Abstract: The subject technology discloses configurations for receiving a response message from a server in which the response message includes image content in a first image format. The subject technology detects an unsupported image content type corresponding to the image content in the response message. Responsive to detecting the unsupported image content type, the image content is decoded. The decoded image content is transcoded to a second image format. The subject technology modifies an image content type header in the response message to indicate an image content type of the second image format. A content length header is set in the response message to a size of the transcoded image content. The subject technology transmits the response message to an application.
TRANSCODING IMAGES AT A NETWORK STACK LEVEL

BACKGROUND

[0001] The subject technology generally relates to image processing and transcoding of an image for display on a computing device.

SUMMARY

[0002] The subject technology provides for a computer-implemented method, the method including: receiving a response message from a server, wherein the response message includes image content in a first image format; detecting an unsupported image content type corresponding to the image content in the response message; responsive to detecting the unsupported image content type, decoding the image content; transcoding the image content to a second image format; modifying an image content type header in the response message to indicate an image content type of the second image format; setting a content length header in the response message to a size of the transoded image content; and transmitting the response message to an application.

[0003] Yet another aspect of the subject technology provides a system. The system includes one or more processors, and a memory including instructions stored therein, which when executed by the one or more processors, cause the processors to perform operations including: receiving a response message from a server, wherein the response message includes image content in a first image format; detecting an unsupported image content type corresponding to the image content in the response message; responsive to detecting the unsupported image content type, decoding the image content; transcoding the image content to a second image format; modifying an image content type header in the response message to indicate an image content type of the second image format; setting a content length header in the response message to a size of the transoded image content; and transmitting the response message to an application.

[0004] The subject technology further provides for a non-transitory machine-readable medium comprising instructions stored therein, which when executed by a machine, cause the machine to perform operations including: receiving a response message from a server, wherein the response message includes image content in a first image format; detecting an unsupported image content type corresponding to the image content in the response message; responsive to detecting
the unsupported image content type, decoding the image content; transcoding the image content to a second image format; modifying an image content type header in the response message to indicate an image content type of the second image format; setting a content length header in the response message to a size of the transcoded image content; and transmitting the response message to an application.

[0005] It is understood that other configurations of the subject technology will become readily apparent from the following detailed description, where various configurations of the subject technology are shown and described by way of illustration. As will be realized, the subject technology is capable of other and different configurations and its several details are capable of modification in various other respects, all without departing from the scope of the subject technology. Accordingly, the drawings and detailed description are to be regarded as illustrative in nature and not as restrictive.

**BRIEF DESCRIPTION OF THE DRAWINGS**

[0006] The features of the subject technology are set forth in the appended claims. However, for purpose of explanation, several configurations of the subject technology are set forth in the following figures.

[0007] **Figure 1** conceptually illustrates an example computing architecture in which some configurations of the subject technology may be implemented.

[0008] **Figure 2** conceptually illustrates an exemplary process for transcoding an image at a network stack level.

[0009] **Figure 3** is an example of a mobile device architecture in which some configurations of the subject technology may be implemented.

[0010] **Figure 4** conceptually illustrates a system with which some implementations of the subject technology may be implemented.
DETAILED DESCRIPTION

[0011] The detailed description set forth below is intended as a description of various configurations of the subject technology and is not intended to represent the only configurations in which the subject technology may be practiced. The appended drawings are incorporated herein and constitute a part of the detailed description. The detailed description includes specific details for the purpose of providing a thorough understanding of the subject technology. However, the subject technology is not limited to the specific details set forth herein and may be practiced without these specific details. In some instances, structures and components are shown in block diagram form in order to avoid obscuring the concepts of the subject technology.

[0012] In one example, an application such as a web browser may include support for displaying one or more image formats from data received from web sites or servers. For instance, common image formats including JPG, GIF, PNG and others may be natively supported by the application. In other words, the application may have capability to decode image data in one of the aforementioned formats for display in the application. The image(s), in one example, may be rendered for display in web page data for a requested web page.

[0013] New image formats that provide improved image quality or compression characteristics have been developed for utilization in different areas of computing. For example, WebP is an image format that provides lossless and lossy compression for images on the web. In some examples, WebP lossless images may be 26% smaller in size compared to Portable Network Graphics images (PNGs). In some examples, WebP lossy images may be 25-34% smaller in size compared to JPEG images at equivalent structural similarity (SSIM) index (e.g., a metric for indicating similarity between two images). WebP may support lossless transparency (e.g., alpha channel) with around 22% additional bytes. Transparency is also supported with lossy compression and typically provides 3x smaller file sizes compared to PNG when lossy compression is acceptable for the red/green/blue color channels.
[0014] A web browser or other application running on a mobile computing device may not natively support the latest image formats for display. To enable a web browser to display unsupported images, some existing implementations may provide transcoding of different image formats to an image format compatible with a given client or application. However, some existing implementations may perform transcoding at an application level on a client computing device where an image(s) is received at an application (e.g., via a network message such as a HTTP response) and then stored before transcoding the image by the application and then rendering the transcoded image for display by the application. However, this may result in additional delay in displaying the transcoded image due to further operations to transcode the image(s) by the application. Further, this approach may require ad-hoc integration of a transcoding engine built into a given application that degrades the performance of the application. Other implementations may provide transcoding at a server-side to offset processing at a client. However, a server-side implementation may result in additional delay in a client receiving a response to a request due to further processing at the server-side before transmitting a transcoded image to the client.

[0015] A web browser or other application on a client computing device may transmit a HTTP request message (e.g., for a URL or web address) to a web server. In response to the HTTP request message, the web server may then transmit a HTTP response message to the web browser including the requested content (e.g., the requested URL). As dictated by existing Internet standards, the HTTP response message includes one or more HTTP header fields for describing various aspects of data included in the HTTP response message. For example, a HTTP response message may include a "Content-Type" header field for indicating a MIME type (Internet media type) of content included in the HTTP response message. In some examples, the MIME type may indicate a file format for the content such as an image format, audio format, video format, or type of application. Based on the MIME type in the header fields of the HTTP response message, the web browser may determine an appropriate set of operations for handling or processing the content.

[0016] As discussed above, a given application or web browser may not, however, support all content types in light of new file formats being introduced such as improved image formats.
The subject technology provides techniques for at least 1) intercepting a HTTP response message at a network stack level, 2) transcoding one or more images included in the HTTP response message, 3) and rewriting a MIME type header corresponding to transcoded image content in the HTTP response message. As used herein, the network stack level may refer to a network layer that is utilized for packet forwarding of network traffic to one or more recipients. Such recipients may include a network component (e.g., router, server, client, etc.).

[0017] Figure 1 conceptually illustrates an example computing architecture 100 in which some configurations of the subject technology may be implemented. For instance, the example shown in Figure 1 illustrates an example implementation of transcoding an image(s) at a network stack level. In one example, the computing architecture 100 may be implemented by a client computing device (e.g., mobile device, tablet, laptop, phablet, etc.) or system.

[0018] As illustrated in the example of Figure 1, a processor 110 may execute one or more instructions for an application 115 (e.g., web browser, etc.). During execution of the application 115, a HTTP request message may be transmitted over a network to a server (e.g., web server, etc.). A system component illustrated as a web protocol client 120 may intercept a HTTP response message corresponding to the HTTP request message in some configurations. The web protocol client 120 may monitor network traffic at the network layer over a connection established between the application 115 and the server. Alternatively, in instances in which connectionless communication is utilized (e.g., via UDP), the web protocol client may monitor all network traffic between the application 115 and the server.

[0019] In one example, a HTTP response message 130 is intercepted by the web protocol client 120. The web protocol client 120 detects that the HTTP response message 130 includes a MIME type (e.g., "image/webp") indicating an unsupported image format (e.g., webp) that is not natively supported by the application 115. The web protocol client 120 may extract image content corresponding to the unsupported image format and forward the image content to an image transcoder 125. The image transcoder 125 provides functionality to transcode the image content to a natively supported image format (e.g., TIFF) that may be processed by the application 115. In one example, the image transcoder 125 may initially decode the image
content (e.g., in the unsupported image format) and then transcode the decoded image content to the natively supported image format of the application 115. The natively supported image format may be an uncompressed (or lossless compression) image format in some configurations. In some implementations to produce the transcoded image content, the image transcoder 125 may generate a header corresponding to the natively supported image format and then append the generated header to the decoded image content.

[0020] The image transcoder 125 may store the decoded image content and the transcoded image content in a memory buffer 135 for temporary storage in some configurations. The memory buffer 135 may be configured to be a size large enough to at least store the decoded image content and/or the transcoded image content.

[0021] The web protocol client 120 may overwrite the MIME type header of the HTTP response message 130 to correspond to the image format of the transcoded image content (e.g., "image/tiff") and set a header field indicating a size of the transcoded image content (e.g., "Content-Length"). In some configurations, the size of the transcoded image content is indicated in octets (8-bit bytes). Other header fields in the of the HTTP response message 130 may be left unchanged in some configurations. The web protocol client 120 further modifies the HTTP response message 130 by replacing the image content corresponding to the unsupported image format with the transcoded image content. A modified HTTP response message 145 is then transmitted, by the web protocol client 120, to the application 115 for processing and presentation to an end-user. From the perspective of the application 115, the modified HTTP response message 145 appears as an original (e.g., unmodified) HTTP response message from the server. Thus, the processing by the web protocol client 120 and the image transcoder 125 is transparent to the application 115.

[0022] Although the example described in Figure 1 relates to WebP and TIFF image formats, it is appreciated that other image formats may be processed by the web protocol client 120 and the image transcoder 125 and still be within the scope of the subject technology. For example, JPEG XR, JPEG2000 and/or BMP image formats may be utilized.
[0023] Figure 2 conceptually illustrates an exemplary process for transcoding an image at a network stack level. The process 200 can be performed on one or more computing devices or systems in some configurations. In some configurations, the process 200 is performed by a client computing device (e.g., a mobile device).

[0024] The process 200 begins at 210 by receiving a response message from a server in which the response message includes image content in a first image format. The response message from the server may be responsive to a request message (e.g., HTTP request message) from an application executing on the client computing device. In some configurations, the response message is received by intercepting network traffic between the application and the server. The first image format may be a WebP, JPEG XR, JPEG2000 or a lossy image format in one example.

[0025] At 215, the process 200 detects whether an unsupported image content type corresponding to the image content is included in the response message. For instance, the application is unable to render for display the unsupported image content type. In one example, detecting the unsupported image content type is based on a MIME type header included in the response message. If no unsupported image content type is detected, the process 200 then ends. Alternatively, responsive to detecting the unsupported image content type, the process 200 continues to 220 to decode the image content. In some configurations, the decoded image content is stored in a temporary memory buffer.

[0026] The process 200 at 225 transcodes the image content to a second image format. The second image format may be a TIFF, BMP or a lossless image format in some configurations. The process 200 at 230 modifies an image content type header in the response message to indicate an image content type of the second image format. In some configurations, modifying the image content type header is accomplished by overwriting a MIME type header to indicate the image content type of the second image format. The process 200 at 235 sets a content length header in the response message to a size of the transcoded image content. For example, the content length header is specified in octets (e.g., 8-bit bytes). At 240, the process 200 transmits the response message to an application. The process 200 then ends.
[0027] Figure 3 is an example of a mobile device architecture 300. The implementation of a mobile device can include one or more processing units 305, memory interface 310 and a peripherals interface 315. Each of these components that make up the computing device architecture can be separate components or integrated in one or more integrated circuits. These various components can also be coupled together by one or more communication buses or signal lines.

[0028] The peripherals interface 315 can be coupled to various sensors and subsystems, including a camera subsystem 320, a wireless communication subsystem(s) 325, audio subsystem 330 and Input/Output subsystem 335. The peripherals interface 315 enables communication between processors and peripherals. The peripherals provide different functionality for the mobile device. Peripherals such as an orientation sensor 345 or an acceleration sensor 350 can be coupled to the peripherals interface 315 to facilitate the orientation and acceleration functions. Additionally, the mobile device can include a location sensor 375 to provide different location data. In particular, the location sensor can utilize a Global Positioning System (GPS) to provide different location data such as longitude, latitude and altitude.

[0029] The camera subsystem 320 can be coupled to one or more optical sensors such as a charged coupled device (CCD) optical sensor or a complementary metal-oxide-semiconductor (CMOS) optical sensor. The camera subsystem 320 coupled with the sensors can facilitate camera functions, such as image and/or video data capturing. Wireless communication subsystems 325 can serve to facilitate communication functions. Wireless communication subsystems 325 can include radio frequency receivers and transmitters, and optical receivers and transmitters. The aforementioned receivers and transmitters can be implemented to operate over one or more communication networks such as a Long Term Evolution (LTE), Global System for Mobile Communications (GSM) network, a Wi-Fi network, Bluetooth network, etc. The audio subsystem 330 is coupled to a speaker and a microphone to facilitate voice-enabled functions, such as voice recognition, digital recording, etc.
I/O subsystem 335 involves the transfer between input/output peripheral devices, such as a display, a touchscreen, etc., and the data bus of the processor 305 through the Peripherals Interface. I/O subsystem 335 can include a touchscreen controller 355 and other input controllers 30 to facilitate these functions. Touchscreen controller 355 can be coupled to the touchscreen 35 and detect contact and movement on the screen using any of multiple touch sensitivity technologies. Other input controllers 30 can be coupled to other input/control devices, such as one or more buttons.

Memory interface 310 can be coupled to memory 370, which can include high-speed random access memory and/or non-volatile memory such as flash memory. Memory 370 can store an operating system (OS). The OS can include instructions for handling basic system services and for performing hardware dependent tasks.

By way of example, memory can also include communication instructions to facilitate communicating with one or more additional devices, graphical user interface instructions to facilitate graphic user interface processing, image/video processing instructions to facilitate image/video-related processing and functions, phone instructions to facilitate phone-related processes and functions, media exchange and processing instructions to facilitate media communication and processing-related processes and functions, camera instructions to facilitate camera-related processes and functions, and video conferencing instructions to facilitate video conferencing processes and functions. The above identified instructions need not be implemented as separate software programs or modules. Various functions of mobile device can be implemented in hardware and/or in software, including in one or more signal processing and/or application specific integrated circuits.

Many of the above-described features and applications are implemented as software processes that are specified as a set of instructions recorded on a non-transitory machine readable storage medium (also referred to as computer readable medium). When these instructions are executed by one or more processing unit(s) (e.g., one or more processors, cores of processors, or other processing units), they cause the processing unit(s) to perform the actions indicated in the instructions. Examples of non-transitory machine readable media
include, but are not limited to, CD-ROMs, flash drives, RAM chips, hard drives, EPROMs, etc. The machine readable media does not include carrier waves and electronic signals passing wirelessly or over wired connections.

[0034] In this specification, the term "software" is meant to include firmware residing in read-only memory and/or applications stored in magnetic storage, which can be read into memory for processing by a processor. Also, in some implementations, multiple software components can be implemented as sub-parts of a larger program while remaining distinct software components. In some implementations, multiple software subject components can also be implemented as separate programs. Finally, a combination of separate programs that together implement a software component(s) described here is within the scope of the subject technology. In some implementations, the software programs, when installed to operate on one or more systems, define one or more specific machine implementations that execute and perform the operations of the software programs.

[0035] A computer program (also known as a program, software, software application, script, or code) can be written in a form of programming language, including compiled or interpreted languages, declarative or procedural languages, and it can be deployed in some form, including as a stand alone program or as a module, component, subroutine, object, or other unit suitable for use in a computing environment. A computer program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[0036] Some configurations are implemented as software processes that include one or more application programming interfaces (APIs) in an environment with calling program code interacting with other program code being called through the one or more interfaces. Various function calls, messages or other types of invocations, which can include various kinds of
parameters, can be transferred via the APIs between the calling program and the code being called. In addition, an API can provide the calling program code the ability to use data types or classes defined in the API and implemented in the called program code.

[0037] One or more APIs may be used in some configurations. An API is an interface implemented by a program code component or hardware component ("API implementing component") that allows a different program code component or hardware component ("API calling component") to access and use one or more functions, methods, procedures, data structures, classes, and/or other services provided by the API implementing component. An API can define one or more parameters that are passed between the API calling component and the API implementing component.

[0038] The following description describes an example system in which aspects of the subject technology can be implemented.

[0039] Figure 4 conceptually illustrates a system 400 with which some implementations of the subject technology can be implemented. The system 400 can be a computer, phone, PDA, or another sort of electronic device. In some configurations, the system 400 includes a television with one or more processors embedded therein. Such a system includes various types of computer readable media and interfaces for various other types of computer readable media. The system 400 includes a bus 405, processing unit(s) 410, a system memory 415, a read-only memory 420, a storage device 425, an optional input interface 430, an optional output interface 435, and a network interface 440.

[0040] The bus 405 collectively represents all system, peripheral, and chipset buses that communicatively connect the numerous internal devices of the system 400. For instance, the bus 405 communicatively connects the processing unit(s) 410 with the read-only memory 420, the system memory 415, and the storage device 425.

[0041] From these various memory units, the processing unit(s) 410 retrieves instructions to execute and data to process in order to execute the processes of the subject technology. The
processing unit(s) can be a single processor or a multi-core processor in different implementations.

[0042] The read-only-memory (ROM) 420 stores static data and instructions that are needed by the processing unit(s) 410 and other modules of the system 400. The storage device 425, on the other hand, is a read-and-write memory device. This device is a non-volatile memory unit that stores instructions and data even when the system 400 is off. Some implementations of the subject technology use a mass-storage device (such as a magnetic or optical disk and its corresponding disk drive) as the storage device 425.

[0043] Other implementations use a removable storage device (such as a flash drive, a floppy disk, and its corresponding disk drive) as the storage device 425. Like the storage device 425, the system memory 415 is a read-and-write memory device. However, unlike storage device 425, the system memory 415 is a volatile read-and-write memory, such a random access memory. The system memory 415 stores some of the instructions and data that the processor needs at runtime. In some implementations, the subject technology’s processes are stored in the system memory 415, the storage device 425, and/or the read-only memory 420. For example, the various memory units include instructions for processing multimedia items in accordance with some implementations. From these various memory units, the processing unit(s) 410 retrieves instructions to execute and data to process in order to execute the processes of some implementations.

[0044] The bus 405 also connects to the optional input and output interfaces 430 and 435. The optional input interface 430 enables the user to communicate information and select commands to the system. The optional input interface 430 can interface with alphanumeric keyboards and pointing devices (also called "cursor control devices"). The optional output interface 435 can provide display images generated by the system 400. The optional output interface 435 can interface with printers and display devices, such as cathode ray tubes (CRT) or liquid crystal displays (LCD). Some implementations can interface with devices such as a touchscreen that functions as both input and output devices.
Finally, as shown in Figure 4, bus 405 also couples system 400 to a network interface 440 through a network adapter (not shown). In this manner, the computer can be a part of a network of computers (such as a local area network ("LAN"), a wide area network ("WAN"), or an Intranet, or an interconnected network of networks, such as the Internet. The components of system 400 can be used in conjunction with the subject technology.

These functions described above can be implemented in digital electronic circuitry, in computer software, firmware or hardware. The techniques can be implemented using one or more computer program products. Programmable processors and computers can be included in or packaged as mobile devices. The processes and logic flows can be performed by one or more programmable processors and by one or more programmable logic circuitry. General and special purpose computing devices and storage devices can be interconnected through communication networks.

Some implementations include electronic components, such as microprocessors, storage and memory that store computer program instructions in a non-transitory machine-readable or non-transitory computer-readable medium (alternatively referred to as computer-readable storage media, machine-readable media, or machine-readable storage media). Some examples of such computer-readable media include RAM, ROM, read-only compact discs (CD-ROM), recordable compact discs (CD-R), rewritable compact discs (CD-RW), read-only digital versatile discs (e.g., DVD-ROM, dual-layer DVD-ROM), a variety of recordable/rewritable DVDs (e.g., DVD-RAM, DVD-RW, DVD+RW, etc.), flash memory (e.g., SD cards, mini-SD cards, micro-SD cards, etc.), magnetic and/or solid state hard drives, read-only and recordable Blu-Ray® discs, ultra density optical discs, optical or magnetic media, and floppy disks. The computer-readable media can store a computer program that is executable by at least one processing unit and includes sets of instructions for performing various operations. Examples of computer programs or computer code include machine code, such as is produced by a compiler, and files including higher-level code that are executed by a computer, an electronic component, or a microprocessor using an interpreter.
[0048] While the above discussion primarily refers to microprocessor or multi-core processors that execute software, some implementations are performed by one or more integrated circuits, such as application specific integrated circuits (ASICs) or field programmable gate arrays (FPGAs). In some implementations, such integrated circuits execute instructions that are stored on the circuit itself.

[0049] As used in this specification and the claims of this application, the terms "computer", "server", "processor", and "memory" all refer to electronic or other technological devices. These terms exclude people or groups of people. For the purposes of the specification, the terms display or displaying means displaying on an electronic device. As used in this specification and the claims of this application, the terms "computer readable medium" and "computer readable media" are entirely restricted to non-transitory, tangible, physical objects that store information in a form that is readable by a computer. These terms exclude wireless signals, wired download signals, and other ephemeral signals.

[0050] To provide for interaction with a user, implementations of the subject matter described in this specification 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 information 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. In addition, a computer can interact with a user by sending documents to and receiving documents from a device that is used by the user; for example, by sending web pages to a web browser on a user's client device in response to requests received from the web browser.

[0051] Configurations of the subject matter described in this specification 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 graphical user interface or a Web browser through
which a user can interact with an implementation of the subject matter described in this specification, or a combination of one or more 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") and a wide area network ("AIVAN"), an inter-network (e.g., the Internet), and peer-to-peer networks (e.g., ad hoc peer-to-peer networks).

[0052] 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. In some configurations, a server transmits data (e.g., an HTML page) to a client device (e.g., for purposes of displaying data to and receiving user input from a user interacting with the client device). Data generated at the client device (e.g., a result of the user interaction) can be received from the client device at the server.

[0053] It is understood that a specific order or hierarchy of steps in the processes disclosed is an illustration of example approaches. Based upon design preferences, it is understood that the specific order or hierarchy of steps in the processes can be rearranged, or that all illustrated steps be performed. Some of the steps can be performed simultaneously. For example, in certain circumstances, multitasking and parallel processing can be advantageous. Moreover, the separation of various system components in the configurations described above should not be understood as requiring such separation in all configurations, and it should be understood that the described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

[0054] The previous description is provided to enable a person skilled in the art to practice the various aspects described herein. Various modifications to these aspects will be readily apparent to those skilled in the art, and the generic principles defined herein can be applied to other aspects. Thus, the claims are not intended to be limited to the aspects shown herein, but is to be accorded the full scope consistent with the language claims, wherein reference to an
element in the singular is not intended to mean "one and only one" unless specifically so stated, but rather "one or more." Unless specifically stated otherwise, the term "some" refers to one or more. Pronouns in the masculine (e.g., his) include the feminine and neuter gender (e.g., her and its) and vice versa. Headings and subheadings, if any, are used for convenience only and do not limit the subject technology.

[0055] A phrase such as an "aspect" does not imply that such aspect is essential to the subject technology or that such aspect applies to all configurations of the subject technology. A disclosure relating to an aspect can apply to all configurations, or one or more configurations. A phrase such as an aspect can refer to one or more aspects and vice versa. A phrase such as a "configuration" does not imply that such configuration is essential to the subject technology or that such configuration applies to all configurations of the subject technology. A disclosure relating to a configuration can apply to all configurations, or one or more configurations. A phrase such as a configuration can refer to one or more configurations and vice versa.

[0056] The word "example" is used herein to mean "serving as an example or illustration." An aspect or design described herein as "example" is not necessarily to be construed as preferred or advantageous over other aspects or designs.

[0057] All structural and functional equivalents to the elements of the various aspects described throughout this disclosure that are known or later come to be known to those of ordinary skill in the art are expressly incorporated herein by reference and are intended to be encompassed by the claims.
CLAIMS

What is claimed is:

1. A computer-implemented method, the method comprising:
   receiving a response message from a server, wherein the response message includes image content in a first image format;
   detecting an unsupported image content type corresponding to the image content in the response message;
   responsive to detecting the unsupported image content type, decoding the image content;
   transcoding the decoded image content to a second image format;
   modifying an image content type header in the response message to indicate an image content type of the second image format;
   setting a content length header in the response message to a size of the transcoded image content; and
   transmitting the response message to an application.

2. The method of claim 1, wherein the response message is received by intercepting network traffic between the application and the server.

3. The method of claim 1, wherein the first image format comprises one of a WebP, JPEG XR, JPEG2000 or a lossy image format.

4. The method of claim 1, wherein the second image format comprises one of a TIFF, BMP or a lossless image format.

5. The method of claim 1, wherein detecting the unsupported image content type is based on a MIME type header.

6. The method of claim 1, wherein modifying the image content type header comprises overwriting a MIME type header to indicate the image content type of the second image format.
7. The method of claim 1, wherein content length header is specified in octets.

8. The method of claim 1, wherein the response message from the server is responsive to a request message from the application.

9. The method of claim 1, wherein the application is unable to render for display the unsupported image content type.

10. The method of claim 1, wherein the decoded image content is stored in a temporary memory buffer.

11. A system, the system comprising:
   one or more processors;
   a memory comprising instructions stored therein, which when executed by the one or more processors, cause the processors to perform operations comprising:
   receiving a response message from a server, wherein the response message includes image content in a first image format;
   detecting an unsupported image content type corresponding to the image content in the response message;
   responsive to detecting the unsupported image content type, decoding the image content;
   transcoding the decoded image content to a second image format;
   modifying an image content type header in the response message to indicate an image content type of the second image format;
   setting a content length header in the response message to a size of the transcoded image content; and
   transmitting the response message to an application.

12. The system of claim 11, wherein the response message is received by intercepting network traffic between the application and the server.
13. The system of claim 11, wherein the first image format comprises one of a WebP, JPEG XR, JPEG2000 or a lossy image format.

14. The system of claim 11, wherein the second image format comprises one of a TIFF, BMP or a lossless image format.

15. The system of claim 11, wherein detecting the unsupported image content type is based on a MIME type header.

16. The system of claim 11, wherein modifying the image content type header comprises overwriting a MIME type header to indicate the image content type of the second image format.

17. The system of claim 11, wherein content length header is specified in octets.

18. The system of claim 11, wherein the response message from the server is responsive to a request message from the application.

19. The system of claim 11, wherein the application is unable to render for display the unsupported image content type.

20. The system of claim 11, wherein the decoded image content is stored in a temporary memory buffer.

21. A non-transitory machine-readable medium comprising instructions stored therein, which when executed by a machine, cause the machine to perform operations comprising:
    - receiving a response message from a server, wherein the response message includes image content in a first image format;
    - detecting an unsupported image content type corresponding to the image content in the response message;
    - responsive to detecting the unsupported image content type, decoding the image content;
transcoding the decoded image content to a second image format;
modifying an image content type header in the response message to indicate an image content type of the second image format;
setting a content length header in the response message to a size of the transcoded image content; and
transmitting the response message to an application.
Receive a response message from a server

Detect unsupported image content type?

YES

Decode image content

NO

Transcode decoded image content

Modify an image content type header

Set a content length header

Transmit the response message

End

FIG. 2
FIG. 4
**INTERNATIONAL SEARCH REPORT**

**International application No**
PCT/IB2013/001801

**A. CLASSIFICATION OF SUBJECT MATTER**
INV. H04L29/08 G06F17/30
ADD.

According to International Patent Classification (IPC) and/or both national classification and IPC.

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)
H04L G06F H04N

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

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

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
</table>

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"I" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered involving an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"S" document member of the same patent family

**Date of the actual completion of the international search**
22 January 2014

**Date of mailing of the international search report**
29/01/2014

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

Authorized officer
Kontopodi S., D.
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>RODRIGUEZ J R ET AL: &quot;IBM WebSphere Transcoding Publisher VI.1: Extending Web Applications to the Pervasive World&quot;, IBM INTERNET CITATION, 1 July 2000 (2000-07-01), XP002197984, Retrived from the Internet: URL: <a href="http://publib.boulder.ibm.com/Redbooks/RedbookAbstracts/sg245965.html">http://publib.boulder.ibm.com/Redbooks/RedbookAbstracts/sg245965.html</a> [retrieved on 2000-07-30] sections 1.4 page 13, line 1 - line 4 sections 2.2.1-2.2.2, 2.3.2, 2.5.3.4 figures 2,6,8 -----</td>
<td></td>
</tr>
<tr>
<td>Patent document cited in search report</td>
<td>Publication date</td>
<td>Patent family member(s)</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>us 2004049598 Al</td>
<td>11-03-2004</td>
<td>AU 4321801 A</td>
</tr>
<tr>
<td>US 2004049598 Al</td>
<td>11-03-2004</td>
<td></td>
</tr>
<tr>
<td>WO 0163485 A2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>us 2012265847 Al</td>
<td>18-10-2012</td>
<td>EP 2697967 Al</td>
</tr>
<tr>
<td>EP 2697968 Al</td>
<td></td>
<td></td>
</tr>
<tr>
<td>us 2012265847 Al</td>
<td>18-10-2012</td>
<td></td>
</tr>
<tr>
<td>US 2012265901 Al</td>
<td>18-10-2012</td>
<td></td>
</tr>
<tr>
<td>US 2013282874 Al</td>
<td>24-10-2013</td>
<td></td>
</tr>
<tr>
<td>WO 2012142508 Al</td>
<td>18-10-2012</td>
<td></td>
</tr>
<tr>
<td>WO 2012142510 Al</td>
<td>18-10-2012</td>
<td></td>
</tr>
<tr>
<td>wo 2008098249 Al</td>
<td>14-08-2008</td>
<td>EP 2127230 Al</td>
</tr>
<tr>
<td>EP 2008195761 Al</td>
<td></td>
<td></td>
</tr>
<tr>
<td>US 2014010286 Al</td>
<td>09-01-2014</td>
<td></td>
</tr>
<tr>
<td>WO 2008098249 Al</td>
<td>14-08-2008</td>
<td></td>
</tr>
</tbody>
</table>