

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
27 September 2007 (27.09.2007)

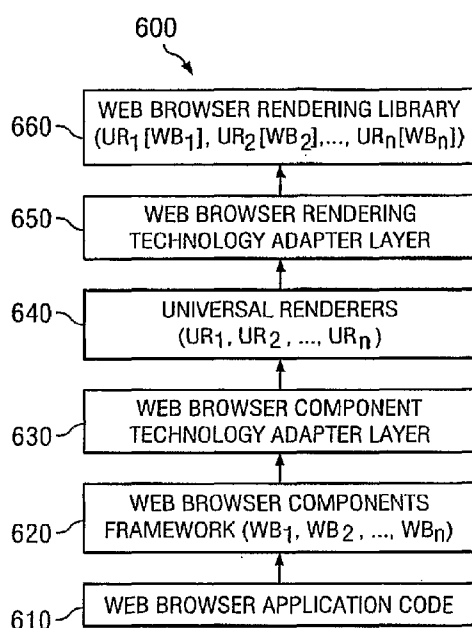
PCT

(10) International Publication Number
WO 2007/109138 A2

- (51) International Patent Classification: Not classified
- (21) International Application Number: PCT/US2007/006643
- (22) International Filing Date: 15 March 2007 (15.03.2007)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
11/385,441 21 March 2006 (21.03.2006) US
- (71) Applicant (for all designated States except US): **COMPUTER ASSOCIATES THINK, INC.** [US/US]; One Computer Associates Plaza, Islandia, NY 11749 (US).
- (72) Inventor; and
- (75) Inventor/Applicant (for US only): **ABRAMS, Howard, A.** [US/US]; 200 Brannan, #441, San Francisco, CA 94107 (US).
- (74) Agent: **PEDERSEN, Luke, K.**; Baker Botts L.L.P., 2001 Ross Avenue, Suite 600, Dallas, TX 75201 (US).
- (81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, 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, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, 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, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHOD AND SYSTEM FOR SHARING USER INTERFACE RENDERING AMONG HETEROGENEOUS ARCHITECTURES



(57) Abstract: According to various illustrative embodiments of the present invention, a method for sharing rendering of a user interface component for an information handling application comprises instructing a plurality of information handling application components, using an information handling application code, to send a particular component to be rendered by one of a plurality of universal renderers, wherein each universal renderer is associated with a respective one of the information handling application components. The method continues by rendering the particular information handling application component to produce a rendered information handling application component. The method concludes by providing the rendered information handling application component to an information handling application rendering library, so that when a plurality of the information handling application components has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components.

**Declarations under Rule 4.17:**

- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*
- *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Published:

- *without international search report and to be republished upon receipt of that report*

METHOD AND SYSTEM FOR SHARING USER INTERFACE RENDERING AMONG HETEROGENEOUS ARCHITECTURES

TECHNICAL FIELD OF THE INVENTION

5 The present invention relates generally to information processing and, more particularly, to systems and methods for improving implementations of a user interface by sharing the renderings of user interface components.

BACKGROUND OF THE INVENTION

10 As the value and use of information continues to increase, individuals and businesses continually seek additional ways to process and store information. Improving the user interface (UI) with underlying applications is an important aspect of facilitating the processing and storing of information.

15 Separating the rendering of what ultimately gets presented to the user by the user interface (UI) from the application data model of a user interface component is one conventional technique that helps decouple the "look and feel" of what is rendered to the user from the underlying application data model, using a component framework as a go-between. This separation allows developers to specify what data the user interface (UI) will ultimately present to the user without embedding any details into the application data
20 model itself, and without specifying how the data will look to the user.

25 For example, as shown at 100 in FIGURE 1, a conventional rendering of what ultimately gets presented to the user by the user interface (UI) from the application data model of a user interface component is illustrated. A Java Swing toolkit application code 110 instructs a Java Swing toolkit components framework 120, comprising a set of
30 Java Swing toolkit components $JST_1, JST_2, \dots, JST_n$, where n may be any non-zero integer, to send a particular component, such as JST_k , where k may be any non-zero integer less than or equal to n , to be rendered by Java Swing toolkit renderers 130. The Java Swing toolkit renderers 130 comprise a set of n Java Swing toolkit renderers $JSTR_1, JSTR_2, \dots, JSTR_n$, each specific to a respective one of the Java Swing toolkit components $JST_1, JST_2, \dots, JST_n$, where n may be any non-zero integer. For example, the Java Swing toolkit
renderer $JSTR_k$ specifically renders the Java Swing toolkit component JST_k , producing a rendered Java Swing toolkit component $JSTR_k[JST_k]$, where k may be any non-zero integer less than or equal to n . The rendered Java Swing toolkit component $JSTR_k[JST_k]$,

where k may be any non-zero integer less than or equal to n , may be provided to a Java Swing toolkit rendering library 140. When every one of the Java Swing toolkit components $JST_1, JST_2, \dots, JST_n$, has been rendered, then the Java Swing toolkit rendering library 140 comprises the set of rendered Java Swing toolkit components $JSTR_1[JST_1], JSTR_2[JST_2], \dots, JSTR_n[JST_n]$, where n may be any non-zero integer.

Similarly, as shown at 200 in FIGURE 2, for example, another conventional rendering of what ultimately gets presented to the user by the user interface (UI) from the application data model of a user interface component is illustrated. A JavaServer Faces standard application code 210 instructs a JavaServer Faces standard components framework 220, comprising a set of JavaServer Faces standard components $JSF_1, JSF_2, \dots, JSF_n$, where n may be any non-zero integer, to send a particular component, such as JSF_k , where k may be any non-zero integer less than or equal to n , to be rendered by JavaServer Faces standard renderers 230. The JavaServer Faces standard renderers 230 comprise a set of n JavaServer Faces standard renderers $JSFR_1, JSFR_2, \dots, JSFR_n$, each specific to a respective one of the JavaServer Faces standard components $JSF_1, JSF_2, \dots, JSF_n$, where n may be any non-zero integer. For example, the JavaServer Faces standard renderer $JSFR_k$ specifically renders the JavaServer Faces standard component JSF_k , producing a rendered JavaServer Faces standard component $JSFR_k[JSF_k]$, where k may be any non-zero integer less than or equal to n . The rendered JavaServer Faces standard component $JSFR_k[JSF_k]$, where k may be any non-zero integer less than or equal to n , may be provided to a JavaServer Faces standard rendering library 240. When every one of the JavaServer Faces standard components $JSF_1, JSF_2, \dots, JSF_n$, has been rendered, then the JavaServer Faces standard rendering library 240 comprises the set of rendered JavaServer Faces standard components $JSFR_1[JSF_1], JSFR_2[JSF_2], \dots, JSFR_n[JSF_n]$, where n may be any non-zero integer.

However, as shown in the Java Swing toolkit and the JavaServer Faces standard rendering implementations described above, whenever a new user interface component is developed, multiple new corresponding component renderers have to be developed as well, one for each toolkit and/or technology in use. For example, if a new Java Swing toolkit component JST_{n+1} , is developed, a corresponding new Java Swing toolkit renderer $JSTR_{n+1}$ has to be developed to be able to render the new rendered Java Swing toolkit component $JSTR_{n+1}[JST_{n+1}]$, to the Java Swing toolkit rendering library 140. Similarly, if a new JavaServer Faces standard component JSF_{n+1} , is developed, a corresponding new

JavaServer Faces standard renderer JSFR_{n+1} has to be developed to be able to render the new rendered JavaServer Faces standard component JSFR_{n+1}[JSF_{n+1}], to the JavaServer Faces standard rendering library 240.

SUMMARY OF THE INVENTION

According to various illustrative embodiments of the present invention, a method for sharing rendering of a user interface component for an information handling application comprises instructing a plurality of information handling application components, using an information handling application code, to send a particular component to be rendered by one of a plurality of universal renderers, wherein each universal renderer is associated with a respective one of the information handling application components. The method continues by rendering the particular information handling application component to produce a rendered information handling application component. The method concludes by providing the rendered information handling application component to an information handling application rendering library, so that when a plurality of the information handling application components has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components.

Various embodiments of the present invention may benefit from numerous advantages. It should be noted that one or more embodiments may benefit from some, none, or all of the advantages discussed below. The system and method disclosed herein are advantageous in providing a mechanism for reusing the renderer for a given component regardless of the underlying user interface framework or technology. In various illustrative embodiments, this may be accomplished through the use of adapters to isolate the renderers from the rendering technology and the respective user interface components. Other technical advantages will be apparent to those of ordinary skill in the art having the benefit of the present disclosure and in view of the following specification, claims, and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding the present invention and its advantages, reference is now made to the following description, taken in conjunction with the accompanying drawings, in which:

FIGURE 1 illustrates a block diagram showing a computer system, according to prior teachings;

FIGURE 2 illustrates a block diagram showing another computer system, according to prior teachings;

FIGURE 3 illustrates a block diagram showing an overview of an Internet-based (client/server) system, according to teachings of the present disclosure;

FIGURE 4 illustrates an exemplary embodiment of a method for sharing renderings of user interface components for a web browser, according to teachings of the present disclosure;

FIGURE 5 illustrates an exemplary embodiment of a method for sharing renderings of user interface components for a desktop application, according to teachings of the present disclosure;

FIGURE 6 illustrates an exemplary embodiment of a method for sharing renderings of user interface components for a handheld computer application, according to teachings of the present disclosure;

FIGURE 7 illustrates an exemplary embodiment of a method for sharing renderings of user interface components for a cell phone, according to teachings of the present disclosure; and

FIGURE 8 illustrates a method for sharing renderings of user interface components, according to teachings of the present disclosure.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS OF THE INVENTION

Illustrative embodiments of the present invention are described in detail below. The following description will focus on various illustrative embodiments, which may be operative in an Internet-connected environment, including, for instance, client machines running under the Microsoft® Windows® environment and connected to an open network, such as a wide area network (WAN) and/or the Internet. The present invention, however, is not limited to any particular one application or any particular environment. Instead, those skilled in the art having the benefit of the present disclosure will find that the system and methods of the present invention may be advantageously applied to a variety of system and application software, including database management systems, word processors, spreadsheets, and the like, operating on a variety of different platforms,

including the Macintosh[®] operating system, the UNIX[®] operating system, the NextStep[®] operating system, and the like. Therefore, the description of the exemplary embodiments that follows is for purposes of illustration and not limitation.

Various illustrative embodiments may generally be embodied on an information processing system including one or more computer systems, such as a computer system 300 of FIGURE 3, operating on a network. The computer system 300 may comprise a central processor 301, a main memory 302, an input/output controller 303, a keyboard 304, a pointing device 305 (*e.g.*, a mouse, a track ball, a pen device, and/or the like), a display and/or screen device 306, and a mass storage 307 (*e.g.*, a hard disk or a fixed disk, a removable floppy disk, an optical disk, a magneto-optical disk, a flash memory, and/or the like), a network interface card or controller 311 (*e.g.*, Ethernet, Firewire, and/or the like), and a modem 312 (*e.g.*, a 56K baud modem, an ISDN modem, a DSL modem and/or the like). Although not shown separately, a real-time system clock may be included with the computer system 300, in a conventional manner. The computer processor 301 may include and/or be coupled to a cache memory 309 for storing frequently accessed information. The cache memory 309 may be an on-chip cache and/or an external cache (as shown). One or more input/output (I/O) device(s) 308, such as a printing device and/or a slide output device, may be included in the computer system 300, as desired. As shown, the various components of the computer system 300 may communicate through a system bus 310 and/or a similar architecture. The computer system 300 itself may communicate with other computer systems via the network interface card 311 and/or the modem 312.

In various illustrative embodiments, a method and system for sharing renderings of user interface components for an information handling application is provided to reuse the renderer for a given component regardless of the underlying user interface framework or technology. In various illustrative embodiments, this may be accomplished through the use of adapters to isolate the renderers from the rendering technology and the respective user interface components. Referring to FIGURE 4, an exemplary embodiment of a method for sharing renderings of user interface components for a web browser is illustrated, according to teachings of the present disclosure, as shown at 600, for example. A web browser application code 610 may instruct a web browser components framework 620, comprising a set of web browser components $WB_1, WB_2, \dots, WB_n$, where n may be any non-zero integer, to send a particular component, such as WB_k , where k may be any non-zero

integer less than or equal to n , to be rendered by universal renderers 640. The universal renderers 640 may comprise a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the web browser components $WB_1, WB_2, \dots, WB_n$, where n may be any non-zero integer. For example, the universal renderer UR_k may specifically render the web browser component WB_k , producing a rendered web browser component $UR_k[WB_k]$, where k may be any non-zero integer less than or equal to n . The rendered web browser component $UR_k[WB_k]$, where k may be any non-zero integer less than or equal to n , may be provided to a web browser rendering library 660. When every one of the web browser components $WB_1, WB_2, \dots, WB_n$, has been rendered, then the web browser rendering library 660 may comprise the set of rendered web browser components $UR_1[WB_1], UR_2[WB_2], \dots, UR_n[WB_n]$, where n may be any non-zero integer.

In various illustrative embodiments, as shown in FIGURE 4, for example, a web browser component technology adapter layer 630 may optionally be disposed between the web browser components framework 620 and the universal renderers 640. In various other illustrative embodiments, a web browser rendering technology adapter layer 650 may optionally be disposed between the universal renderers 640 and the web browser rendering library 660. In still other various illustrative embodiments, the web browser component technology adapter layer 630 may optionally be disposed between the web browser components framework 620 and the universal renderers 640, and the web browser rendering technology adapter layer 650 may also optionally be disposed between the universal renderers 640 and the web browser rendering library 660.

Referring to FIGURE 5, another exemplary embodiment of a method and system for sharing renderings of user interface components for a desktop application is illustrated, according to teachings of the present disclosure, as shown at 700, for example. A desktop application code 710 may instruct a desktop application components framework 720, comprising a set of desktop application components $DA_1, DA_2, \dots, DA_n$, where n may be any non-zero integer, to send a particular component, such as DA_k , where k may be any non-zero integer less than or equal to n , to be rendered by universal renderers 740. The universal renderers 740 may comprise a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the desktop application components $DA_1, DA_2, \dots, DA_n$, where n may be any non-zero integer. For example, the universal renderer UR_k may specifically render the desktop application component DA_k , producing a rendered desktop application component $UR_k[DA_k]$, where k may be any non-zero integer less than or equal

to n . The rendered desktop application component $UR_k[DA_k]$, where k may be any non-zero integer less than or equal to n , may be provided to a desktop application rendering library 760. When every one of the desktop application components $DA_1, DA_2, \dots, DA_n$, has been rendered, then the desktop application rendering library 760 may
5 comprise the set of rendered desktop application components $UR_1[DA_1], UR_2[DA_2], \dots, UR_n[DA_n]$, where n may be any non-zero integer.

In various illustrative embodiments, as shown in FIGURE 5, for example, a desktop application component technology adapter layer 730 may optionally be disposed between the desktop application components framework 720 and the universal
10 renderers 740. In various other illustrative embodiments, a desktop application rendering technology adapter layer 750 may optionally be disposed between the universal renderers 740 and the desktop application rendering library 760. In still other various illustrative embodiments, the desktop application component technology adapter layer 730 may optionally be disposed between the desktop application components framework 720
15 and the universal renderers 740, and the desktop application rendering technology adapter layer 750 may also optionally be disposed between the universal renderers 740 and the desktop application rendering library 760.

Referring to FIGURE 6, another exemplary embodiment of a method and system for sharing renderings of user interface components for a handheld computer application is
20 illustrated, according to teachings of the present disclosure, as shown at 800, for example. A handheld computer application code 810 may instruct a handheld computer application components framework 820, comprising a set of handheld computer application components $HH_1, HH_2, \dots, HH_n$, where n may be any non-zero integer, to send a particular component, such as HH_k , where k may be any non-zero integer less than or equal to n , to
25 be rendered by universal renderers 840. The universal renderers 840 may comprise a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the handheld computer application components $HH_1, HH_2, \dots, HH_n$, where n may be any non-zero integer. For example, the universal renderer UR_k may specifically render the handheld computer application component HH_k , producing a rendered handheld computer
30 application component $UR_k[HH_k]$, where k may be any non-zero integer less than or equal to n . The rendered handheld computer application component $UR_k[HH_k]$, where k may be any non-zero integer less than or equal to n , may be provided to a handheld computer application rendering library 860. When every one of the handheld computer application

components $HH_1, HH_2, \dots, HH_n$, has been rendered, then the handheld computer application rendering library 860 may comprise the set of rendered handheld computer application components $UR_1[HH_1], UR_2[HH_2], \dots, UR_n[HH_n]$, where n may be any non-zero integer.

5 In various illustrative embodiments, as shown in FIGURE 6, for example, a handheld computer application component technology adapter layer 830 may optionally be disposed between the handheld computer application components framework 820 and the universal renderers 840. In various other illustrative embodiments, a handheld computer application rendering technology adapter layer 850 may optionally be disposed
10 between the universal renderers 840 and the handheld computer application rendering library 860. In still other various illustrative embodiments, the handheld computer application component technology adapter layer 830 may optionally be disposed between the handheld computer application components framework 820 and the universal renderers 840, and the handheld computer application rendering technology adapter layer 850 may also optionally be disposed between the universal renderers 840 and the
15 handheld computer application rendering library 860.

Referring to FIGURE 7, another exemplary embodiment of a method and system for sharing renderings of user interface components for a cell phone is illustrated, according to teachings of the present disclosure, as shown at 900, for example. A cell
20 phone application code 910 may instruct a cell phone components framework 920, comprising a set of cell phone components $CP_1, CP_2, \dots, CP_n$, where n may be any non-zero integer, to send a particular component, such as CP_k , where k may be any non-zero integer less than or equal to n , to be rendered by universal renderers 940. The universal renderers 940 may comprise a set of n universal renderers $UR_1, UR_2, \dots, UR_n$,
25 each applicable to a respective one of the cell phone components $CP_1, CP_2, \dots, CP_n$, where n may be any non-zero integer. For example, the universal renderer UR_k may specifically render the cell phone component CP_k , producing a rendered cell phone component $UR_k[CP_k]$, where k may be any non-zero integer less than or equal to n . The rendered cell phone component $UR_k[CP_k]$, where k may be any non-zero integer less than or equal to n ,
30 may be provided to a cell phone rendering library 960. When every one of the cell phone components $CP_1, CP_2, \dots, CP_n$, has been rendered, then the cell phone rendering library 960 may comprise the set of rendered cell phone components $UR_1[CP_1], UR_2[CP_2], \dots, UR_n[CP_n]$, where n may be any non-zero integer.

In various illustrative embodiments, as shown in FIGURE 7, for example, a cell phone component technology adapter layer 930 may optionally be disposed between the cell phone components framework 920 and the universal renderers 940. In various other illustrative embodiments, a cell phone rendering technology adapter layer 950 may optionally be disposed between the universal renderers 940 and the cell phone rendering library 960. In still other various illustrative embodiments, the cell phone component technology adapter layer 930 may optionally be disposed between the cell phone components framework 920 and the universal renderers 940, and the cell phone rendering technology adapter layer 950 may also optionally be disposed between the universal renderers 940 and the cell phone rendering library 960.

FIGURE 8 illustrates a method 1000 for sharing renderings of user interface components, according to teachings of the present disclosure. The method 1000 for sharing rendering of a user interface component for an information handling application may comprise instructing an information handling application components framework, comprising a set of information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer, using an information handling application code, to send a particular component, such as IH_k , where k is any non-zero integer less than or equal to n , to be rendered by universal renderers, comprising a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer, so that the universal renderer UR_k specifically renders the information handling application component IH_k , producing a rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n , as indicated at 1010. The method may also comprise providing the rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n , to an information handling application rendering library, so that when every one of the information handling application components $IH_1, IH_2, \dots, IH_n$, has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components $UR_1[IH_1], UR_2[IH_2], \dots, UR_n[IH_n]$, where n is any non-zero integer, as indicated at 1020.

Although various illustrative embodiments of the present invention and their advantages are described in detail, a person skilled in the art having the benefit of the present disclosure could make various alterations, additions, and/or omissions without

departing from the spirit and scope of the present invention, as defined by the appended claims.

WHAT IS CLAIMED IS:

1. A method for sharing rendering of a user interface component for an information handling application, the method comprising:

instructing an information handling application components framework, comprising a set of information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer, using an information handling application code, to send a particular component, such as IH_k , where k is any non-zero integer less than or equal to n , to be rendered by universal renderers, comprising a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer, so that the universal renderer UR_k specifically renders the information handling application component IH_k , producing a rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n ; and

providing the rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n , to an information handling application rendering library, so that when a plurality of the information handling application components $IH_1, IH_2, \dots, IH_n$, has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components $UR_1[IH_1], UR_2[IH_2], \dots, UR_n[IH_n]$, where n is any non-zero integer.

2. The method of Claim 1, further comprising disposing an information handling application component technology adapter layer between the information handling application components framework and the universal renderers.

3. The method of Claim 1, further comprising disposing an information handling application rendering technology adapter layer between the universal renderers and the information handling application rendering library.

4. The method of Claim 1, wherein the information handling application comprises at least one of a web browser, a desktop application, a handheld computer application, and a cell phone.

5. A system for sharing rendering of a user interface component for an information handling application, the system comprising:

an information handling application components framework, comprising a set of information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer;

universal renderers, comprising a set of n universal renderers $UR_1, UR_2, \dots, UR_n$, each applicable to a respective one of the information handling application components $IH_1, IH_2, \dots, IH_n$, where n is any non-zero integer, so that a universal renderer UR_k is capable of specifically rendering a particular information handling application component IH_k , producing a rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n ;

a memory operable to store:

an information handling application code capable of instructing the information handling application components framework to send the particular component IH_k , where k is any non-zero integer less than or equal to n , to be rendered; and

an information handling application rendering library capable of being provided the rendered information handling application component $UR_k[IH_k]$, where k is any non-zero integer less than or equal to n , so that when a plurality of the information handling application components $IH_1, IH_2, \dots, IH_n$, has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components $UR_1[IH_1], UR_2[IH_2], \dots, UR_n[IH_n]$, where n is any non-zero integer.

6. The system of Claim 5, further comprising an information handling application component technology adapter layer disposed between the information handling application components framework and the universal renderers.

7. The system of Claim 5, further comprising an information handling application rendering technology adapter layer disposed between the universal renderers and the information handling application rendering library.

8. The system of Claim 5, wherein the information handling application comprises at least one of a web browser, a desktop application, a handheld computer application, and a cell phone.

5 9. A method for sharing rendering of a user interface component for an information handling application, the method comprising:

instructing a plurality of information handling application components, using an information handling application code, to send a particular component to be rendered by one of a plurality of universal renderers, wherein each universal renderer is associated
10 with a respective one of the information handling application components;

rendering the particular information handling application component to produce a rendered information handling application component; and

providing the rendered information handling application component to an information handling application rendering library, so that when a plurality of the information handling application components has been rendered, then the information
15 handling application rendering library will comprise the set of rendered information handling application components.

10. The method of Claim 9, further comprising disposing an information
20 handling application component technology adapter layer between the information handling application components framework and the universal renderers.

11. The method of Claim 9, further comprising disposing an information
25 handling application rendering technology adapter layer between the universal renderers and the information handling application rendering library.

12. The method of Claim 9, wherein the information handling application
30 comprises at least one of a web browser, a desktop application, a handheld computer application, and a cell phone.

13. A system for sharing rendering of a user interface component for an information handling application, the system comprising:

an information handling application components framework, comprising a plurality of information handling application components;

a plurality of universal renderers, wherein each universal renderer is applicable to a respective one of the information handling application components, so that a particular universal renderer is capable of rendering a particular information handling application component to thereby produce a rendered information handling application component;

a memory operable to store:

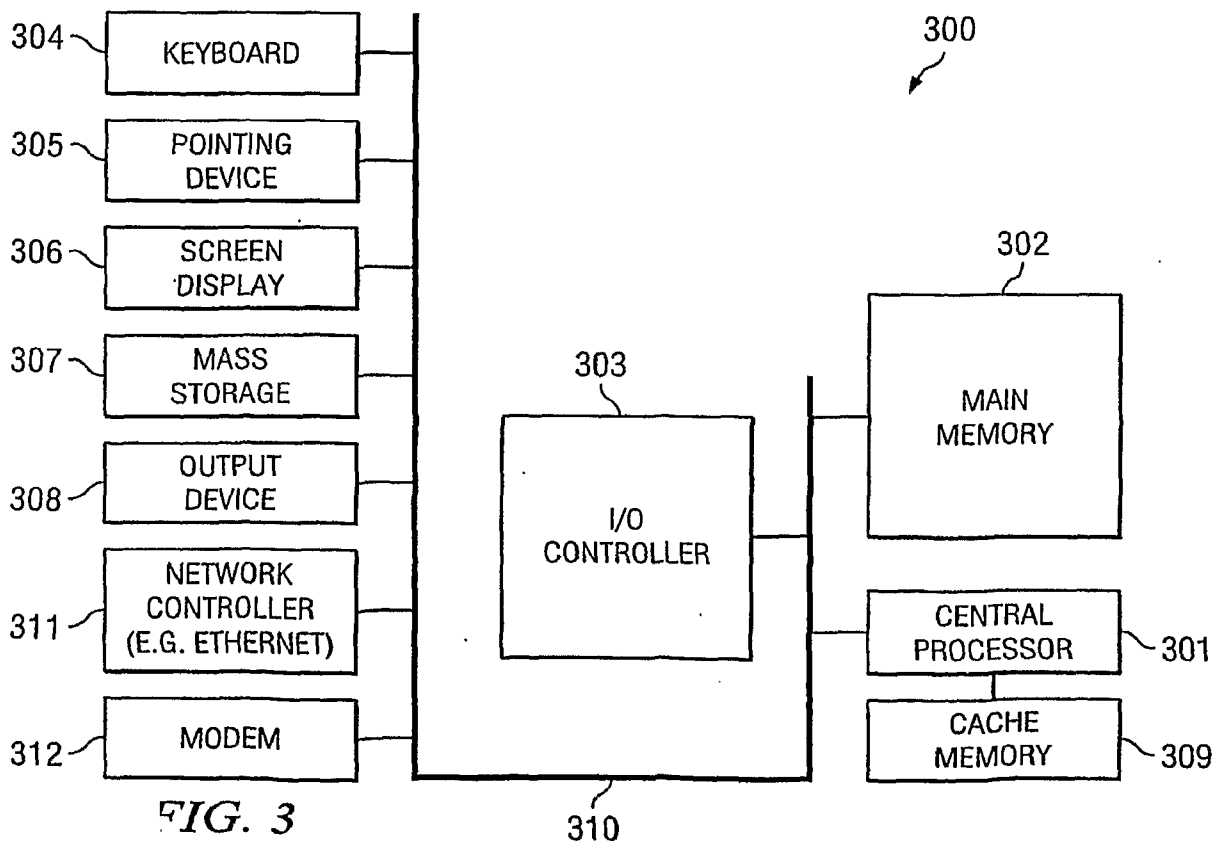
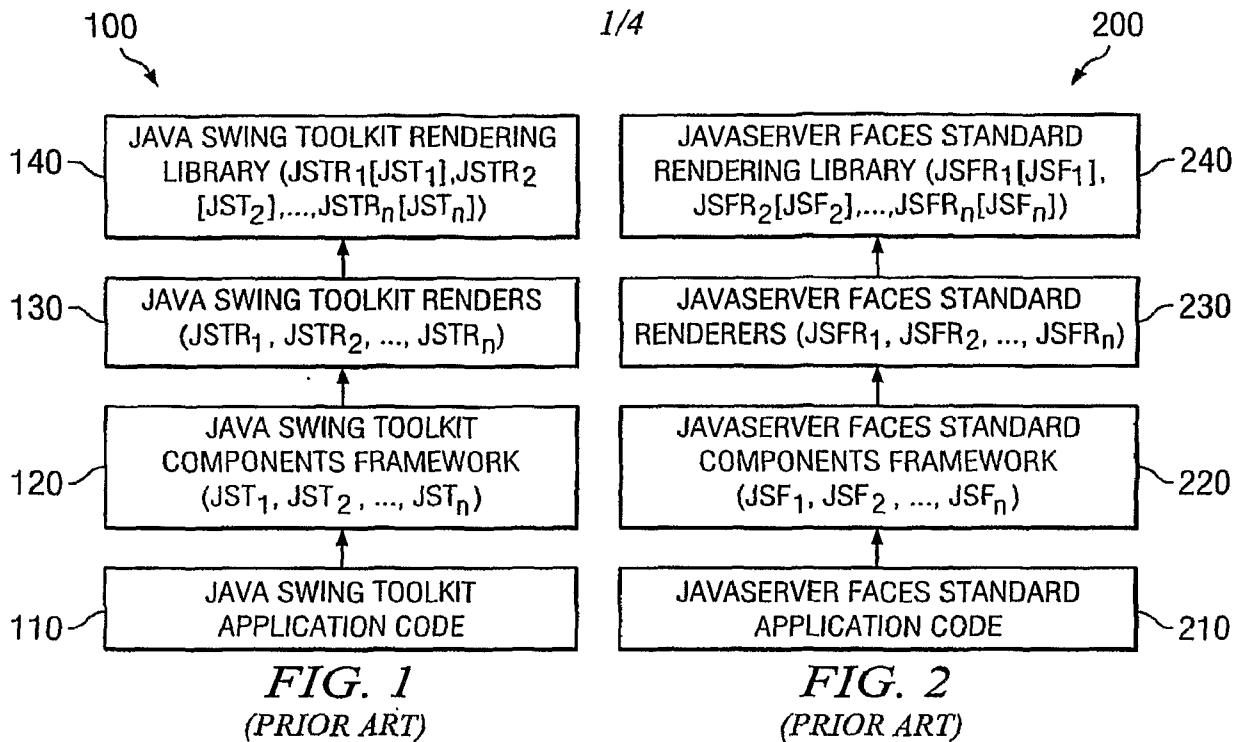
information handling application code capable of instructing the information handling application components framework to send the particular information handling application component to be rendered; and

an information handling application rendering library capable of being provided the rendered information handling application component so that when a plurality of the information handling application components has been rendered, then the information handling application rendering library will comprise the set of rendered information handling application components.

14. The system of Claim 13, further comprising an information handling application component technology adapter layer disposed between the information handling application components framework and the universal renderers.

15. The system of Claim 13, further comprising an information handling application rendering technology adapter layer disposed between the universal renderers and the information handling application rendering library.

16. The system of Claim 13, wherein the information handling application comprises at least one of a web browser, a desktop application, a handheld computer application, and a cell phone.



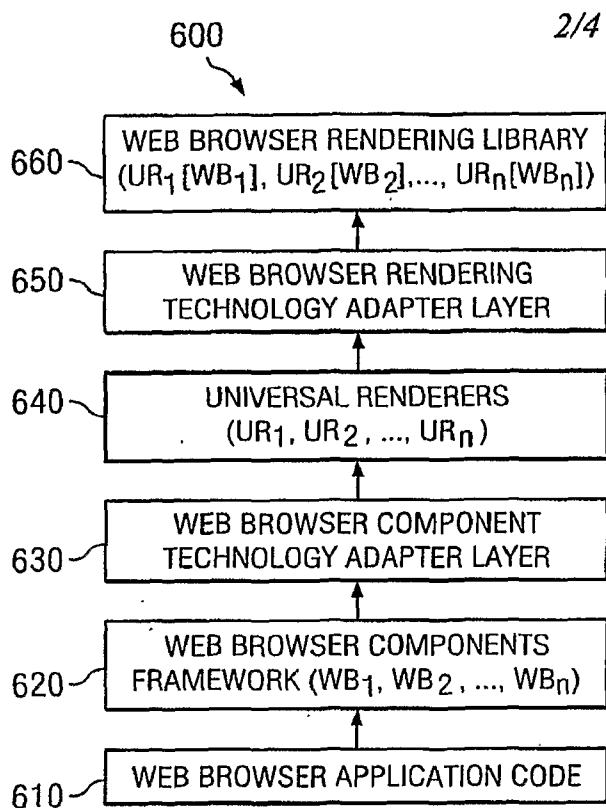


FIG. 4

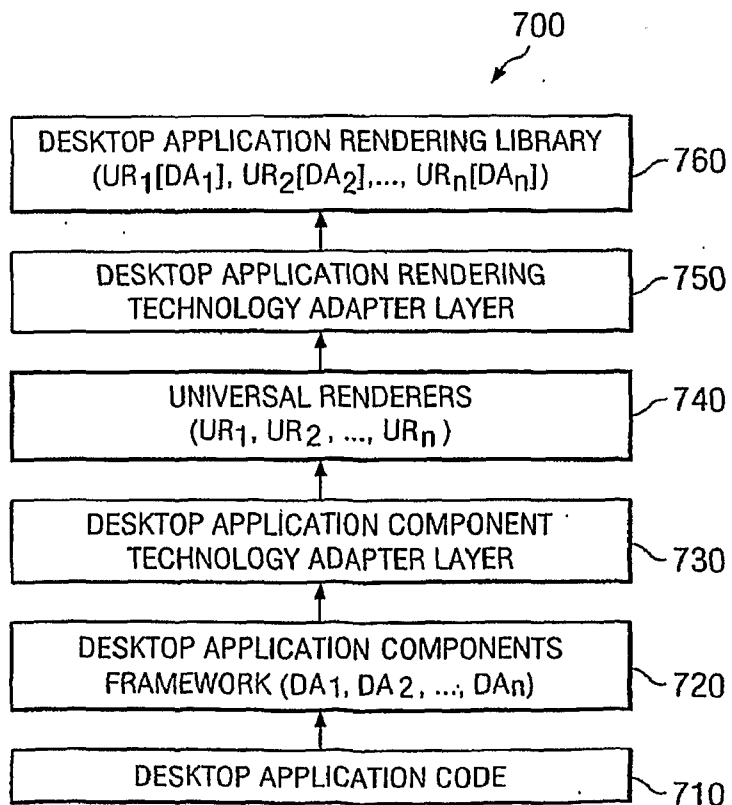


FIG. 5

3/4

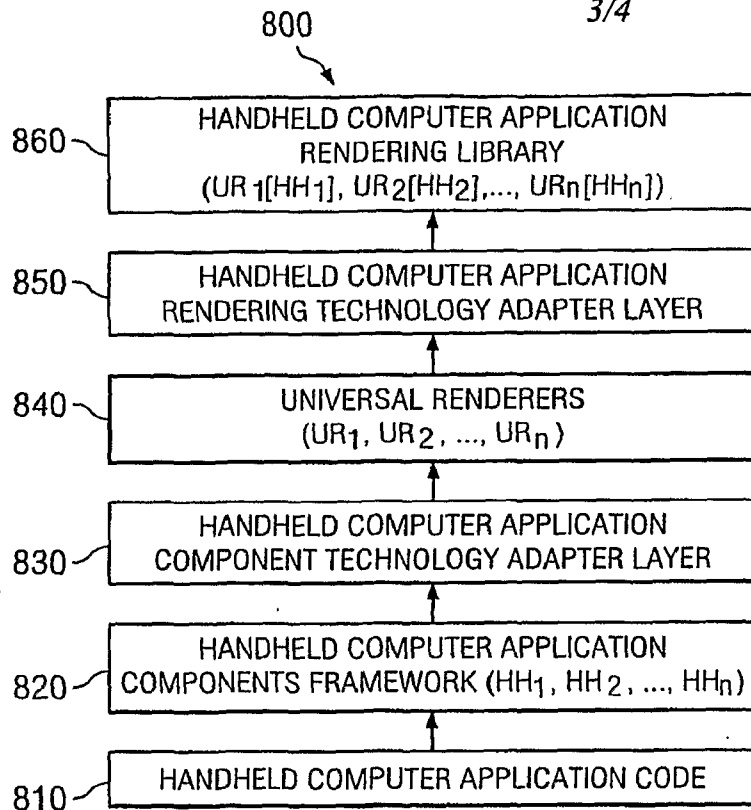


FIG. 6

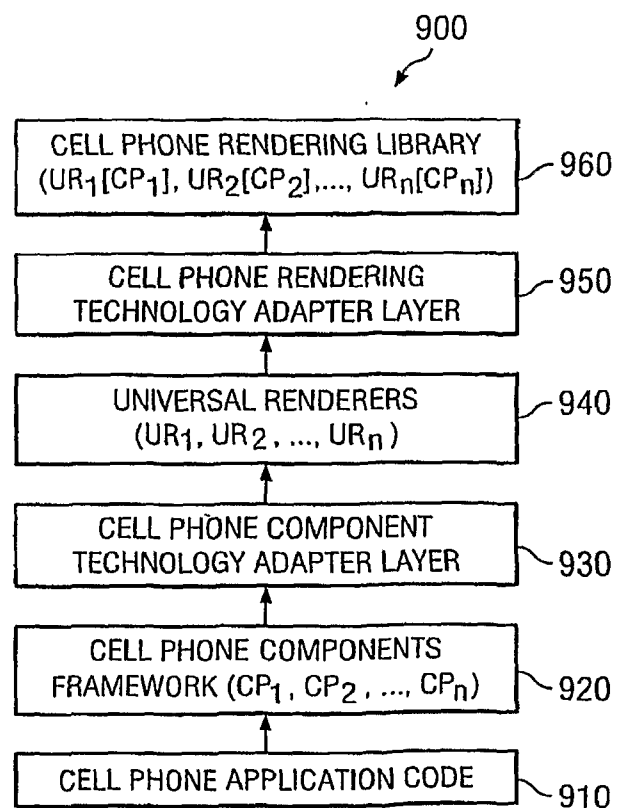


FIG. 7

4/4

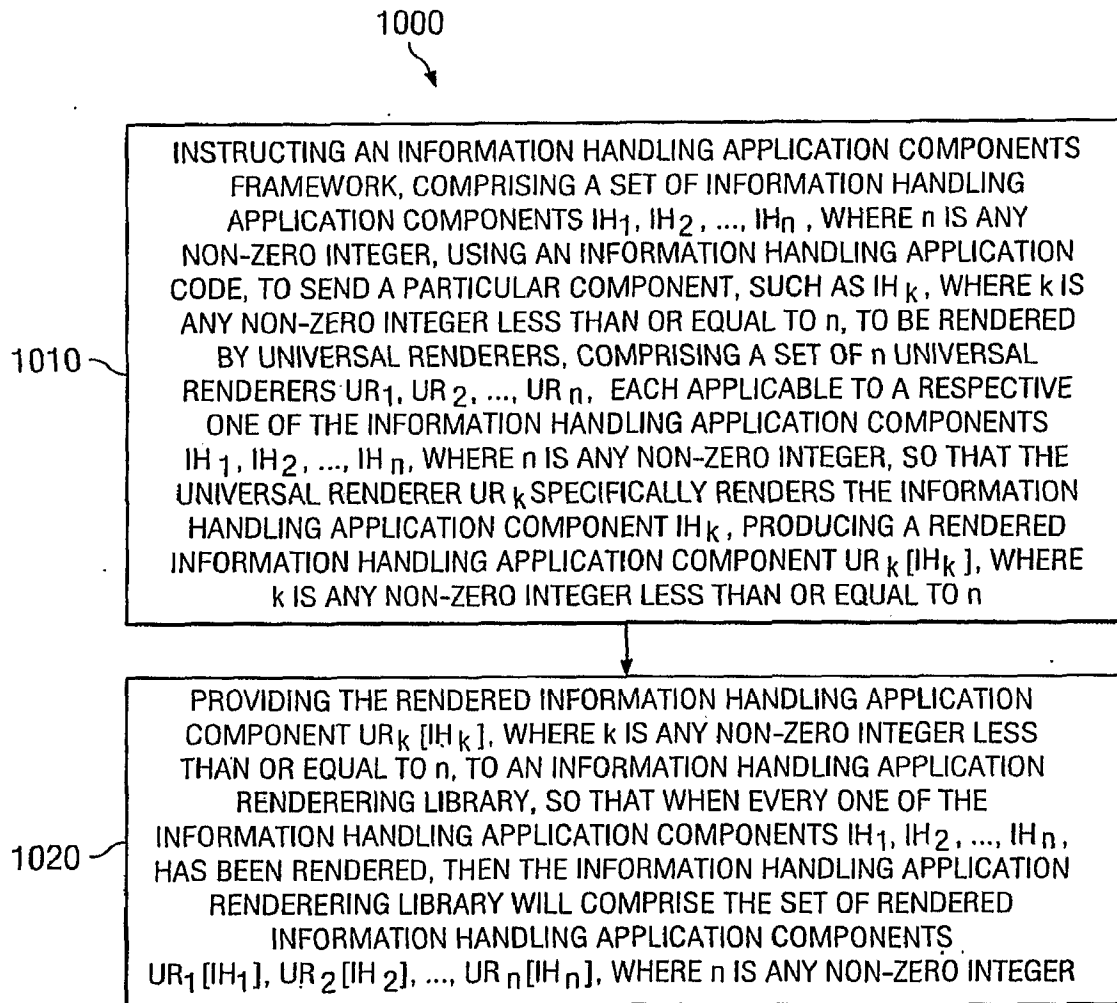


FIG. 8