 106 for provision to the web server 108. The web server 108 may then transmit the translated contents to the remote device 102 using a web protocol.

[0043] Although, it is described that the content translator 210 is implemented external to the pre-OS program 106, in an example implementation, the content translator 210 may be implemented inside the pre-OS program 106.

[0044] FIG. 3 illustrates a block diagram of a computing system 100 for transmission of contents of the pre-OS program 106, according to another example of the present subject matter.

[0045] In addition to the processor(s) 104, the interface(s) 202, the memory

204, the pre-OS program 106, the web server 108, the module(s) 206, and the data 208, the computing system 100 also includes a controller 302. The controller 302 may be implemented as, signal processor(s), state machine(s), logic circuitries, and/or any other device or component that manipulates signals based on operational instructions. Further, the controller 302 can be implemented by hardware, by computer-readable instructions executed by a processing unit, or by a combination thereof.

[0046] In an example, the controller 302 may be a Baseboard Management Controller (BMC). The controller 302 monitors a physical state of the computing system 100, a network server, and other hardware devices. The controller 302 may be in communication with the pre-OS program 106. Further, as shown in the FIG. 3, the web server 108 is provided inside the controller 302. In the current example, the web server 108 is executable by the controller 302.

[0047] According to an example implementation, the web server 108 may receive a request from the remote device 102 to access the contents of the screen of the pre-OS program 106. On receiving the request, the web server 108 may send a request to the pre-OS program 106 for the translated contents. In a case when the pre-OS program 106 is inactive, for example, when the computing system 100 is powered OFF or when a connection between the computing system 100 and the remote device 102 is not established, the pre-OS program 106 may not receive the request from or provide the translated contents to the web server 108. In such a case, a default web page may be displayed on the remote device 102 since the controller 302, and thereby the web server 108, may be active independent of the pre-OS program 106. In an example, if the pre-OS program 106 is inactive, a default web page showing a message “Waiting for the system to be powered ON” may be displayed on a web browser of the remote device 102. Therefore, no error message, such as “Network error” or “HTTP 503 Service Unavailable” may be displayed on the remote device 102 unless the controller 302 is not ready, for example, when the controller 302 is powered OFF.

[0048] Although, it is described that the content translator 210 is implemented external to the pre-OS program 106, in an example implementation, the content

translator 210 may be implemented inside the pre-OS program 106. Further, according to an example implementation, the receipt of the request, the translation of the contents of the screen of the pre-OS program 106, transmission of the translated contents, the receipt of an input from a user of the remote device 102, processing of the input into a computer processable command, and the like, may happen in a similar manner as described in conjunction with FIG. 1 and FIG. 2.

[0049] FIG. 4 illustrates a call flow diagram 400 for transmitting the contents of the pre-OS program 106 running on the computing system 100 to the remote device 102, according to an example of the present subject matter.

[0050] As shown in FIG. 4, at step 402, the remote device 102 sends a request to the web server 108 to access contents of a screen of the pre-OS program 106. In an example, a user of the remote device 102 sends the request to the web server 108 from a web browser of the remote device 102 using an API, such as XHR. The user may be a system administrator who wishes to view or update settings for the computing system 100 using the pre-OS program 106. In an example, the pre-OS program 106 is a BIOS program and the screen is a BIOS setup screen.

[0051] In case the web server 108 is executable by the pre-OS program 106 and is inactive, an error message, such as “HTTP 503 Service Unavailable” may be displayed on the web browser of the remote device 102. The web server 108 is said to be inactive if the computing system 100 is powered OFF, a connection between the computing system 100 and the remote device 102 is not established, a network capability for the web server 108 is not ready, or when the web server 108 is itself not ready.

[0052] In case the web server 108 is executable by the controller 302, such as a BMC, no error message, such as “Network error” or “HTTP 503 Service Unavailable” is displayed on the remote device 102 unless the controller 302 is not ready, as the web server 108 is always active inside the controller 302. However, in this case, if the pre-OS program 106 is inactive, for example, when the computing system 100 is powered OFF or when a connection between the computing system 100 and the remote device 102 is not established, the pre-

OS program 106 may not receive the request from or provide the translated contents to the web server 108. In such a case, a default web page may be displayed on the remote device 102. In an example, if the pre-OS program 106 is inactive, a default web page showing a message “Waiting for the system to be powered ON” may be displayed on the web browser of the remote device 102.

[0053] At step 404, the web server 108 sends a request to the pre-OS program 106 for translated contents. The translated contents are contents of the screen of the pre-OS program 106 that are translated into a web-based language.

[0054] Subsequently, at step 406, the pre-OS program 106 sends a command to the content translator 210. The command may be a request for translating the contents of the screen of the pre-OS program 106 into a web-based language.

[0055] At step 408, the content translator 210 translates the contents of the screen of pre-OS program 106 into the web-based language. According to an example, the web-based language is one of a HTML language, a JS language, an XML language, and an AJAX language. In an example, the content translator 210 may also embed a code into the translated content. The code may capture an activity performed on the web browser of the remote device 102 and translate the activity into a web-based language. In an example, the translated contents, i.e., the web-based contents, may correspond to a web page or a web application. Further, according to an example, the translated contents include web elements, such as HTML elements. The HTML elements may include image elements, button elements, static text elements, checkbox elements, and text-input elements.

[0056] At step 410, the content translator 210 sends the translated contents to the pre-OS program 106 for being provided to the web server 108.

[0057] At step 412, the pre-OS program 106 provides the translated contents to the web server 108.

[0058] At step 414, the web server 108 provides the translated contents to the web browser of the remote device 102. In an example, the web server 108

may provide the translated contents to the web browser using a web protocol, such as HTTP. In an example, representation of the contents of the pre-OS program screen on the web browser of the remote device 102 may be different from representation of the contents of the pre-OS program screen on the computing system 100.

[0059] At step 416, an input, generated at the remote device 102, is captured by the embedded code. The input is further translated into a web-based language. The input may correspond to a user event at the remote device 102. For instance, when the user clicks on the button element provided on the translated contents of the pre-OS program screen using a mouse, the input is captured. The user may click on the button element to change or update the settings of the pre-OS program 106.

[0060] At step 418, the remote device 102 provides the input to the web server 108. In an example, the remote device 102 may provide the input to the web server using a web protocol, such as HTTP.

[0061] At step 420, the web server 108 provides the input to the pre-OS program 106 for translation. In an example, the web server 108 provides the input to the pre-OS program 106 for translation into a computer processable command. The computer processable command may be for example, a GUI message.

[0062] At step 422, the pre-OS program 106 sends an instruction to the content translator 210 to translate the input into the computer processable command.

[0063] At step 424, the content translator 210 translates the input into the computer processable command. In an example, the computer processable command corresponds to receipt of the input from one of a mouse of the remote device 102, a keyboard of the remote device 102, and a touchscreen of the remote device 102. The input may be translated into a data structure format, such as a JavaScript Object Notation (JSON) format or an XML format.

[0064] At step 426, the content translator 210 provides the computer processable command to the pre-OS program 106. On receiving the computer processable command, the pre-OS program 106 executes the computer

processable command and thus changes or modifications made by the user through the contents of the pre-program screen are reflected on the pre-OS program 106 running on the computing system 100.

[0065] FIGS. 5 and 6 illustrate flowcharts of methods 500 and 600, respectively, for providing contents of a pre-OS program running on a computing system to a remote device, according to an example of the present subject matter. The order in which the methods are described is not intended to be construed as a limitation, and any number of the described method blocks may be combined in any order to implement the aforementioned methods, or an alternative method. Furthermore, methods 500 and 600 may be implemented by processing resource or computing device(s) through any suitable hardware, non-transitory machine readable instructions, or combination thereof.

[0066] It may also be understood that methods 500 and 600 may be performed by programmed computing devices, such as the computing system 100 as depicted in FIGS. 1-3. Furthermore, the methods 500 and 600 may be executed based on instructions stored in a non-transitory computer readable medium. The non-transitory computer readable medium may include, for example, digital memories, magnetic storage media, such as one or more magnetic disks and magnetic tapes, hard drives, or optically readable digital data storage media. Although, the methods 500 and 600 are described below with reference to the computing system 100 as described above, other suitable systems for the execution of these methods can also be utilized. Additionally, implementation of these methods is not limited to such examples.

[0067] With reference to the method 500 as depicted in FIG. 5, at block 502, the method 500 includes receiving, at a computing system, a request from a remote device to access contents of a pre-OS screen of a pre-OS program on a web browser of the remote device, where the pre-OS program is executable prior to loading an OS. The pre-OS program may be a Basic Input/Output System (BIOS) program. Further, examples of the pre-OS screen include BIOS setup screen, BIOS Power-On Self-Test (POST) screen, text mode screen, and Graphical User Interface (GUI) application screen.

[0068] In an example, the request to access the contents of the pre-OS

screen may be received from a user of the remote device, for example, a system administrator. The contents of the pre-OS screen include one of graphical contents, text contents, and a combination of the graphical contents and the text contents. In an example, the web server 108 of the computing system 100 may receive the request from the remote device 102 to access the contents of the pre-OS screen of the pre-OS program 106 running on the computing system 100.

[0069] At block 504, the contents of the pre-OS screen are translated into a web-based language. The web-based language may be one of a HyperText Markup Language (HTML) language, a JavaScript (JS) language, an EXtensible Markup Language (XML) language, and an Asynchronous JavaScript and XML (AJAX) language. In an example, the content translator 210 may translate the contents of the pre-OS screen of the pre-OS program 106 into the web-based language.

[0070] At block 506, the translated contents are provided to the web browser of the remote device using a web protocol. The web protocol may be a Hypertext Transfer Protocol (HTTP). Therefore, the contents of the pre-OS screen of the pre-OS program running on the computing system are directed to the web browser of the remote device. In an example implementation, the web server 108 provides the translated contents to the web browser of the remote device 102 using the web protocol.

[0071] At block 508, an input is received from the web browser of the remote device. The input may correspond to a user event at the remote device. Once the translated contents of the pre-OS screen are provided to the web browser, the user of the remote device may interact with the pre-OS screen using input devices of the remote device, such as a mouse, a keyboard, a touchscreen, and a touchpad. In an example, when the user clicks on the mouse, or types on the keyboard, or touches a screen of the remote device, the user event occurs. In an example, the web server 108 may receive the input from the web browser of the remote device 102.

[0072] At block 510, an action is performed at the computing system based on the input. In an example, the action may include translation of the input into a

computer processable command and execution of the computer processable format to update contents of the pre-OS screen of the pre-OS program. In an example, the computer processable command corresponds to receipt of the input from one of the mouse of the remote device, the keyboard of the remote device, and the touchscreen of the remote device.

[0073] With reference to method 600 as depicted in FIG. 6, at block 602, the method 600 includes receiving, at a computing system, a request from a remote device to access contents of a pre-OS screen of a pre-OS program on a web browser of the remote device. The pre-OS program is executable prior to loading an OS. The pre-OS program may be a BIOS program, a UEFI program and other programs, such as a network card Option Read Only Memory (OPROM) executed during a pre-OS environment. Further, examples of the pre-OS screen include a BIOS setup screen, a BIOS POST screen, a text mode screen, and a GUI application screen. In an example, the request to access the contents of the pre-OS screen may be received from a user of the remote device, for example, a system administrator. Further, the contents of the pre-OS screen may include one of graphical contents, text contents, and a combination of the graphical contents and the text contents.

[0074] In an example implementation, the web server 108 of the computing system 100 may receive the request from the remote device 102 to access the contents of the pre-OS screen of the pre-OS program 106 running on the computing system 100. In case the web server 108 is executable by the pre-OS program 106 and is inactive, an error message, such as “HTTP 503 Service Unavailable” or “Network error” may be displayed on the web browser of the remote device 102.

[0075] In case the web server 108 is executable by the controller 302, such as a BMC, no error message, such as “Network error” or “HTTP 503 Service Unavailable” is displayed on the remote device 102 unless the controller 302 is not ready, as the web server 108 is always active inside the controller 302. However, if the pre-OS program 106 is inactive, for example, when the computing system 100 is powered OFF or when a connection between the computing system 100 and the remote device 102 is not established, a default

web page showing a message "Waiting for the system to be powered ON" may be displayed on the web browser of the remote device 102.

[0076] At block 604, the contents of the pre-OS screen are translated into a web-based language. The web-based language may be one of a HTML language, a JS language, an XML language, and an AJAX language. In an example, the translated contents include web elements in the web-based language format. For instance, the translated contents may include HTML elements, such as image elements, button elements, static text elements, checkbox elements, and text-input elements. In an example, the content translator 210 may translate the contents of the pre-OS screen of the pre-OS program 106 into the web-based language. In an example, the content translator 210 may also embed a code in the translated content for receiving a user input. The code may capture an activity performed on the web browser of the remote device 102, such as an input provided by a user, and may translate the activity into a web-based language. In an example, the translated contents, i.e., the web-based contents may correspond to a web page or a web application.

[0077] At block 606, the translated contents are provided to the web browser on the remote device using a web protocol. The web protocol may be HTTP. Therefore, the contents of the pre-OS screen of the pre-OS program running on the computing system are directed to the web browser of the remote device. In an example implementation, the web server 108 provides the translated contents to the web browser of the remote device 102 using the web protocol.

[0078] At block 608, an input is received from the remote device, where the input corresponds to a user event at the remote device. Once the contents of the pre-OS screen are provided to the remote device, the user of the remote device may interact with the pre-OS program screen using input devices of the remote device, such as a mouse and a keyboard. In an example, when the user clicks on the mouse or types on the keyboard, the user event is generated. The user event is further translated into a web-based language. In an example, the web server 108 may receive the input from the remote device 102.

[0079] At block 610, the input is provided to the pre-OS program for

translation into a computer processable command and execution of the computer processable command to update contents of the pre-OS screen of the pre-OS program. The computer processable command corresponds to receipt of the input from one of the mouse of the remote device, the keyboard of the remote device, and a touchscreen of the remote device. In an example, the pre-OS screen of the pre-OS program is updated when changes or modifications made by the user of the remote device to the contents of the pre-OS program screen are reflected on the pre-OS program running on the computing system. In an example, the web server 108 may provide the input to the pre-OS program 106 for translation into the computer processable command related to the contents of the pre-OS screen of the pre-OS program 106 and execution of the computer processable command.

[0080] At block 612, the updated contents of the pre-OS screen are translated and the translated updated contents are provided to the web browser of the remote device using the web protocol. In an example, on updating the contents of the pre-OS screen, the updated contents of the pre-OS screen are translated into the web-based language.

[0081] FIG. 7 is a block diagram of a network environment 700 implementing a non-transitory computer-readable medium, for providing contents of the pre-OS program 106 running on a computing system to a remote device, according to an example of the present subject matter. The network environment 700 may comprise at least a portion of a public networking environment or a private networking environment, or a combination thereof. In one implementation, the network environment 700 includes a processing resource 702 communicatively coupled to a non-transitory computer readable medium 704, hereinafter referred to as computer readable medium 704, through a communication link 706. In an example, the processing resource 702 can be a computing device, such as the computing system 100.

[0082] The computer readable medium 704 can be, for example, an internal memory device of the computing device or an external memory device. In one implementation, the communication link 706 may be a direct communication link, such as any memory read/write interface. In another implementation, the

communication link 706 may be an indirect communication link, such as a network interface. In such a case, the processing resource 702 can access the computer readable medium 704 through a network 708. The network 708 may be a single network or a combination of multiple networks and may use a variety of different communication protocols.

[0083] The processing resource 702 and the computer readable medium 704 may also be coupled to data sources 710 through the communication link 706, and/or to communication devices 712 over the network 708. The coupling with the data sources 710 enables in receiving the requested data in an offline environment, and the coupling with the communication devices 712 enables in receiving the requested data in an online environment.

[0084] In one implementation, the computer readable medium 704 includes a set of computer readable instructions, implementing the pre-OS program 106 and the web server 108. The set of computer readable instructions, referred to as instructions hereinafter, can be accessed by the processing resource 702 through the communication link 706 and subsequently executed to perform acts for providing the contents of the screen of the pre-OS program 106 to the remote device 102. For discussion purposes, the execution of the instructions by the processing resource 702 has been described with reference to various components introduced earlier with reference to description of FIGS. 1, 2, and 3.

[0085] On execution by the processing resource 702, the web server 108 may receive a request from the remote device 102 to access contents of a screen of the pre-OS program 106 on a web browser of the remote device 102. The pre-OS program 106 is executable prior to loading of OS. The pre-OS program 106 may be a BIOS program. Further, the contents of the screen of the pre-OS program 106 may include one of graphical contents, text contents, and a combination of the graphical contents and the text contents.

[0086] Thereafter, the web server 108 may initiate translation of the contents of the screen of the pre-OS program 106 into a web-based language. In an example, the web server 108 may initiate translation of the contents of the screen by sending a request to the pre-OS program 106 for translated contents

of the screen of the pre-OS program 106. The pre-OS program 106 may then provide the contents of the screen to the content translator 210 for translation. The content translator 210 may translate the contents into a web-based language. In an example, the web-based language may be one of a HTML language, a JS language, an XML language, and an AJAX language. The translated contents may also include an embedded code to capture a user input. For example, the code may capture an activity performed on the web browser of the remote device 102, such as an input provided by a user, and may translate the activity into a web-based language.

[0087] Further, the content translator 210 may provide the translated contents to the pre-OS program 106 for provision to the web server 108. Subsequently, the web server 108 provides the translated contents to the web browser of the remote device 102 using a web protocol, such as a Hypertext Transfer Protocol (HTTP). Once the translated contents are provided to the web browser on the remote device 102, the web server 108 may receive the user input from the web browser of the remote device 102. The user input may correspond to a user event at the web browser. Further, the web server 108 may provide user input to the pre-OS program 106 for translation into a computer processable command related to the contents of the screen of the pre-OS program 106 and execution of the computer processable command.

[0088] Although implementations of pre-OS content transmission have been described in language specific to structural features and/or methods, it is to be understood that the present subject matter is not necessarily limited to the specific features or methods described. Rather, the specific features and methods are disclosed and explained in the context of a few implementations for pre-OS content transmission.

We claim:

1. A computing system comprising:

a processor;

a pre-Operating System (pre-OS) program executable by the processor, wherein the pre-OS program is executable prior to loading an OS; and

a web server executable by the processor to:

receive a request from a remote device to access contents of a screen of the pre-OS program;

send a request to the pre-OS program for translated contents, wherein the translated contents are contents of the screen of the pre-OS program that are translated into a web-based language; and

transmit the translated contents to the remote device using a web protocol.

2. The computing system as claimed in claim 1, wherein the web server is executable by the pre-OS program and an error message is displayed on the remote device when a connection between the computing system and the remote device is not established.

3. The computing system as claimed in claim 1, wherein the web server is executable by a controller of the computing system and a default web page is displayed on the remote device when a connection between the computing system and the remote device is not established.

4. The computing system as claimed in claim 1, wherein the pre-OS program is a Basic Input/Output System (BIOS) program.

5. The computing system as claimed in claim 1, wherein the computing system further comprises a content translator capable of communicating with the pre-OS program, and wherein the content translator is to:

translate the contents of the screen of the pre-OS program into the web-based language; and

provide the translated contents to the pre-OS program for provision to the web server.

6. The computing system as claimed in claim 1, wherein the web server is further to:

receive an input from the remote device, wherein the input corresponds to a user event at the remote device; and

provide the input to the pre-OS program for translation into a computer processable command related to the contents of the screen of the pre-OS program.

7. The computing system as claimed in claim 6, wherein the computer processable command corresponds to receipt of an input from one of a mouse of the remote device, a keyboard of the remote device, a touchpad of the remote device, and a touchscreen of the remote device.

8. A method comprising:

receiving, at a computing system, a request from a remote device to access contents of a pre-Operating System (pre-OS) screen of a pre-OS program on a web browser of the remote device, wherein the pre-OS program is executable prior to loading an OS;

translating the contents of the pre-OS screen into a web-based language;

providing the translated contents to the web browser of the remote device using a web protocol;

receiving an input from the web browser of the remote device; and

performing an action at the computing system based on the input.

9. The method as claimed in claim 8, wherein the action includes translation of the input into a computer processable command and execution of the computer processable command to update contents of the pre-OS screen of the pre-OS program.

10. The method as claimed in claim 9, wherein, on updating the contents of the pre-OS screen, the method further comprises:

translating the updated contents of the pre-OS screen into the web-based language; and

providing the translated updated contents to the web browser of the remote device using the web protocol.

11. The method as claimed in claim 8, wherein the contents of the pre-OS screen include one of graphical contents, text contents, and a combination of the graphical contents and the text contents.

12. The method as claimed in claim 8, wherein the web-based language is one of a HyperText Markup Language (HTML) language, a JavaScript (JS) language, an EXtensible Markup Language (XML) language, and an Asynchronous JavaScript and XML (AJAX) language.

13. A non-transitory computer-readable medium encoded with instructions executable by a processing resource to:

receive a request from a remote device to access contents of a screen of a pre-Operating System (pre-OS) program on a web browser of the remote device, wherein the pre-OS program is executable prior to loading an OS;

translate the contents of the screen of the pre-OS program into a web-based language, wherein the translated contents include an embedded code to capture a user input; and

provide the translated contents to the web browser of the remote device using a web protocol.

14. The non-transitory computer-readable medium as claimed in claim 13, wherein the web protocol is a Hypertext Transfer Protocol (HTTP).

15. The non-transitory computer-readable medium as claimed in claim 13, wherein the instructions are further executable to:

receive the user input from the web browser of the remote device, wherein the user input corresponds to a user event at the web browser; and

provide the user input to the pre-OS program for translation into a computer processable command related to the contents of the screen of the pre-OS program.

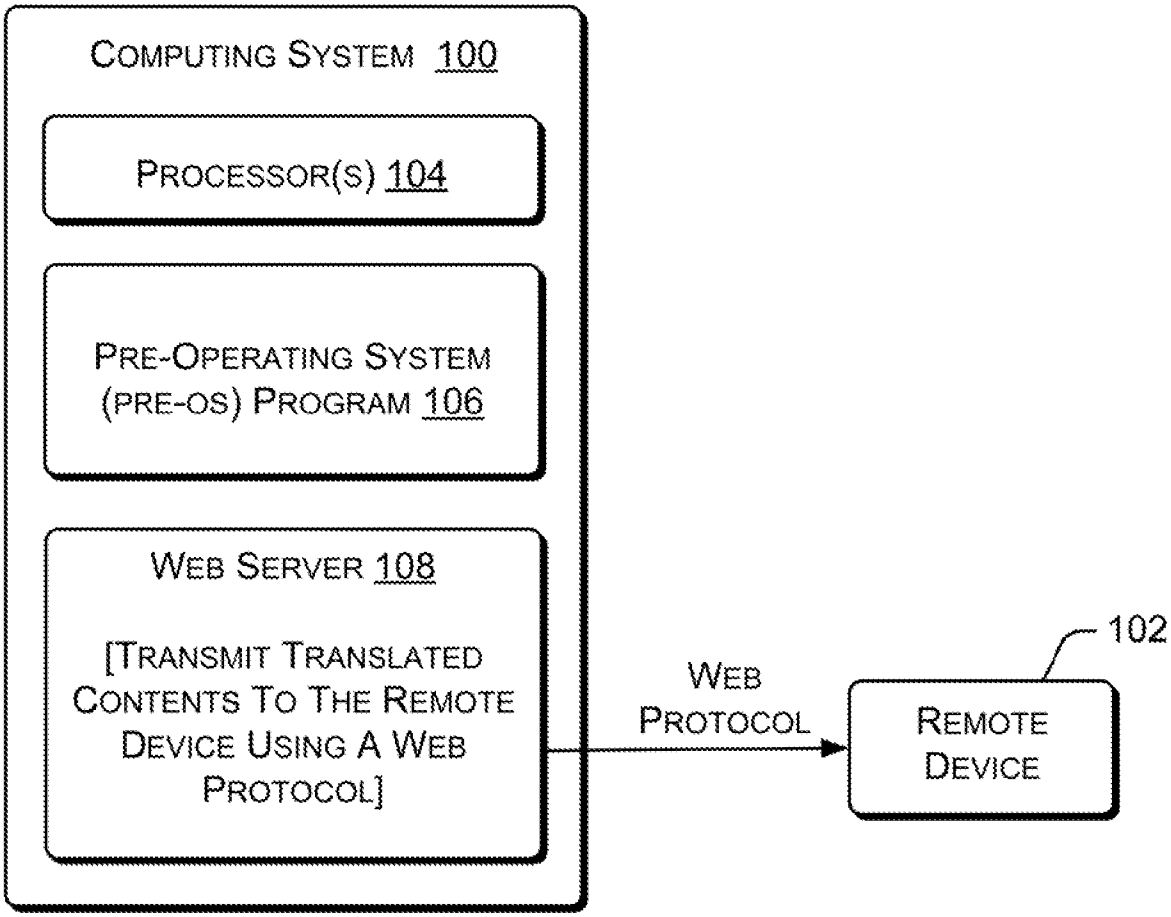
COMPUTING SYSTEM 100
PROCESSOR(S) 104
PRE-OPERATING SYSTEM (PRE-OS) PROGRAM 106
WEB SERVER 108
[TRANSMIT TRANSLATED CONTENTS TO THE REMOTE DEVICE USING A WEB PROTOCOL]
WEB PROTOCOL
REMOTE DEVICE
102

FIG. 1

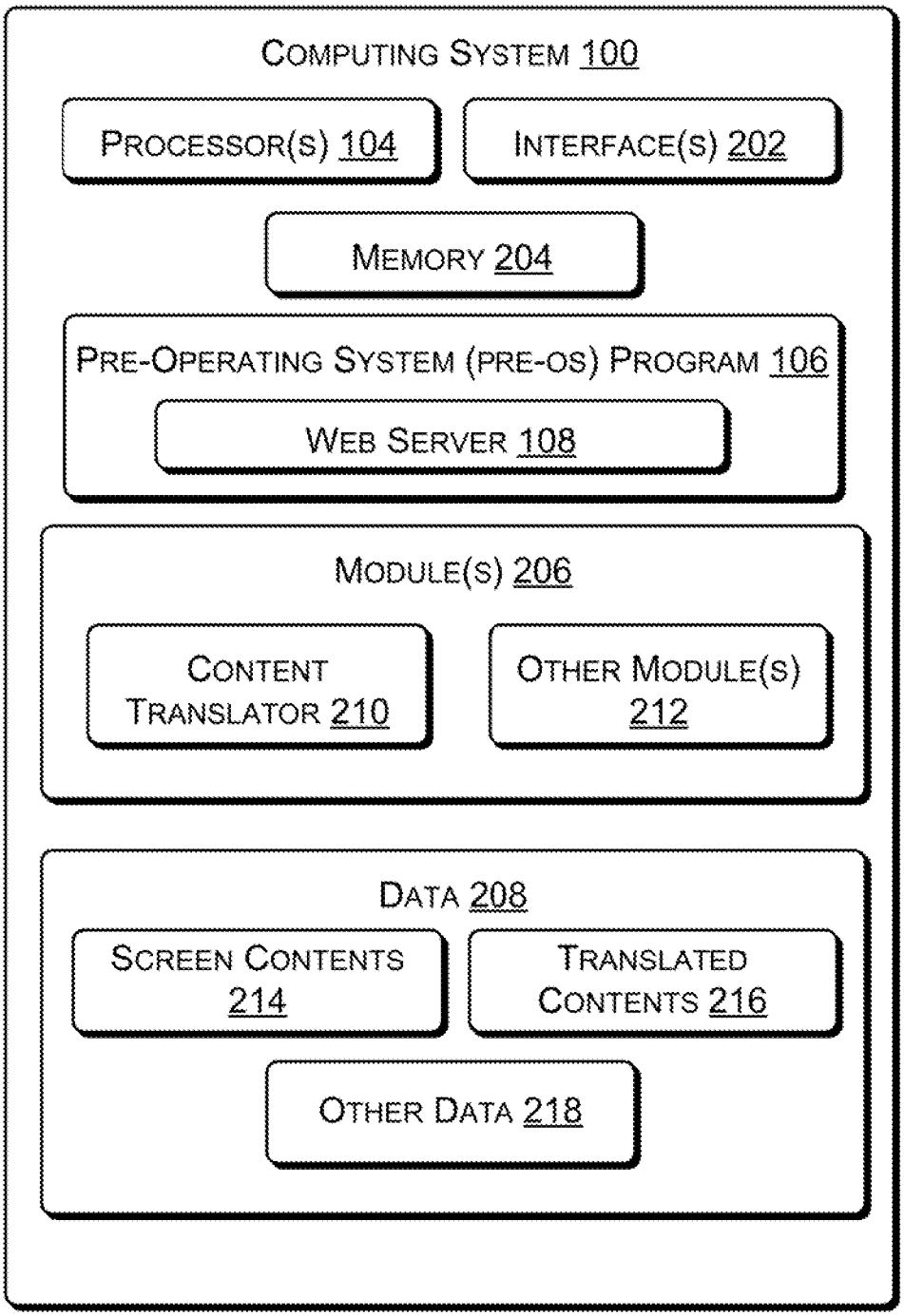
COMPUTING SYSTEM 100
PROCESSOR(S) 104
INTERFACE(S) 202
MEMORY 204
PRE-OPERATING SYSTEM (PRE-OS) PROGRAM 106
WEB SERVER 108
MODULE(S) 206
CONTENT TRANSLATOR 210
OTHER MODULE(S) 212
DATA 208
SCREEN CONTENTS 214
TRANSLATED CONTENTS 216
OTHER DATA 218

FIG. 2

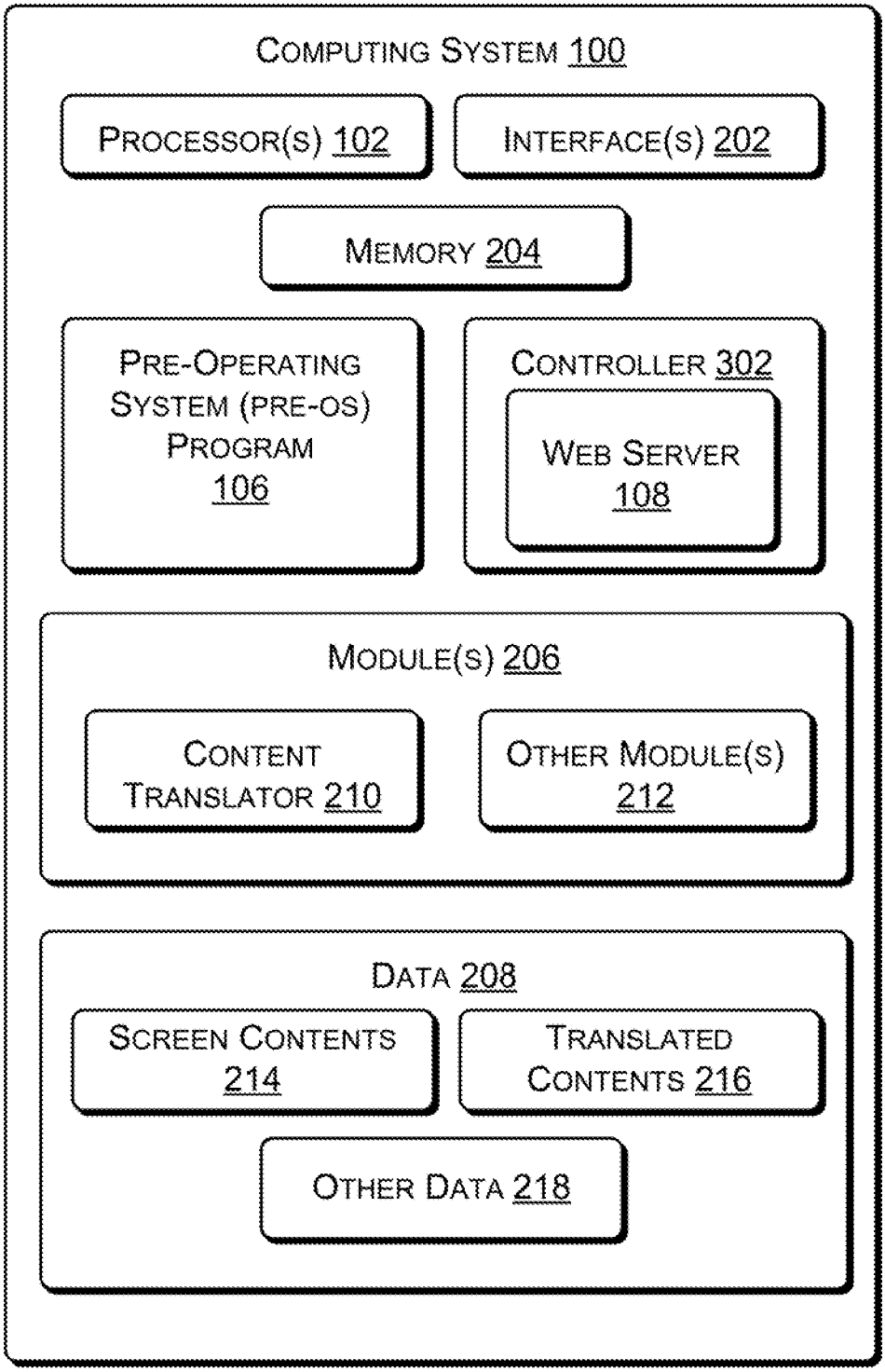
COMPUTING SYSTEM 100
PROCESSOR(S) 102
INTERFACE(S) 202
MEMORY 204
PRE-OPERATING SYSTEM (PRE-OS) PROGRAM 106
CONTROLLER 302
WEB SERVER 108
MODULE(S) 206
CONTENT TRANSLATOR 210
OTHER MODULE(S) 212
DATA 208
SCREEN CONTENTS 214
TRANSLATED CONTENTS 216
OTHER DATA 218

FIG. 3

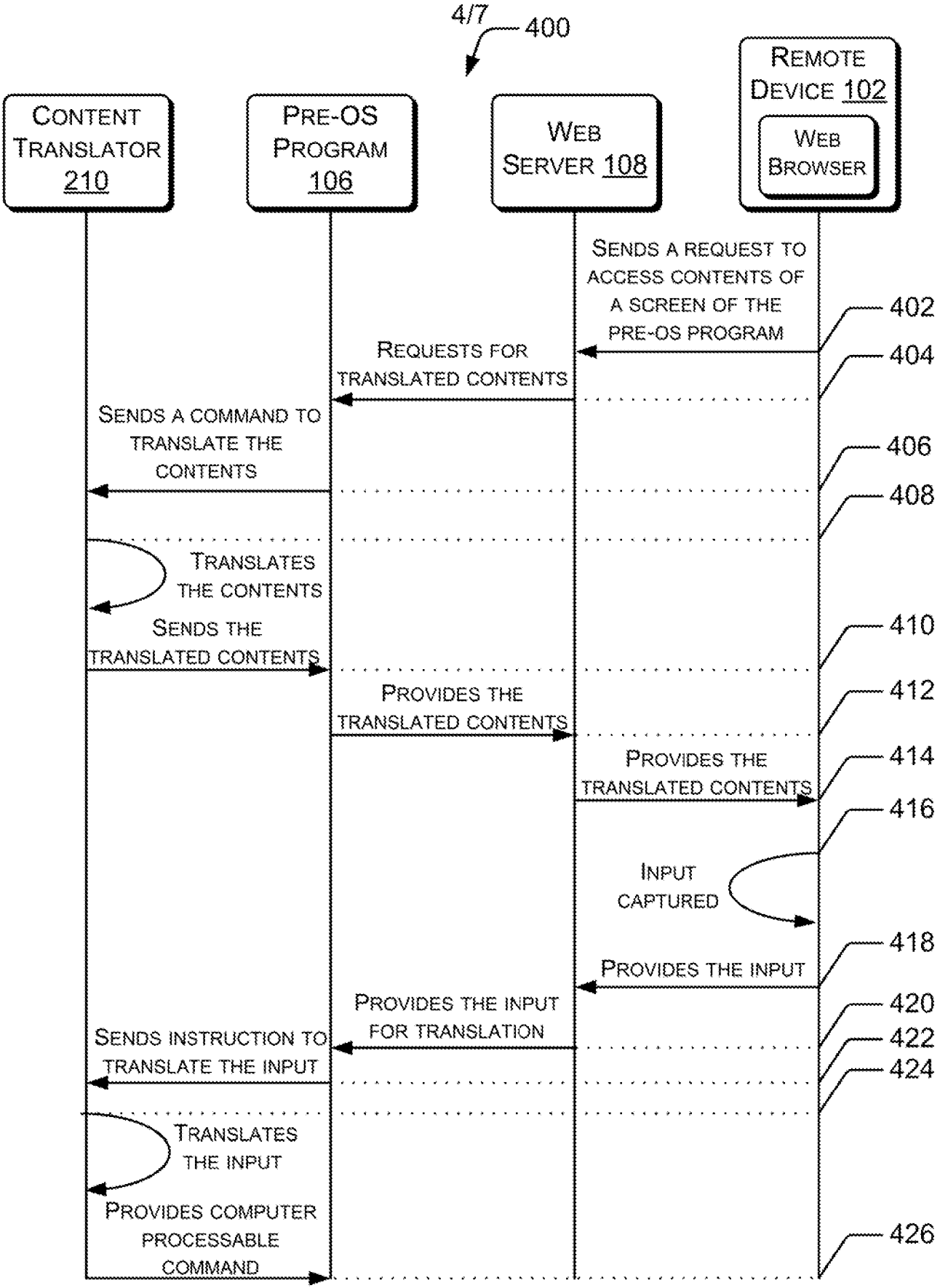
4/7
400
CONTENT TRANSLATOR 210
PRE-OS PROGRAM 106
WEB SERVER 108
REMOTE DEVICE 102
WEB BROWSER
SENDS A REQUEST TO ACCESS CONTENTS OF A SCREEN OF THE PRE-OS PROGRAM
402
REQUESTS FOR TRANSLATED CONTENTS
404
SENDS A COMMAND TO TRANSLATE THE CONTENTS
406
408
TRANSLATES THE CONTENTS
SENDS THE TRANSLATED CONTENTS
410
PROVIDES THE TRANSLATED CONTENTS
412
PROVIDES THE TRANSLATED CONTENTS
414
416
INPUT CAPTURED
418
PROVIDES THE INPUT
PROVIDES THE INPUT FOR TRANSLATION
420
SENDS INSTRUCTION TO TRANSLATE THE INPUT
422
424
TRANSLATES THE INPUT
PROVIDES COMPUTER PROCESSABLE COMMAND
426

FIG. 4

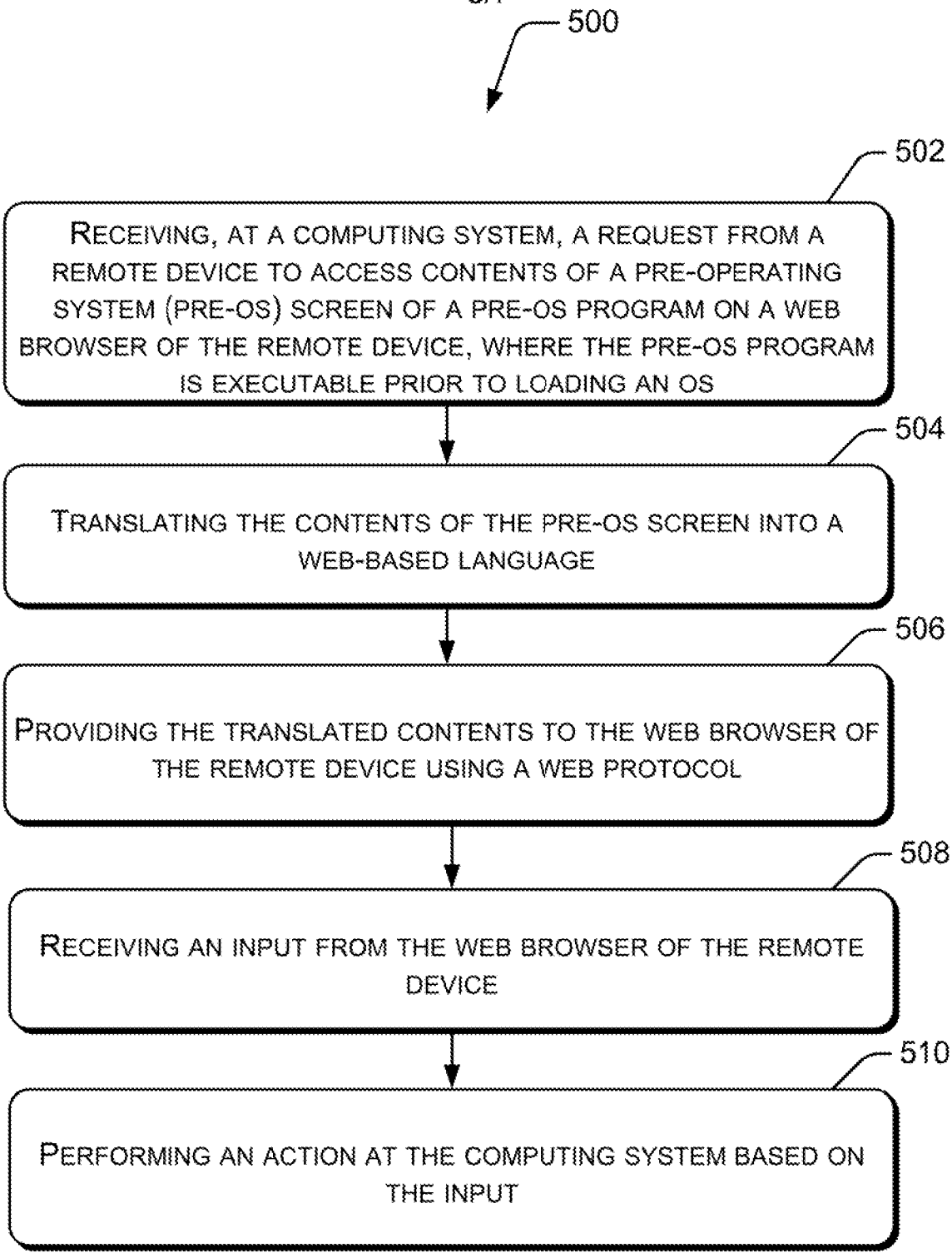
500
502
RECEIVING, AT A COMPUTING SYSTEM, A REQUEST FROM A REMOTE DEVICE TO ACCESS CONTENTS OF A PRE-OPERATING SYSTEM (PRE-OS) SCREEN OF A PRE-OS PROGRAM ON A WEB BROWSER OF THE REMOTE DEVICE, WHERE THE PRE-OS PROGRAM IS EXECUTABLE PRIOR TO LOADING AN OS
504
TRANSLATING THE CONTENTS OF THE PRE-OS SCREEN INTO A WEB-BASED LANGUAGE
506
PROVIDING THE TRANSLATED CONTENTS TO THE WEB BROWSER OF THE REMOTE DEVICE USING A WEB PROTOCOL
508
RECEIVING AN INPUT FROM THE WEB BROWSER OF THE REMOTE DEVICE
510
PERFORMING AN ACTION AT THE COMPUTING SYSTEM BASED ON THE INPUT

FIG. 5

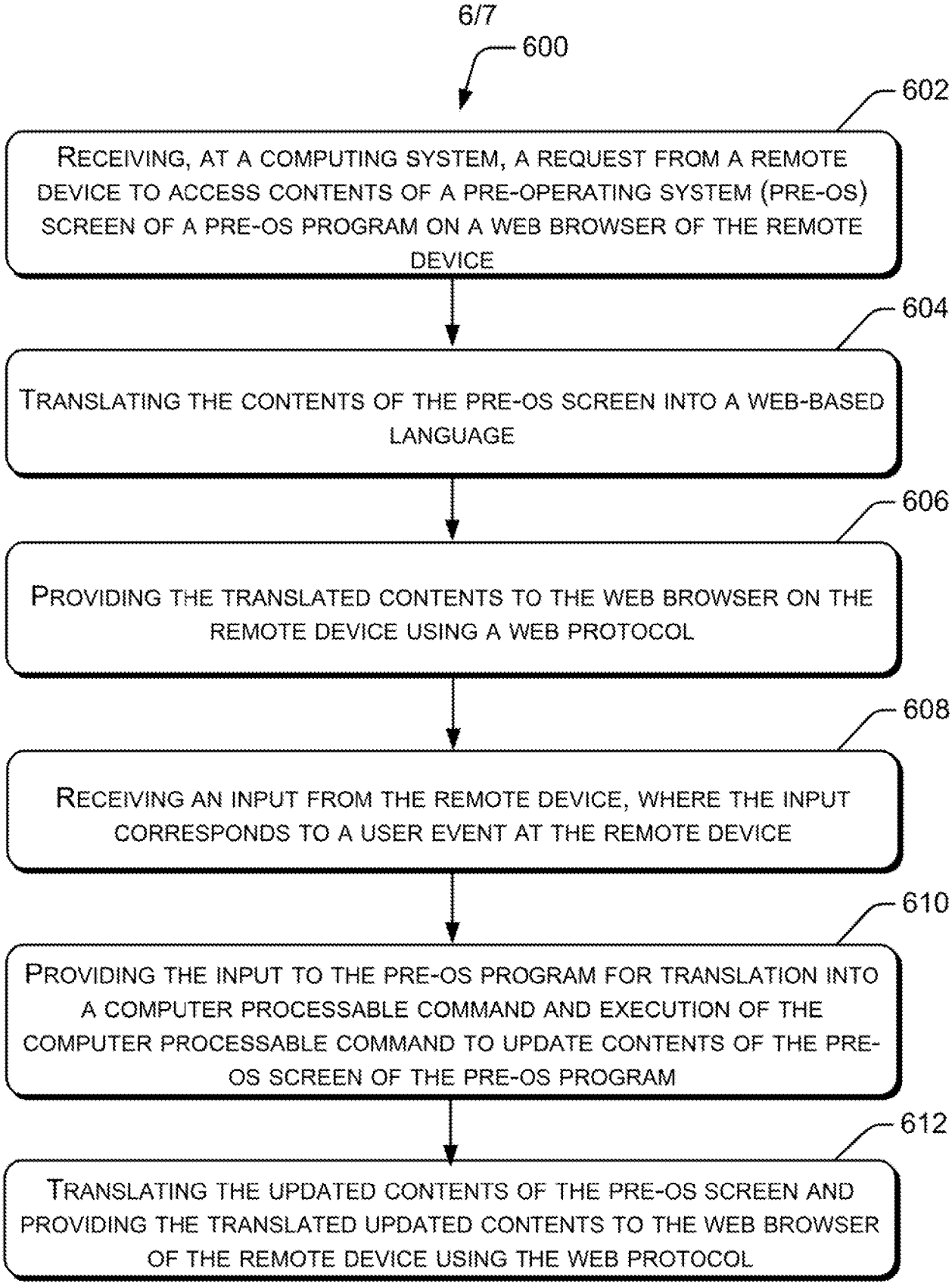
600
602
RECEIVING, AT A COMPUTING SYSTEM, A REQUEST FROM A REMOTE DEVICE TO ACCESS CONTENTS OF A PRE-OPERATING SYSTEM (PRE-OS) SCREEN OF A PRE-OS PROGRAM ON A WEB BROWSER OF THE REMOTE DEVICE
604
TRANSLATING THE CONTENTS OF THE PRE-OS SCREEN INTO A WEB-BASED LANGUAGE
606
PROVIDING THE TRANSLATED CONTENTS TO THE WEB BROWSER ON THE REMOTE DEVICE USING A WEB PROTOCOL
608
RECEIVING AN INPUT FROM THE REMOTE DEVICE, WHERE THE INPUT CORRESPONDS TO A USER EVENT AT THE REMOTE DEVICE
610
PROVIDING THE INPUT TO THE PRE-OS PROGRAM FOR TRANSLATION INTO A COMPUTER PROCESSABLE COMMAND AND EXECUTION OF THE COMPUTER PROCESSABLE COMMAND TO UPDATE CONTENTS OF THE PRE-OS SCREEN OF THE PRE-OS PROGRAM
612
TRANSLATING THE UPDATED CONTENTS OF THE PRE-OS SCREEN AND PROVIDING THE TRANSLATED UPDATED CONTENTS TO THE WEB BROWSER OF THE REMOTE DEVICE USING THE WEB PROTOCOL

FIG. 6

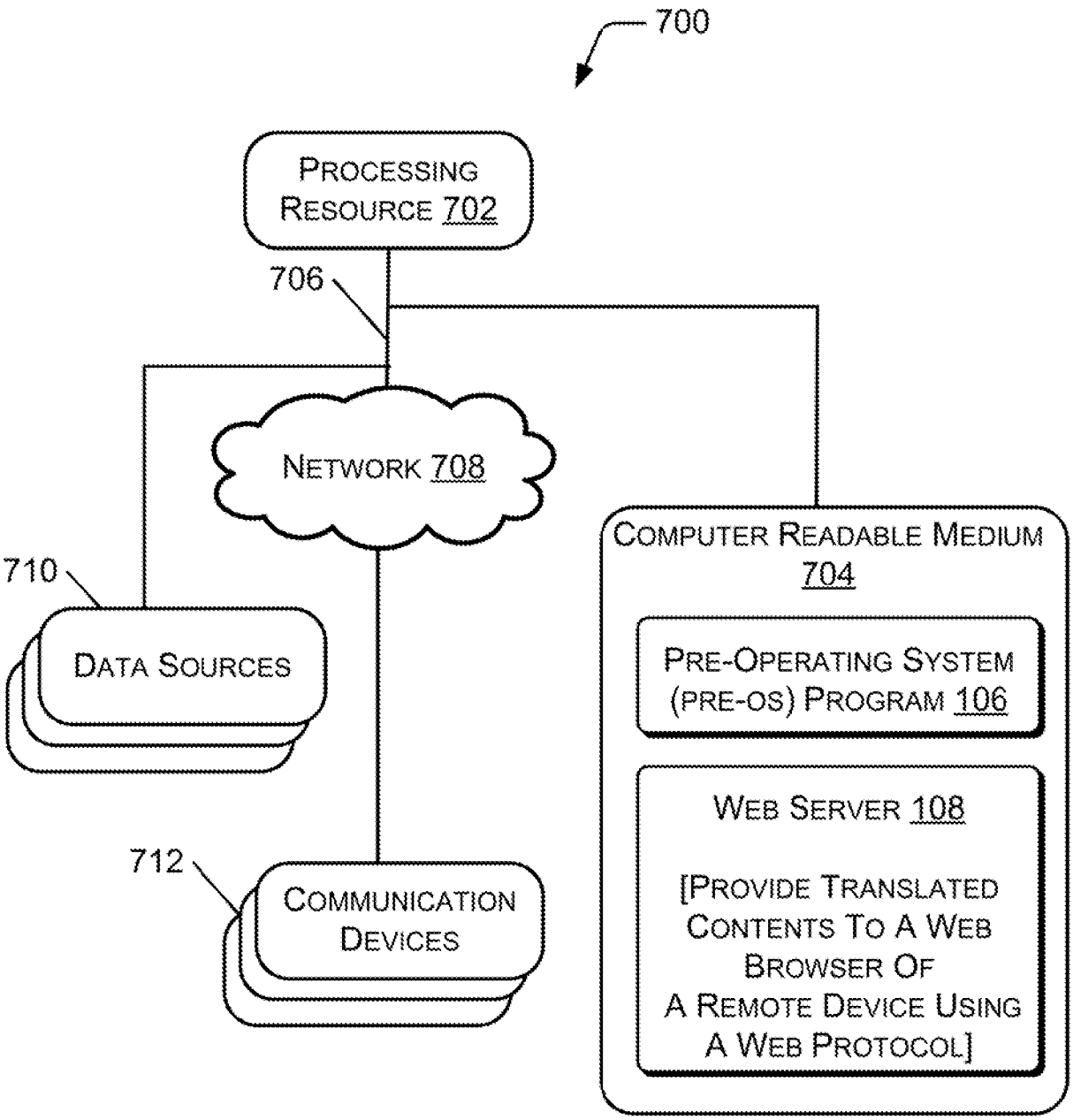
700
PROCESSING RESOURCE 702
706
NETWORK 708
710
DATA SOURCES
712
COMMUNICATION DEVICES
COMPUTER READABLE MEDIUM 704
PRE-OPERATING SYSTEM (PRE-OS) PROGRAM 106
WEB SERVER 108
[PROVIDE TRANSLATED CONTENTS TO A WEB BROWSER OF A REMOTE DEVICE USING A WEB PROTOCOL]

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/029533

A. CLASSIFICATION OF SUBJECT MATTER

G06F 15/16(2006.01)i, G06F 9/22(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 15/16; G06F 13/40; G06F 17/20; G06F 9/44; G06F 9/22; G06F 9/24; G06F 17/28; G06F 15/177

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: pre-OS, boot, CMOS, access screen, translate

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2010-0057440 A1 (RANDALL S. SPRINGFIELD et al.) 04 March 2010 See figures 4-5; paragraphs [0032][0034][0035].	1-15
A	US 2014-0278347 A1 (GOOGLE INC.) 18 September 2014 See abstract; figure 4; paragraph [0004] and claim 1.	1-15
A	US 2014-0380034 A1 (INTEL CORPORATION) 25 December 2014 See abstract; figure 4 and paragraphs [0040][0041].	1-15
A	US 2006-0070053 A1 (GREGORY ANDERSEN et al.) 30 March 2006 See abstract and claims 1,2.	1-15
A	WO 2015-060853 A1 (INTEL CORPORATION et al.) 30 April 2015 See abstract.	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
08 January 2016 (08.01.2016)	**11 January 2016 (11.01.2016)**

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
YOON, Hye Sook
Telephone No. +82-42-481-8370

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/029533

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2010-0057440 A1	04/03/2010	US 8340954 B2	25/12/2012
US 2014-0278347 A1	18/09/2014	KR 10-2015-0130476 A	23/11/2015
		US 9195651 B2	24/11/2015
		WO 2014-150407 A1	25/09/2014
US 2014-0380034 A1	25/12/2014	US 2008-133794 A1	05/06/2008
		US 8862785 B2	14/10/2014
US 2006-0070053 A1	30/03/2006	DE 102005040075 A1	06/04/2006
		JP 2006-092544A	06/04/2006
		US 7730472 B2	01/06/2010
WO 2015-060853 A1	30/04/2015	US 2015-0212828 A1	30/07/2015