



- (51) **International Patent Classification:**
G06F 17/21 (2006.01)
- (21) **International Application Number:**
PCT/US2013/076917
- (22) **International Filing Date:**
20 December 2013 (20.12.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
61/745,153 21 December 2012 (21.12.2012) US
- (71) **Applicant:** MONOTYPE IMAGING INC. [US/US]; 500 Unicorn Park Drive, Woburn, Massachusetts 01801 (US).
- (72) **Inventors:** KAASILA, Sampo Juhani; 6 Squirrel Run Road, Plaistow, New Hampshire 03865 (US). ORCUTT, Daniel Robert; 10 Depot Road, Boxford, Massachusetts 01921 (US). LEPPER, Mark Peter; 20 Ashmont Road, Wellesley, Massachusetts 02481 (US).
- (74) **Agent:** BARCLAY, Jeffrey J.; Fish & Richardson P.C., P.O. Box 1099, Minneapolis, Minnesota 55440-1022 (US).
- (81) **Designated States** (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM,

AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

- (84) **Designated States** (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— without international search report and to be republished upon receipt of that report (Rule 48.2(g))

(54) **Title:** SUPPORTING COLOR FONTS

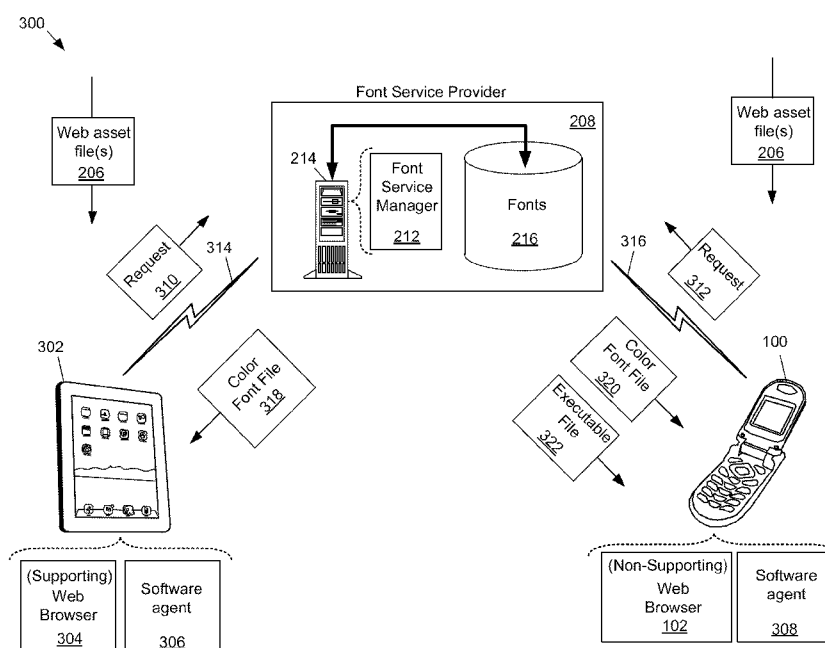


FIG. 3

(57) **Abstract:** A system includes a computing device that includes a memory configured to store instructions. The computing device also includes a processor to execute the instructions to perform operations that include receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset. In response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.

Supporting Color Fonts

CLAIM OF PRIORITY

This application claims priority under 35 USC §119(e) to U.S. Patent Application Serial No. 61/745,153, filed on December 21, 2012, the entire contents of which are hereby incorporated by reference.

BACKGROUND

[001] This description relates to techniques for allowing computing devices to support color fonts for presentation.

[002] In the ever-expanding connectivity and information sharing capabilities provided by computer networks such as the Internet, various types of web assets such as websites, webpages, etc. have been developed to assist with the transfer of information. Along with the almost explosive development of web assets, the content being presented by the assets has similarly grown. Text, audio, video, etc. is being incorporated into web assets to provide a more efficient and enjoyable viewing experience. For example, different languages, imagery, etc. may be included in web assets to tailor content for characteristics of a viewer such as their geographical location. However, providing such rich content does not come without some constraints. For example, the functionality of computing devices, operating systems, software, etc. may limit a viewer's ability to view and enjoy all of the content types that could be provided.

SUMMARY

[003] The systems and techniques described here relate to determining if an asset presenter (e.g., a web browser, a device, etc.) is capable of supporting fonts that are capable of employing multiple colors. Appropriate action can be taken based upon the determination. For example, if capable of supporting such color fonts, appropriate font information may be provided to the asset presenter for presenting the font (e.g., in an asset such as a webpage, website, etc.). If not supported by the asset presenter, font information may be prepared and provided such that the asset presenter can present the color fonts. By providing the ability to present such richly colored content, viewer experiences may be improved along with their interest in the asset (e.g., webpage, website, etc.) being presented.

[004] In one aspect, a computer-implemented method includes receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device. In response to receiving the information, the method include sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device. The color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.

[005] Implementations may include one or more of the following features. The executable file, when executed by the computing device, may adjust a document object model. Adjusting the document object model may include replicating glyphs of the web asset in a node-based structure representation of the document object model. Adjusting the document object model may include assigning one or more colors to the replicated glyphs. The color font file may include a font subset for presenting the content of the web asset. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[006] In another aspect, a system includes a computing device that includes a memory configured to store instructions. The computing device also includes a processor to execute the instructions to perform operations that include receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device. Operations also include, in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device. The color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.

[007] Implementations may include one or more of the following features. The executable file, when executed by the computing device, may adjust a document object model. Adjusting the document object model may include replicating glyphs of the web asset in a node-based structure representation of the document object model. Adjusting the document object model may include

assigning one or more colors to the replicated glyphs. The color font file may include a font subset for presenting the content of the web asset. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[008] In another aspect, one or more computer readable media storing instructions that are executable by a processing device, and upon such execution cause the processing device to perform operations that include receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device. Operations also include, in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device. The color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.

[009] Implementations may include one or more of the following features. The executable file, when executed by the computing device, may adjust a document object model. Adjusting the document object model may include replicating glyphs of the web asset in a node-based structure representation of the document object model. Adjusting the document object model may include assigning one or more colors to the replicated glyphs. The color font file may include a font subset for presenting the content of the web asset. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[010] In another aspect, a computer-implemented method that includes receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset. In response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.

[011] Implementations may include one or more of the following features. Sending the font information may include sending an executable file to the computing device. Sending the font information may include sending a color font file to the computing device. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[012] In another aspect, a system includes a computing device that includes a memory configured to store instructions. The computing device also includes a processor to execute the instructions to perform operations that include receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset. In response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.

[013] Implementations may include one or more of the following features. Sending the font information may include sending an executable file to the computing device. Sending the font information may include sending a color font file to the computing device. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[014] In another aspect, one or more computer readable media storing instructions that are executable by a processing device, and upon such execution cause the processing device to perform operations that include receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset. In response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.

[015] Implementations may include one or more of the following features. Sending the font information may include sending an executable file to the computing device. Sending the font information may include sending a color font file to the computing device. The color font file may include scalable vector graphics. Sending the color font information may include sending a file to the computing device that includes glyph spacing information and is absent glyph marking information. Positioning of the glyph spacing information as presented by the computing device may be adjustable based upon user interaction.

[016] These and other aspects and features and various combinations of them may be expressed as methods, apparatus, systems, means for performing functions, program products, and in other ways.

[017] Other features and advantages will be apparent from the description and the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 illustrates a mobile device presenting text in a single color.

FIG. 2 is a block diagram of an Internet based computer network.

FIG. 3 illustrates a font service provider for distributing font information to computing devices.

FIG. 4 illustrates presentable content that employs multiple colors.

FIG. 5 illustrates glyphs represented by spacing and non-marking information.

FIGs. 6a and b is an example flow chart of operations of a font service manager.

FIG. 7 is a block diagram showing an example of a system for providing hosted storage and accessing the hosted storage from a client device.

FIG. 8 illustrates an example of a computing device and a mobile computing device that can be used to implement the techniques described here.

DETAILED DESCRIPTION

[018] Referring to FIG. 1, many types of computing devices are capable of presenting various types of graphical content such as text, images, video, etc. To present text and similar graphics,

various types of font families (e.g., Times New Roman, Arial, etc.) may be used that typically include a set of fonts, e.g., regular, italic, bold, bold italic, etc. Each font generally includes a set of individual character shapes called glyphs and the glyphs generally share various design features (e.g., geometry, stroke thickness, serifs, size, etc.) associated with the font. One or more techniques may be utilized for representing such fonts; for example, outline-based representations may be adopted in which lines and curves are used to define the borders of glyphs. Such fonts may be scalable for a variety of sizes (e.g., for rendering by various computing devices) and may be represented in one or more formats. For example, scalable outline fonts may be represented in a format that includes data structures capable of supporting a variety of typographic visual symbols of many languages. Along with the size and shape of the glyphs used by the font to represent a character, a color is also generally applied. For example, relatively dark colors (e.g., black, blue, etc.) are typically implemented for presenting the font characters. However, other colors such as lighter shaded colors (e.g., white, yellow, etc.) may also be implemented (e.g., when a darker background such as a black background is being presented).

[019] As illustrated in the figure, some devices, operating systems, software applications etc. may be capable of presenting font characters and features that employ a single color. However, often such devices, operating systems, applications, etc. are unable to support presenting such characters and features in multiple colors. For example, it may be desired to present an individual font with multiple colors such as one portion of a character in one color (e.g., the horizontal crossbar of the character “T”) and another portion of the character (e.g., the vertical post of the character “T”) in another color. In another example, different characters may be presented in different colors, for example the font character “A” may be presented in the color red while the font character “B” may be presented in the color blue. In this illustrated example, a cellular telephone 100 is executing an asset presenter (e.g., web browser application 102) to present content transmitted over one or more networks (e.g., the Internet) from a variety of sources. In this situation, the web browser 102 does not support the ability to present font characters in multiple colors. For example, each character of the presented phrase “Hi” 104 may only be presented by the web browser 102 in one color (e.g., the color black) and not in multiple colors. Based upon the browser being executed by a user computing device, the ability to use multiple colors for individual characters may or may not be supported and multi-color content

from an author may not be presented as originally prepared. However, by knowing the capabilities of a device, its operating system, executed applications, etc., information may be prepared and provided to the device so it can be capable of properly presenting such multi-color content as originally intended.

[020] Referring to FIG. 2, a computing environment 200 includes a computing device (e.g., the cellular telephone 100) that a user may interact with (e.g., using a keypad, etc.) to identify a target web asset (e.g., website, webpage, etc.) for being presented by the computing device. For example, the web browser 102 or other type of asset presenter (e.g., a software application) may be executed by the cellular telephone 100 for the user to target one or more webpages. Upon being identified, operations of the web browser 102 may include requesting, via the Internet 202, content from one or more webpage sources 204 a,b,c for the target webpage(s). As illustrated, in this particular example, a webpage page is requested from web asset source 204a and a corresponding web asset file or files 206 are sent from the source through the Internet 202 to the cellular telephone 100. In one arrangement, the web asset files 206 include a hypertext markup language (HTML) file that includes instructions for presenting the asset and a cascading style sheet (CSS) file that provides presentation semantics for the asset being provided by the HTML file.

[021] To determine if the web browser 102 is capable of processing and presenting multi-color character fonts, one or more techniques and methodologies may be implemented. For example, operations may be executed at one or more locations for making the determination. In the illustrated example, operations may be executed by the cellular telephone 100 (or other types of user computing devices) and a font service provider 208, which is in communication with the computing device through one or more networks (e.g., the Internet 202). The font service provider 208 may incorporate one or more architectures, layouts, etc. For example, the font service provider 208 may incorporate a relatively large distributions of systems, computing devices (e.g., servers), etc. deployed in one or more locations (e.g., different geographical locations) and can be considered a content delivery network. Once determined if the web browser is capable of supporting multi-color character fonts or not, the font service provider 208 may execute operations to provide appropriate information to the computing device for presenting the multi-color content for either situation (e.g., the web browser supports multi-color

fonts or the web browser does not support multi-color fonts). As illustrated in the figure, a color font file 210 represents the multi-color information that is sent from the font service provider 208 to the user device (i.e., the cellular telephone 100). Once the multi-color information (e.g., contained in the color font file 210) is received, the information is used by the user device (e.g., executed by an asset presenter such as a web browser) to present the multi-color character fonts. In the illustrated example, a determination is made whether a web browser supports color fonts, however such a determination may be made for other types of web asset presenters. For example, along with web browsers, a web asset presenter may be considered as one or more applications (referred to as a web-based application) that can access or be accessed over a network such as the Internet, an intranet, etc. Such web-based applications may also be considered as software applications that are hosted over a network and coded in a browser supported programming language (such as JavaScript, combined with a browser-rendered markup language such as HTML, etc.). A web asset presenter may also be one or more applications (e.g., native application) executed on a computing device (or multiple devices) such as a user device (e.g., the cellular telephone 100) that provide a view of a web asset (e.g., a web view). Similar applications executed locally, remotely, or in combination among multiple locations may be considered as a web asset presenter. Similar to making the determination for web asset presenters, such a determination may be made for asset presenters that do not communicate with networks such as the Internet. For example, an asset presenter may be considered as one or more applications locally executed by a computing device that is capable of presenting network based assets such webpages, websites, etc. without being in communication with the Internet.

[022] One or more architectures may be implemented by the font service provider 208 for determining if an asset presenter is capable of presenting color fonts along with other functionality. In the illustrated arrangement, a font service manager 212 is executed by a server 214 located at the font service provider 208. To prepare and provide the color font file 210 (or files) to the user device, the font service manager 212 may access information from one or more sources such as a storage device 216 (e.g., one or more hard drives, CD-ROMs, etc.) located at the font service provider 208. However, the font information provided by the server 214 may also be collected from one or more other sources located internal or external to the font service provider 208. In one arrangement, upon receiving and executing the web asset file(s) 206 (e.g.,

the HTML file), the web browser 102 may initiate a request to be delivered to the font service provider 208 that asks for a software agent to be sent to the user device (e.g., the cellular telephone 100). Such agents can be considered as a software module that may be executable in a substantially autonomous manner. For example, upon being provided to the user computer device (e.g., the cellular telephone 100), a software agent may operate without considerable user interaction. By operating in a somewhat flexible manner, the software agent can adaptively identify if the web browser 102 being executed by the user device is capable of supporting color fonts such as multiple colors being applied to one or more font characters. In one arrangement, the software agent may be implemented as a file that includes scripting language that is capable of supporting a variety of programming styles (e.g., JavaScript). Once received, the software agent may be executed (e.g., by the web browser 102) to determine if the executed browser 102 is capable of presenting color fonts. To make such a determination, one or more techniques may be implemented. For example, by identifying the browser being executed (e.g., from information such as browser type, version, manufacturer, etc.) the software agent may use one or more predefined rules (e.g., Microsoft Internet Explorer version 10 supports color fonts, Microsoft Internet Explorer version 8 does not support color fonts, etc.) to determine if the web browser is capable of presenting such color fonts such as multi-color characters. Along with a rules-based determination being used by the software agent to determine if a web browser supports color fonts, other techniques and methodology may be implemented. For example, the software agent may poll the asset presenter (e.g., web browser), test the asset presenter, etc. to determine if the asset presenter supports color fonts such as multi-color font characters.

[023] Operations may be executed remotely from the user device to determine if an asset presenter (e.g., executed by the user device) is capable of supporting color fonts such as multi-color font characters. For example, operations may be executed at the font service provider 208 (with or without operations being executed on the user device) for making such determinations. In one arrangement, the font service provider 208 may use information provided by the user device to execute the determinations. For example, software (e.g., a user agent) operating in association with asset presenter (e.g., the web browser 102) may provide information to the font service provider 208. Along with identifying information associated with the asset presenter (e.g., web browser name, type, version, capabilities, etc.), the user agent may also provide other information such as device information (e.g., type of user device, version, capabilities, etc.),

other software executed by the device (e.g., operating system, applications, etc.) and other types of information (e.g., protocols being used by the web browser, etc.). From the provided information, the font service provider 208 may determine if the user device (e.g., cellular telephone 100) is capable of supporting color fonts.

[024] Software agents provided to the user device may also provide other type of functionality. For example, a software agent may also gather other information for having a target web asset (that uses color fonts) to be presented by the user computing device. For example, the software agent may identify the particular color-related information from the web asset that may be needed from the font service provider 208 to present the asset. In one arrangement, the software agent may scan the CSS file provided with the web asset file(s) 206 of the web asset to determine such color-related information included in the web asset. The software agent may also review the HTML file provided with the web asset file(s) 206 to identify color-related information such as the particular characters, glyphs, etc. and other types of typographical elements (e.g., ligatures, a single glyph fractions, subscripts, superscripts, etc.) being used by the asset. Other types of information may also be collected for being provided to the font service provider 208. For example, information associated with the user computing device, operating system used by the device, application(s) executed by the device (e.g., the type of web browser being executed by the device), etc. Once identified, this information may be provided to the font service provider 208 for processing and preparing the needed information for delivery to the user computing device (e.g., the cellular telephone 100) for presenting the content of the asset. Along with color-related information, other types of information may be provided to the font service provider 208. For example, the particular fonts, characters, typographical features, etc. included in the content of the web asset may be identified (e.g., by a software agent) and provided. In response to being provided this information, the font service manager 212 may prepare one or more subsets of appropriate font data (along with the color-related information) such that the only font information provided to the user device is the information needed to present the web asset and no additional font information (e.g., font characters not included in the web asset) is transmitted from the font service provider 208 to the user computing device.

[025] Referring to FIG. 3, a diagram 300 graphically illustrates data transfers such that appropriate information (e.g., color font information) is provided to asset presenters (e.g., web

browsers) for presenting web assets based upon whether or not the asset presenter supports color fonts. In this illustration, one computing device (i.e., the cellular telephone 100) does not support this capability and another computing device (i.e., a tablet computing device 302) is capable of presenting such color fonts. Both devices respectively execute browsers 102, 304, which in this example, are directed to the same target web asset. From the source of the web asset, the corresponding web asset file(s) 206 are provided to each device 100, 302. Upon receiving the file(s), software agents 306, 308 are respectively provided to the devices (e.g., from the font service provider 208) or are previously residing and are executed (e.g., to determine if each corresponding device is capable of providing support for such color fonts). To initiate the delivery of the software agents 306, 308, one or more techniques may be implemented. For example, once delivered, the web asset file(s) 206 (e.g., an HTML file and a CSS file) may be used (e.g., executed) by the recipient device to initiate a request being sent to the font service provider 208. Receiving the request, a software agent (e.g., a JavaScript file) may be sent as a reply from the font service manager 212 being executed by the server 214 at the font service provider 208. Along with sending the software agent, the font service provider 208 may perform other operations associated with the software agent. For example, at predefined times (e.g., intervals, event triggered times, etc.) the agent may be updated by the font service provider 208 as information is collected. Information regarding which types of browsers is capable of supporting color fonts, incapable of supporting color fonts, etc. may be gathered and included in updated versions of the software agent. The font service manager 212 may execute other operations in some arrangements. For example, the font service manager 212 may determine if a web browser is capable of supporting or not supporting color fonts and other potential features (e.g., scalable font formats such as OpenType fonts). Receiving information from a device (e.g., the type of browser being executed by the device in a request such as request 310), the font service manager 212 may be able to determine if color fonts are supported by the device and execute appropriate operations. For example, information associated with the asset presenter (e.g., web browser 304) and collected by a user agent may be provided in the request (being sent from the device to the font service provider 208). From the provided information and possibly other information (e.g., present at the font service provider 208), the font service manager 212 may determine the color font capabilities of the asset presenter executed by the device and take appropriate action.

[026] The software agents 306, 308 may be capable of performing other operations. For example, operations may be respectively executed by the software agents 306, 308 to collect information regarding color information, typographical features and other content of the web asset being provided by the web asset file(s) 206. In one example, each software agent may read the CSS file included with the web asset file(s) 206 to identify which font information (e.g., color information, classes, etc.) may be used by the asset. The software agent may also read the HTML file included with the web asset file(s) 206 to identify the particular colors, characters, features, etc. that may be used by the web asset. In some arrangements, the software agent may read information of the HTML file from other sources, for example, information from a document object model (DOM), a node-based structure used to organize the information (e.g., a DOM tree), etc. By identifying data such as the colors, characters, etc. used by the web asset, one or more advantageous operations may be executed. For example, if a relatively small number of colors, characters, glyphs, etc. for a particular font are identified as being presented for the web asset, an appropriate font subset that only includes those identified colors, characters, glyphs, typographical features (e.g., OpenType font features) etc. may be provided by the font service provider 208, thereby conserving processing time and memory by not having font information absent from the web asset being sent to the corresponding user device. Incremental subsetting techniques may also be employed. For example, after a subset is produced for the color fonts, characters, typographical features, etc. present in a first page of a web asset, this identified information may not be used for producing subset(s) for subsequent pages (e.g., a second page) of the web asset. By filtering out color fonts, characters, typographical features, etc. already present in a subset, less processing time and memory may be consumed in a redundant manner.

[027] In some arrangements, operations of the software agents 306, 308 may be initiated by one or more conditions being satisfied, one or more events occurring, etc. For example, the software agents may not use the contents of the web asset file(s) 206 until appropriately signaled that one or more events have occurred. In one arrangement, applications such as a text-to-speech application may process the contents of the web asset file(s) 206 prior to being used by the software agents 306, 308. For example, once textual content of the web asset file(s) 206 (e.g., an HTML file) has been read by the text-to-speech application (e.g., for conversion to audible speech), one or more signals may be sent (or other type of signaling technique employed) to

notify the software agents 306, 308 that the web asset files(s) 206 may be accessed and used. In some arrangements, operations of the software agent may execute substantially in parallel with operations of other applications that also use the contents of the web asset file(s) 206 (e.g., the software agent and one or more other applications may access content of the web asset file(s) in parallel).

[028] To provide the collected information (e.g., information collected by a user agent, does the browser support color fonts, etc.; what color information, color fonts, etc. are included in the web asset, etc.) one or more techniques or methodologies may be implemented. For example, to provide the collected information to the font service provider 208, each of the user devices 100, 302 may send a corresponding request 310, 312 to the font service provider. In this arrangement, each device is wirelessly connected to the one or more networks (e.g., the Internet) for exchanging information with the font service provider 208, as represented with the respective graphics 314, 316. Upon receiving the requests 310, 312, the font service manager 212 prepares color font information for the devices based upon the information provided in each respective request. For example, dependent upon the capability of the corresponding browser, different types of information may be respectively prepared and sent by the font service provider 208. In the illustrated arrangement, a color font file 318 is prepared and sent to the device 302 executing the web browser 304 that assists the device in supporting color fonts. In general, the information included in the color font file 318 allows the device 302 to express the color font of the web asset as authored. In one arrangement, the color font file 318 may include color font formats that are directly supported by the web browser 304. For example, such a color font format may be implemented in a format that includes data structures capable of supporting a variety of typographic visual symbols of many languages. A format such as a variant of the OpenType format may be capable of defining color fonts along with other types of typographical features for one or more scalable computer fonts. Generally using a file or files of relatively reduced size, such formats may be advantageous.

[029] In some arrangements, the color font file 318 may include other types of color font information. For example, one or more other formats may be implemented for providing color font information. In one arrangement, the asset presenter (e.g., web browser) of the device may be compatible with a family of specifications of an extensible markup language (XML) based

file format for two-dimensional vector graphics such as scalable vector graphics (SVG). Formats such as SVG, inline SVG, SVG Tiny (e.g., for mobile devices), etc. can be used for static imagery (e.g., individual images) and dynamic imagery (e.g., interactive, animated imagery, etc.). Similar to SVG, other implementations may also be employed that use layering techniques (e.g., apply one color to a background layer, apply a second color to a foreground layer, etc.) for providing color font information (e.g., in the color font file 318) to the asset presenter (e.g., web browser 304).

[030] Some web browsers or other types of asset presenters are unable to support color fonts formats such as SVG. In such situations, one or more techniques may be implemented to enable color fonts. For example, one or more executable files may be provided to the asset presenter (e.g., a non-supporting web browser) such that when executed each character present in the web asset is replicated into multiple glyphs such that each may be assigned a specific color. For example, one or more JavaScript files may be provided and when executed, the DOM tree associated with a target web asset is altered at run-time such that each character present in the asset is replicated into multiple glyphs and assigned a corresponding color. Two or more of the replicated glyphs may share a common font, or, two or more different fonts may be used by the replicated glyphs. Assigned colors, the executable files may layer the replicated glyphs such that a multi-color character may be presented in one or more fonts (e.g., web fonts). In some arrangements, the executed file or files (e.g. JavaScript files), a software agent, etc. may insert, adjust, etc. data in the DOM tree.

[031] In this example, recognizing that the web browser 102 of the device 100 does not support color fonts (e.g., from the information provided by the request 312), different color font information may be sent to the user device (compared to the color font file 318 sent to the user device 302 capable of supporting such color fonts). One or more techniques may be implemented to provide color font information to non-supporting web browsers for presenting, for example, multi-color font characters. In the illustrated example, based upon the information included in the request 312, multiple files may be sent from the font service provider 208 to the non-supporting browser 102 of user device 100. In particular, a color font file 320 is delivered from the font service provider 208 that includes some color font information needed to present the color font content of the web asset. In some examples, information is only provided for the

content included in the web asset to be presented. As such, a subset of information (needed to present the asset) may be provided (e.g., font color information for characters present in the web asset), thereby conserving sources such as memory, processing time, throughput (and/or other transmission characteristics), etc.

[032] In addition to the color font file 320, an executable file 322 may be provided by the font service provider 208. Typically, the executable file 322 (e.g., a JavaScript file) includes logic for producing, processing, etc. color font information. For example, the executable file 322 may include logic for adjusting the DOM tree of the target web asset for replicating characters and assigning colors to the corresponding glyphs. Additionally, the executable file may include logic for layering the replicated glyphs for producing multi-color characters, for example. Similar to the color font files 318, 320, the contents of the executable file 322 may be reduced based upon the subset of material included in the web asset. For example, logic may be absent from the executable file 322 if corresponding characters and colors are not included in the content of the web asset. Further, multiple executable files may be sent from the font service provider 208 to the device rather than a single executable file. For example, multiple files may be used for delivering the logic for providing color font information. In some arrangements subsetting color font information may be delivered in an individual file or concatenated from multiple files into a single file. In some arrangements, the content of files may be combined. For example, by including the content (e.g., SVG data) of the color font file 320 and the executable file 322 (e.g., a JavaScript file), a single file (e.g., an executable file) that contains the content needed by an asset presenter (e.g., the non-supporting web browser 102) may be provided. Along with using the information provided from the font service provider 208 for presenting color fonts (e.g., with asset presenters that support or do not support such features), the information (e.g., provided by one or more font files, executable files, etc.) may be used for other applications. For example, this information may be utilized by an editor, executed locally (e.g., by a user device) or remotely (e.g., at the font service provider or other location), for corresponding applications (e.g., creating content, editing content, managing content, etc.). For example, an editor that allows a user to select various types of characters (e.g., to create different types of presentations, assets, applications, etc.) may utilize the color font information. Similar to providing characters for selection, the editor may also provide colors and various color patterns and palettes for content creation. Along with different types of editors, various types of user interfaces may be

implemented. For example, interfaces may be implemented that are directed toward use on a local computer. Similarly, interfaces may be implemented for remotely located computer systems (e.g., cloud-based computer systems and services) and other types of computer and network architectures.

[033] Referring to FIG. 4, a graphical representation 400 is presented that includes font characters that use multiple colors. In particular, along with shapes being used, the multiple colors applied to each character give the appearance that each letter of the word “HELLO” is made of bamboo sticks.

[034] Along with information for presenting multi-color characters, other types of information may be included in the color font information. For example, similar to solid colors being assigned to portions of a character, gradients of one or more colors may be represented in the color font information. By representing such gradients, a variety of effects (e.g., lighting effects, etc.) may be simulated. For example, by applying color gradients to portions of the letters included in the representation 400, the incident angle of a light source may be simulated (and potentially adjusted). Physical texture may also be represented by in color font information. For example, one or more representations of physical textures (e.g., wood, fur, sand, smooth metal, glass, leather, etc.) may be represented in the information (e.g., the color font file 318 provided to an asset presenter capable of supporting color fonts, the color font file 320 and/or the executable file 322 provided to an asset presenter unable to support color fonts). Similar to solid colors, such gradients, textures, etc. may be represented by vector graphics (e.g., SVG, inline SVG, SVG Tiny, etc.) or other techniques that use geometrical primitives for representing images and allow the images to be scaled without significant degrading of clarity. Textures may be represented along one or more axes, for example, to create texture patterns. A graphical element that represents a texture element may be mirrored along an x-axis and a y-axis to produce a two-dimensional representation of the texture. By mirroring the graphical element along a single axis (e.g., an x-axis), a one-dimensional texture path may be produced that extends along the single axis. Through the use of vector graphics, the texture representation may be adjusted (e.g., scaled up, scaled down, offset, bent, etc.) to provide the visual effect of interest (e.g., have a texture path follow a straight line, curved line, etc.; represent a two-dimensional surface with the texture, etc.).

[035] Similar to producing two-dimensional characters, such color information could also be used for producing three-dimensional characters. For example, color font information representing solid colors, color gradients, textures, etc. may be used to define three-dimensional characters based upon the location of one or more virtual light sources. Along with using the geometry of the virtual light source (or sources), other geometrical locations may be used for defining the shading, tone, etc. of three dimensional characters (or combinations of two and three dimensional characters). For example, the perspective of one or more viewers may be defined and used to determine the colors, gradients, textures, etc. to be applied to portions of the characters. For example, colors, gradients and textures may be applied to provide a certain esoteric visual effect at the perspective of the viewer. Along with static views, use of a virtual light source or light sources, one or more viewing perspectives, etc. may be used for dynamically changing characters. For example, characters that incorporate one or more animations (e.g., the crossbar of a letter “T” spins about an axis defined by the vertical bar of the letter), may use colors, gradients, textures, etc. to present the animated characters from a particular viewing perspective while being illuminated by one or more virtual light sources. Similar to virtual light sources, data representing real light sources may be used for determining colors, gradients, textures, etc. to be applied to portions of characters. For example, data from one or more sensors may be used to determine parameters (e.g., incident angle, location, orientation, etc.), properties (e.g., light intensity, polarization, etc.), etc. of one or more real light sources (e.g., the sun, street lights, surface reflections, etc.) to determine appropriate colors, gradients, textures, etc. for applying to characters. Viewer perspective and other viewing characteristics may also be included in defining the presentable characters.

[036] Referring to FIG. 5, information may be provided from the font service provider 208 such that the presented text (e.g., represented in a multi-color font) may take advantage of other capabilities of an asset presenter such as a web browser. For example, along with presenting text (e.g., in a webpage), the web browser may be capable of interacting with the text to allow a user to select a portion of the text (e.g., highlight the text through pointing device operations), copy and paste the text (e.g., into another executed application), search the content of the text (e.g., for user-provided search terms), etc. In some instances, such as when the web browser is not capable of supporting color fonts, additional information may be provided by the font service provider 208 to allow for such interactions along with assisting with color font support to present

the text. In one example technique, additional data being provided may represent spatial information associated with the font glyphs but not include any additional data for visual information (e.g., character outlines or fill). As such, glyph spacing information may be provided but glyph marking information may be absent. Regardless of whether color fonts are supported or not, the web browser is able to recognize and use this spatial information. For example, the spatial information may assist the web browser with positioning a cursor between the text characters. Made aware of the spatial boundaries of the character, the web browser can allow the text to be searched, portions selected, copied (for pasting in other applications), etc. regardless of whether color fonts are supported by the browser.

[037] One or more techniques may be employed to provide such an empty or invisible font that provides spacing information but is absent additional visual information. For example, such invisible font information may be included in one or more files provided by the font service provider 208. With reference to FIG. 3, along with providing the color font file 320 and the executable file 322 (for use with the web browser 102) the font service provider 208 may also provide a file that includes such invisible font information. For the situation in which the web browser supports color fonts (e.g., web browser 304), a similar file that includes invisible font information may be sent along with a file with color font information (e.g., color font file 318) from the font service provider 208. Other combinations of files may be employed for providing the invisible font information. For example, rather than being sent in a separate file, the invisible font information may be combined with the color font information and sent in a single file (such as the color font file 320). The invisible font information may also be sent from other sources. For example, such invisible font information may be included in the web asset file(s) (e.g., included in the HTML file defining a webpage). In an HTML file, a node may be defined for each word (to be displayed) for instructing that the word be presented in such an invisible font.

[038] Once provided the color font information (e.g., SVG data in the color font file 320) and the invisible font information (e.g., in a separate file from the font service provider 208), the web browser can overprint the invisible font with a color font. As such, the web browser can be assisted in presenting a color font and the web browser is also capable of allowing text to be searched, selected, copied (for pasting into another application), etc. by using the invisible font data. As illustrated in the figure, two characters 500, 502 are presented that employ a font that

uses multiple colors (represented in shades of gray) for each character. Additionally, an invisible font for each character (represented by hash-lined boxes 504, 506) provides spatial information such as for positioning a cursor on either side or between the characters.

[039] To assist with allowing the spatial information provided by the invisible font to be utilized, the layering position of the color font and the invisible font may be adjusted. For example, the invisible font may be initially presented and overprinted by the color font. To assist with the selection of the characters, the layer position of the invisible font and the color font may be reversed such that the invisible font lies over the color font. Once the positions have changed, selection of one or more characters may be easier for the user (e.g., operating a pointing device). To initiate the reversing of the positions, one or more techniques may be implemented. For example, based upon a pointing device activity being detected (e.g., the user left-clicks on a mouse), the positions of the invisible font and the color font may reverse. Once reversed, the user can then select the desired character(s) for other operations (e.g., copying, cutting, pasting, etc.).

[040] Referring to FIG. 6a a flowchart 600 represents operations of a font service manager (e.g., the font service manager 212 shown in FIG. 2). Operations of the font service manager are typically executed by a single computing device (e.g., the server 214 also shown in FIG. 3); however, operations of the font service manager may be executed by multiple computing devices. Along with being executed at a single site (e.g., the font service provider 208 shown in FIG. 3), operation execution may be distributed among two or more locations.

[041] Operations of the font service manager may include receiving 602 receiving information that requests one or more colored fonts of a web asset for an asset presenter (e.g., a web browser) being executed by a computing device. In some arrangements the received information may include information about the computing device (e.g., indicate whether the asset presenter being executed by the computing device is capable of presenting the one or more colored fonts of the web asset. For example, a request may be sent from the computing device to the font service provider (where the font service manager is executed) that contains information indicative of whether a web browser executed by the device supports color fonts (e.g., information that identifies web browser, the computing device, etc.). In some arrangements, the request may be

absent information indicative of whether the asset presenter is capable (or not) of supporting colored fonts. Operations may also include, in response to receiving the information, sending 604 color font information to the computing device to allow content (e.g., characters, etc.) of the web asset represented in colored fonts to be presented by the computing device. The color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file. For example, a color font file may be sent by the font service provider that includes inline SVG information along with an executable file (e.g., a JavaScript file) for producing color font characters. In some arrangements, the contents of these two files may be combined. For example, an executable file (e.g., a JavaScript file) may be provided that also includes the content of the color font file (e.g., inline SVG information) so that the color font characters may be produced for presentation from a single file. Other file combinations and file content combinations may be realized. For example, subsets of color font information may be produced and packaged prior to sending the information to the computing device.

[042] Referring to FIG. 6b a flowchart 606 represents other potential operations of a font service manager (e.g., the font service manager 212 shown in FIG. 2). As mentioned above, operations of the font service manager are typically executed by a single computing device (e.g., the server 214 also shown in FIG. 3); however, operations of the font service manager may be executed by multiple computing devices. Along with being executed at a single site (e.g., the font service provider 208 shown in FIG. 3), operation execution may be distributed among two or more locations.

[043] Operations of the font service manager may include receiving 608 information that indicates whether an asset presenter (e.g., a web browser) being executed by a computing device is capable of presenting one or more colored fonts. For example, a request may be sent from the computing device to the font service provider (where the font service manager is executed) that contains information indicative of whether a web browser executed by the device supports color fonts (e.g., information that identifies web browser, the computing device, etc.). Operations may also include, in response to receiving the information, sending 610 color font information to the computing device to allow content (e.g., characters, etc.) of the web asset represented in colored fonts to be presented by the computing device. For example, a color font file may be sent by the

font service provider that includes inline SVG information, an executable file (e.g., a JavaScript file) and/or other information for producing color font characters. Other operations may also subset the color font information prior to sending the information to the computing device.

[044] FIG. 7 is a block diagram showing an example of a system 700 for providing hosted storage and accessing the hosted storage from a client device 702. In some implementations, a hosted storage service 720 may provide access to stored data (e.g., font information) by applications (e.g., web browsers) running on computing devices operating separately from one another, provide offsite data backup and restore functionality, provide data storage to a computing device with limited storage capabilities, and/or provide storage functionality not implemented on a computing device.

[045] The system 700 may provide scalable stores for storing data resources. The client device 702 may upload data resources to the hosted storage service 720 and control access to the uploaded data resources. Access control may include a range of sharing levels (e.g., private, shared with one or more individuals, shared with one or more groups, public, etc.). Data stored in hosted storage service 720 can be secured from unauthorized access. The hosted storage service 720 can use a simple and consistent application programming interface, or API, which can allow arbitrary quantities of structured or unstructured data to be kept private or shared between individuals, organizations, or with the world at large. The client device 702 may access, retrieve, be provided, store, etc. data in the hosted storage service 720 for any number of a variety of reasons. For example, data may be stored for business reasons (e.g., provide identification information to attain access clearance for font data at the hosted storage service 720), or for use in data processing by other services.

[046] The client device 702 may be implemented using a computing device, such as the computing device 800 or the mobile device 850 described with respect to FIG. 8. The client device 702 may communicate with the hosted storage service 720 via a network 704, such as the Internet. The client device 702 may communicate across the network using communication protocols such as, for example, one or more of Transmission Control Protocol/Internet Protocol (TCP/IP), Hypertext Transfer Protocol (HTTP), Secure Shell Remote Protocol (SSH), or Application Program Interfaces (API). While only a single client device 702 is shown, there may

be multiple client devices communicating across the network 704 with the hosted storage service 720 and/or other services and devices.

[047] The hosted storage service 720 may be implemented such that client applications executing on client device 702, such as a client application 703, may store, retrieve, or otherwise manipulate data resources in the hosted storage service 720. The hosted storage service 720 may be implemented by one or more server devices, which may be implemented using a computing device, such as the computing device 800 or mobile device 850 described with respect to FIG. 8. For example, the hosted storage service 720 may be implemented by multiple server devices operating in the same, or different, data centers.

[048] The hosted storage service 720 generally includes an interface frontend 706, an interface backend 708, a storage backend 710, and metadata 716 for resources stored in the storage backend 710. The hosted storage service 720 may also include an authenticator 709 to verify that a user requesting one or more fonts should be provided access to the fonts (e.g., based on a service subscription, rental period, etc.).

[049] In general, the interface frontend 706 may receive requests from and send responses to the client device 702. For instance, the hosted storage service 720 may be implemented as a Web Service with a corresponding set of Web Service Application Programming Interfaces (APIs). The Web Service APIs may be implemented, for example, as a Representational State Transfer (REST)-based HTTP interface or a Simple Object Access Protocol (SOAP)-based interface. Interface frontend 706 may receive messages from the client 702 and parse the requests into a format usable by the hosted storage service 720, such as a remote procedure call (RPC) to an interface backend 708. The interface frontend 706 may write responses generated by the hosted storage service 720 for transmission to the client 702. In some implementations, multiple interface frontends 706 may be implemented, for example to support multiple access protocols.

[050] The interface frontend 706 may include a graphical front end, for example to display on a web browser for data access. The interface frontend 706 may include a sub-system to enable managed uploads and downloads of large files (e.g., for functionality such as pause, resume, and recover from time-out). The interface frontend 706 may monitor load information and update logs, for example to track and protect against denial of service (DOS) attacks.

[051] As described above, the Web Service API may be a REST-based HTTP interface. In a REST-based interface, a data resource is accessed as a resource, uniquely named using a uniform resource identifier (URI), and the client application 703 and service 720 exchange representations of resource state using a defined set of operations. For example, requested actions may be represented as verbs, such as by HTTP GET, PUT, POST, HEAD, and DELETE verbs. The GET verb may be used to retrieve a resource, while the HEAD verb may be used to retrieve information about a resource without retrieving the resource itself. The DELETE verb may be used to delete a resource from the hosted storage service 720. The PUT and POST verbs may be used to upload a resource to the service 720. PUT requests may come from the client 702 and contain authentication and authorization credentials and resource metadata in a header, such as an HTTP header. POST requests may be received when a client 702 wants to upload from a web browser form. The form POST upload protocol for the hosted storage service 720 may involve multiple form fields to provide authentication, authorization, and resource metadata. More generally, any of the API requests may include credentials for authentication and authorization, for example in a header of the request. An authorization header may be included in the REST requests, which may include an access key to identify the entity sending the request.

[052] Alternatively, or additionally, a user may be authenticated based on credentials stored in a browser cookie, which may be appended to the API requests. If no valid cookie is present, a redirect to an authentication frontend may be generated, and the authentication frontend may be used to generate the browser cookie. The authentication frontend may be used by systems and services in addition to the hosted storage service 720 (e.g., if the organization operating the hosted storage service 720 also operates other web services such as email service). A user may also or alternatively be authenticated based on authentication credentials from an external credentialing service or an external service that includes credentialing functionality. User or group identifier information may be calculated from the external service's credential information. Requests sent by the client 702 to the interface frontend 706 may be translated and forwarded to the external service for authentication.

[053] In general, resources stored in the hosted storage service 720 may be referenced by resource identifiers. The hosted storage service 720 may define namespaces to which a valid resource identifier must conform. For example, the namespace may require that resource

identifiers be a sequence of Unicode characters whose UTF-8 encoding is at most 1024 bytes long. As another example, the namespace may require that resource identifiers be globally unique identifiers (GUIDs), which may be 128-bit integers.

[054] Resources (e.g., objects such as font data) may be stored in hosted storage service 720 in buckets. In some examples, each bucket is uniquely named in the hosted storage service 720, each data resource is uniquely named in a bucket, and every bucket and data resource combination is unique. Data resources may be uniquely identified by a URI that includes the bucket name and the resource name, and identifies the hosted storage service 720.

[055] The interface backend 708 along with the authenticator 709 may handle request authentication and authorization, may manage data and metadata, and may track activity such as for billing. As one example, the interface backend 608 may query the authenticator 709 when a request for one or more fonts is received. The interface backend 708 may also provide additional or alternative functionality. For example, the interface backend 708 may provide functionality for independent frontend / backend scaling for resource utilization and responsiveness under localized heavy loads. Data management may be encapsulated in the interface backend 708 while communication serving may be encapsulated in the interface frontend 706. The interface backend 708 may isolate certain security mechanisms from the client-facing interface frontend 706.

[056] The interface backend 708 may expose an interface usable by both the interface frontend 706 and other systems. In some examples, some features of the interface backend 708 are accessible only by an interface frontend (not shown) used by the owners of the hosted storage service 720 (internal users). Such features may include those needed for administrative tasks (e.g., resolving a resource reference to a low level disk address). The interface backend 708 may handle request authentication (e.g., ensuring a user's credentials are valid) and authorization (e.g., verifying that a requested operation is permitted). The interface backend may also provide encryption and decryption services to prevent unauthorized access to data, even by internal users.

[057] The interface backend 708 may manage metadata 716 associated with data resources, for example in a MySQL database or BigTable. User-specified names labeling the buckets can be completely defined within the metadata 716, and resource metadata 716 can map a resource

name to one or more datastores 712 storing the resource. The metadata 716 can also contain bucket and resource creation times, resource sizes, hashes, and access control lists 718 (ACL 718) for both buckets and resources. The interface backend 708 can log activity and track storage consumption to support accounting for billing and chargebacks. In some examples, this includes quota monitoring in each dimension in which customers are charged (e.g., reads, writes, network transfers, total storage in use).

[058] The ACLs 718 may generally define who is authorized to perform actions on corresponding buckets or resources, and the nature of the permitted actions. The ACLs 718 may be an unordered list of {scope, role} pairs, plus Boolean flags. The scope may define a user or group of users and the role may define the access permissions for the user or group. In some examples, the union of all {scope, role} pairs may define access rights. In some examples, more specific {scope, role} pairs override more general ones.

[059] The storage backend 710 may contain multiple datastores 712a-712c. Although three datastores 712 are shown, more or fewer are possible. Each of the datastores 712a-712c may store data resources 714a-714c in a particular format. For example, data store 712a may store a data resource 714a as a Binary Large Object (BLOB), data store 712b may store a data resource 714b in a distributed file system (e.g., Network File System), and data store 712c may store a data resource 714c in a database (e.g., MySQL).

[060] FIG. 8 shows an example of example computer device 800 and example mobile computer device 850, which can be used to implement the techniques described herein. For example, a portion or all of the operations of the font service manager 212 (shown in FIG. 2) may be executed by the computer device 800 and/or the mobile computer device 850. Computing device 800 is intended to represent various forms of digital computers, including, e.g., laptops, desktops, workstations, personal digital assistants, servers, blade servers, mainframes, and other appropriate computers. Computing device 850 is intended to represent various forms of mobile devices, including, e.g., personal digital assistants, tablet computing devices, cellular telephones, smartphones, and other similar computing devices. The components shown here, their connections and relationships, and their functions, are meant to be examples only, and are not meant to limit implementations of the techniques described and/or claimed in this document.

[061] Computing device 800 includes processor 802, memory 804, storage device 806, high-speed interface 808 connecting to memory 804 and high-speed expansion ports 810, and low speed interface 812 connecting to low speed bus 814 and storage device 806. Each of components 802, 804, 806, 808, 810, and 812, are interconnected using various busses, and can be mounted on a common motherboard or in other manners as appropriate. Processor 802 can process instructions for execution within computing device 800, including instructions stored in memory 804 or on storage device 806 to display graphical data for a GUI on an external input/output device, including, e.g., display 816 coupled to high speed interface 808. In other implementations, multiple processors and/or multiple buses can be used, as appropriate, along with multiple memories and types of memory. Also, multiple computing devices 800 can be connected, with each device providing portions of the necessary operations (e.g., as a server bank, a group of blade servers, or a multi-processor system).

[062] Memory 804 stores data within computing device 800. In one implementation, memory 804 is a volatile memory unit or units. In another implementation, memory 804 is a non-volatile memory unit or units. Memory 804 also can be another form of computer-readable medium, including, e.g., a magnetic or optical disk.

[063] Storage device 806 is capable of providing mass storage for computing device 800. In one implementation, storage device 806 can be or contain a computer-readable medium, including, e.g., a floppy disk device, a hard disk device, an optical disk device, or a tape device, a flash memory or other similar solid state memory device, or an array of devices, including devices in a storage area network or other configurations. A computer program product can be tangibly embodied in a data carrier. The computer program product also can contain instructions that, when executed, perform one or more methods, including, e.g., those described above. The data carrier is a computer- or machine-readable medium, including, e.g., memory 804, storage device 806, memory on processor 802, and the like.

[064] High-speed controller 808 manages bandwidth-intensive operations for computing device 800, while low speed controller 812 manages lower bandwidth-intensive operations. Such allocation of functions is an example only. In one implementation, high-speed controller 808 is coupled to memory 804, display 816 (e.g., through a graphics processor or accelerator), and to

high-speed expansion ports 810, which can accept various expansion cards (not shown). In the implementation, low-speed controller 812 is coupled to storage device 806 and low-speed expansion port 814. The low-speed expansion port, which can include various communication ports (e.g., USB, Bluetooth®, Ethernet, wireless Ethernet), can be coupled to one or more input/output devices, including, e.g., a keyboard, a pointing device, a scanner, or a networking device including, e.g., a switch or router, e.g., through a network adapter.

[065] Computing device 800 can be implemented in a number of different forms, as shown in the figure. For example, it can be implemented as standard server 820, or multiple times in a group of such servers. It also can be implemented as part of rack server system 824. In addition or as an alternative, it can be implemented in a personal computer including, e.g., laptop computer 822. In some examples, components from computing device 800 can be combined with other components in a mobile device (not shown), including, e.g., device 850. Each of such devices can contain one or more of computing device 800, 850, and an entire system can be made up of multiple computing devices 800, 850 communicating with each other.

[066] Computing device 850 includes processor 852, memory 864, an input/output device including, e.g., display 854, communication interface 866, and transceiver 868, among other components. Device 850 also can be provided with a storage device, including, e.g., a microdrive or other device, to provide additional storage. Each of components 850, 852, 864, 854, 866, and 868, are interconnected using various buses, and several of the components can be mounted on a common motherboard or in other manners as appropriate.

[067] Processor 852 can execute instructions within computing device 850, including instructions stored in memory 864. The processor can be implemented as a chipset of chips that include separate and multiple analog and digital processors. The processor can provide, for example, for coordination of the other components of device 850, including, e.g., control of user interfaces, applications run by device 850, and wireless communication by device 850.

[068] Processor 852 can communicate with a user through control interface 858 and display interface 856 coupled to display 854. Display 854 can be, for example, a TFT LCD (Thin-Film-Transistor Liquid Crystal Display) or an OLED (Organic Light Emitting Diode) display, or other appropriate display technology. Display interface 856 can comprise appropriate circuitry for

driving display 854 to present graphical and other data to a user. Control interface 858 can receive commands from a user and convert them for submission to processor 852. In addition, external interface 862 can communicate with processor 842, so as to enable near area communication of device 850 with other devices. External interface 862 can provide, for example, for wired communication in some implementations, or for wireless communication in other implementations, and multiple interfaces also can be used.

[069] Memory 864 stores data within computing device 850. Memory 864 can be implemented as one or more of a computer-readable medium or media, a volatile memory unit or units, or a non-volatile memory unit or units. Expansion memory 874 also can be provided and connected to device 850 through expansion interface 872, which can include, for example, a SIMM (Single In Line Memory Module) card interface. Such expansion memory 874 can provide extra storage space for device 850, or also can store applications or other data for device 850. Specifically, expansion memory 874 can include instructions to carry out or supplement the processes described above, and can include secure data also. Thus, for example, expansion memory 874 can be provided as a security module for device 850, and can be programmed with instructions that permit secure use of device 850. In addition, secure applications can be provided through the SIMM cards, along with additional data, including, e.g., placing identifying data on the SIMM card in a non-hackable manner.

[070] The memory can include, for example, flash memory and/or NVRAM memory, as discussed below. In one implementation, a computer program product is tangibly embodied in a data carrier. The computer program product contains instructions that, when executed, perform one or more methods, including, e.g., those described above. The data carrier is a computer- or machine-readable medium, including, e.g., memory 864, expansion memory 874, and/or memory on processor 852, which can be received, for example, over transceiver 868 or external interface 862.

[071] Device 850 can communicate wirelessly through communication interface 866, which can include digital signal processing circuitry where necessary. Communication interface 866 can provide for communications under various modes or protocols, including, e.g., GSM voice calls, SMS, EMS, or MMS messaging, CDMA, TDMA, PDC, WCDMA, CDMA2000, or GPRS,

among others. Such communication can occur, for example, through radio-frequency transceiver 868. In addition, short-range communication can occur, including, e.g., using a Bluetooth®, WiFi, or other such transceiver (not shown). In addition, GPS (Global Positioning System) receiver module 870 can provide additional navigation- and location-related wireless data to device 850, which can be used as appropriate by applications running on device 850. Sensors and modules such as cameras, microphones, compasses, accelerators (for orientation sensing), etc. maybe included in the device.

[072] Device 850 also can communicate audibly using audio codec 860, which can receive spoken data from a user and convert it to usable digital data. Audio codec 860 can likewise generate audible sound for a user, including, e.g., through a speaker, e.g., in a handset of device 850. Such sound can include sound from voice telephone calls, can include recorded sound (e.g., voice messages, music files, and the like) and also can include sound generated by applications operating on device 850.

[073] Computing device 850 can be implemented in a number of different forms, as shown in the figure. For example, it can be implemented as cellular telephone 880. It also can be implemented as part of smartphone 882, personal digital assistant, or other similar mobile device.

[074] Various implementations of the systems and techniques described here can be realized in digital electronic circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These various implementations can include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which can be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device.

[075] These computer programs (also known as programs, software, software applications or code) include machine instructions for a programmable processor, and can be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the terms machine-readable medium and computer-readable medium refer to a computer program product, apparatus and/or device (e.g., magnetic discs, optical disks,

memory, Programmable Logic Devices (PLDs)) used to provide machine instructions and/or data to a programmable processor, including a machine-readable medium that receives machine instructions.

[076] To provide for interaction with a user, the systems and techniques described here can be implemented on a computer having a display device (e.g., a CRT (cathode ray tube) or LCD (liquid crystal display) monitor) for displaying data to the user and a keyboard and a pointing device (e.g., a mouse or a trackball) by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be a form of sensory feedback (e.g., visual feedback, auditory feedback, or tactile feedback); and input from the user can be received in a form, including acoustic, speech, or tactile input.

[077] The systems and techniques described here can be implemented in a computing system that includes a back end component (e.g., as a data server), or that includes a middleware component (e.g., an application server), or that includes a front end component (e.g., a client computer having a user interface or a Web browser through which a user can interact with an implementation of the systems and techniques described here), or a combination of such back end, middleware, or front end components. The components of the system can be interconnected by a form or medium of digital data communication (e.g., a communication network). Examples of communication networks include a local area network (LAN), a wide area network (WAN), and the Internet.

[078] The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[079] In some implementations, the engines described herein can be separated, combined or incorporated into a single or combined engine. The engines depicted in the figures are not intended to limit the systems described here to the software architectures shown in the figures.

[080] A number of embodiments have been described. Nevertheless, it will be understood that various modifications can be made without departing from the spirit and scope of the processes and techniques described herein. In addition, the logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other steps can be provided, or steps can be eliminated, from the described flows, and other components can be added to, or removed from, the described systems. Accordingly, other embodiments are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A computer-implemented method comprising:
receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device; and
in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device, wherein the color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.
2. The computer-implemented method of claim 1, wherein the executable file, when executed by the computing device, adjusts a document object model.
3. The computer-implemented method of claim 2, wherein adjusting the document object model includes replicating glyphs of the web asset in a node-based structure representation of the document object model.
4. The computer-implemented method of claim 3, wherein adjusting the document object model includes assigning one or more colors to the replicated glyphs.
5. The computer-implemented method of claim 1, wherein the color font file includes a font subset for presenting the content of the web asset.
6. The computer-implemented method of claim 1, wherein the color font file includes scalable vector graphics.
7. The computer-implemented method of claim 1, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.
8. The computer-implemented method of claim 7, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.

9. A system comprising:
a computing device comprising:
a memory configured to store instructions; and
a processor to execute the instructions to perform operations comprising:
receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device; and
in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device, wherein the color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.
10. The system of claim 9, wherein the executable file, when executed by the computing device, adjusts a document object model.
11. The system of claim 10, wherein adjusting the document object model includes replicating glyphs of the web asset in a node-based structure representation of the document object model.
12. The system of claim 11, wherein adjusting the document object model includes assigning one or more colors to the replicated glyphs.
13. The system of claim 9, wherein the color font file includes a font subset for presenting the content of the web asset.
14. The system of claim 9, wherein the color font file includes scalable vector graphics.
15. The system of claim 9, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.

16. The system of claim 15, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.

17. One or more computer readable media storing instructions that are executable by a processing device, and upon such execution cause the processing device to perform operations comprising:

receiving information that requests one or more colored fonts of a web asset for an asset presenter being executed by a computing device; and

in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device, wherein the color font information is provided by an executable file and a color font file, or, an executable file that includes content of the color font file.

18. The computer readable media of claim 17, wherein the executable file, when executed by the computing device, adjusts a document object model.

19. The computer readable media of claim 18, wherein adjusting the document object model includes replicating glyphs of the web asset in a node-based structure representation of the document object model.

20. The computer readable media of claim 18, wherein adjusting the document object model includes assigning one or more colors to the replicated glyphs.

21. The computer readable media of claim 17, wherein the color font file includes a font subset for presenting the content of the web asset.

22. The computer readable media of claim 17, wherein the color font file includes scalable vector graphics.

23. The computer readable media of claim 17, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.
24. The computer readable media of claim 23, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.
25. A computer-implemented method comprising:
receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset; and
in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.
26. The computer-implemented method of claim 25, wherein sending the font information includes sending an executable file to the computing device.
27. The computer-implemented method of claim 25, wherein sending the font information includes sending a color font file to the computing device.
28. The computer-implemented method of claim 27, wherein the color font file includes scalable vector graphics.
29. The computer-implemented method of claim 25, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.
30. The computer-implemented method of claim 29, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.

31. A system comprising:
a computing device comprising:
a memory configured to store instructions; and
a processor to execute the instructions to perform operations comprising:
receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset; and
in response to receiving the information, sending color font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.
32. The system of claim 31, wherein sending the font information includes sending an executable file to the computing device.
33. The system of claim 31, wherein sending the font information includes sending a color font file to the computing device.
34. The system of claim 33, wherein the color font file includes scalable vector graphics.
35. The system of claim 31, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.
36. The system of claim 35, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.
37. One or more computer readable media storing instructions that are executable by a processing device, and upon such execution cause the processing device to perform operations comprising:
receiving information that indicates whether an asset presenter being executed by a computing device is capable of presenting one or more colored fonts of a web asset; and

in response to receiving the information, sending font information to the computing device to allow content of the web asset represented in colored fonts to be presented by the computing device.

38. The computer readable media of claim 37, wherein sending the font information includes sending an executable file to the computing device.

39. The computer readable media of claim 37, wherein sending the font information includes sending a color font file to the computing device.

40. The computer readable media of claim 39, wherein the color font file includes scalable vector graphics.

41. The computer readable media of claim 37, wherein sending the color font information includes sending a file to the computing device that includes glyph spacing information and is absent glyph marking information.

42. The computer readable media of claim 41, wherein positioning of the glyph spacing information as presented by the computing device is adjustable based upon user interaction.

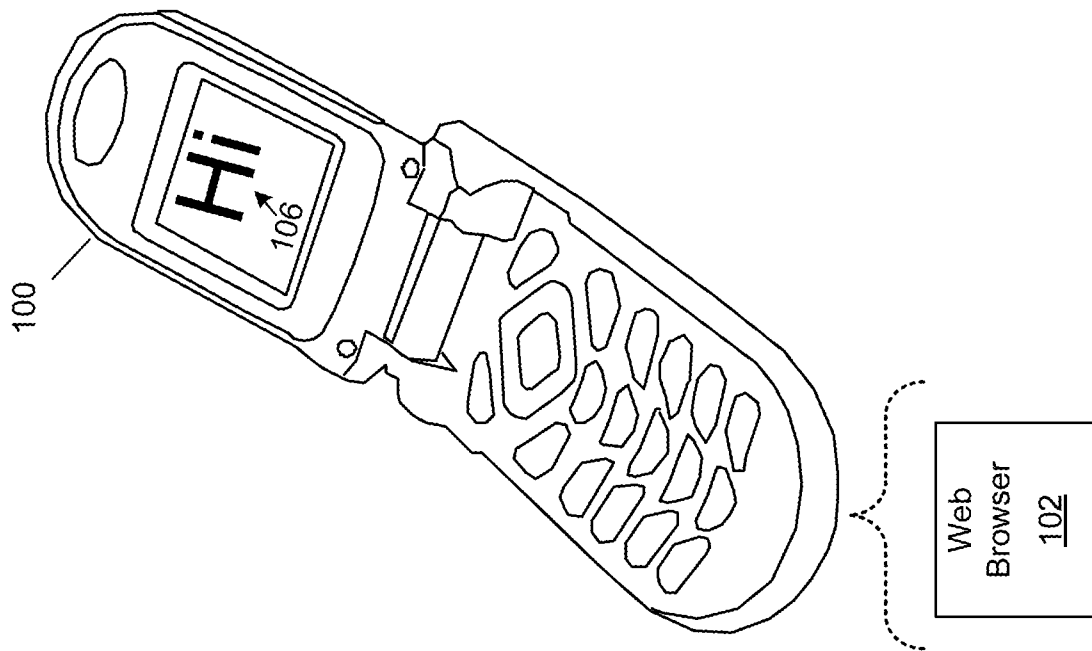


FIG. 1

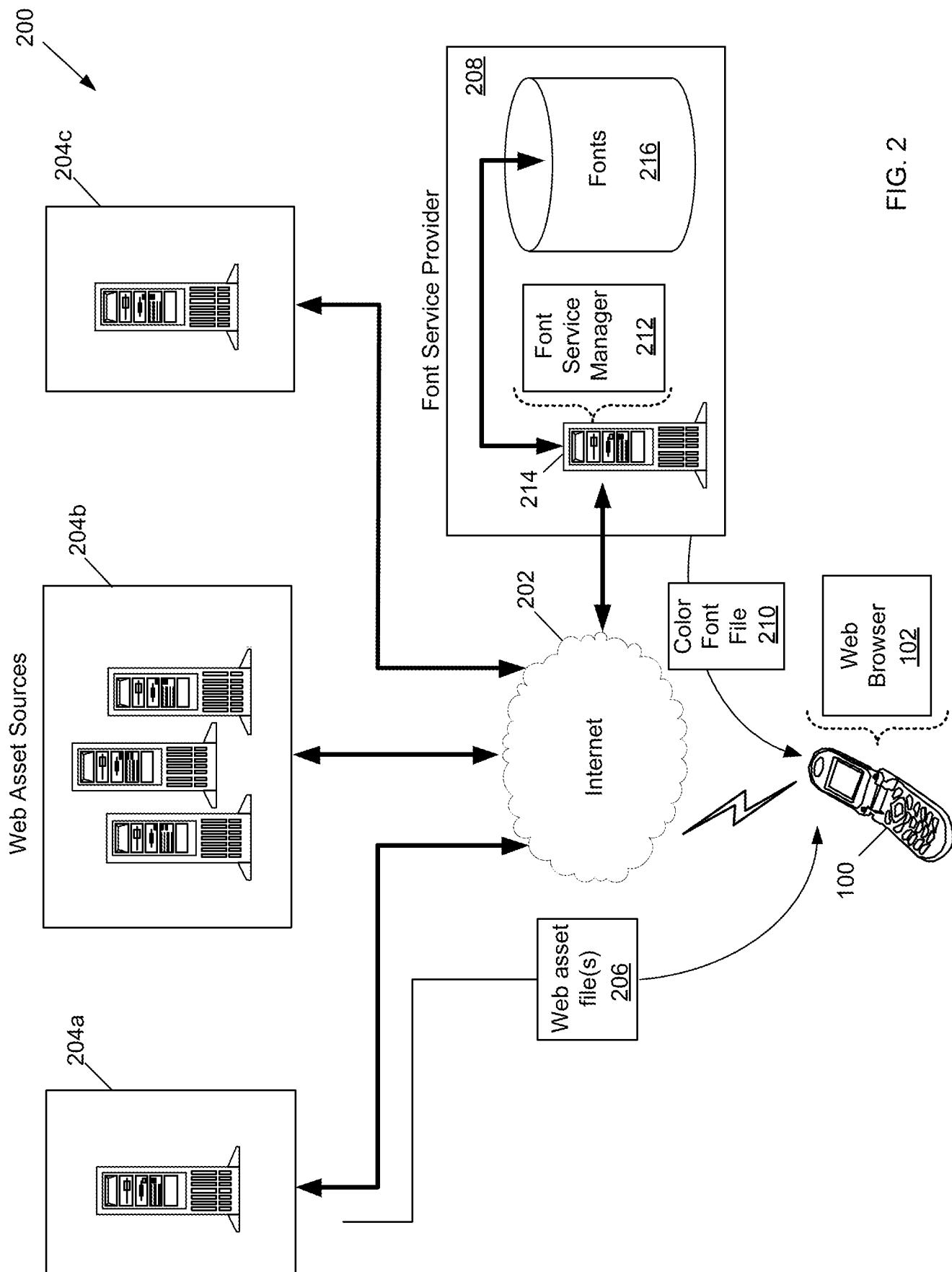


FIG. 2

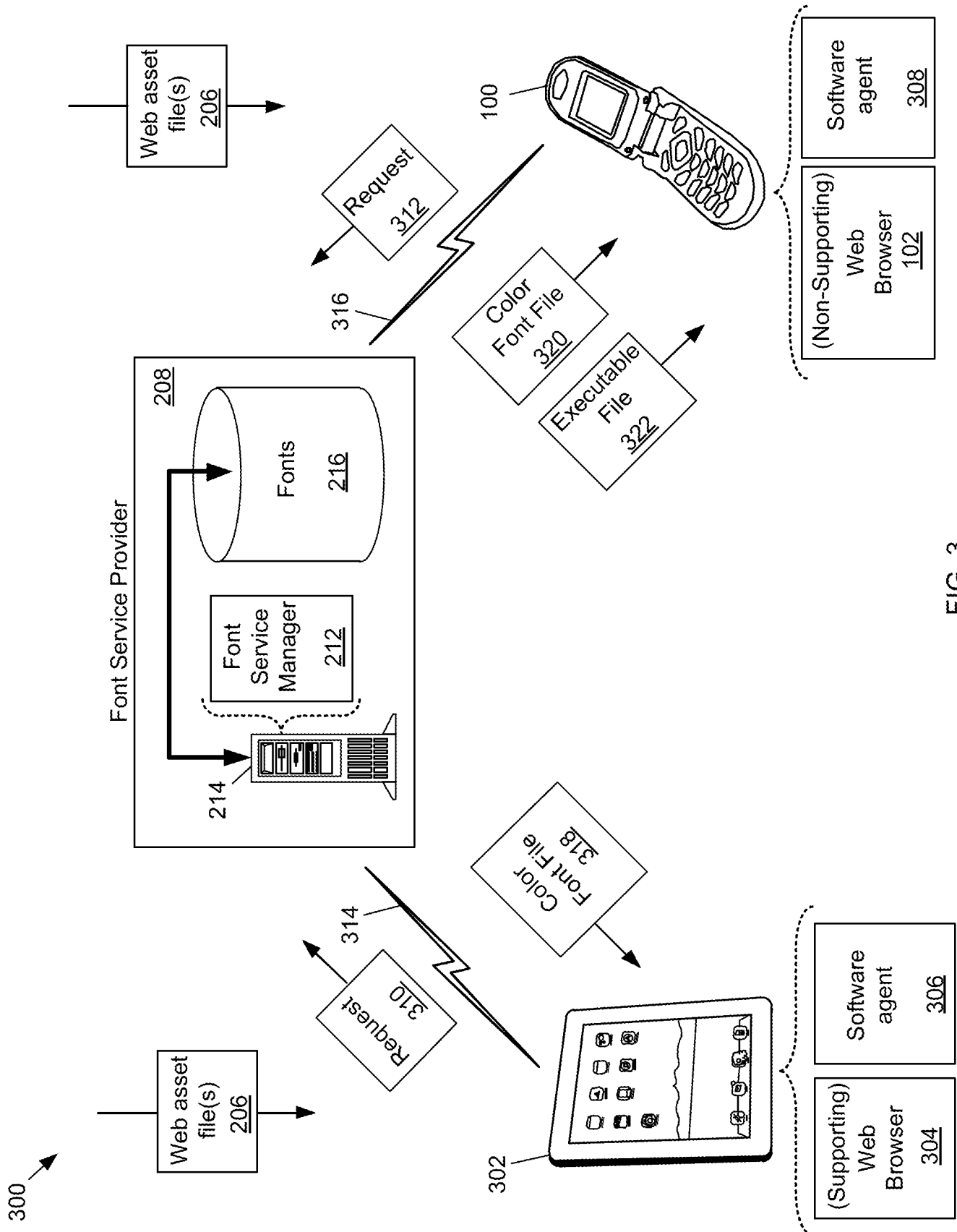


FIG. 3

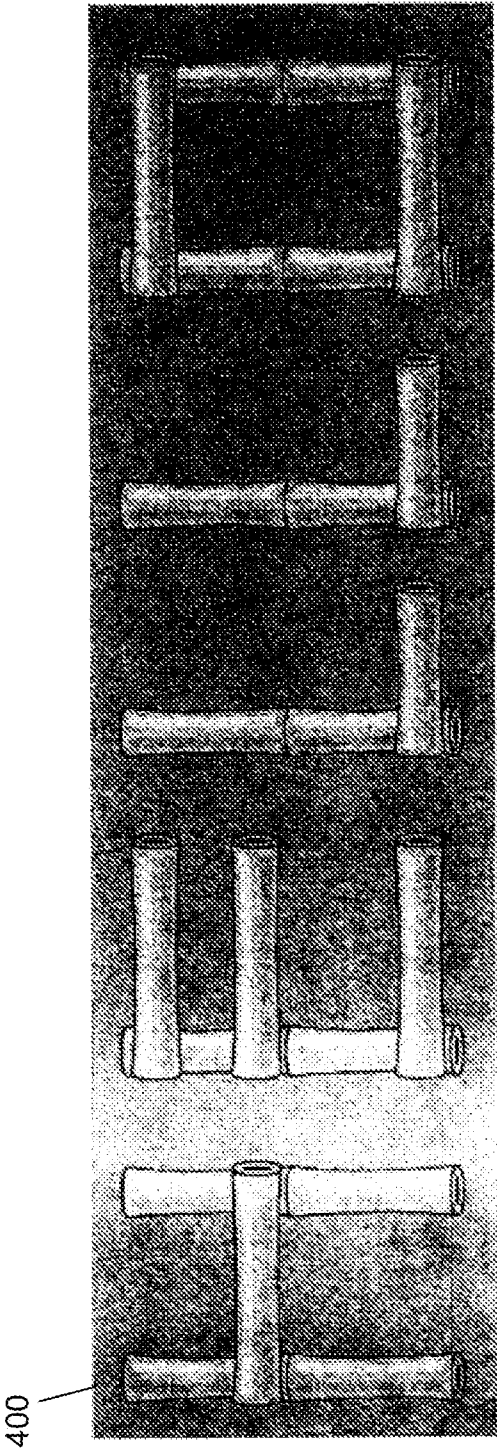


FIG. 4

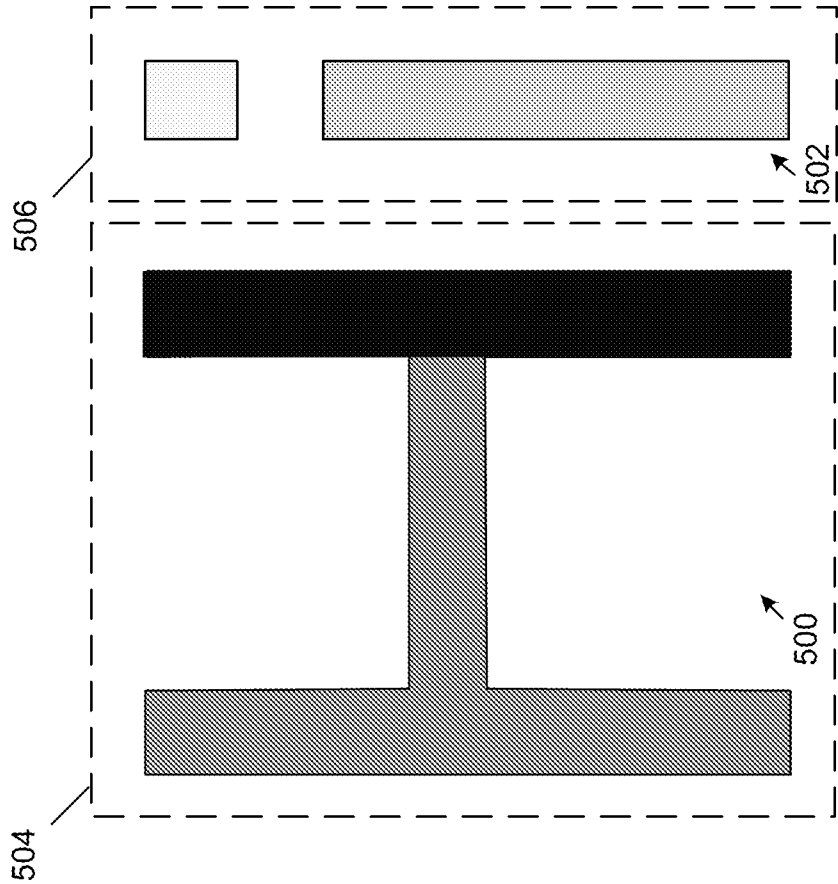


FIG. 5

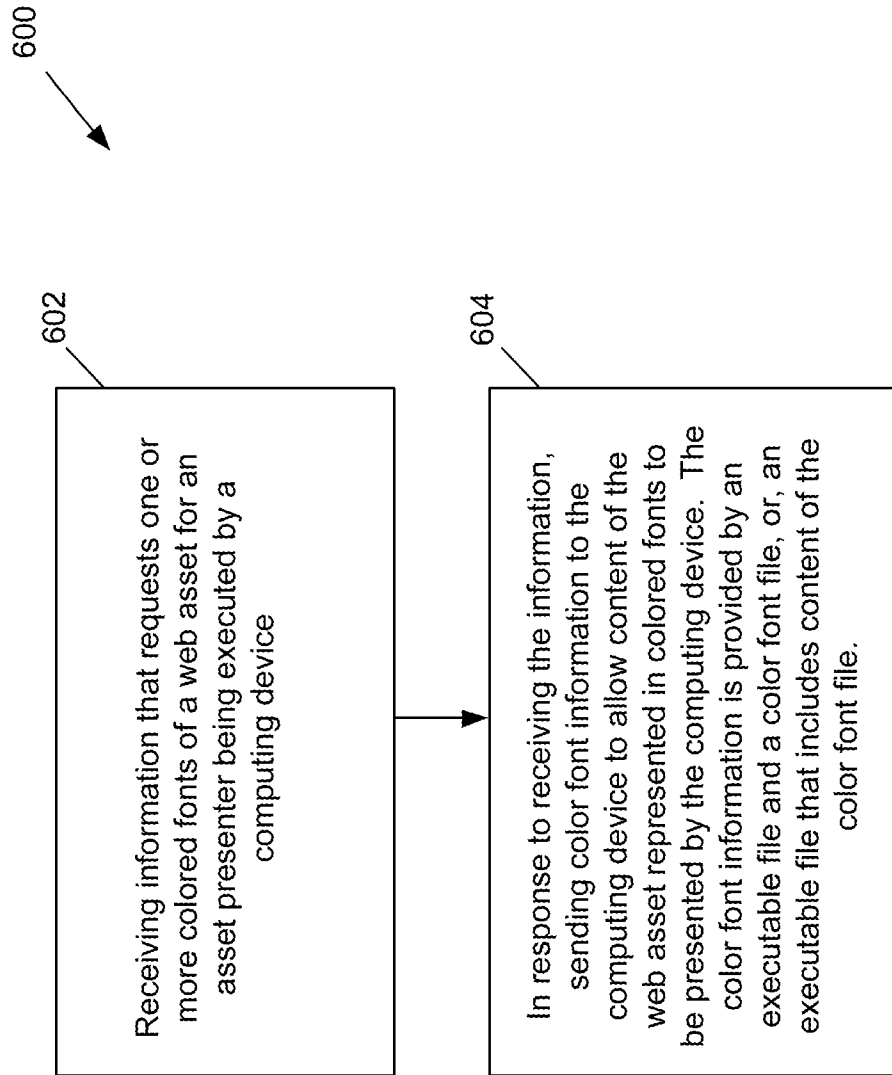


FIG. 6a

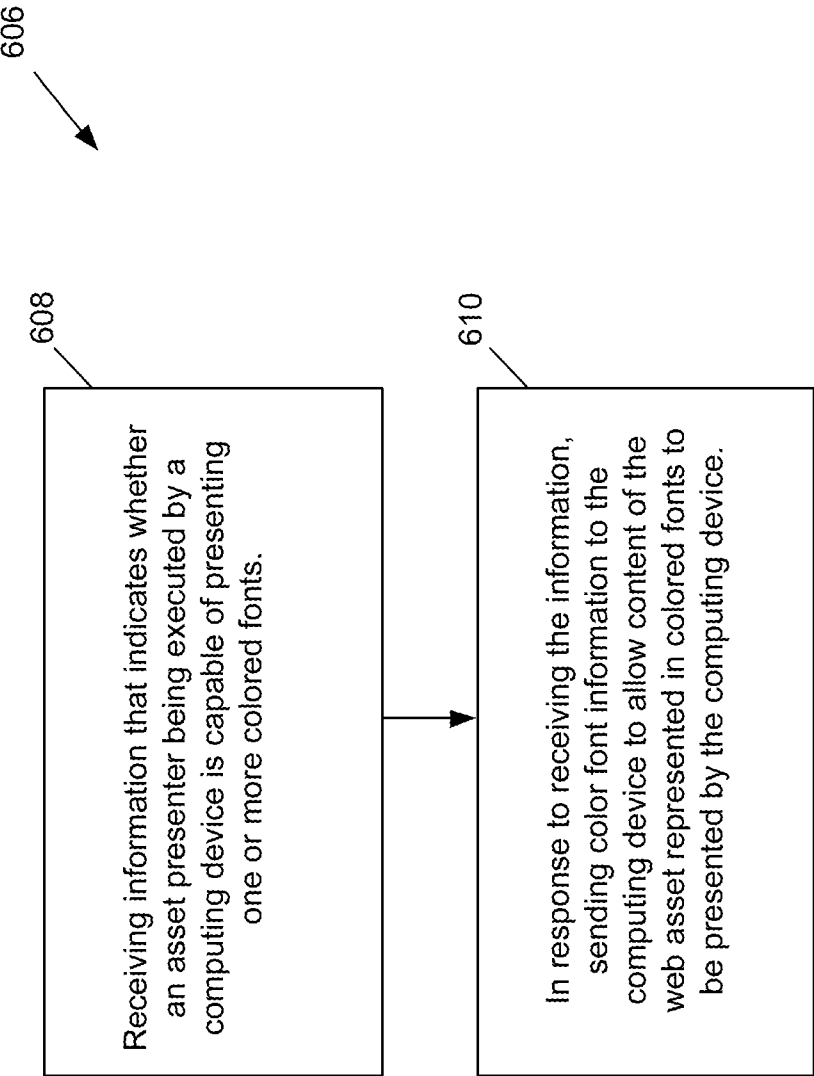


FIG. 6b

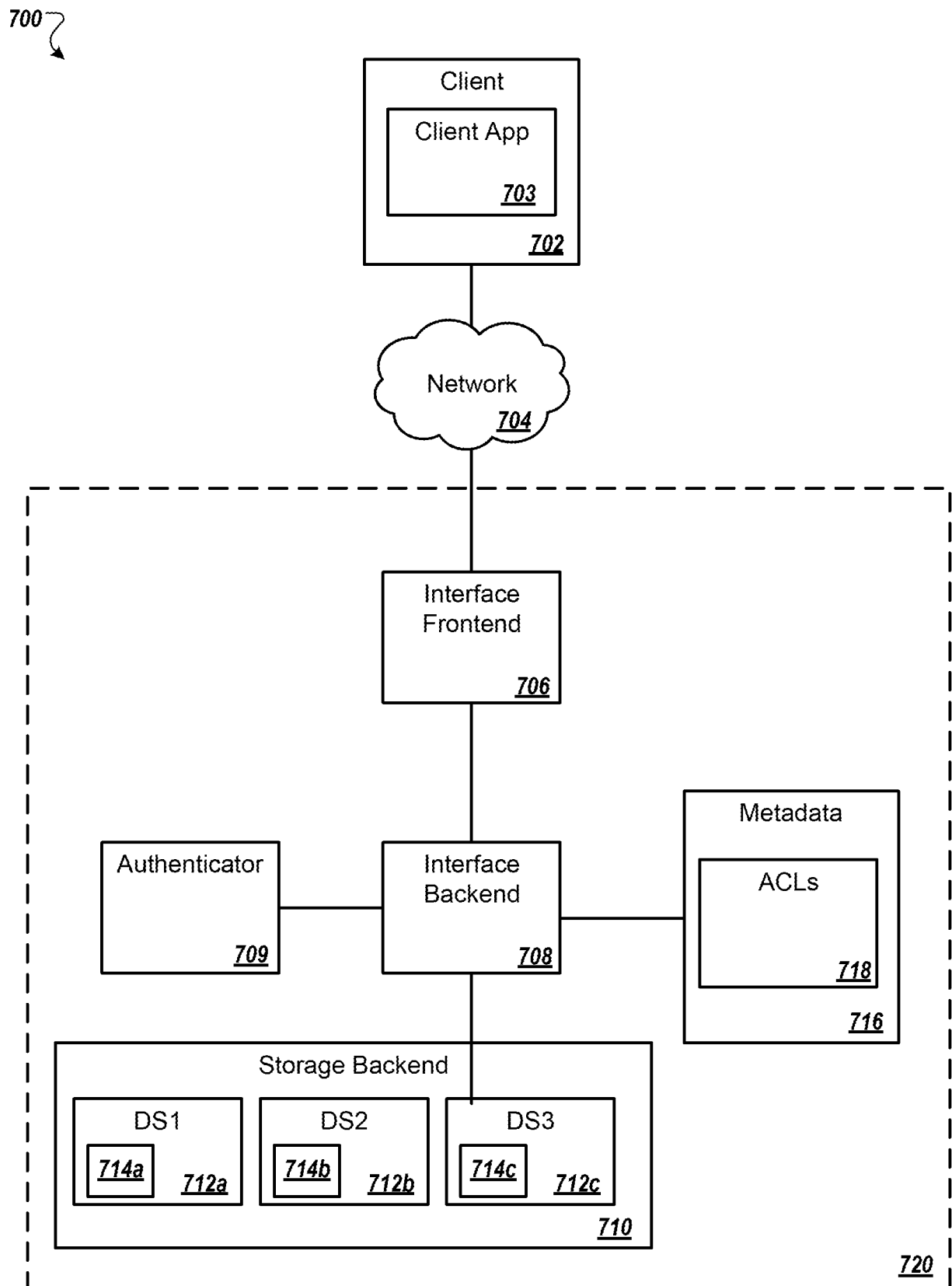


FIG. 7

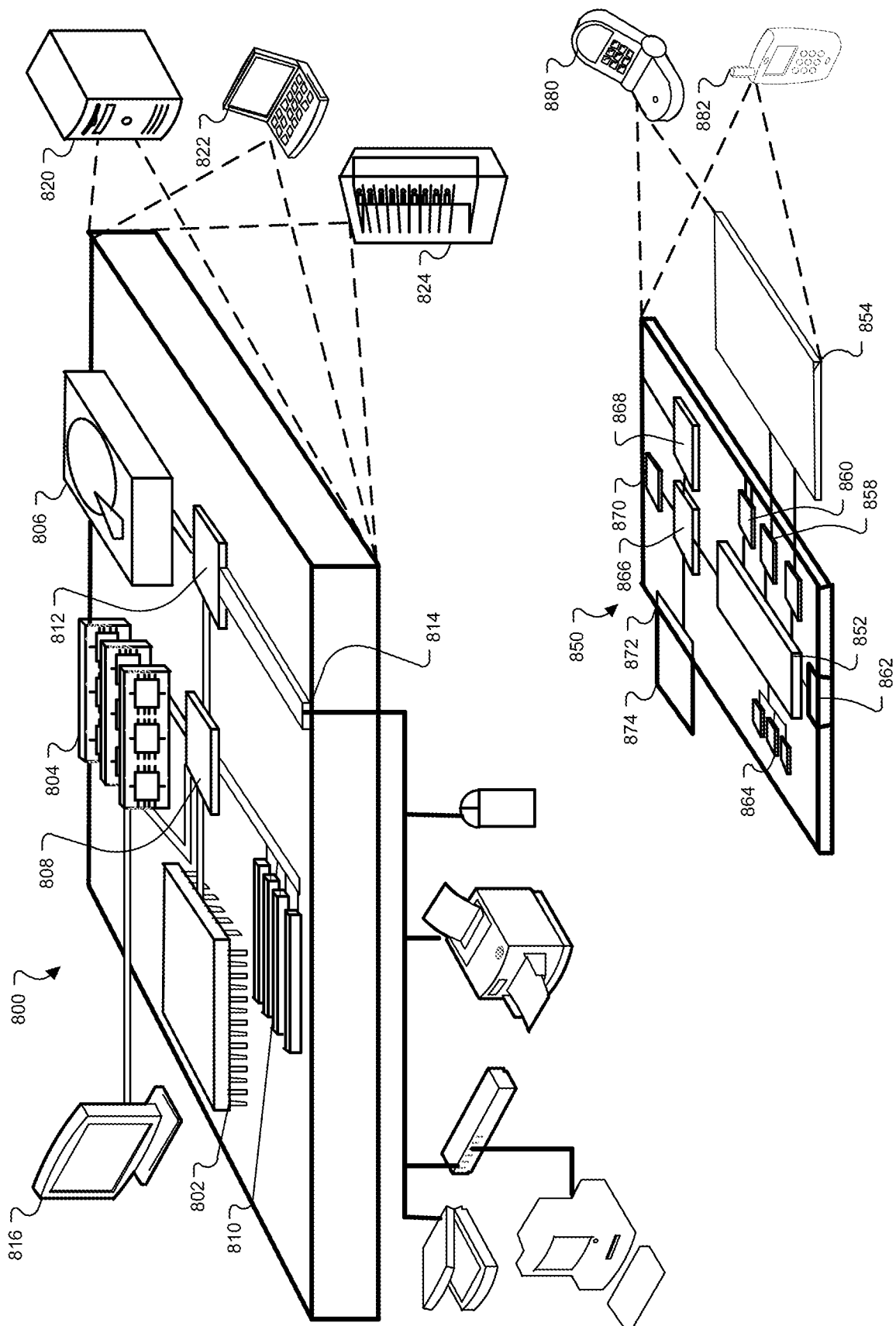


FIG. 8