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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 October 2011 (13.10.2011)

PCT

(10) International Publication Number
WO 2011/124284 A1

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

(21) International Application Number:
PCT/EP2010/070295

(22) International Filing Date:
20 December 2010 (20.12.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/322,325 9 April 2010 (09.04.2010) US

(71) Applicant *(for all designated States except US)*: **TOMTOM INTERNATIONAL B.V.** [NL/NL]; IP Creation, 35 Rembrandtplein, NL-1017 CT Amsterdam (NL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **VAN DER MOLEN, Henk** [NL/NL]; Dertien Loten 85, NL-5223 ZX 's-Hertogenbosch (NL). **VAN HULTEN, Peter** [NL/NL]; Vuurijzer 12, NL-5406 BD Uden (NL). **SLEGERS, Walter** [NL/NL]; Clara Wichmannstraat 148, NL-5612 RR Eindhoven (NL). **BOSCHKER, Breght** [NL/NL]; Rigelstraat 6, NL-Hilversum Hilversum (NL).

(74) Agent: **MOORE, Nicholas**; TomTom International B.V., IP Creation, Oosterdoksstraat 114, NL-1011 DK Amsterdam (NL).

(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, 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, PE, PG, PH, PL, PT, RO, RS, RU, 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, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, 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,

[Continued on next page]

(54) Title: BRIDGE MODULE GENERATOR APPARATUS, BRIDGE MODULE UPDATE SYSTEM, METHOD OF GENERATING A BRIDGE MODULE AND METHOD OF UPDATING A BRIDGE MODULE

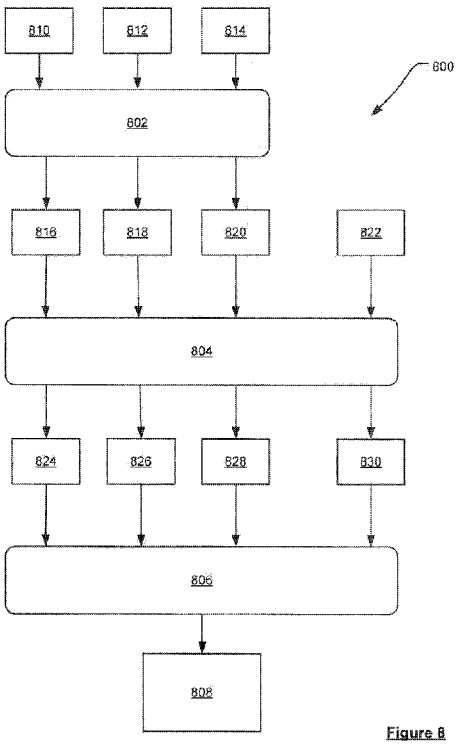

Figure 8

(57) Abstract: A bridge module generator apparatus comprises a processing resource arranged to support, when in use, an operational environment. The operational environment supports a bridge generator (800). The bridge generator (800) is capable of accessing an interface description (810, 812, 814) associated with an application and to use the interface description (810, 812, 814) to generate interface-specific code (816, 818, 820) associated with the application. The bridge generator (800) is arranged to generate a bridge module (808) using the interface-specific code (816, 818, 820) for facilitating communication between a script engine of a HyperText Markup Language engine and an application.

WO 2011/124284 A1

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))*

Published:

— *with international search report (Art. 21(3))*

BRIDGE MODULE GENERATOR APPARATUS, BRIDGE MODULE UPDATE SYSTEM, METHOD OF GENERATING A BRIDGE MODULE AND METHOD OF UPDATING A BRIDGE MODULE

Field of the Invention

The present invention relates to bridge module generator apparatus of the type that, for example, generates a bridge module for a browser engine. The present invention also relates to a bridge module update system of the type that, for example, supports communication between a script engine and an application following change to an interface associated with an application. The present invention further relates to a method of generating a bridge module, the method being of the type that generates the bridge module to support communication between a script engine and an application. The present invention also relates to a method of updating a bridge module, the method being of the type that, for example, supports communication between a script engine and an application following change to an interface associated with an application.

Background to the Invention

Computing devices, for example portable computing devices, such as Portable Navigation Devices (PNDs) that include GPS (Global Positioning System) signal reception and processing functionality are well known and are widely employed as in-car or other vehicle navigation systems.

In general terms, a modern PND comprises a processor, memory and map data stored within said memory. The processor and memory cooperate to provide an execution environment in which a software operating system is typically established, and additionally it is commonplace for one or more additional software programs to be provided to enable the functionality of the PND to be controlled, and to provide various other functions.

Typically, these devices further comprise one or more input interfaces that allow a user to interact with and control the device, and one or more output interfaces by means of which information may be relayed to the user. Illustrative examples of output interfaces include a visual display and a speaker for audible output. Illustrative examples of input interfaces include one or more physical buttons to control on/off operation or other features of the device (which buttons need not necessarily be on the device itself but can be on a steering wheel if the device is built into a vehicle), and a microphone for detecting user speech. In one particular arrangement, the output

interface display may be configured as a touch sensitive display (by means of a touch sensitive overlay or otherwise) additionally to provide an input interface by means of which a user can operate the device by touch.

Devices of this type will also often include one or more physical connector interfaces by means of which power and optionally data signals can be transmitted to and received from the device, and optionally one or more wireless transmitters/receivers to allow communication over cellular telecommunications and other signal and data networks, for example Bluetooth, Wi-Fi, Wi-Max, GSM, UMTS and the like.

PNDs of this type also include a GPS antenna by means of which satellite-broadcast signals, including location data, can be received and subsequently processed to determine a current location of the device.

The PND may also include electronic gyroscopes and accelerometers which produce signals that can be processed to determine the current angular and linear acceleration, and in turn, and in conjunction with location information derived from the GPS signal, velocity and relative displacement of the device and thus the vehicle in which it is mounted. Typically, such features are most commonly provided in in-vehicle navigation systems, but may also be provided in PNDs if it is expedient to do so.

The utility of such PNDs is manifested primarily in their ability to determine a route between a first location (typically a start or current location) and a second location (typically a destination). These locations can be input by a user of the device, by any of a wide variety of different methods, for example by postcode, street name and house number, previously stored “well known” destinations (such as famous locations, municipal locations (such as sports grounds or swimming baths) or other points of interest), and favourite or recently visited destinations.

Typically, the PND is enabled by software for computing a "best" or "optimum" route between the start and destination address locations from the map data. A "best" or "optimum" route is determined on the basis of predetermined criteria and need not necessarily be the fastest or shortest route. The selection of the route along which to guide the driver can be very sophisticated, and the selected route may take into account existing, predicted and dynamically and/or wirelessly received traffic and road information, historical information about road speeds, and the driver's own preferences for the factors determining road choice (for example the driver may specify that the route should not include motorways or toll roads).

The device may continually monitor road and traffic conditions, and offer to or choose to change the route over which the remainder of the journey is to be made due to changed conditions. Real time traffic monitoring systems, based on various

technologies (e.g. mobile phone data exchanges, fixed cameras, GPS fleet tracking), are being used to identify traffic delays and to feed the information into notification systems.

PNDs of this type may typically be mounted on the dashboard or windscreen of a vehicle, but may also be formed as part of an on-board computer of the vehicle radio or indeed as part of the control system of the vehicle itself. The navigation device may also be part of a hand-held system, such as a PDA (Portable Digital Assistant), a media player, a mobile telephone or the like, and in these cases, the normal functionality of the hand-held system is extended by means of the installation of software on the device to perform both route calculation and navigation along a calculated route.

Once a route has been calculated by a PND, the user interacts with the navigation device to select the desired calculated route, optionally from a list of proposed routes. Optionally, the user may intervene in, or guide the route selection process, for example by specifying that certain routes, roads, locations or criteria are to be avoided or are mandatory for a particular journey. The route calculation aspect of the PND forms one primary function, and navigation along such a route is another primary function.

Although the route calculation and navigation functions are fundamental to the overall utility of PNDs, it is possible to use the device purely for information display, or "free-driving", in which only map information relevant to the current device location is displayed, and in which no route has been calculated and no navigation is currently being performed by the device. Such a mode of operation is often applicable when the user already knows the route along which it is desired to travel and does not require navigation assistance.

During navigation along a calculated route, it is usual for such PNDs to provide visual and/or audible instructions to guide the user along a chosen route to the end of that route, i.e. the desired destination. It is also usual for PNDs to display map information on-screen during the navigation, such information regularly being updated on-screen so that the map information displayed is representative of the current location of the device, and thus of the user or user's vehicle if the device is being used for in-vehicle navigation.

An icon displayed on-screen typically denotes the current device location, and is centred with the map information of current and surrounding roads in the vicinity of the current device location and other map features also being displayed. Additionally, navigation information may be displayed, optionally in a status bar above, below or to one side of the displayed map information, examples of navigation information include a distance to the next deviation from the current road required to be taken by the user, the nature of that deviation possibly being represented by a further icon suggestive of the

particular type of deviation, for example a left or right turn. The navigation function also determines the content, duration and timing of audible instructions by means of which the user can be guided along the route. As can be appreciated, a simple instruction such as "turn left in 100 m" requires significant processing and analysis. As previously mentioned, user interaction with the device may be by a touch screen, or additionally or alternately by steering column mounted remote control, by voice activation or by any other suitable method.

Devices of the type described above, for example the GO 940 LIVE model manufactured and supplied by TomTom International B.V., provide a reliable means for enabling users to navigate from one position to another. Such devices are of great utility when the user is not familiar with the route to the destination to which they are navigating.

It is clear from the above-described operation that the navigation apparatus comprises a user interface. In this regard, it is known that web browsers capable of running JavaScript code are powerful tools for creating complex and hence feature-rich user interfaces. One known technique to support interaction between the web browser and the application employs so-called browser "plug-ins" to translate between script code used and the application.

However, support for communications between the user interface and existing applications running in different processes or on different processors or devices is usually complex. Consequently, adaptation of interfaces to changes made to the application is not straightforward. In this respect, changes or extensions to the application, the script and/or the plug-in require the application, the script and/or the plug-in to be rewritten, depending upon the change made. This activity is time consuming and also susceptible to human error.

As an alternative to the use of plug-ins, it is known to use a web server disposed between a web browser and the existing application. In this respect, script to be executed by the web browser comprises so-called "XMLHttpRequests", which are received by the web server and acted upon, the web server communicating with the existing application using, for example, existing Remote Procedure Calls (RPCs). This technique is employed when writing Asynchronous JavaScript and XML (AJAX) applications. However, in the event that the interface of the application is changed, the web server has to be manually reconfigured and the script has to be manually updated. This technique is therefore technically cumbersome. Also, the interface has a performance overhead associated therewith as a result of the need for the web server as an intermediary.

Summary of the Invention

According to a first aspect of the present invention, there is provided a bridge module generator apparatus comprising: a processing resource arranged to support, when in use, an operational environment, the operational environment supporting a bridge generator; wherein the bridge generator is capable of accessing an interface description associated with an application and to use the interface description to generate interface-specific code associated with the application; and the bridge generator is arranged to generate a bridge module using the interface-specific code for facilitating communication between a script engine of a browser engine and an application.

The bridge module may be a plug-in. The script engine may be a JavaScript engine.

The bridge generator may comprise a first compiler; the first compiler may be arranged to generate the interface-specific code.

The apparatus may further comprise: framework code for the bridge module; and a second compiler and linker arranged to compile the interface-specific code and the framework code and generate the bridge module.

The interface description may be expressed in an interface description language.

According to a second aspect of the present invention, there is provided a plug-in generator for a browser engine comprising the bridge module generator apparatus as set forth above in relation to the first aspect of the invention.

According to a third aspect of the present invention, there is provided a method of generating a bridge module for supporting communication between a script engine of a browser engine and an application, the method comprising: a bridge generator accessing an interface description associated with the application and using the interface description to generate interface-specific code associated with the application; and the bridge generator generating a bridge module using the interface-specific code for supporting communication between the script engine of the browser engine and the application.

According to a fourth aspect of the present invention, there is provided a bridge module update system comprising: a processing resource arranged to support, when in use, an operational environment, the operational environment supporting a browser engine and a bridge module therefor, the browser engine comprising a script engine for executing a script and the script engine being arranged to communicate with an application; wherein the bridge module is arranged to cooperate, when in use, with the

script engine in order to translate a communication between the script engine and the application; and the browser engine is arranged to receive an updated bridge module in response to an interface reconfiguration of the application, the updated bridge module being installable in relation to the browser engine in place of the bridge module.

The bridge module may be a plug-in. The script engine may be a JavaScript engine and/or the script is a JavaScript.

The application may be arranged to support a service accessible by the browser engine.

The service may be a location-based service.

The bridge module may be arranged to communicate with the application using a predefined interface, the predefined interface being an Intra-Process Communication, an Inter-Process Communication (IPC), and/or an Inter-Processor Communication. The Inter-Process Communication may support by a Remote Procedure Call (RPC).

The bridge module may be arranged to translate a script function call and associated argument so to be usable by the Inter-Process Communication.

The bridge module may be arranged to translate an Intra-Process Communication call and associated argument to a script function call and/or an Inter-Process Communication call and associated argument to a script function call and/or an Inter-Processor call and associated argument to a script function call.

According to a fifth aspect of the present invention, there is provided a user interface supported by the bridge module update system as set forth above in relation to the fourth aspect of the invention.

According to a sixth aspect of the present invention, there is provided a navigation apparatus comprising: a bridge module update system as set forth above in relation to the fourth aspect of the invention; and a navigation engine constituting the application, the application being supported by the operational environment; a web browser arranged to support a user interface system; and an update module arranged to receive the bridge module and install the bridge module with respect to the web browser.

According to a seventh aspect of the present invention, there is provided a method of updating a bridge module for a browser engine having a script engine arranged to communicate with an application, the method comprising: providing a bridge module cooperating with the browser engine in order to translate a communication between the script engine and the application; and receiving an updated bridge module in response to an interface reconfiguration of the application; and installing the updated bridge module in relation to the browser engine in place of the bridge module.

According to a eighth aspect of the present invention, there is provided a

computer program element comprising computer program code means to make a computer execute the method as set forth above in relation to the third aspect and/or the seventh aspect of the invention.

The computer program element may be embodied on a computer readable medium.

It is thus possible to provide a method and apparatus that accommodates changes or extensions to existing applications without a need to update manually the bridge module used by a script engine of a web browser to communicate with the existing application. Consequently, well-known Application Programming Interfaces (APIs) of existing applications can be used by a script engine in order to build feature-rich applications. Source code for a bridge module, for example a plug-in, is generated automatically using an interface description in relation to the application. Furthermore, the apparatus and method support, if desired, legacy interfaces using the bridge module. The method and apparatus also obviates the need for the provision of a web server or other process between the existing application and the script engine of, for example, a web browser. In the context of a navigation apparatus, when a new interface implementation is added to a navigation engine application, an interface description is created and added to a plug-in project description stored in a project file, thereby simply necessitating a new build of the plug-in in order to generate the necessary code to support a new interface with the navigation engine in response to changes to the navigation engine.

Other advantages of these embodiments are set out hereafter, and further details and features of each of these embodiments are defined in the accompanying dependent claims and elsewhere in the following detailed description.

Brief Description of the Drawings

At least one embodiment of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a schematic illustration of an exemplary part of a Global Positioning System (GPS) usable by a navigation apparatus;

Figure 2 is a schematic diagram of a communications system for communication between a navigation apparatus and a server apparatus;

Figure 3 is a schematic illustration of components of the navigation apparatus of Figure 2 or any other suitable navigation apparatus;

Figure 4 is a schematic diagram of an arrangement of mounting and/or docking a

navigation apparatus;

Figure 5 is a schematic representation of an architectural stack employed by the navigation apparatus of Figure 3;

Figure 6 is a schematic diagram of an HTML engine capable of communicating with an application;

Figure 7 is a schematic diagram of a computing arrangement;

Figure 8 is a schematic diagram of a bridge processing application supported by the computing arrangement of Figure 7;

Figure 9 is flow diagram of a method of generating a bridge module constituting another embodiment of the invention;

Figure 10 is a schematic diagram of another bridge processing application constituting a further embodiment of the invention; and

Figure 11 is a schematic diagram of a web browser employing a bridge module generated by the bridge processing application of Figure 10.

Detailed Description of Preferred Embodiments

Throughout the following description identical reference numerals will be used to identify like parts.

Embodiments of the present invention will now be described with particular reference to a PND. It should be remembered, however, that the teachings herein are not limited to PNDs but are instead universally applicable to any type of processing device that is configured to execute software, for example navigation software in a portable and/or mobile manner so as to provide route planning and navigation functionality. It follows therefore that in the context of the embodiments set forth herein, a navigation apparatus is intended to include (without limitation) any type of route planning and navigation apparatus, irrespective of whether that device is embodied as a PND, a vehicle such as an automobile, or indeed a portable computing resource, for example a portable personal computer (PC), a mobile telephone or a Personal Digital Assistant (PDA) executing route planning and navigation software. However, the skilled person should appreciate that the embodiments described herein are not limited to use in connection with software to support navigation-related functionality. Indeed, the embodiments can be employed in relation to an in-vehicle arrangement, whereby a web browser or HTML engine application is employed and communicates with an existing application. The existing application can be supported by the same processor or processing resource in the vehicle and the browser, but can be supported elsewhere in

the vehicle, communication between the web browser and the existing application being, for example, via an in-vehicle communications network, for example a Flexray™ or CAN network.

With the above provisos in mind and in the context of navigation, the Global Positioning System (GPS) of Figure 1 and the like are used for a variety of purposes. In general, the GPS is a satellite-radio based navigation system capable of determining continuous position, velocity, time, and in some instances direction information for an unlimited number of users. Formerly known as NAVSTAR, the GPS incorporates a plurality of satellites which orbit the earth in extremely precise orbits. Based on these precise orbits, GPS satellites can relay their location to any number of receiving units.

The GPS system is implemented when a device, specially equipped to receive GPS data, begins scanning radio frequencies for GPS satellite signals. Upon receiving a radio signal from a GPS satellite, the device determines the precise location of that satellite via one of a plurality of different conventional methods. The device will continue scanning, in most instances, for signals until it has acquired at least three different satellite signals (noting that position is not normally, but can be determined, with only two signals using other triangulation techniques). Implementing geometric triangulation, the receiver uses the three known positions to determine its own two-dimensional position relative to the satellites. This can be done in a known manner. Additionally, acquiring a fourth satellite signal allows the receiving device to calculate its three dimensional position by the same geometrical calculation in a known manner. The position and velocity data can be updated in real time on a continuous basis by an unlimited number of users.

As shown in Figure 1, the GPS system 100 comprises a plurality of satellites 102 orbiting about the earth 104. A GPS receiver 106 receives spread spectrum GPS satellite data signals 108 from a number of the plurality of satellites 102. The spread spectrum data signals 108 are continuously transmitted from each satellite 102, the spread spectrum data signals 108 transmitted each comprise a data stream including information identifying a particular satellite 102 from which the data stream originates. The GPS receiver 106 generally requires spread spectrum data signals 108 from at least three satellites 102 in order to be able to calculate a two-dimensional position. Receipt of a fourth spread spectrum data signal enables the GPS receiver 106 to calculate, using a known technique, a three-dimensional position.

In Figure 2, a navigation system comprises a navigation apparatus 200 capable of communicating, if desired in an embodiment, with a server 150 via a communications channel 152 supported by a communications network, for example a Wide Area Network

(WAN), such as the Internet, that can be implemented by any of a number of different arrangements. The communication channel 152 generically represents the propagating medium or path that connects the navigation apparatus 200 and the server 150. The server 150 and the navigation apparatus 200 can communicate when a connection via the communications channel 152 is established between the server 150 and the navigation apparatus 200 (noting that such a connection can be a data connection via mobile device, a direct connection via personal computer via the Internet, etc.).

The communication channel 152 is not limited to a particular communication technology. Additionally, the communication channel 152 is not limited to a single communication technology; that is, the channel 152 may include several communication links that use a variety of technologies. For example, the communication channel 152 can be adapted to provide a path for electrical, optical, and/or electromagnetic communications, etc. As such, the communication channel 152 includes, but is not limited to, one or a combination of the following: electric circuits, electrical conductors such as wires and coaxial cables, fibre optic cables, converters, radio-frequency (RF) waves, the atmosphere, free space, etc. Furthermore, the communication channel 152 can include intermediate devices, for example: routers, repeaters, buffers, transmitters, and receivers.

In one illustrative arrangement, the communication channel 152 is supported by telephone and computer networks. Furthermore, the communication channel 152 may be capable of accommodating wireless communication, for example, infrared communications, radio frequency communications, such as microwave frequency communications, etc. Additionally, the communication channel 152 can accommodate satellite communication.

Both digital and analogue signals can be transmitted through the communication channel 152. These signals can be modulated, encrypted and/or compressed signals as can be desirable for the communication technology.

In this example, the navigation apparatus 200 comprises mobile telephone technology, more details of which will be described later herein. As such, the mobile telephone technology within the navigation apparatus 200 can establish a network connection between the navigation apparatus 200 and the server 150, via the Internet for example, in a manner similar to that of any wireless communications-enabled terminal. However, the navigation apparatus 200 comprising or coupled to the GPS receiver device 106, is capable of establishing a data session, if required, with the network hardware of the communications network via a wireless communications terminal (not shown), such as a mobile telephone, PDA, and/or any device equipped

with mobile telephone technology, in order to establish a digital connection, for example a digital connection via known Bluetooth technology. Thereafter, through its network service provider, the mobile terminal can establish the network connection (through the Internet for example) with the server 150.

As such, the "mobile" network connection can be established between the navigation apparatus 200 (which can be, and often times is, mobile as it travels alone and/or in a vehicle) and the server 150 and constitutes the communication channel 152.

Further, establishing the network connection between the mobile device (via a service provider) and another device such as the server 150, using the Internet for example, can be achieved using any number of appropriate data communications protocols, for example the TCP/IP layered protocol. Furthermore, the mobile device can use any number of communication standards such as CDMA2000, GSM, IEEE 802.11 a/b/c/g/n, etc.

Hence, it can be seen that the Internet connection may be used, which can be achieved via a data connection using the mobile telephone or mobile telephone technology.

The server 150 includes, in addition to other components which may not be illustrated, a processor 154 operatively connected to a memory 156 and further operatively connected, via a wired or wireless connection 158, to a mass data storage device 160. The mass storage device 160 contains, for example, a store of navigation data and map information, and can again be a separate device from the server 150 or can be incorporated into the server 150. The processor 154 is further operatively connected to transmitter 162 and receiver 164, to transmit and receive information to and from navigation apparatus 200 via the communications channel 152. The signals sent and received may include data, communication, and/or other propagated signals. The transmitter 162 and receiver 164 may be selected or designed according to the communications requirement and communication technology used in the communication design for the navigation system 200. Further, it should be noted that the functions of transmitter 162 and receiver 164 may be combined into a single transceiver.

As mentioned above, the navigation apparatus 200 can be arranged to communicate with the server 150 through communications channel 152, using transmitter 166 and receiver 168 to send and receive signals and/or data through the communications channel 152, noting that these devices can further be used to communicate with devices other than the server 150, for example other servers (not shown). Further, the transmitter 166 and receiver 168 constitute a communications unit and are selected or designed according to communication requirements and

communication technology used in the communication design for the navigation apparatus 200 and the functions of the transmitter 166 and receiver 168 may be combined into a single transceiver. Of course, the navigation apparatus 200 comprises other hardware and/or functional parts, which will be described later herein in further detail.

Software stored in server memory 156 provides instructions for the processor 154 and allows the server 150 to provide services to the navigation apparatus 200.

Referring to Figure 3, it should be noted that the block diagram of the navigation apparatus 200 is not inclusive of all components of the navigation apparatus, but is only representative of many example components. The navigation apparatus 200 is located within a housing (not shown). The navigation apparatus 200 includes a processing resource comprising, for example, the processor 202, the processor 202 being coupled to an input device 204 and a display device, for example a display screen 206. Although reference is made here to the input device 204 in the singular, the skilled person should appreciate that the input device 204 represents any number of input devices, including a keyboard device, voice input device, touch panel and/or any other known input device utilised to input information. Likewise, the display screen 206 can include any type of display screen such as a Liquid Crystal Display (LCD), for example.

In one arrangement, one aspect of the input device 204, the touch panel, and the display screen 206 are integrated so as to provide an integrated input and display device, including a touchpad or touchscreen input 250 (Figure 4) to enable both input of information (via direct input, menu selection, etc.) and display of information through the touch panel screen so that a user need only touch a portion of the display screen 206 to select one of a plurality of display choices or to activate one of a plurality of virtual or “soft” buttons. In this respect, the processor 202 supports a Graphical User Interface (GUI) that operates in conjunction with the touchscreen, further details of which will be described later herein.

In the navigation apparatus 200, the processor 202 is operatively connected to and capable of receiving input information from the input device 204 via a connection 210. The navigation apparatus 200 can include an output device 208, for example an audible output device (e.g. a loudspeaker). As the output device 208 can produce audible information for a user of the navigation apparatus 200, it should equally be understood that input device 204 can include a microphone and software for receiving input voice commands as well. The processor 202 is operatively connected to at least one of the display screen 206 and the output device 208, via respective output connections 212, to output information thereto. Further, the navigation apparatus 200

can also include any additional input device and/or any additional output device, such as audio input/output devices for example. The processor 202 is operably coupled to a memory resource 214 via connection 216 and is further arranged to receive/send information from/to input/output (I/O) ports 218 via connection 220, wherein the I/O port 218 is connectible to an I/O device 222 external to the navigation apparatus 200. The memory resource 214 comprises, for example, a volatile memory, such as a Random Access Memory (RAM) and a non-volatile memory, for example a digital memory, such as a flash memory. The external I/O device 222 can include, but is not limited to an external listening device, such as an earpiece for example. The connection to I/O device 222 can further be a wired or wireless connection to any other external device such as a car stereo unit for hands-free operation and/or for voice activated operation for example, for connection to an earpiece or headphones, and/or for connection to a mobile telephone for example, wherein the mobile telephone connection can be used to establish, for example, the data connection between the navigation apparatus 200 and the server 150 via the Internet or any other network for example. However, in this example internal mobile telephony technology is employed to support data communication.

Figure 3 further illustrates an operative connection between the processor 202 and an antenna/receiver 224 via connection 226, wherein the antenna/receiver 224 can be a GPS antenna/receiver for example. It should be understood that the antenna and receiver designated by reference numeral 224 are combined schematically for illustration, but that the antenna and receiver may be separately located components, and that the antenna may be a GPS patch antenna or helical antenna for example.

As mentioned above, in order to support communications in a Universal Mobile Telecommunications System (UMTS), the processor 202 is also coupled to a cellular communications module 228 constituting the mobile telephone technology. The cellular communications module 228 supports a communications interface 229 for transmitting and receiving data wirelessly. The cellular communications module 228 comprises a Subscriber Identity Module (SIM) 230 coupled thereto having a data subscription associated therewith. The subscription is, in this example, for a limited data usage over a pre-determined period of time, for example a calendar month. For example, the data limit may be 1GB per month. In other embodiments, the subscription need not have a data usage limit. The cellular communications module 228 supports a bidirectional data communications service, for example a packet switched data service, such as a General Packet Radio Service (GPRS) supported by the GSM communications network and/or a High Speed Downlink Packet Access (HSDPA) service supported by the UMTS network.

The communications interface 229 is therefore compatible with the bidirectional data communications service. The bidirectional data communications service supports an Internet Protocol (IP) for data communications although use of other protocols is contemplated.

It will, of course, be understood by one of ordinary skill in the art that the electronic components shown in Figure 3 are powered by one or more power sources (not shown) in a conventional manner. As will be understood by one of ordinary skill in the art, different configurations of the components shown in Figure 3 are contemplated. For example, the components shown in Figure 3 may be in communication with one another via wired and/or wireless connections and the like. Thus, the navigation apparatus 200 described herein can be a portable or handheld navigation device 200.

In addition, the portable or handheld navigation device 200 of Figure 3 can be connected or "docked" in a known manner to a vehicle such as a bicycle, a motorbike, a car or a boat for example. Such a navigation device 200 is then removable from the docked location for portable or handheld navigation use.

Referring to Figure 4, the navigation apparatus 200 can be a unit that includes the integrated input and display device 206 and the other components of Figure 2 (including, but not limited to, the internal GPS receiver 224, the microprocessor 202, a power supply (not shown), memory systems 214, etc.).

The navigation apparatus 200 can sit on an arm 252, which itself may be secured to a vehicle dashboard/window/etc. using a suction cup 254. This arm 252 is one example of a docking station to which the navigation apparatus 200 can be docked. The navigation apparatus 200 can be docked or otherwise connected to the arm 252 of the docking station by snap connecting the navigation apparatus 200 to the arm 252 for example. The navigation apparatus 200 may then be rotatable on the arm 252. To release the connection between the navigation apparatus 200 and the docking station, a button (not shown) on the navigation apparatus 200 may be pressed, for example. Other equally suitable arrangements for coupling and decoupling the navigation apparatus 200 to a docking station are well known to persons of ordinary skill in the art.

Turning to Figure 5, the memory resource 214 stores a boot loader program (not shown) that is executed by the processor 202 in order to load an operating system 262 from the memory resource 214 for execution by functional hardware components 260, which provides an environment in which application software 264 can run. The application software 264 provides an operational environment including the GUI 266 that supports core functions of the navigation apparatus 200, for example map viewing, route planning, navigation functions and any other functions associated therewith. In this

example, part of the application software 264 also comprises a navigation engine application 268 that supports navigation-related functionality, for example identification of a current location, route planning, route following, and any other suitable functionality useful for navigation.

Referring to Figure 6, in a generic example, an adaptable managed interface update system comprises a web browser engine 600, which is an example of a HyperText Mark-up Language (HTML) engine, is supported by an operational environment, which in turn is supported by a processing resource, for example one or more processors. The HTML engine supports a script engine 602, the script engine being used to execute script code received by the web browser engine, for example JavaScript code. The script is typically received by the browser engine 600. Of course, it should be appreciated that the embodiments herein can be applied in relation to other scripting languages, for example Ecmascript.

An application or application process 604 with which the web browser 600 needs to communicate is also, in this example, supported by the processing resource. However, the skilled person should appreciate that the application or application process 604 need not be supported by the same processor or even processing resource. In this respect, the application or application process can be executing on a different processor to the processor supporting the web browser 600. In this respect, the processor or processing resource supporting the application need not be co-located with the processor or processing resource supporting the web browser 600 and the application can be executing remotely from the web browser 600, for example on the server 150 via the communication channel 152. In this example, communication with the application 604 is achieved using Inter-Process Communications (IPCs). In this example, the IPC is a standardised IPC. The IPC supports a Remote Procedure Call (RPC). The application 604 is arranged to support a service accessible by the browser engine 600, for example a navigation-related service, such as a location-based service or any other type of navigation-related functionality described above.

The browser engine 600 supports the use of a bridge module 606 and, in this respect, the bridge module 606 cooperates with the browser engine 600 so as to support an interface between the script engine 602 and the application 604. The bridge module 606 serves to translate a communication associated with script code executed by the script engine 602 into IPC calls and arguments used to communicate with the application 600. In this respect, script code is executed and function calls (and any arguments) that result from execution of the script code are intercepted by the bridge module, changed to IPC calls and forwarded to the application. The arguments, which are stored as an

object in the memory space (not shown) of the web browser are used by the bridge module and the processed arguments are stored in the memory space (also not shown) of the bridge module 606 as an object for subsequent communication to the application. Of course, the converse occurs in respect of communications in the opposite direction. However, in the case of the presently described direction of communication, the bridge module 606 therefore translates, in this example, a script function call and any associated arguments so as to be usable in an IPC. Similarly, the bridge module 606, in a reverse sense, translates an IPC call and any associated arguments to a script function call. In this example, the bridge module 606 provides bidirectional translation. The bridge module 606 therefore interfaces 608 with the script engine 602 and interfaces with the application 604.

Referring to Figure 7, an overview is given of a computing arrangement 700 comprising a processing resource 702, for example a processor, such as a microprocessor.

The processor 702 is coupled to a plurality of storage components, including a hard disk drive 704, a Read Only Memory (ROM) 706, a digital memory, for example a flash memory 708, and a Random Access Memory (RAM) 710. Not all of the memory types described above need necessarily be provided. Moreover, these storage components need not be located physically close to the processor 702, and can be located remotely from the processor 702.

The processor 702 is also coupled to one or more input devices for inputting instructions and data by a user, for example a keyboard 712 and a mouse 714. Other input devices, for example a touch screen input unit, a trackball and/or a voice recognition unit, or any other input device, known to persons skilled in the art, can also be provided.

A removable media unit 716 coupled to the processor 702 is provided. The removable media unit 716 is arranged to read data from and possibly write data to a removable data carrier or removable storage medium, for example a Compact Disc-ReWritable (CD-RW) disc 718. In other examples, the removable data carriers can be, for example: tapes, DVDs, CD-Rs, DVD-Rs, CD-ROMs, DVD-RAMs, SD Cards or memory sticks as is known to persons skilled in the art.

The processor 702 can be coupled to a printer 720 for printing output data on paper, as well as being coupled to a display 722, for instance, a monitor, such as an LCD (Liquid Crystal Display) monitor, or any other type of display known to persons skilled in the art. The processor 702 can also be coupled to a loudspeaker 724. Furthermore, the processor 702 can be coupled to a communications network 726, such

as a data network, by means of a data communications interface 728. The processor 702 can therefore be arranged to communicate with other communication-enabled equipment through the network 726.

The data carrier 718 can comprise a computer program product in the form of data and/or instructions arranged to provide the processor 702 with the capacity to perform a method as described later herein. However, such a computer program product may, alternatively, be downloaded via the communications network 726.

The processing resource 702 can be implemented as a standalone system, or as a plurality of parallel operating processors each arranged to carry out sub-tasks of a larger computer program, or as one or more main processors with several sub-processors.

Furthermore, parts of the functionality described herein can even be carried out by remote processors communicating with the processor 702 through the communications network 726.

The components contained in the computing arrangement 700 of Figure 7 are those typically found in general purpose computer systems, and are intended to represent a broad category of such computer components that are well known in the art. Thus, the computing arrangement 700 of Figure 7 can be a Personal Computer (PC), workstation, minicomputer, mainframe computer, etc. The computer can also include different bus configurations, networked platforms, multi-processor platforms, etc. Various operating systems can be used including UNIX, Solaris, Linux, Windows, Macintosh OS, or any other suitable operating system.

Turning to Figure 8, the computing arrangement of Figure 7 supports an operational environment that, in turn supports, a bridge processing application 800. The bridge processing application 800 comprises an interface compiler module 802, a bridge module compiler module 804 and a linker module 806. In this example, the bridge module 808 to be generated needs to support three interfaces. The interface compiler module 802 therefore has access to data stored on the hard drive 704, the data being a first interface description file 810, a second interface description file 812 and a third interface description file 814, respectively relating to a first interface of the application 604, a second interface of the application 604 and a third interface of the application 604. The first, second and third interface description files 810, 812, 814 are each written in an interface description language. The interface description language can be any suitable language intended to define an interface. An example of an interface description language is the Android Interface Description Language (AIDL). The AIDL can be used, either modified or unmodified, in order to write the interface description files 810, 812,

814. The Microsoft Interface Description Language is another example of a suitable language.

In this example, the interface compiler module 802 is a custom-written compiler, the bridge module compiler module 804 and the linker module 806 being any suitable compiler and linker applications, for example the GNU's Not Unix (GNU) C++ compiler and linker. In relation to the interface compiler module 802, this is a custom-written compiler module that generates code, for example C++ code, from the interface definition information stored in the interface description files 810, 812, 814. The code generated for a given interface serves to generate scriptable objects for the arguments of a given function and to invoke appropriate script functions for function calls from a native application to a script domain in which script code is executed, and to "marshal" scriptable objects and function calls for function calls from the script domain to the application.

In operation (Figure 9), the interface description files are manually updated (Step 900) in respect of the first, second and third interfaces and then stored on the hard drive 704. In the respect, any suitable application can be used to write and/or edit the interface description files, for example using any suitable text editor application.

Thereafter, the bridge processing module 800 is initiated and the interface compiler module 802 of the bridge processing module 800 accesses the first, second and third interface description files 810, 812, 814 and compiles (Step 902) the first, second and third interface description files 810, 812, 814 to yield first interface-specific code, second interface-specific code and third interface specific code, which are stored as a first source interface file 816, a second source interface file 818, and a third source interface file 820. A framework data file 822, comprising framework data constituting the basis of a standardised interface for the bridge module 808, is also stored on the hard drive 704. The framework data is a manually written data file that defines the core structure and functionality of the bridge module 808, which is to be supplemented by the functionality for the first, second and third interfaces of the application 604. The framework data is the code that is applicable to all interfaces in order to make functions available to script code executing in the script domain. Examples of the core functionality addressed by the framework data includes: setting up a connection to another process, for example including authentication; handing function calls, for example function calls from both sides of a communications link; and dispatching of function calls.

Once the first, second and third source interface files 816, 818, 820 have been generated, the bridge module compiler 804 accesses the first, second and third source files 816, 818, 820 and the framework data file 822 and compiles (Step 904) the data

contained in these files in order to generate a first interface object file 824, a second interface object file 826, a third interface object file 828 and a framework object file 830 respectively associated with the first interface, the second interface, the third interface and the framework data file 822.

Thereafter, the linker module 806 accesses the first interface object file 828, the second interface object file 826, the third interface object file 828 and the framework object file 830 and combines (Step 906) these files into a single executable program, namely the bridge module 808.

Once generated, the bridge module 808 can be installed (Step 908) into the web browser 600. After installation, the bridge module 606 is available to the script engine 602 so that the script engine 602 can access the application 604 via the IPC.

The above steps, from initial compilation (Step 902) to installation (Step 908) can be implemented using the "make" utility supplied with a Unix operating system.

Although the above-described example is in the context of providing support for three interfaces, the skilled person should understand that the above example is equally applicable to providing support for a different number of interfaces, for example two interfaces or four interfaces.

In another embodiment (Figure 10), libraries can be employed in order to enhance the flexibility of the bridge module 808. In this respect, the above-described process of the preceding embodiment is adapted so that the linker module 806, in addition to generating the bridge module 808, generates a first interface library file 832, a second interface library file 834 and a third interface library file 836. The benefit of the use of library files is that when an interface is changed or added, a new interface library has to be generated as opposed to the whole bridge module 808.

Consequently, referring to Figure 11, the bridge module 808, integrated with the browser 600, comprises the first interface library file 832, the second interface library file 834 and the third interface library file 836.

In the event of an interface modification associated with the application, the bridge module 808 simply has to be regenerated after re-writing or amending the interface description 704 in the manner described above in relation to Figure 8. The updated bridge module is then received by the browser engine in response to the reconfiguration of the interface or a part thereof of the application, and the bridge module is replaced by the updated bridge module. Of course, in the example employing replaceable libraries, the library associated with the changed interface simply has to be regenerated.

In one embodiment, the bridge module 808 is a plug-in for the web browser 600.

In this regard, the plug-in comprises a shared library with a standardised interface.

In another embodiment, supported by the processing resource of Figure 2, the user interface 266 comprises a web browser engine of the kind described above. The web browser 600 needs to communicate with the navigation engine 268, which constitutes the application 604. The web browser 600 comprises a JavaScript engine 602 that serves to execute "web pages" comprising JavaScript code, HyperText Mark-up Language (HTML) code, and/or Cascading Style Sheet (CSS) data.

In this respect, a first interface and a second interface exist between the web browser 600 and the navigation engine 268. The first and second interfaces are implemented in the web browser 600 using the JavaScript, HTML code and/or the CSS data. A plug-in 606 for the web browser 600 is generated in the manner already described above.

Consequently, when a new interface implementation is added to the navigation engine 268, the corresponding interface description file simply needs to be updated and the plug-in build, as described above, only needs to be performed to make the new interface available to the web browser 600. The updated plug-ins can then be communicated to the navigation apparatus 200 for installation using any suitable software update facility, for example an Over The Air (OTA) update or using the TomTom HOME application for obtaining and installing the update.

In a further example, the navigation engine 268 supports a location service that is accessible via the user interface 266 using the web browser engine 600. The managed interface, known as iLocation herein, between the web browser 600 and the navigation engine 268 supports procedure calls, for example: GetCurrentLocation and Get LocationByAddress in order to obtain current location data and identify a location in response to provision of address data, respectively. In this respect, these procedure calls are part of a bespoke protocol implemented in respect of the navigation apparatus 200. As the details of this protocol are not core to the embodiments herein, the protocol will not be described in any further detail in order not to distract from the teachings of the embodiments.

The managed interface (iLocation) is described by an interface description as mentioned above. The plug-in 604 can be generated in the manner already described above in relation to previous embodiments. Hence, as mentioned above, when the interface is updated, the script and the plug-in simply needs updating in order to maintain compatibility between the script engine, in this example the JavaScript engine 602, and the navigation engine 604. In this respect, the script and the interface description simply needs to be amended or rewritten and the plug-in rebuilt.

Although some of the above examples have been described in the context of a user interface, the skilled person should appreciate that the above examples are also applicable to situations where a program written in a scripting language needs to communicate with another program using a formally specified interface. For example, in a vehicle tracking implementation, a script can be written to retrieve, periodically, a current location using a specified interface, such as the iLocation interface described above. Of course, examples of services, other than those supported by the iLocation interface, also exist. For example, services to provide: content, such as images, video, temperature data and/or screen layout; vehicle data, such as vehicle speed and/or error messages; changes to settings, such as language settings; and/or authentication can be employed in relation to the embodiments described herein.

It will also be appreciated that whilst various aspects and embodiments of the present invention have heretofore been described, the scope of the present invention is not limited to the particular arrangements set out herein and instead extends to encompass all arrangements, and modifications and alterations thereto, which fall within the scope of the appended claims.

For example, although reference is made herein to Inter-Process Communications, the skilled person should appreciate that embodiments described herein also apply to an Intra-Process Communication and/or an Inter-Processor Communication. By way of another example, although substantial reference has been made herein to user interfaces, the skilled person should appreciate that the embodiments described herein are applicable direct communication between a script engine and another device, albeit possibly via a firewall. For example, such communication could take place in a vehicle using an in-vehicle communications network for example a Flexray™ or CAN network. One example of a communication is a function call that activates or deactivates headlights of a vehicle.

Whilst embodiments described in the foregoing detailed description refer to GPS, it should be noted that the navigation apparatus may utilise any kind of position sensing technology as an alternative to (or indeed in addition to) GPS. For example the navigation apparatus may utilise using other global navigation satellite systems such as the European Galileo system. Equally, it is not limited to satellite based but could readily function using ground based beacons or any other kind of system that enables the device to determine its geographic location.

Alternative embodiments of the invention can be implemented as a computer program product for use with a computer system, the computer program product being, for example, a series of computer instructions stored on a tangible data recording

medium, such as a diskette, CD-ROM, ROM, or fixed disk, or embodied in a computer data signal, the signal being transmitted over a tangible medium or a wireless medium, for example, microwave or infrared. The series of computer instructions can constitute all or part of the functionality described above, and can also be stored in any memory device, volatile or non-volatile, such as semiconductor, magnetic, optical or other memory device.

It will also be well understood by persons of ordinary skill in the art that whilst the preferred embodiment implements certain functionality by means of software, that functionality could equally be implemented solely in hardware (for example by means of one or more ASICs (application specific integrated circuit)) or indeed by a mix of hardware and software. As such, the scope of the present invention should not be interpreted as being limited only to being implemented in software.

Lastly, it should also be noted that whilst the accompanying claims set out particular combinations of features described herein, the scope of the present invention is not limited to the particular combinations hereafter claimed, but instead extends to encompass any combination of features or embodiments herein disclosed irrespective of whether or not that particular combination has been specifically enumerated in the accompanying claims at this time.

CLAIMS

1. A bridge module generator apparatus comprising:

a processing resource arranged to support, when in use, an operational environment, the operational environment supporting a bridge generator; wherein

the bridge generator is capable of accessing an interface description associated with an application and to use the interface description to generate interface-specific code associated with the application; and

the bridge generator is arranged to generate a bridge module using the interface-specific code for facilitating communication between a script engine of a browser engine and an application.

2. An apparatus as claimed in Claim 1, wherein the bridge module is a plug-in.

3. An apparatus as claimed in Claim 1 or Claim 2, wherein the script engine is a JavaScript engine.

4. An apparatus as claimed in Claim 1 or Claim 2 or Claim 3, wherein the bridge generator comprises a first compiler, the first compiler being arranged to generate the interface-specific code.

5. An apparatus as claimed in Claim 4, further comprising:

framework code for the bridge module; and

a second compiler and linker arranged to compile the interface-specific code and the framework code and generate the bridge module.

6. An apparatus as claimed in any one of the preceding claims, wherein the interface description is expressed in an interface description language.

7. A plug-in generator for a browser engine comprising the bridge module generator apparatus as claimed in any one of the preceding claims.

8. A method of generating a bridge module for supporting communication between a script engine of a browser engine and an application, the method comprising:

a bridge generator accessing an interface description associated with the application and using the interface description to generate interface-specific code associated with the application; and

the bridge generator generating a bridge module using the interface-specific code for supporting communication between the script engine of the browser engine and the application.

9. A bridge module update system comprising:

a processing resource arranged to support, when in use, an operational environment, the operational environment supporting a browser engine and a bridge module therefor, the browser engine comprising a script engine for executing a script and the script engine being arranged to communicate with an application; wherein

the bridge module is arranged to cooperate, when in use, with the script engine in order to translate a communication between the script engine and the application; and

the browser engine is arranged to receive an updated bridge module in response to an interface reconfiguration of the application, the updated bridge module being installable in relation to the browser engine in place of the bridge module.

10. A system as claimed in Claim 9, wherein the bridge module is a plug-in.

11. A system as claimed in Claim 9 or Claim 10, wherein the script engine is a JavaScript engine and/or the script is a JavaScript.

12. A system as claimed in any one of Claims 9 to 11, wherein the application is arranged to support a service accessible by the browser engine.

13. A system as claimed in Claim 12, wherein the service is a location-based service.

14. A system as claimed in Claims 9 to 13, wherein the bridge module is arranged to communicate with the application using a predefined interface, the predefined interface being an Intra-Process Communication, an Inter-Process Communication (IPC), and/or an Inter-Processor Communication.

15. A system as claimed in Claim 14, wherein the Inter-Process Communication supports by a Remote Procedure Call (RPC).

16. A system as claimed in Claim 14, wherein the bridge module is arranged to translate a script function call and associated argument so to be usable by the Inter-Process Communication.

17. A system as claimed in Claim 14, wherein the bridge module is arranged to translate an Intra-Process Communication call and associated argument to a script function call and/or an Inter-Process Communication call and associated argument to a script function call and/or an Inter-Processor call and associated argument to a script function call.

18. A user interface supported by the bridge module update system as claimed in any one of Claims 9 to 17.

19. A navigation apparatus comprising:

a bridge module update system as claimed in any one of Claims 9 to 17; and

a navigation engine constituting the application, the application being supported by the operational environment;

a web browser arranged to support a user interface system; and

an update module arranged to receive the bridge module and install the bridge module with respect to the web browser.

20. A method of updating a bridge module for a browser engine having a script engine arranged to communicate with an application, the method comprising:

providing a bridge module cooperating with the browser engine in order to translate a communication between the script engine and the application; and

receiving an updated bridge module in response to an interface reconfiguration of the application; and

installing the updated bridge module in relation to the browser engine in place of the bridge module.

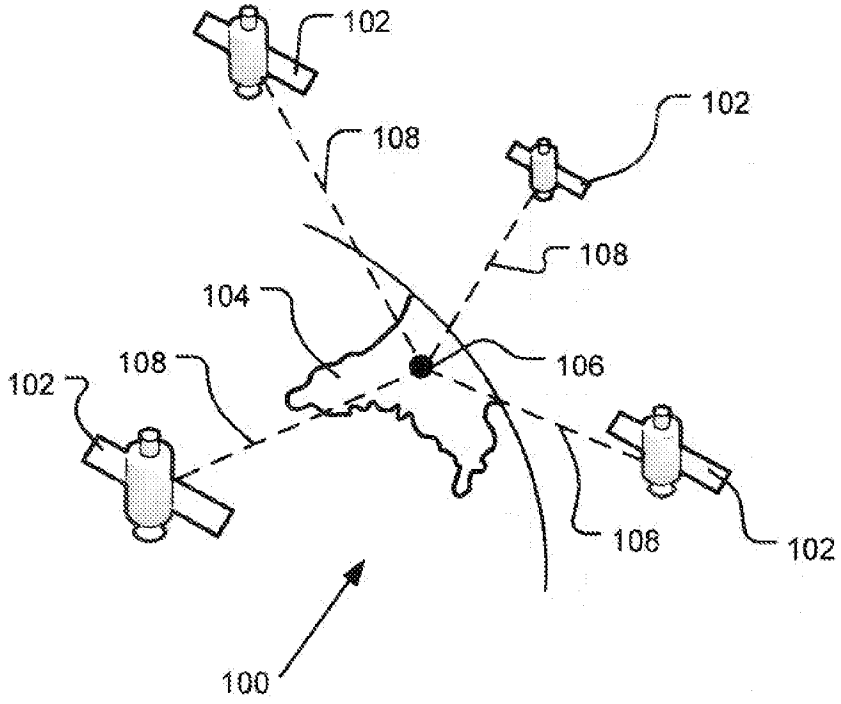

Figure 1

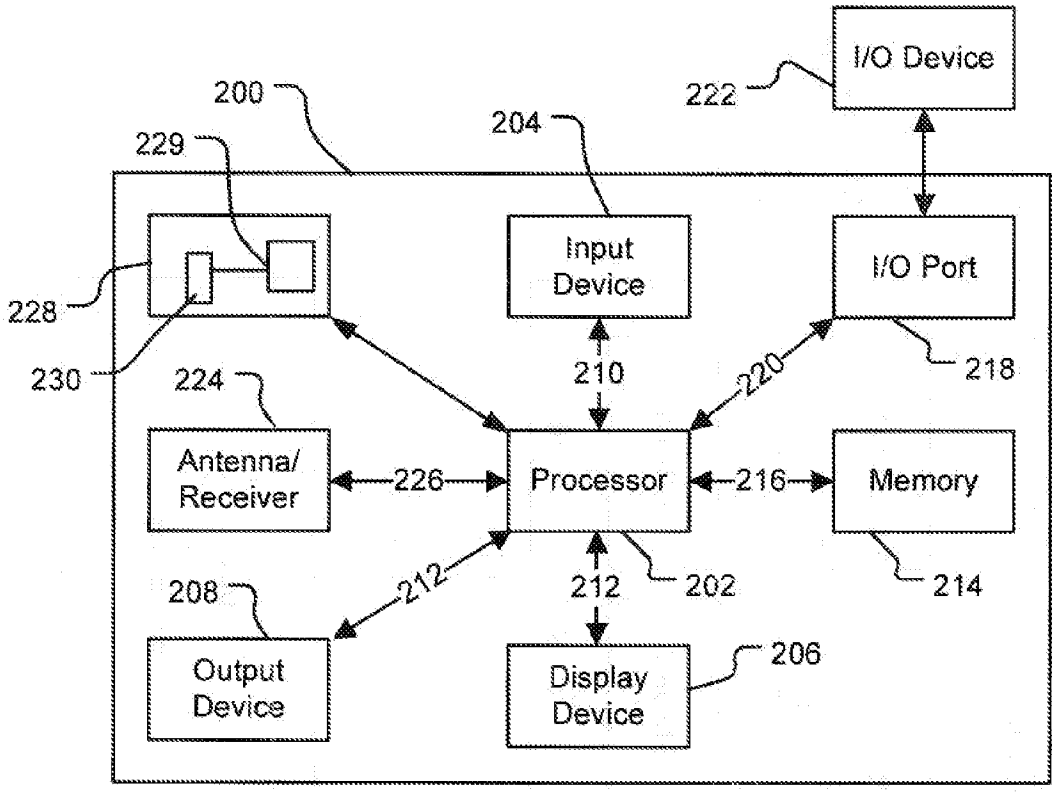

Figure 3

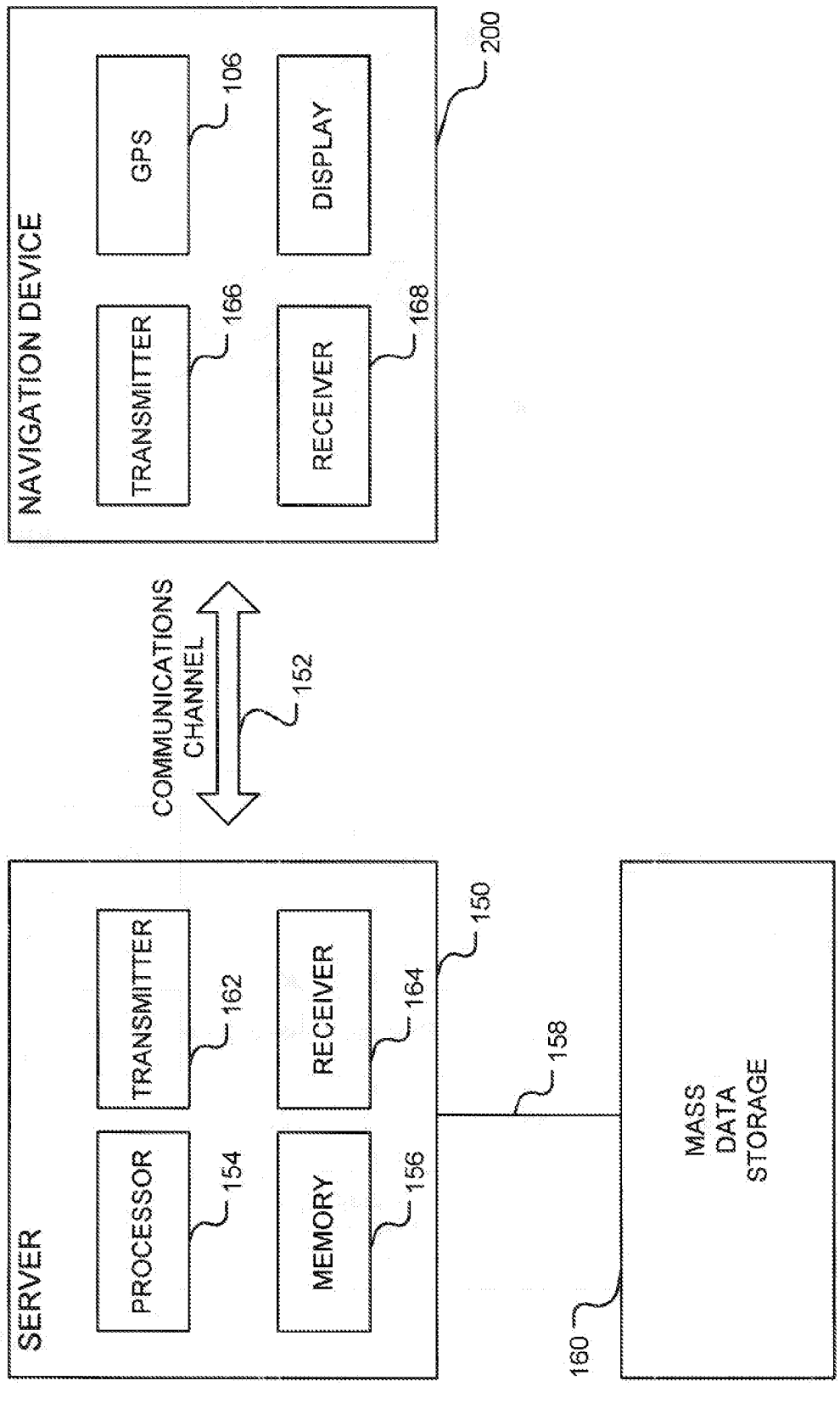
NAVIGATION DEVICE
GPS
106
DISPLAY
TRANSMITTER
166
RECEIVER
168
200
COMMUNICATIONS CHANNEL
152
SERVER
PROCESSOR
154
TRANSMITTER
162
MEMORY
156
RECEIVER
164
150
158
160
MASS DATA STORAGE

Figure 2

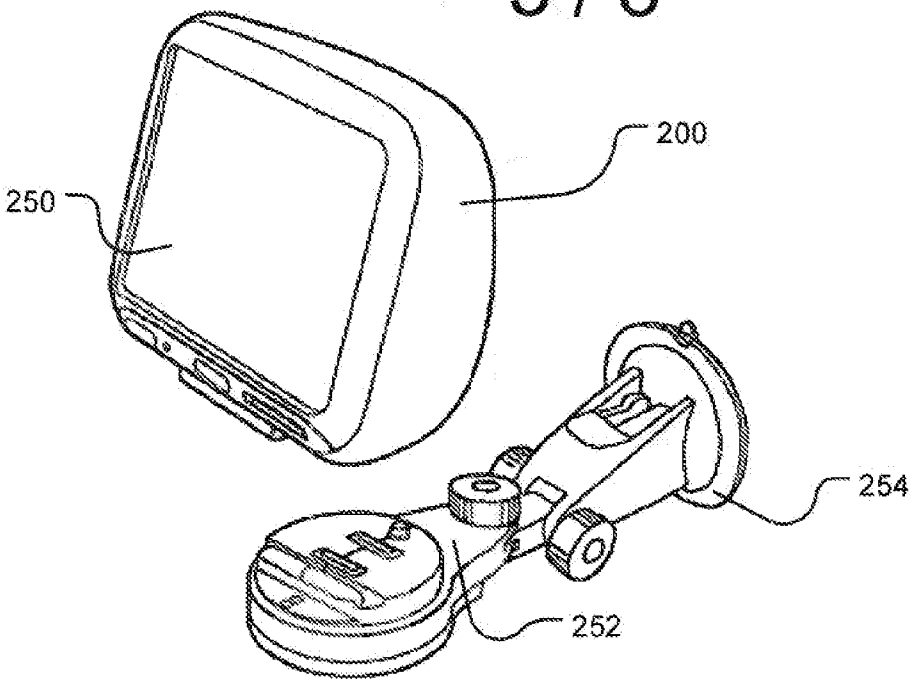

Figure 4

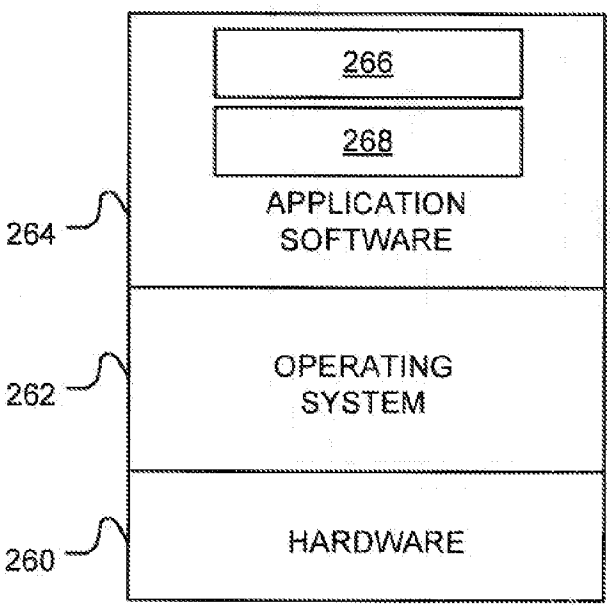

Figure 5

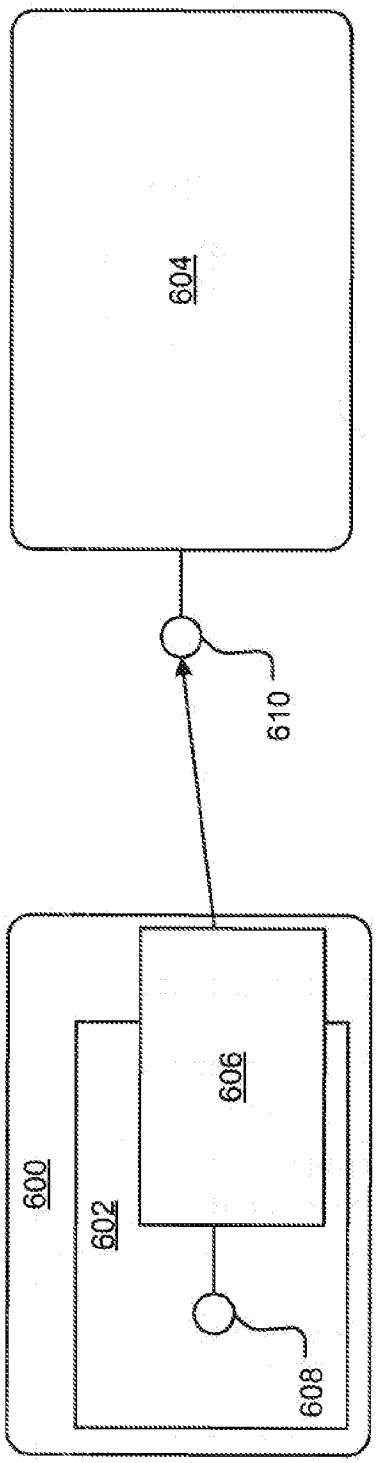

Figure 6

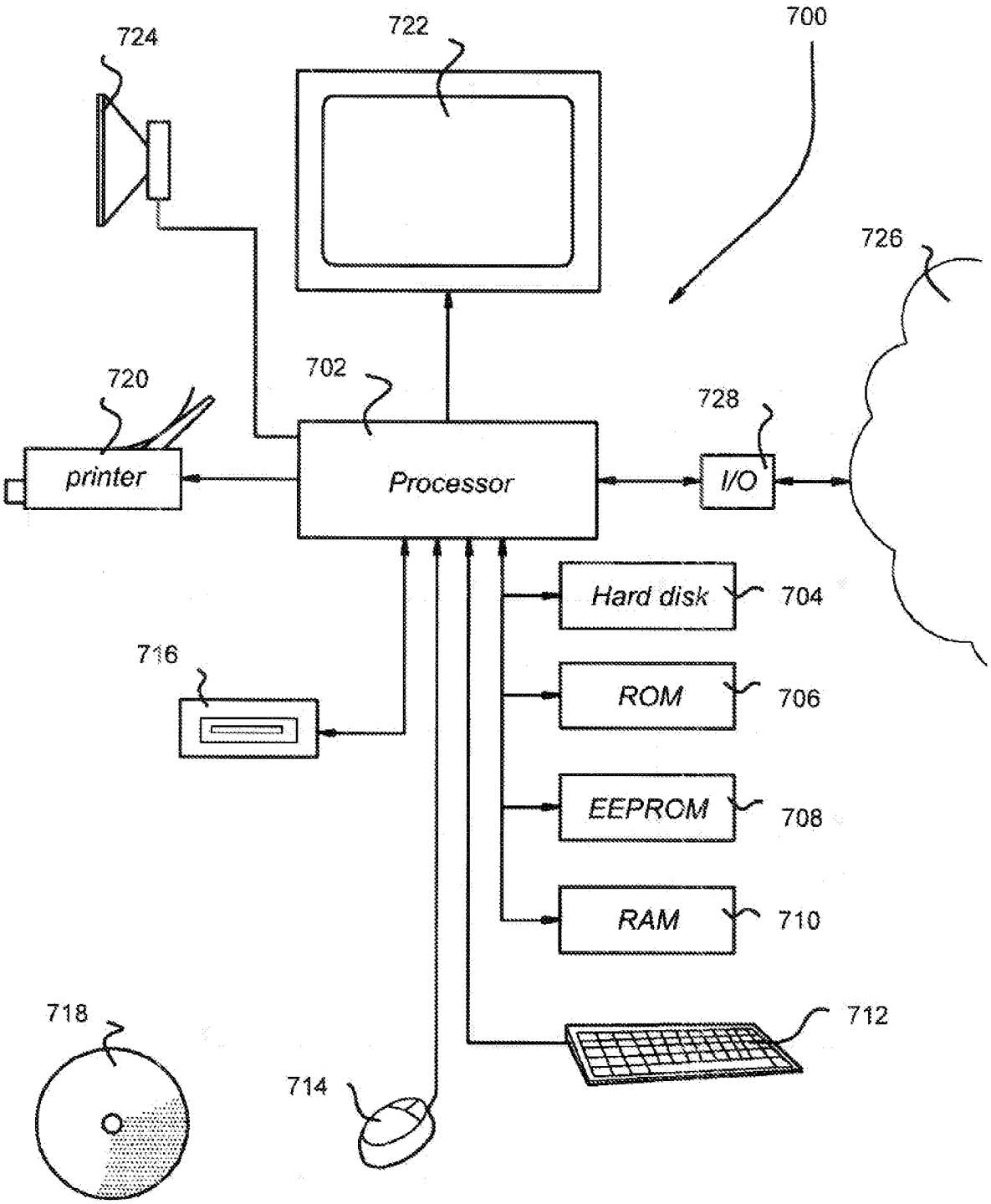
724
722
700
726
720
702
728
printer
Processor
I/O
Hard disk
704
716
ROM
706
EEPROM
708
RAM
710
718
712
714

Figure 7

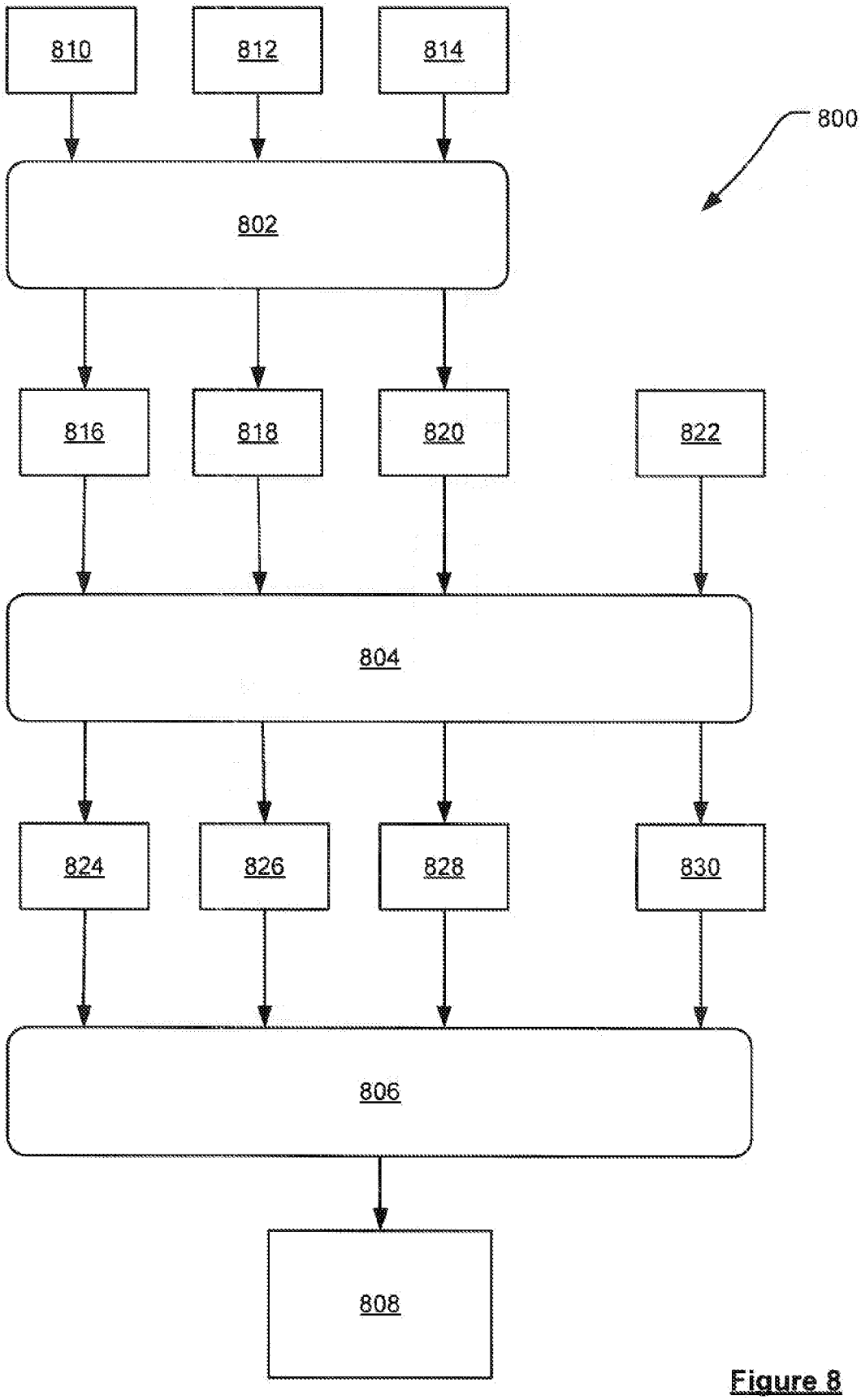

Figure 8

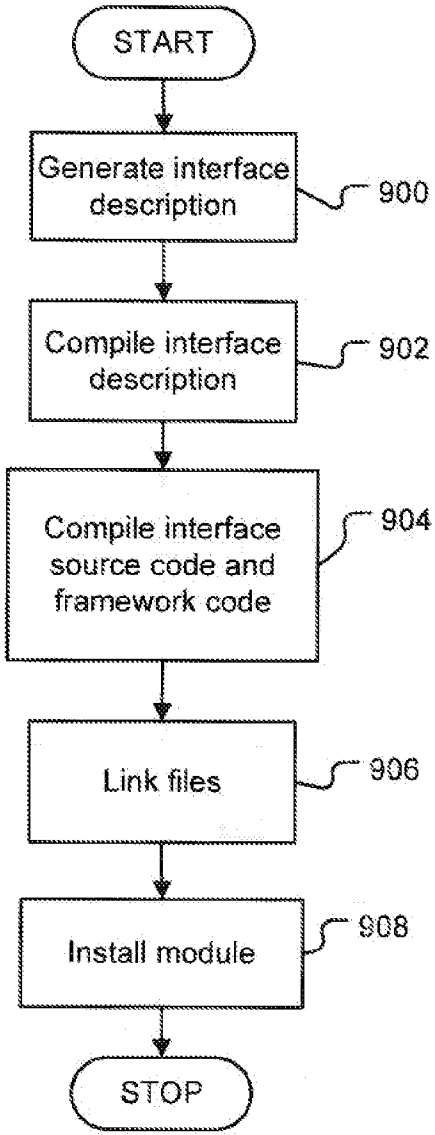

Figure 9

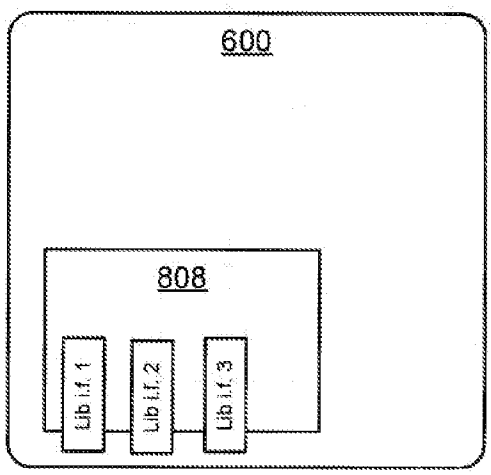

Figure 11

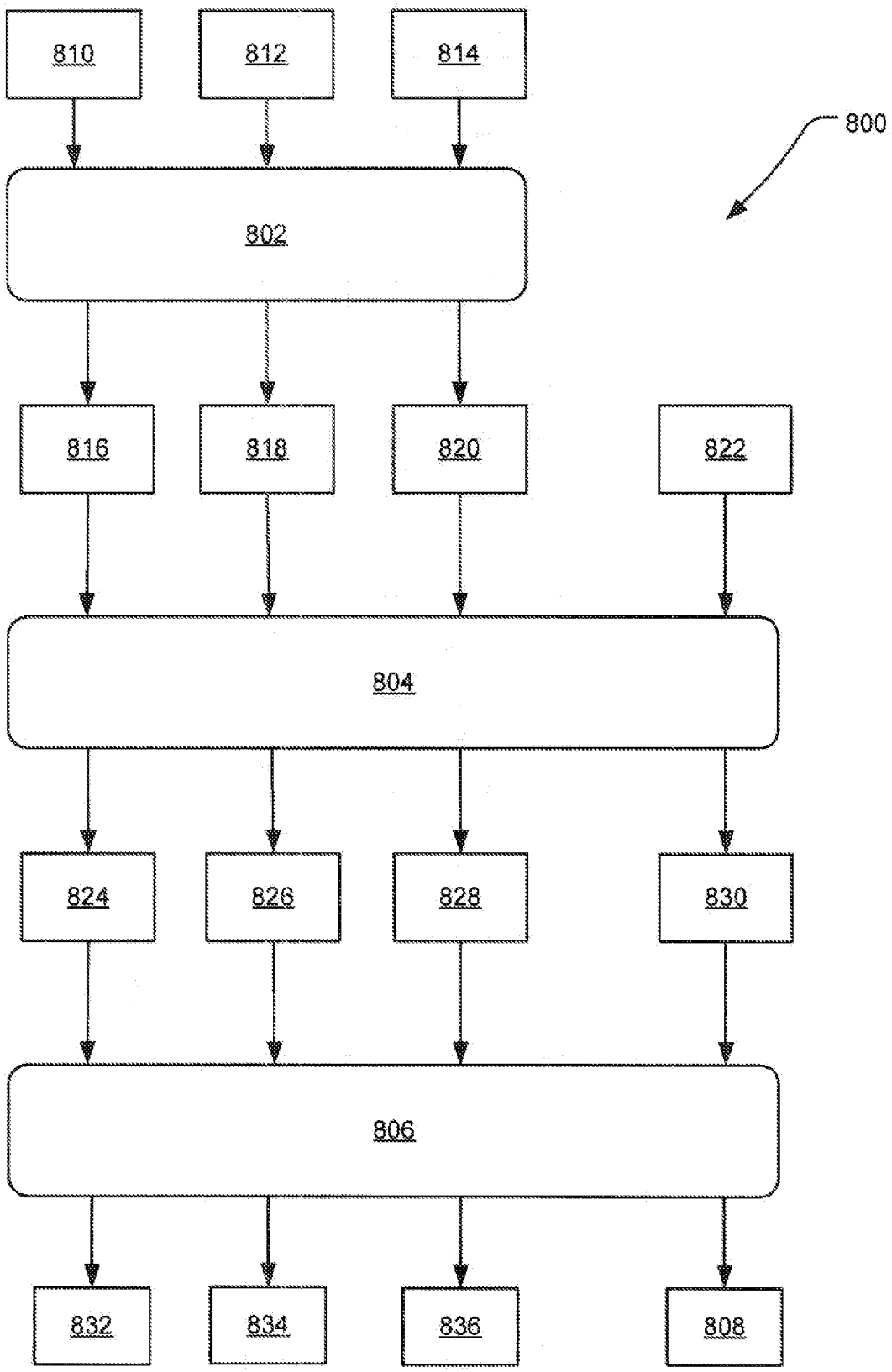
810
812
814
800
802
816
818
820
822
804
824
826
828
830
806
832
834
836
808

Figure 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/070295

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/44
ADD.

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

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

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 7 673 285 B1 (MICHEL RUBEN [US] ET AL) 2 March 2010 (2010-03-02) column 4, lines 25-35; figure 1 column 5, line 48 - column 6, line 17 column 5, lines 25-26 column 6, lines 18-26 -----	1-20

☐ 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 document 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
2 March 2011	18/03/2011

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Dieben, Marc

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2010/070295

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 7673285 B1	02-03-2010	NONE	