UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DROPBOX, INC.,

Petitioner

v.

MOTION OFFENSE, LLC,

Patent Owner

IPR2024-00287
U.S. Patent No. 11,611,520

DECLARATION OF DR. MICHAEL SMITH

TABLE OF CONTENTS

I. INTRODUCTION 1
II. QUALIFICATIONS AND EXPERIENCE 1
III. MATERIALS CONSIDERED 4
IV. LEGAL STANDARDS 5
V. BACKGROUND 8
A. Priority Applications 8
B. Overview of the '520 Patent 8
C. Challenged Claims 17
D. Benefits of File Request Embodiments Over the Prior Art 18
E. Cited References 21
1. *Riepling* 21
2. *Manzano* 21
3. *Meisels* 23
4. *Garcia* 24
VI. CLAIM CONSTRUCTION 26
A. Legal Principles 26
B. "representation of the at least one file" 28
1. The Claim Language Supports Patent Owner's Proposed Construction 29
2. The Specification Supports Patent Owner's Proposed Construction 34
VII. LEGAL PRINCIPLES 37
A. Priority and Written Description 37
B. Prior Art Available Under 35 U.S.C. § 311(b) ..... **Error! Bookmark not defined.**
C. Obviousness 38

VIII. Petitioner Has Not Met Its Burden for Any Claim 39
A. The Challenged Claims Are Entitled to At Least a September 25, 2012 Priority Date 39
B. Grounds 1 and 3: Petitioner Has Failed to Show that Claims 9, 10, 14, and 15 Are Obvious Over *Riepling* or *Riepling* and *Meisels* 40
1. *Riepling* Does Not Teach Claim Element 9[b] Because It Does Not Teach "a file description via the first user interface element that includes a textbox, utilizing the first interface, the file description describing the at least one file in connection with the file request" 41
2. *Riepling* Does Not Teach Claim Element 9[g] Because It Does Not Teach a "first message" That Indicates "at least one of the date or the time on which the file request is based" and "the file description describing the at least one file in connection with the file request" 45
3. *Riepling* Does Not Teach Claim Element 9[l] Because It Does Not Teach "a second message that includes an eighth user interface element, and that does not include a file attachment with the second message" 52
4. *Riepling* Does Not Teach Claim 10 55
5. *Riepling* Does Not Teach Claim 14 55
6. *Riepling* Does Not Teach Claim 15 56
7. For Ground 3, *Meisels* Does Not Cure *Riepling*'s Deficiencies 57
C. Grounds 3 and 4: Petitioner Has Failed to Show that Claims 9, 10, 14, and 15 Are Obvious Over *Riepling* and *Meisels*, and that Claims 11–13, 16, 22–24 Are Obvious over *Riepling*, *Manzano*, and *Meisels* under 35 U.S.C. § 103 59
D. Grounds 2, 4, 5 and 6: Petitioner Has Failed to Show that Claims 11–13, 16, and 22–24 Are Obvious over any combination of *Riepling*, *Manzano*, *Meisels* and *Garcia* under 35 U.S.C. § 103 61

1. These Grounds Fail for the Same Reasons as Grounds 1 and 3 with Respect to *Riepling* and *Meisels* 61
2. Grounds 2, 4, 5 and 6 Also Fail Because No Reference Discloses the Creation of a Representation of the at Least one File in a Location Among One or More Folders on the File Explorer Interface as Claims 63

IX. RESERVATION OF RIGHTS 66

I. INTRODUCTION

1. I am currently an independent consultant.

2. This declaration is based on my own personal knowledge. I make this declaration in support of Patent Owner Motion Offense, LLC's in ("Patent Owner" or "Motion Offense") Patent Owner Response to the Petition in IPR2024-00287 regarding U.S. Patent No. 11,611,520 ("the '520 patent") before the Patent Trial and Appeal Board.

II. QUALIFICATIONS AND EXPERIENCE

3. I hold two degrees from Cambridge University: (1) a Bachelor of Arts degree in Electrical Sciences earned in 1978; and (2) a Master of Arts degree in Electrical Sciences earned in 1978. I also hold two degrees from Stanford University: (1) a Master of Sciences in Electrical Engineering earned in 1981 and (2) a Doctor of Philosophy (Ph.D.) in Electrical Engineering earned in in 1985.

4. From 2000–2004, I acted as CTO of a fabless networking semiconductor start-up, iReady. I was responsible for: the basic technology ideas; recruiting theengineering team; all aspects of design: architecture and IP, design methodology,emulation, verification; system debug, testing, bringup; and standards work. I developed the iReady patent portfolio. I helped sell iReady and its IP to Nvidia in 2004.

5. From 2004–2007, I was employed with Nvidia, acting as a Chief Architect. I was involved in all aspects of systems design: chipset, graphics, memory, peripherals, networking, software (drivers, firmware) from product conception through to high-volume production. I worked with marketing to specify products, with software and hardware engineers on design, and with sales in the field.

6. From 2007–2009, I was Chief Architect and CTO of MetaRAM, a KPCB fabless semiconductor start-up. I hired the initial MetaRAM team and then managed all ofengineering except silicon design. I wrote and filed 30 patents in 6 months and helped sell the MetaRAM IP to Google in 2010.

7. From 2009–2010, I consulted with Google on IP strategy, IP portfolio development, patent prosecution, and high-value litigation.

8. At present, I am involved with two entities. Since 2011, I have acted as Founder and owner of P4TENTS1. This entity consults on all aspects of IP prosecution and litigation, analyzes and values IP and patent portfolios. P4TENTS1 also develops IP including complete patent portfolios. Therefore, I have a deep understanding of patents from a technical perspective.

9. Since 2014, and in addition to my work with P4TENTS1, I have worked with Weaved, renamed remote.it as part-time CTO. This team is made up

of ex-iReady team members. Through my work, I help this start-up provide networking for the Internet of Things using custom network protocols.

10. In addition, prior to 2000, I served in the following capacities:

a. ***IBM (United Kingdom) Ltd., Hursley, England and IBM T.J. Watson Research Center, New York***: Worked on DRAM and analog CMOS ASICs.

b. ***University of Hawaii***: Professor. I created and developed the first undergraduate ASIC program in the US. I also received a National Science Foundation Presidential Young Investigator award from President Ronald Reagan. My principle area of research was hardware for networking and internet protocols. I also advised 45 graduate students. As part of my research I worked with the following companies, either part-time, while on sabbatical, or as unpaid collaboration:

c. ***Eastman Kodak, New York***: I worked to create some of the first CMOS digital camera IC designs.

d. ***University of California, Berkeley***: I worked to create the first undergraduate course on IC design.

e. ***Xilinx, San Jose***: I helped develop the Xilinx University Program and, with NSF, introduced the FPGA into IC design courses worldwide; I helped develop the first FPGA development platform with Stanford and Actel.

f. ***VLSI Technology andCompass Design Automation, San Jose***: As Chief Scientist; I designed one of the first ASICs using logic synthesis, a Viterbi decoder; architect for the first ASIC cell library for synthesis.

g. ***Apple Computer***: I worked on education and with research projects in the Advanced Technology Group on ASIC design; created some of the first university telelectures ever given in collaboration with Stanford and Apple. Developed a website with one of the first online textbooks, "ASICs on the Web."

III. MATERIALS CONSIDERED

11. I have considered the materials listed in this declaration as well as the '520 patent and its file history. I have also reviewed the Petition in this proceeding as well as the references cited by Petitioner and Dr. Mowry in his declaration. I have also reviewed the Board's institution decision.

12. I am being compensated at my regular and established rate of $500 per hour. My compensation is not dependent on the outcome of this proceeding.

IV. LEGAL STANDARDS

13. I am not an attorney. I have been informed regarding certain legal principles. I am not opining on the law, however. I set forth my understanding of the legal principles as they have been explained to me below.

14. I have been informed that claim construction and patentability is generally analyzed from the perspective of a person or ordinary skill in the art at the time of the invention ("POSITA").

15. I have been asked to assume for purposes of my analyses that a POSITA is defined as the Board has defined it in the Institution Decision. I understand that Petitioner proposed a formulation for a POSITA in its Petition. (Pet. at 3–4.) The Board adopted this formulation in its Institution Decision (Paper 11 at 15–16.) For purposes of my analyses and opinions, I have applied this formulation and have been asked to do so assuming the time of invention was in the September 2012 timeframe, up to and including September 25, 2012.

16. Based on my education and professional experience in the field of the claimed inventions, I am familiar with the level and abilities of a POSITA as defined by the Board. I had at least the qualification of a POSITA under this

definition in September 2012. I meet these qualifications today, as well. My opinions and analysis herein are from the perspective of the POSITA as defined by the Board.

17. I have been informed that claim terms are to be given their plain and ordinary meaning as a POSITA would have understood them at the time of the invention when read in light of the claims and in view of the specification and prosecution history.

18. I have been informed that a petitioner must clearly set forth the basis for its challenges in the petition. A petitioner may not rely on the Board to substitute its own reasoning to remedy deficiencies in a petition.

19. I have also been informed that to make a prima facie showing of obviousness, a petition must, among other requirements, demonstrate that the cited references, either alone or together, disclose each element of a challenged claim.

20. I have been informed that Petitioner also has the burden to show there would have been some motivation to combine the asserted references, and that a POSITA would have had a reasonable expectation of success in making such a combination. In other words, obviousness requires more than a mere showing that the prior art includes separate references covering each separate limitation in a claim under examination. Further, even if individual modifications or choices

were obvious, a petition must explain why making all of the changes at once would have been obvious. I understand that such secondary considerations include: a long-felt need, commercial success, unexpected results, praise of the invention, licensing, copying, failure of others, and skepticism by experts.

21. It is also my understanding that hindsight analysis is also inappropriate; obviousness must be measured at the time the invention was made. Consequently, a petitioner may not use the patent as a roadmap. In other words, a petition must demonstrate a rationale to combine prior art references without relying on the patent disclosure itself.

22. I have been informed that a combination of elements in a patent claim may be obvious when all of those elements were known in the prior art and there was a reason for a POSITA to combine or modify the prior art to obtain the elements as claimed, with no change in their respective functions. Furthermore, I understand that a patent claim would not be obvious if a proposed modification would render the prior art unsuitable for its intended purpose or if the proposed modification would change the principle of operation of the prior art. I have also been told that a reference may "teach away" from a proposed combination where it expressly rules out or discourages such a combination.

V. BACKGROUND

A. Priority Applications

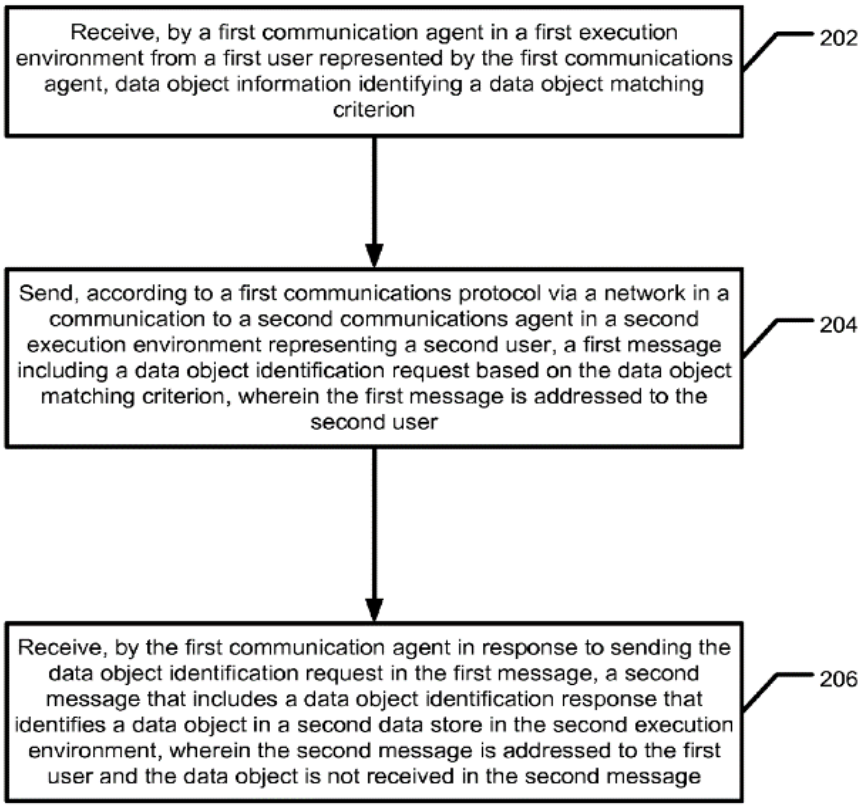
23. The ’520 Patent claims priority to U.S. Patent Application Nos. 13/624,906 and 13/626,635 (the “’906 application” and “’635 application” respectively, and collectively the “Original Applications”, attached hereto as EXS 2004, 2002), filed on September 22, 2012 and September 25, 2012 respectively. The ’635 application incorporates the ’906 application by reference, (EX2002, ¶¶3, 34), and both Original Applications are incorporated by reference in the ’520 patent, (EX1001 at 1:52–62.). The Original Applications were also incorporated by reference in all subsequent applications leading to the ’520 patent. (EX2007.) The Original Applications provide written description support for the challenged claims.

B. Overview of the ’520 Patent

24. The ’520 patent generally relates to methods and systems to request a file to be shared via a network without attaching the specific files, folders etc. to a message and instead allow remote access to the contents of the file (referred to as the “File Request embodiments”), which were disclosed in the Original Applications. At a high level, the File Request embodiments disclose that through a combination of capabilities or operations involving a message with a file request,

the requested file may be shared without an attachment included with another message that is responsive to the foregoing file request. (’906 application, EX2004, at Abstract.) Instead, the file(s) is stored remotely and may be downloaded to the recipient’s local drive only in response to detection of an indication to open or access the file(s) in the folders(s). (’906 application, EX2004 at [0150] , [0151] , [0162])

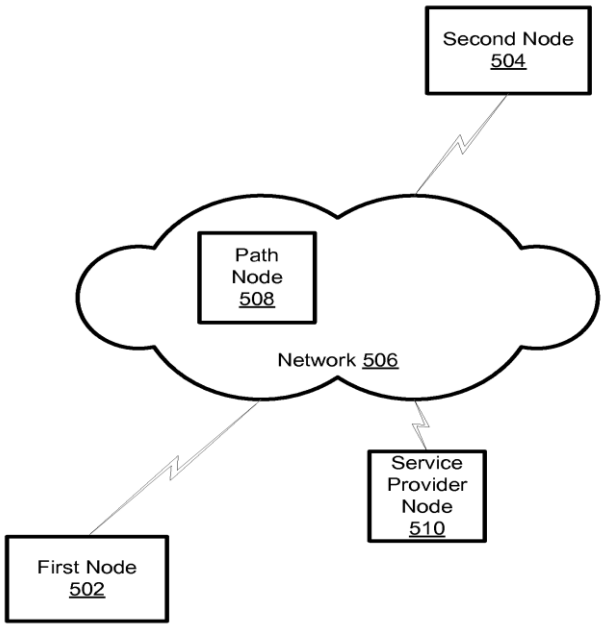
25. The ’906 application describes techniques for causing the sending of “a first message including a data object identification request … wherein the first message is addressed to the second user.” (’906 application, EX2004, at [0127]). In response to such first message, the ’906 application discloses “a second message that includes a data object identification response that identifies a data object …, wherein the second message is addressed to the first user and the data object is not received in the second message.” (’906 application, EX2004, at [0137] and Fig. 2A, reproduced below.)



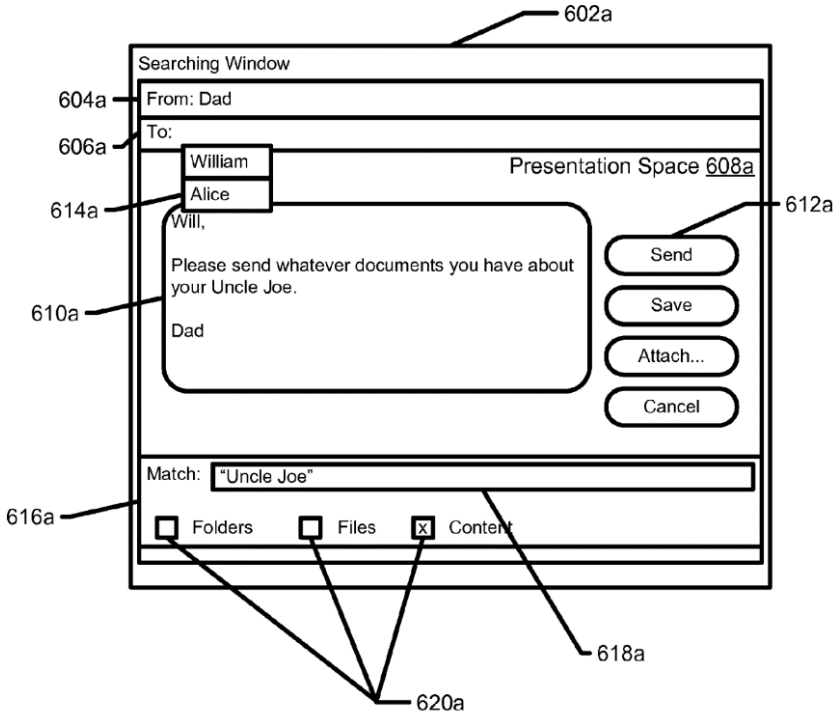
26. This apparatus comprises at least one non-transitory memory storing instructions and at least one processor that can execute these instructions. ('906 application, EX2004, at [0035], [0038], [0052]–[0053], [0209]–[0210].) In embodiments, the software is stored in remote storage accessible via a network. ('906 application, EX2004 at [0039], [0050] , [0102] , [0124], [0162] , [0171].).

27. In some embodiments, a web services provider node may interoperate with users' client nodes, for example, through users' browsers. ('906 application,

EX2004 at [0123]–[0124]) This is generally shown in Figure 5 of the '906 application, reproduced below.



28. When the apparatus such as the web services node executes these instructions, it causes, at a first node, the display of an interface for use in creating a file request for at least one file. ('906 application, EX2004, at [0100], [0111], [0115]–[0117].) An example of this user interface is included in FIG. 6A of the '906 application, copied below:

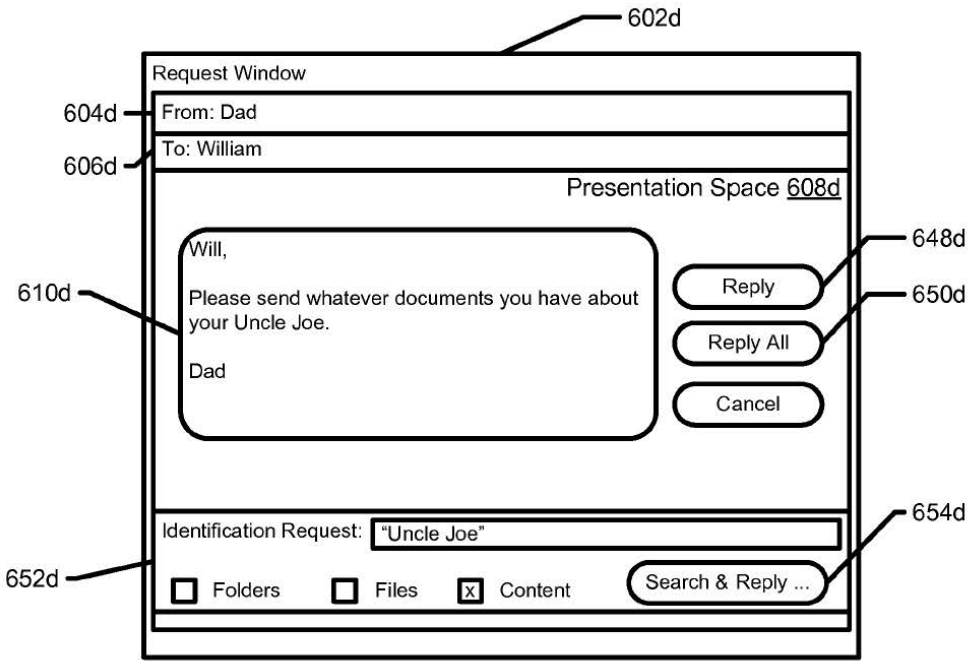


29. The interface (*e.g.*, a Match window 616a in Fig. 6A above) includes a user interface element (that includes a textbox) to receive a file description describing the at least one file in connection with the file request (*e.g.*, text box 618a above). ('906 application, EX2004, at [0125]–[0126].) The interface may also include a second user interface element to receive an indication of a date or a time, so that the file request may be based on the date or the time as, for example, "additional matching criteria." ('906 application, EX2004, at [0092]–[0095], [0133], [0175], [0180].)

30. Also displayed is an interface (*e.g.*, Presentation Space 608a in Fig. 6A above) with a user interface element (*e.g.*, a contactee user interface element

606a) to receive indicia associated with at least one email address, so that the email address is used to send the file request. The interface is also equipped with a user interface element (e.*g.*, a Send button user interface element 612a), where a receipt of an indication of a selection of the user interface element indicates that the file request is to be caused to be sent. (’906 application, EX2004, at [0111], [0115].)

31. In response to the receipt of the indication of the selection of that user interface element (e.*g.*, a Send button user interface element 612a), and based on the file description, the indication of the at least one of the date or the time, and the indicia associated with the at least one email address; a first message addressed to the at least one email address and including the file request, is caused to be generated and sent. To inform the user of the second node, the first message may indicate the date or the time on which the file request is based. (’906 application, EX2004, at [0092]–[0093], [0133], [0175], [0180].) An example of this first message is seen in FIG. 6D of the ’906 application, copied below:



32. As disclosed, the first message may include a sentence including the file description describing the at least one file in connection with the file request. (’906 application, EX2004, at [0100], [0111], [0162], [0166]–[0170].)

33. By this design, an indication of a selection of the user interface element of the first message may be received, which indicates that the file(s) is to be uploaded in response to the file request. In response, an interface is displayed for use in uploading the at least one file in response to the file request. (’906 application, EX2004, at Figs. 6D, 6E, 6F and description thereof.) Such third interface (*e.g.*, Located Resources Dialog interface 630b in Fig. 6B) includes a user interface element for receiving an indication of a selection the at least one file (e.*g.*, the displayed list of files within 632b) and a user interface element for receiving an

indication that the at least one file is to be caused to be uploaded (*e.g.*, an Access or Access All user interface element 634b or 636b). An indication of a selection of the latter user interface element results in the generation and sending of a second message indicating that the file request has been responded to, where the second message includes an eighth user interface element, and does not include a file attachment with the second message. ('906 application, EX2004, at [0150].) An example of this second message is seen in FIGS. 6C and/or 6G of the '906 application, reproduced below:

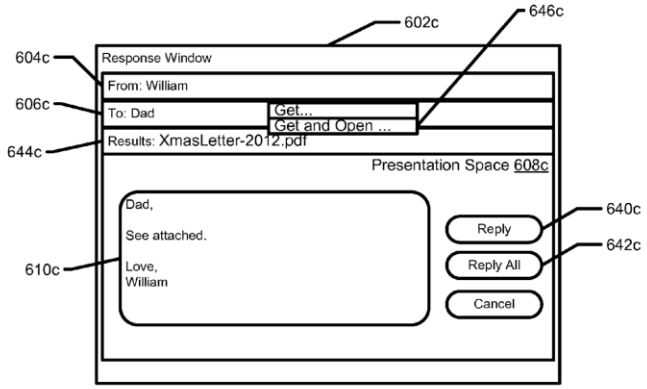


FIG. 6C

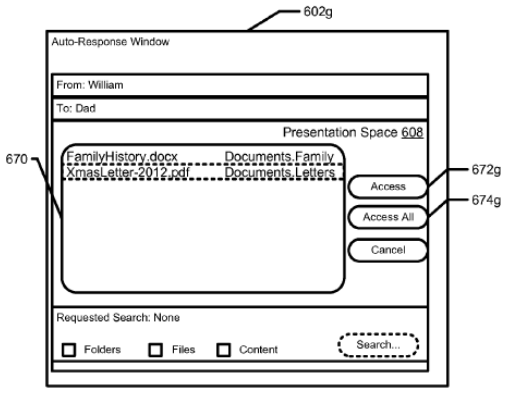


FIG. 6G

34. In response to a receipt of an indication of the selection of the eighth user interface element of the second message, a fourth interface is displayed with a reference to the at least one file for allowing access to the at least one file at the

first node. (’906 application, EX2004, at [0142], [0144], [0147], [0150]–[0151], [0179]), [0201].)

35. The requestor is able to view the file(s) uploaded in response to the file request within the same type of client-based file explorer interface as if a folder containing that file had been shared with them. Specifically, code stored at the first node and configured to cooperate with the client-based file explorer interface may create a representation of the newly uploaded file(s) and display it in a location among other folders on the first node. When the representation of the new file is created, the first node does not store the files, which were not sent as attachments within the generated email message, within the folder. (’635 application, EX2002, at [0109]–[0119].)

36. Instead, the client-based file explorer interface shown in Fig. 6B of the ’635 application, reproduced above (*see* ’635 application, EX2002, at [107]), may cooperate with file systems or file system clients such as the XtreemFS client, in the same manner as if a folder had been shared with the requestor. (’635 application, EX2002, at [116], [118].) A representation of the uploaded file is displayed at the first node among other possible folders in the client-based file explorer interface. Exemplary third-party file explorer interfaces may include Samba, content management interoperability services (CMIS), Web Distributed

Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, the Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). ('635 application, EX2002, at [115], [118], [119].)

37. When the requestor indicates to the apparatus to open the file or edit it, for example, by double-clicking it, an indication to open a file within the represented folder may be detected which, at that point, causes retrieval of the file over the network to allow display of the file at the second node. ('635 application, EX2002, at Fig. 6C and description thereof.)

38. The above features are broadly covered via the challenged claims of the '520 Patent.

C. Challenged Claims

39. I understand that Petitioner challenges claims 9–16 and 22–24 of the '520 patent. Of those, the challenged independent claims are claims 9, 14, and 22.

40. The Board stated that claim 9 is illustrative in its Institution Decision. (Paper 11 at 9.) I disagree. For example, claim 11 depends on claim 9 and further requires that the apparatus is configured such that in response to the receipt of an indication of the selection of the seventh user interface element, a creation of a representation of the at least one file is caused at the first node in a location among

one or more folders on a file explorer interface that is not a web page and utilizes a client-based file explorer application such that the file is accessible via the second message and the file explorer interface. While implicit in claim 9 (regarding the reference) and claim 11 (regarding the representation), claim 12 clarifies that the first node does not store the at least one file until it is requested by a user at the first node using the representation (of claim 11) or reference (of claim 9). Claim 22 and its dependent claims, and dependent claim 16 likewise include limitations directed to representations and must be evaluated independently of claim 9.

D. Benefits of File Request Embodiments Over the Prior Art

41. The ’520 patent discloses inventions to simply and efficiently process file requests. (’906 application, EX2004, at [0003].) It addresses the above problems by, among other things, allowing a requester to specify deadline criteria and provide a description of the requested file that is communicated to the recipient of the file request along with a link for uploading the requested file. A message is sent to the requestor upon uploading in which the message allows access to the requested file. In some embodiments, a web page is provided to the file requestor with a representation of the requested file and providing code to the requestor’s computer that is configured to be stored on the requestor’s computer and to cooperate with the file explorer interface of the requestor’s client-based file

explorer application to cause a representation of the requested file to be created among folders stored on the requestor's computer and displayed via his local file explorer application (e.g., Windows Explorer). The file is not downloaded when the representation is caused to be created. The contents of the file(s) are only downloaded (and stored on the requestor's local drive) in response to a detection of an indication to open them.

42. Prior art systems did not effectively communicate file request deadlines or file descriptions or allow responsive files to be sent as representations through multiple means and be organized within a requester's file system. To overcome this, the '520 patent requires a "first message" that, in addition to including a file description describing the file in connection with the file request, such "first message indicat[es] the at least one of the date or the time on which the file request is based." By this design, the initial indication (to the request-"ee") of an existence of the file request (of the request-"or") takes the form of the "first message" and, importantly, such "first message" itself indicates "the date or the time on which the file request is based." This gives the recipient of the file request (the request-"ee") optimal notice of not only the request for the file itself, but also a timeframe for such file request.

43. In response to the file request, the requestor may receive a representation of the requested file, which may be displayed via a web page and/or locally in the requestor's local file system among other folders and files. A "representation" of the file does not include its contents, which are stored remotely such as on a server and are not communicated to the user till the user initiates the communication. While the files appear to the user to be present locally at the requestor's computer, (EX2003 ¶ [0201]), the contents of such files exist remotely and are not downloaded until the user wishes to interact with them, for example, by viewing or editing them. In other words, these "representations" of the files within the folder act as placeholders for the files and are created at the second node, for example, in response to a file request being responded to. The contents of any of the files are not shared with the requesting node until the user attempts to open the file or access it.

E. Cited References

1. *Riepling*

44. Petitioner relies on U.S. Patent App. Pub. No. 2013/0046833 to Riepling et al. ("*Riepling*") (EX1107), "filed on August 16, 2011 and published on February 21, 2013, as § 102(e) prior art." (Pet. at 5.) *Riepling* was abandoned during prosecution. (EX2014.)

45. *Riepling* is titled "Method and System for Sending a Digital Invitation Requesting a Data Upload" and purports to address a way for registered users of a content sharing service to receive content from unregistered users "in a convenient and secure manner" through the use of invitations. (EX1107 at Abstr., ¶¶ [0002], [0007], [0008].) It purports to do so while protecting registered users from security concerns such as, among other things, unsolicited messages from unregistered accounts. (*Id.* ¶ [0004].)

2. *Manzano*

46. Petitioner relies on U.S. Patent App. Pub. No. 2010/0005138 to Manzano et al. ("*Manzano*") (EX1105), "published on January 7, 2010, as § 102(b) prior art." (Pet. at 5.) *Manzano* is titled "Electronic File Sharing" and purports to address problems involving "exchang[ing] large files using awkward prior-art methods such as e-mail attachments . . . ." (EX1105 ¶ [0003].) *Manzano* teaches

sharing files between devices using "mobile agents" or a "mobile agent object" that exists on a first computer system and can "transplant itself" to a second computer system, while preserving its current execution state. (*Id.* ¶ [0016].) This is performed by the mobile object creating another mobile object that functions as a "proxy object" on the server. (*Id.* ¶ [0035].)

47. Specifically, *Manzano*'s Figure 3 describes that a user may share files between a first computer and a second computer using a "file transfer application" with "file transfer clients" running at each computer device. (*Id.* ¶ [0032].) *Manzano* teaches that files may be shared with the second computer by placing them—such as by dragging and dropping them—in a designated folder. (*Id.* ¶¶ [0033], [0046]–[0049].) The server transfers portions of the shared file to the second computer by creating a separate mobile object that informs the recipient client of the file transfer. (*Id.* ¶¶ [0036], [0039].) In the process, the recipient client creates an electronic file folder in which the transferred file will be placed (*id.*, ¶ [0041]), and displays a progress meter at the sending client to indicate the status of the file transfer, (*id.* ¶ [0046]), and informs the sending client application of the file transfer completion, (*id.* ¶ [0043]).

48. In this context, if a file does not already exist at the second computer in the electronic file folder, it is indicated by a "ghosted" icon. (*Id.* ¶ [0050].) The

user may then selectively download all such files that do not already exist on the second computer. (*Id.* ¶ [0051].) The recipient-side operation is further shown in Figure 5, which describes that the recipient computer creates a mobile object, which in turn creates a proxy object at the server to establish a network connection. (*Id.* at Fig. 5; ¶¶ [0059]–[0061].) The proxy object provides each portion of a file to be transferred to the recipient client in the specially created electronic folder. (*Id.* ¶¶ [0041], [0062].)

3. *Meisels*

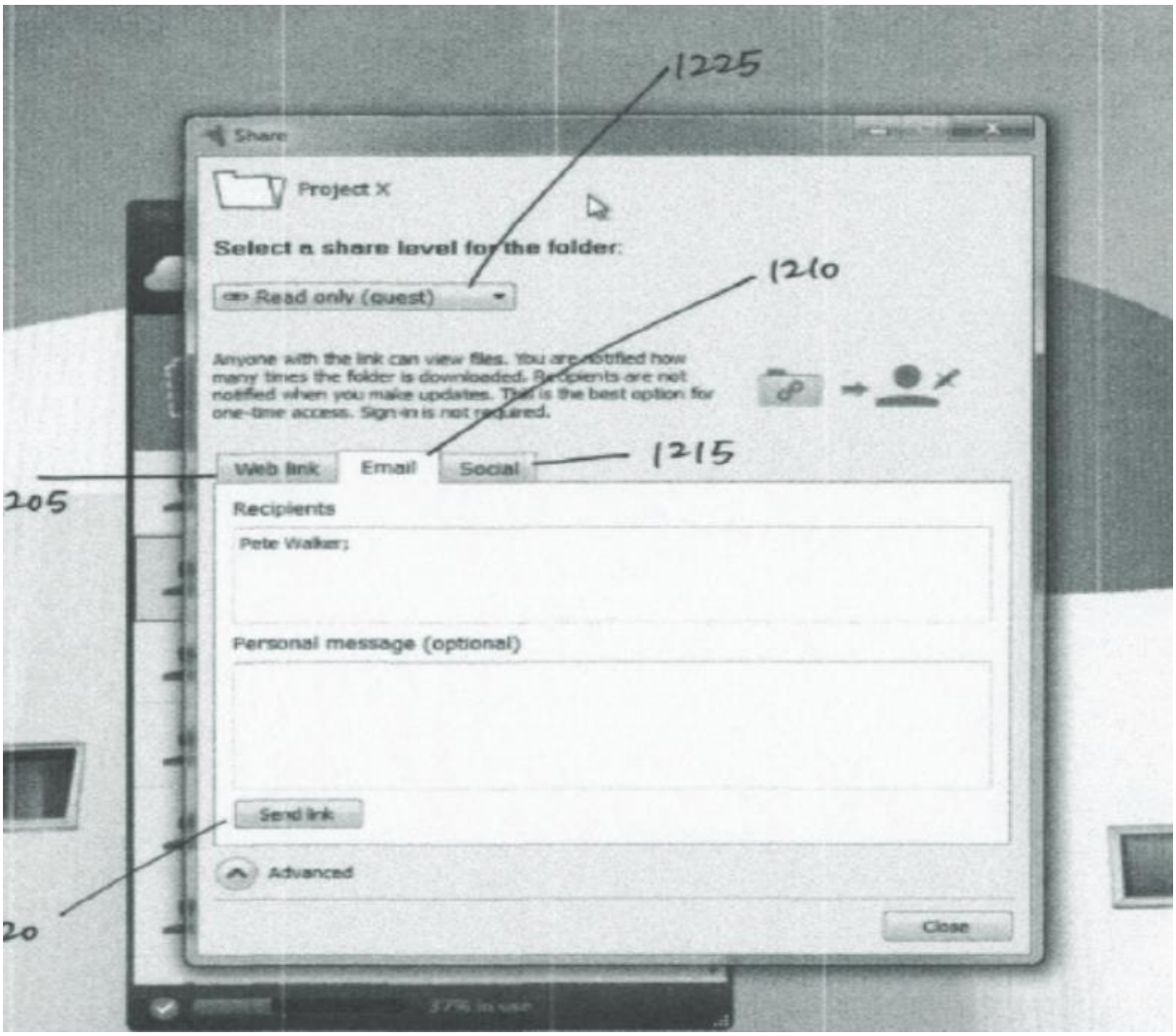
49. Petitioner relies on U.S. Patent App. Pub. No. 2012/0278404 to Meisels et al. ("*Meisels*") (EX1108), "filed on April 28, 2011 and published on November 1, 2012, as § 102(e) prior art." (Pet. at 6.) *Meisels* is titled "Upload of attachment and insertion of link into electronic messages," and relates to creating a link for an email attachment that may be sent in the email message "along with or in place of" the attachment. (EX1108 at Abstr., ¶ [0003].) The attachment is uploaded to a network location, and selecting the link "generally retrieves the content that is pointed to by the link," i.e., the attachment. (*Id.* ¶¶ [0023], [0028], [0034], [0035].)

4. *Garcia*

50. Petitioner relies on U.S. Patent No. 9,633,125 to Garcia et al. ("*Garcia*") (EX1104), "filed on August 10, 2012 and issued on April 25, 2017, as § 102(e) prior art." (Pet. at 6.) *Garcia* is a patent originally assigned to SugarSync, Inc. that was assigned to Dropbox in 2014. (EX2008–2009.) *Garcia* purports to describe the operation of the then-existing SugarSync. (EX1104 at Figs. 3–19.)

51. *Garcia* relates to "a system, method, and computer program . . . for enabling a user to synchronize, manage, and share files and folders across a plurality of client devices and a synchronization server." (EX1104 at Abstract, 1:12–15, 1:39–42.) *Garcia* further describes that a client can share a folder with another client by generating and sending an email with a link to and identifying the shared folder, instead of an attachment. (EX1104 at 10:32–34, 10:38–40, 11:39–43, Figs. 11–12.) While *Garcia* does not describe the process of a recipient receiving such an email or accessing a shared folder in much detail, it does state that "in one embodiment, when a user drags a folder to a particular contact, ***the folder*** or ***link to the folder*** is automatically sent (via email, text, or other means) to the contact," (EX1104, 11:16–19 (emphasis added)), and "when the contact clicks on the link to folder 115, the ***contents*** of the folder are shared with the contact,"

(EX1104 at 11:29–31 (emphasis added)). (*See also* EX1104 at 10:32–43 (web link to download).) While Garcia's FIG. 12 is not legible, a higher resolution of the figure is available as supplemental content on PAIR and reproduced below:



(EX2006.) The window shown in Garcia FIG. 12 has "email" selected and states "Anyone with the link can view files. You are notified how many times the folder is downloaded." (*Id.*) At the bottom, there is a button stating "Send Link."

(EX2006.) Thus, Garcia discloses that the recipient of the shared folder downloads the ***contents*** of the folder when the link is accessed. It does ***not*** disclose the creation of a representation of the folder at the receiving node or representations of any files. At most, it suggests that the receiving user's ***other devices (i.e., other nodes)*** may show the shared folder as a representation.

VI. CLAIM CONSTRUCTION

52. I understand that Patent Owner proposes that the Board construe the term "representation of the at least one file," which appears in claims 11–12, 16, and 22:

Term	Patent Owner's proposed construction
"representation of the at least one file"	a file that can be displayed at the user's node, but whose at least one file content is stored remotely and is not communicated to the user till the user initiates the communication

53. I agree with this proposed construction, and further agree that it is necessary to construe this term because Petitioner's arguments and the Board's Institution Decision rely on an incorrect assumption about what it means.

A. Legal Principles

54. I have been informed that, in an *inter partes* review proceeding, a claim of a patent shall be construed using the same claim construction standard that

would be used to construe the claim in a civil action, including construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent. Under this standard, the Board must determine how a POSITA would understand a claim term in the context of the entire patent, including the specification.

55. I have also been informed that the claims themselves provide substantial guidance as to the meaning of particular claim terms. The context in which a term is used in the asserted claim can be highly instructive.

56. Further, I understand that the claims do not stand alone; instead, they are part of a fully integrated written instrument, consisting principally of a specification. For that reason, claims must be read in view of the specification, of which they are a part. I understand that the specification is always highly relevant to claim construction; it is the single best guide to the meaning of a disputed term.

57. I also understand that the Board may consider other claims of the patent in question, both asserted and unasserted, as valuable sources of enlightenment as to the meaning of a claim term. The usage of a term in one claim can often illuminate the meaning of the same term in other claims.

58. I have been informed that a claim term should be construed consistently throughout a patent. Where multiple patents derive from the same parent application and share many common terms, the Board must interpret the claims consistently across all asserted patents.

B. "representation of the at least one file"

Term	Patent Owner's proposed construction
"representation of the at least one file"	a file that can be displayed at the user's node, but whose at least one file content is stored remotely and is not communicated to the user till the user initiates the communication

59. The '520 patent claims recite two "representation[s] of the at least one file": a "first" representation and a "second" representation. Each of these representations of a file can be displayed at the user's node in a location among one or more folders, just like an ordinary file, either via a web page or in the requesting node's client-based file explorer application. But each representation of a file is different than an ordinary file for two reasons: (1) the representation's file contents are stored not at the user's node but instead remotely, despite the representation being displayed at the user's node; and (2) the representation's file contents are not communicated to the user till the user initiates a request for them.

60. For the reasons discussed below, it is my opinion that this proposed construction is consistent with the claim language of challenged claims 11–12, 16, and 22–24, as well as the claim language of claims 1 and 7–8, and the specification, including the Original Applications incorporated by reference.

1. The Claim Language Supports Patent Owner's Proposed Construction

61. The term "representation of the at least one file" appears at least in elements 22[h], [j], and [k] of independent claim 22:

> 22[h]: receiving, from the second node and at at least one server, a signal for causing creation of a first **representation of the at least one file**, in a location among one or more folders, that is stored at the at least one server and that is displayable via at least one web page;
>
> 22[j]: causing, at the first node, receipt of the at least one web page, that results in display, at the first node and via the at least one web page, of a file structure that permits access to the first **representation of the at least one file** that is stored at the at least one server;
>
> 22[k]: causing, at the first node, receipt of code for storage at the first node and cooperation with a file explorer interface of a client-based file explorer application, for being utilized to: based on the detection of the indication of the selection of the seventh user interface element, cause creation of a second **representation of the at least one file**, in a location among one or more folders, that is stored at the first node and that is displayable via the file explorer interface of the client-based file explorer application.

62. These claim elements provide a highly instructive starting point for understanding the term. In particular, these elements show that there are two "representation[s] of the at least one file" (i.e., a "first" and "second" representation), each of which is created in response to different inputs under different circumstances, and each of which can be displayed at the user's node by different means.

63. First, these claim elements show that a "representation of the at least one file" is itself a file because it can be stored "in a location among one or more folders," either at the "at least one server" (in 22[h] and [j]) or at the "first node" (in 22[k]).

64. Second, these claim elements show that there are two representations of a file—a "first" and "second" representation—and both representations can be displayed at the user's node. The first representation is "displayable" at the user's node (i.e., the "second node") "via at least one web page" (in 22[h] and [j]), and the second representation is "displayable" at the user's node "via the file explorer interface of the client-based file explorer application" (in 22[k]).

65. Third, these claim elements show that the first and second representations of the file are created by in response to different inputs, at different times, in different places. The first representation is created in response to "a

signal for causing creation of a first representation of the at least one file" (in 22[h]). The first representation then stored at "the at least one server," and is "display[ed] . . . via at least one web page" at the first node (in 22[h] and [j]). On the other hand, the second representation is caused to be created "based on the detection of the indication of the selection of the seventh user interface element," utilizing "code" that is received at the first node (in 22[k]). The code is stored at the first node and caused to be used "with a file explorer interface of a client-based file explorer application" to create the second representation and "display[]" it via a file-explorer interface at the first node.

66. The "representation[s] of the at least one file" do not contain the contents of the file(s) when they are first created. This is shown by the language of claim element 22[i]. In claim element 22[i], the user at the first node has already indicated that it wants to request a file from a user at the second node. So after the requested file has been uploaded by a user at the second node, the method generates an email notifying the requestor that its request has been responded to, and sends it to the requestor at the first node. The email identifies information about the uploaded file and includes an HTTP link, but—importantly—"does not include a file attachment." Claim element 22[i] explains why:

> 22[i]: . . . for **permitting avoidance of the at least one file from being communicated to and stored at the first node via the second email**,

With this, claim element 22[i] provides two key pieces of information about the "representation[s] of the at least one file": (1) the contents of the files corresponding to the representation (i.e., "**at least one file**") are "**store[d]**" on the "**at least one server**" (see 22[j]); and (2) those contents are not automatically "**communicated**" to "**the first node**," nor are they "**stored at the first node**." Accordingly, Patent Owner incorporates this information in its proposed construction of "representation of the at least one file": i.e., "a file that can be displayed at the user's node, but whose at least one file content is **stored remotely** and is **not communicated to the user till the user initiates the communication**."

67. These aspects of Patent Owner's proposed construction are further supported by the language of claim 16. *See Phillips*, 415 F.3d at 1314 ("[o]ther claims of the patent in question, both asserted and unasserted," are "valuable sources of enlightenment as to the meaning of a claim term"); *see also Rexnord*, 274 F.3d at 1342 (claim terms should be construed consistently throughout a patent). Specifically, the term "representation of the at least one file" appears in dependent claim 16 in the following context:

> 16. The apparatus of claim 15, wherein the apparatus is configured such that,

> in response to the receipt of the indication of the selection of the fifth user interface element, **a creation of a representation of the at least one file is caused**, at the first node, in a location among one or more folders on a file explorer interface that is not a web page and is displayed utilizing a client-based file explorer application,
>
> such that the at least one file is accessible via both the second message and the file explorer interface, wherein the apparatus is further configured such that the second message is sent **without including the at least one file as an attachment of the second message**,
>
> and **a storage at the first node does not store the at least one file when the creation of the representation of the at least one file is caused**, both **for avoiding the at least one file from having to be communicated to and stored at the first node until the at least one file is requested by a user at the first node using the representation or the reference**.

68. Therefore, claim 16 (via independent claim 14) specifies that: (1) the contents of the representation of the at least one file are not "**stored**" at the first node, despite the representation existing among one or more folders in the first node's file-explorer interface; and (2) the contents of the representation of the at least one file are not communicated to the user at the first node "**until the at least one file is requested by a user at the first node using the representation or the reference.**"

69. And as explained above, this avoidance of communicating the contents of the uploaded file automatically to the first node is recited by and consistent with element 22[i].

70. Likewise, claim 12 (via independent claim 9) further supports Patent Owner's proposed construction for the same reasons: i.e., the content of the representation of the at least one file are not "**stored**" at the first node, nor are they "**communicated to**" the user at the first node till the user "**request[s]**" it:

> 12. The apparatus of claim 11, wherein the apparatus is configured such that the second message is sent **without including the at least one file as an attachment of the second message**, and
>
> **a storage at the first node does not store the at least one file when the creation of the representation of the at least one file is caused**,
>
> **both for avoiding the at least one file from having to be communicated to** and **stored at the first node until the at least one file is requested by a user at the first node using the representation or the reference**.

2. The Specification Supports Patent Owner's Proposed Construction

71. The '520 patent's specification further supports Patent Owner's proposed construction of "representation of the at least one file": i.e., "a file that can be displayed at the user's node, but whose at least one file content is **stored**

remotely and is **not communicated to the user till the user initiates the communication**." In particular, the '635 application—which the '520 patent claims priority to and incorporates by reference (EX1001 at 1:52–62; *see also* EX2003 ¶¶ 3, 34)—makes clear that the contents of the representations' files are **stored remotely** rather than at the user's node and are **not communicated to the user till the user initiates the communication.**

72. For example, claim 1 of the '635 application claims a method in which a second user shares "a data object **in a second data store**" with a first user:

> 1. A method for sharing a data object in a data store via a communication, the method comprising:
>
> receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies **a data object in a second data store in the second execution environment**;
>
> detecting the mount descriptor;
>
> determining a first location in a first data store in the first execution environment; and

> creating, based on the mount descriptor, **a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store**,
>
> wherein performing at least one of the preceding actions comprising the method includes execution of an instruction by a processor.

73. This claim has two steps that are relevant here. First, a "representation of the data object" is created at the first user's "first location in a first data store"—which corresponds to claims 22 (as well as 11 and 16) of the '520 patent's creation of a "[second] representation of the at least one file" at the first node. And second, the first user "**access[es] the representation**" at the first user's location, which "**access[es] the data object from the second data store.**" In other words, the contents of the file representation are **stored remotely** (i.e., "in a second data store") rather than at the user's node and are not **accessed** by the user till the user **accesses the representation.**

74. Claims aside, the '635 application's specification also supports Patent Owner's proposed construction because it repeatedly describes a "representation, of the data object" as being located at the user's node ("in the first data storage"). (*See, e.g.*, EX2002 ¶¶ 8–13, 109, 135.) Further, "**when [the representation is] accessed[,]**" the user "**access[es] the data object from the second data store**"

(i.e., the server) rather than from within the representation at the user's node. (*See*, *e.g.*, *id.*)

VII. LEGAL PRINCIPLES

A. Priority and Written Description

75. I have been informed that the priority date for later-added patent claims depends on when the claimed subject matter first appeared in the chain of patent applications from which the claims arose. For claims to be entitled to a priority date of an earlier-filed application, the application must provide adequate written description support for the later-claimed limitations.

76. I have also been informed that the test for written description involves an objective inquiry into the four corners of the specification from the perspective of a person of ordinary skill in the art. The invention is, for purposes of the written description inquiry, whatever is now claimed. Further, I understand that a specification adequately describes an invention when it reasonably conveys to those skilled in the art that the inventor had possession of the claimed subject matter as of the filing date. The invention need not be described using the same terms.

77. I have been informed that originally filed claims are considered in the § 112 analysis. The written description inquiry also requires that the specification

and any references incorporated by reference be evaluated together. By incorporating material by reference, the material is effectively part of the host document as if it were explicitly contained therein.

B. Obviousness

78. I have been informed that to make a prima facie showing of obviousness, a petition must, among other requirements, demonstrate that the cited references, either alone or together, disclose each element of a challenged claim.

79. I have been informed that Petitioner also has the burden to show there would have been some motivation to combine the asserted references, and that a POSITA would have had a reasonable expectation of success in making such a combination. In other words, obviousness requires more than a mere showing that the prior art includes separate references covering each separate limitation in a claim under examination. Further, even if individual modifications or choices were obvious, a petition must explain why making all of the changes at once would have been obvious. I understand that such secondary considerations include: a long-felt need, commercial success, unexpected results, praise of the invention, licensing, copying, failure of others, and skepticism by experts.

80. It is also my understanding that hindsight analysis is also inappropriate; obviousness must be measured at the time the invention was made.

Consequently, a petitioner may not use the patent as a roadmap. In other words, a petition must demonstrate a rationale to combine prior art references without relying on the patent disclosure itself.

81. I have been informed that a combination of elements in a patent claim may be obvious when all of those elements were known in the prior art and there was a reason for a POSITA to combine or modify the prior art to obtain the elements as claimed, with no change in their respective functions. Furthermore, I understand that a patent claim would not be obvious if a proposed modification would render the prior art unsuitable for its intended purpose or if the proposed modification would change the principle of operation of the prior art. I have also been told that a reference may "teach away" from a proposed combination where it expressly rules out or discourages such a combination.

VIII. PETITIONER HAS NOT MET ITS BURDEN FOR ANY CLAIM

82. In my opinion, neither Petitioner nor Dr. Mowry have demonstrated that a POSITA would have found the challenged claims obvious on any ground.

A. The Challenged Claims Are Entitled to At Least a September 25, 2012 Priority Date

83. In my opinion, the challenged claims have written-description support in U.S. Patent Application Nos. 13/624,906 and 13/626,635, both of which the '520 patent claims priority to and incorporates by reference in their entirety. For

purposes of this proceeding, Patent Owner relies on the September 25, 2012 filing date of the '635 application, which incorporated by reference the entire contents of the '906 application. With this declaration, I have provided exemplary supporting citations for each challenged claim.

84. In a prior litigation, Nos. 20-cv-00251-ADA and 21-cv-758-ADA (W.D. Tex.) (the "Dropbox litigation"), Patent Owner alleged that Dropbox infringed predecessor patents to the '520 patent. In that litigation, Dropbox filed a motion for partial summary judgment that those predecessor patents are not entitled to the September 25, 2012, priority date. I provided a signed declaration (EX2011) in support of Patent Owner's opposition (EX2010). I was also deposed. (EX2012.) The District Court denied Dropbox's motion. (EX2013.) Attached hereto is an appendix providing support for the '520 patent's September 25, 2012, priority date. (Appx. 1.)

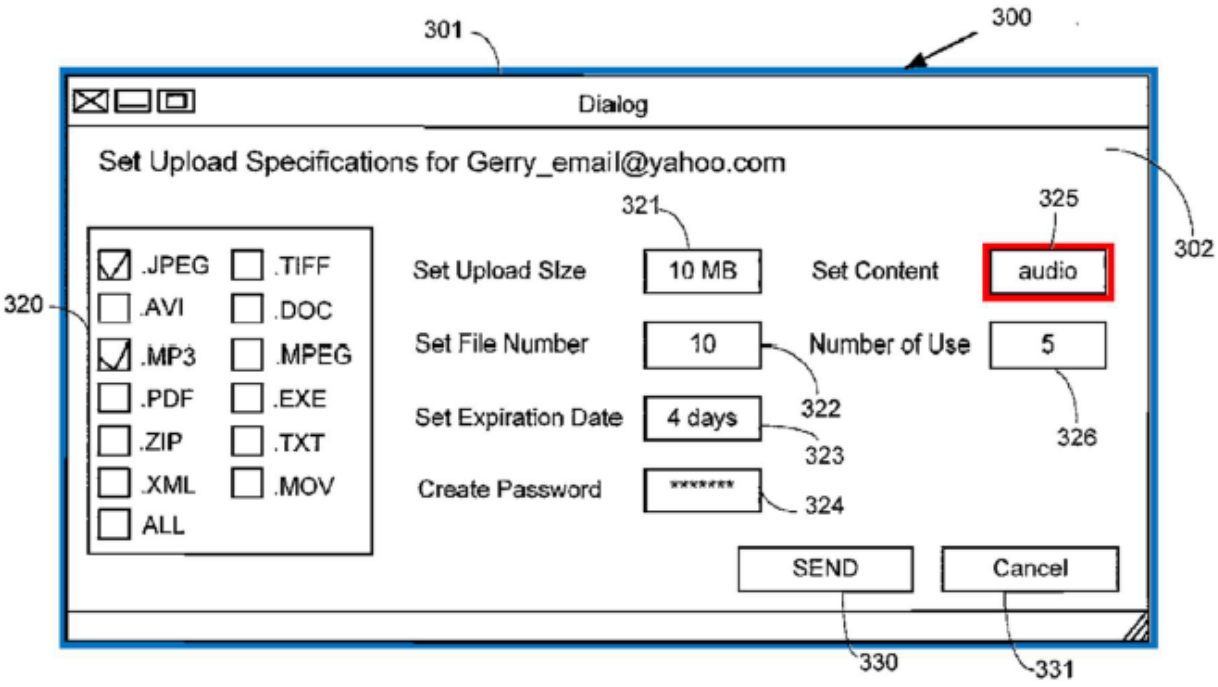
85. Therefore, in my opinion, the challenged claims are entitled to a priority date as least as early as September 25, 2012.

B. Grounds 1 and 3: Petitioner Has Failed to Show that Claims 9, 10, 14, and 15 Are Obvious Over *Riepling* or *Riepling* and *Meisels*

86. In my opinion, Petitioner has failed to show that *Riepling* alone or in combination with *Meisels* renders claims 9, 10, 14 and 15 obvious.

1. *Riepling* Does Not Teach Claim Element 9[b] Because It Does Not Teach "a file description via the first user interface element that includes a textbox, utilizing the first interface, the file description describing the at least one file in connection with the file request"

87. I understand that Dr. Mowry believes that *Riepling* teaches claim element 9[b]. (EX1102 ¶ 70.) I disagree. *Riepling*'s Figure 3 shows a "**dialog 301**" with an "upload parameter interface 302." Within that upload parameter interface 302, *Riepling* shows—but does not describe—what Petitioner calls "**content field 325**":



(EX1107 at Fig. 3 (annotated).) Petitioner argues that this "**content field 325**" in Figure 3 teaches claim element 9[b]'s "file description via the first user interface element that includes a textbox, . . . the file description describing the at least one file in connection with the file request." (Pet. at 18–19.) But *Riepling* does not teach that Petitioner's "**content field 325**" is a file description, nor does *Riepling* teach anything about it at all.

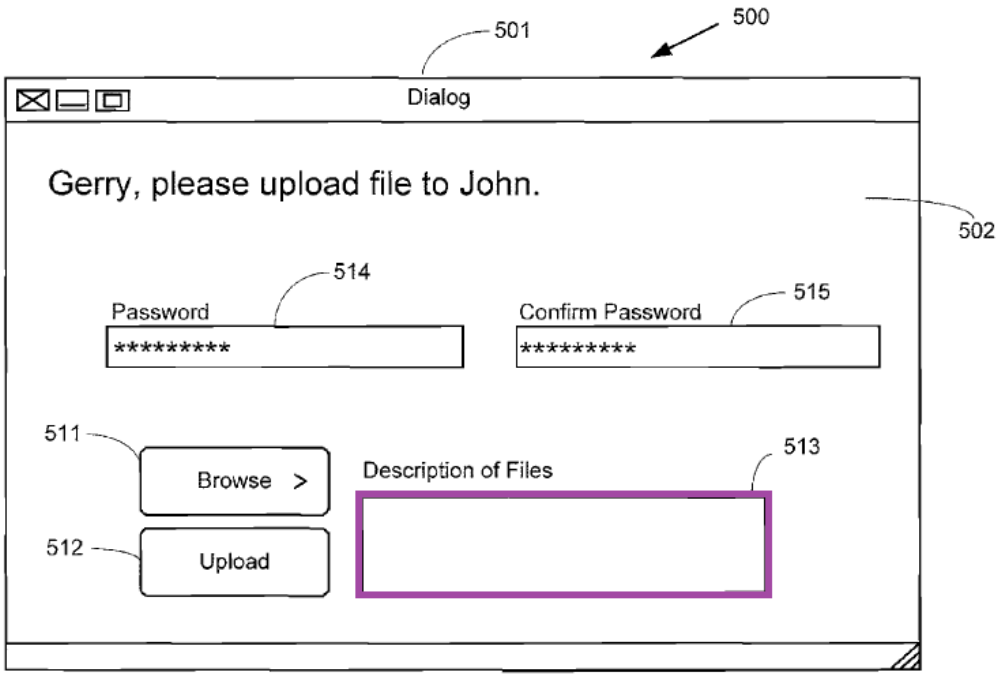
88. First, *Riepling* does not describe Figure 3's "**325**" anywhere in its specification. At most, *Riepling* provides two words next to it in Figure 3: "Set Content." (EX1107 at Fig. 3.) The phrase "Set Content" does not appear in the specification either.

89. Second, the word "audio," which is inside "**content field 325**," appears only twice in the specification, both times describing the MP3 audio format. (EX1107 ¶¶ 30, 32.) Neither the "**content field 325**" nor its contents are mentioned in the specification outside of being shown in Figure 3.

90. Third, *Riepling* does not teach that "**content field 325**" includes a textbox. At most, *Riepling*'s paragraph 32 of *Riepling* notes that "upload parameter interface 302" can ensure that "files uploaded into their registered user accounts contain only a single type of data." (EX1107 ¶ 32.) But it does not specify that "**content field 325**" includes a "textbox" (e.g., it could be a dropdown

list or other type of selection area), or "describ[es] the at least one file[.]" In fact, in Figure 3, "**content field 325**" includes the word "audio" while field 320 has a checkmark next to ".JPEG," which is a file format that does not include audio. Contrary to Dr. Mowry's opinion in Paragraph 70 of his declaration, a POSITA would not have understood field "**325**" to "describ[e] the at least one file" being requested—it does not even describe the checked box in the same file-request interface.

91. Fourth, *Riepling* teaches that file descriptions are included in the response to the file request instead of the file request itself. This is shown in Figure 5, in "**file description text field 513**," labeled "Description of Files":



(EX1107 at Fig. 5, ¶¶ 41, 43.) In particular, *Riepling* states that "**[f]ile description text field 513** may receive a textual description of the one or more data files chosen for uploading." (*Id.* ¶ 43.) In other words, *Riepling*'s file-request recipient (i.e., the uploader) at the second node must add the "**file description**" for the files he is uploading, not the file requestor at the first node. But this is the opposite of the '520 patent's teaching. The '520 patent states:

> To request an attachment, a user typically sends the request as voice and/or text data in a voice and/or text message heard and/or read by another user. . . . ***The other user must interpret the request and find a file or other data object that seems to match the request.*** The other user in many cases locates a "matching" resource using a program or application other than the communications agent that received the request.

(EX1001 at 2:1–10) (emphasis added). Accordingly, the '520 patent's claimed apparatus receives the file description from the file requestor, not the file-request recipient, so that the file-request recipient can "interpret the request and find a file or other data object that seems to match the request." (*See id.*) *Riepling* does not address this problem and is otherwise silent as to Petitioner's "**content field 325**." Therefore, it is my opinion that *Riepling* does not teach claim element 9[b], and I further believe that Dr. Mowry's opinion is incorrect.

92. Nonetheless, the Board's Institution Decision found that *Riepling*'s inclusion of a file description in "the Figure 5 response" to the file request "does

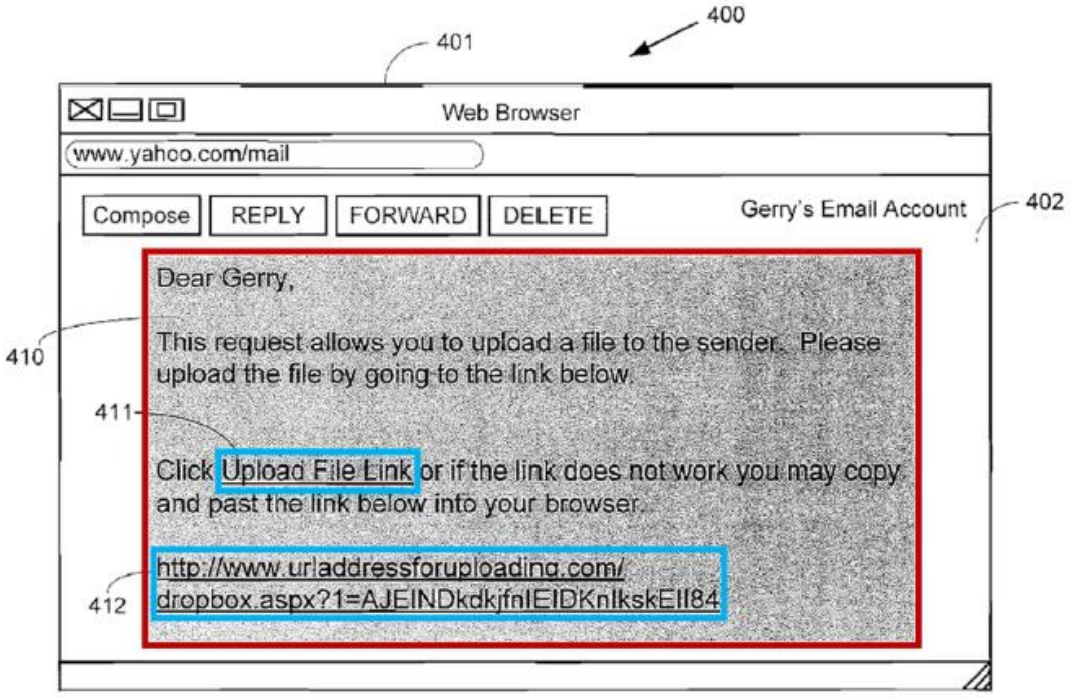
not undercut Petitioner's reliance on set content field 325" as a file description in the file request itself. (*See* Paper 11 at 29.) But the Board overlooks *Riepling*'s insistence that file descriptions be included in the response to the file request (i.e., in Figure 5) rather than the file request itself (i.e., in Figure 3). (*See* EX1107 at Fig. 5, ¶ 43.) In other words, *Riepling* does not teach a system with file descriptions in both the file request and response messages—rather, *Riepling* teaches only one "**file description text field 513**," and it appears only in the Figure 5 response. Moreover, a POSITA would not have understood Figure 3's file request to include a "file description" because "**content field 325**" says only "audio," which does not describe the requested .JPEG files because, by definition, .JPEG files do not include audio. The Board's determination makes no sense in particular when looking at claim element 9[g], which requires "a first message" that includes "the file description describing the at least one file in connection with the file request."

2. *Riepling* Does Not Teach Claim Element 9[g] Because It Does Not Teach a "first message" That Indicates "at least one of the date or the time on which the file request is based" and "the file description describing the at least one file in connection with the file request"

93. Dr. Mowry admits that *Riepling* does not teach two limitations of claim element 9[g]. (EX1102 ¶ 95 (noting that *Riepling* "does not explicitly

indicate" claim element 9[g]'s "date or the time" limitation, and "does not explicitly include" claim element 9[g]'s "file description" limitation).) Moreover, claim element 9[g] recites that its "first message" is "based on" those two missing limitations. Therefore, in my opinion, *Riepling* fails to teach claim element 9[g].

94. Dr. Mowry nonetheless opines that *Riepling*'s "**digital invitation 410**," shown in Figure 4, corresponds to the claimed "first message":

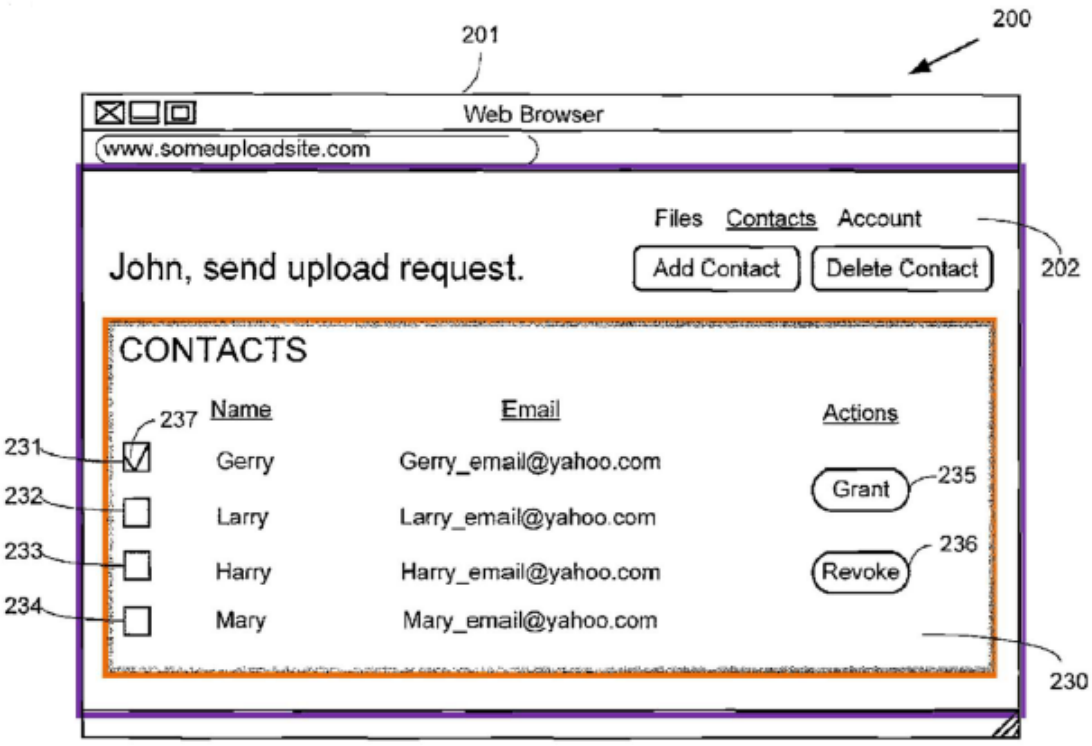


(EX1007 at Fig. 4.) I disagree. This **digital invitation 410** does not include the "date or time on which the file request is based," nor the "file description describing the at least one file in connection with the file request," nor any other information collected by the interface shown in Figure 3's file request. Indeed,

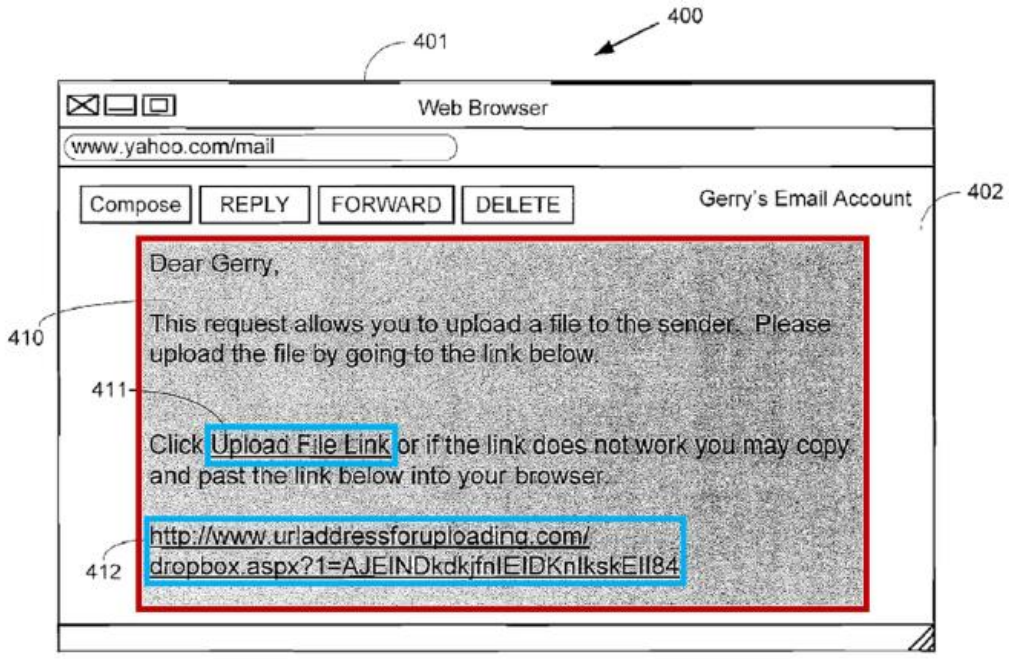
even if the text from Figure 3's **content field 325** (i.e., "audio") were considered to be a "file description," that file description does not appear in the **digital invitation 410**.

95. Nonetheless, Dr. Mowry opines that *Riepling* teaches that its **digital invitation 410** may be "restricted by the one or more upload parameters specified in upload parameter dialog 301." (EX1102 ¶ 95.) But claim element 9[g] claims more than merely "restrict[ing]" the first message: i.e., the first message must "indicat[e] . . . the date or the time on which the file request is based," and "further includ[e] . . . at least one sentence including the file description[.]" *Riepling*'s **digital invitation 410** does neither. Indeed, antecedent basis for "the file description" requires that it be not just any file description but the file description from claim element 9[b], which was absent there and is absent again here.

96. In fact, *Riepling*'s **digital invitation 410** may be generated after the user selects which recipients will receive the file request via the **request interface 202** shown in *Riepling*'s Figure 2, without regard to any information collected by *Riepling*'s file-request interface in Figure 3. (EX1107 ¶ 26 ("Once grant access button 235 has been used, content sharing service 105, using processor 102, may begin creating a digital invitation for transmission to the contact".) Yet in Figure 2's **request interface 202**, only a **list of contact information 230** is selectable:



(EX1007 at Fig. 2.) No "date or time" or "file description" is possible to add or select. Thus, in my opinion, any "digital invitation" that is created based on Figure 2's **request interface 202** would not include these limitations of claim element 9[g]. And *Riepling*'s **digital invitation 410** does not include them either:



(EX1107 at Fig. 3.)

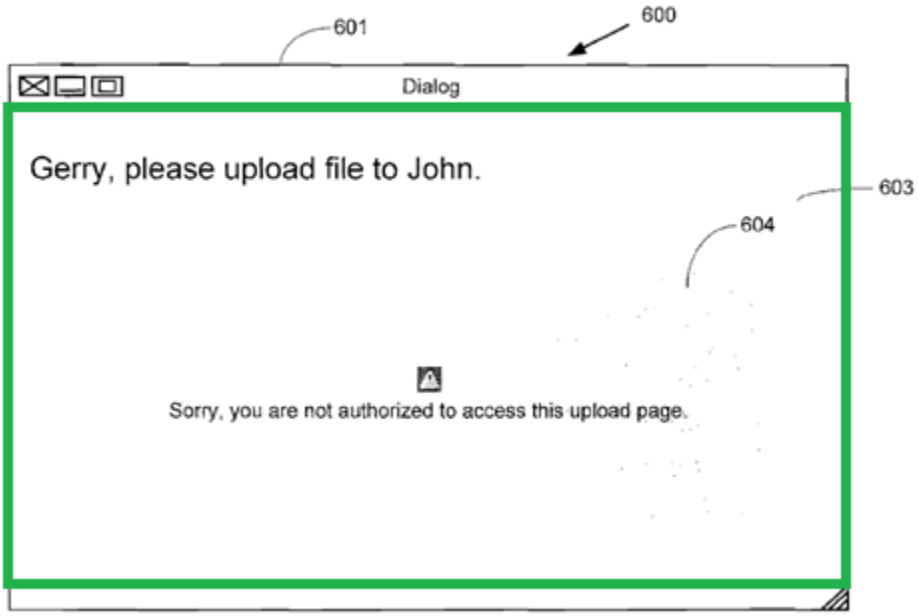
97. The Board noted that "[a]t this stage, Patent Owner appears to be incorrect in asserting that digital invitation 410 is sent after grant access button 235 [in Figure 2] is selected" (Paper 11 at 30.) But I do not understand Patent Owner to argue that selecting the grant access button 235 in Figure 2's **request interface 202** ***sends*** the **digital invitation 410**. Rather, I understand that Patent Owner argues that selecting the grant access button 235 allows the content sharing service 105 to "***begin creating***" it, as stated in *Riepling*'s ¶ 26, which is also consistent with my understanding of *Riepling*. (POPR at 28–29.) And at that point in *Riepling*, neither of claim 9[g]'s required limitations (i.e., "date or time" or "file description") would have been specified.

98. Dr. Mowry admits that *Riepling*'s "**digital invitation 410**," shown in FIG. 4, does not indicate the claimed "date or time" or "file description," but alleges that it would be "an obvious design choice." (EX1102 ¶ 96.) Yet Dr. Mowry fails to describe any motivation to modify *Riepling* in such a way or why such a design choice would be obvious. For this reason, I do not find Dr. Mowry's opinion persuasive on this point.

99. At most, Dr. Mowry cites to a line from the specification indicating that the **digital invitation 410**, shown by FIG. 4, may be "restricted by the one or more upload parameters specified in upload parameter dialog 301 [shown by FIG. 3]." (EX1102 ¶ 95, citing EX1107 ¶ 34.) But whether upload parameters are ***restricted*** has no bearing on what is ***indicated*** in the invitation. Indeed, *Riepling* accounts for restrictions using a separate interface, **blocked interface 603**:

> [A]s previously explained, some upload parameters may also place restrictions on the digital invitation. **Blocked interface 603** may occur when a user such as, for example, sender 117 of FIG. 1 may attempt to access content sharing service 105 through a unique location identifier of a revoked digital invitation.

(EX1107 ¶ 47.) **Blocked interface 603** is shown by FIG. 6:



100. In sum, it is my opinion that *Riepling* does not teach that the "date or time on which the file request is based" or "file description describing the at least one file in connection with the file request" could or would be included in **digital invitation 410**, as required by the claim element 9[g]. Instead, Petitioner uses the '520 patent as a roadmap to support its argument, which is improper hindsight. ('520 patent, EX1001, at 2:1–10.) Notably, Petitioner filed patent applications on file-request functionality years later, which suggests that the file-request inventions of the '520 patent were not obvious. (*See*, *e.g.*, U.S. Patent No. 10,713,966, EX2003.)

101. I do not find Dr. Mowry's opinion persuasive on any of these points. First, the "file description" could not have been included in *Riepling*'s invitation

because it does not exist at the request stage, as described above. Second, the digital invitation is created before the upload parameters are set, and there is otherwise ***no indication in the specification*** that any such parameters are included in an invitation. Indeed, the message sent requesting a file from a contact does not include the "file description" in the message or an indication of the date/time on which it is based. Petitioner and Dr. Mowry fail to adequately describe how or why such a design choice would be obvious given *Riepling*'s shortcomings.

3. *Riepling* Does Not Teach Claim Element 9[l] Because It Does Not Teach "a second message that includes an eighth user interface element, and that does not include a file attachment with the second message"

102. Dr. Mowry admits that *Riepling* does not disclose this element. (EX1102 ¶ 109). Nonetheless, he cites to *Riepling*'s description that, "one or more user accounts may receive . . . [a] notification message alerting the other accounts of the data upload. . . . The notification messages may permit the downloading of the uploaded data through the one or more user accounts." (EX1101 ¶ 111.) But nothing in these statements, or anywhere else in *Riepling,* teaches or suggests that any such "notification message" includes "an eighth user interface element" ***and*** "does not include a file attachment," as required by the claim language.

103. Petitioner dismisses this as obvious because "*Riepling* expressly describes including 'unique location identifier 411,' which may be a clickable link,

in its digital invitation 410," and so "[a]ccordingly, it would have been obvious to a POSITA to implement *Riepling*'s notification message the same way[.]" (EX1101 ¶ 111.) I disagree. When *Riepling*'s initial digital invitation is created and sent, no files have been uploaded. So no files exist that could be accessed or included as attachments. Moreover, the initial digital invitation is being sent to a person that ***does not have a registered user account.*** (EX1107 ¶¶ 21, 29, 30, 56.) The alert message is provided to ***a "user account"*** in contrast. (EX1107 ¶ 45.) The opposite is true ***after*** files have been uploaded, and the specification is silent regarding the contents of any "notification message."

104. In my opinion, the specification is explicit that a user must access uploaded files through a specific interface: "Once a data upload has been successfully performed, the owner of the registered user account may view the uploaded data by accessing data availability interface 702[.]" (EX1107 ¶ 49.) There is no indication that this interface may be accessed via a "notification message."

105. At best, the specification indicates that receipt of a notification "may ***permit*** the downloading of the uploaded data ***through the one or more user accounts***." (EX1107 ¶ 45.) Such permission is given by "the owner of the registered user account" that created the file request, as this owner "may specify

which other user accounts are to receive the notifications." (EX1107 ¶ 45.) That a notification ***indicates permission*** to access an uploaded file does not mean that the notification message ***includes any elements***, let alone the "eighth user interface element" required by the claim language. In particular, the language of claim 9 requires that the "eighth user interface element" may be selected such that "in response to the receipt of the indication of the selection of the eighth user interface element of the second message," the claimed apparatus "cause[s], at the first node, display of at least one fourth interface with a reference to the at least one file for allowing access to the at least one file at the first node." (*See* claim element 9[o], Pet. at 86–87.) No such teaching is found in *Riepling*.

106. Though Dr. Mowry opines that "it would have been obvious to a POSITA to implement *Riepling*'s notification message the same way that *Riepling* expressly taught for the digital invitation 410" (EX1102 ¶ 109), Dr. Mowry fails to describe ***any*** motivation to modify *Riepling* in such a way or why such a design choice would be obvious.

107. Therefore, I disagree with Dr. Mowry, and further believe that *Riepling* does not teach or suggest this claim element, and it would not be obvious.

4. *Riepling* Does Not Teach Claim 10

108. Claim 10 depends on claim 9. In my opinion, for at least the reasons set forth above with respect to claim 9, it is my opinion that Petitioner has not put forward sufficient evidence that *Riepling* renders claim 10 obvious.

5. *Riepling* Does Not Teach Claim 14

a. *Riepling* Does Not Teach Claim Element 14[b]

109. In my opinion, *Riepling* does not teach or suggest this claim element for at least the same reasons that *Riepling* does not teach or suggest the "file description" aspects described above for element 9[b].

b. *Riepling* Does Not Teach Claim Element 14[d]

110. In my opinion, *Riepling* does not teach or suggest this claim element for at least the same reasons that *Riepling* does not teach or suggest the aspects described above for element 9[g].

c. *Riepling* Does Not Teach Claim Element 14[j]

111. In my opinion, *Riepling* does not teach or suggest this claim element for at least the same reasons that *Riepling* does not teach or suggest the aspects described above for element 9[l].

6. *Riepling* Does Not Teach Claim 15

112. Claim 15 depends on claim 14. In my opinion, for at least the reasons set forth above with respect to claim 14, Petitioner has not put forward sufficient evidence that *Riepling* renders claim 15 obvious.

113. In addition, claim 15 recites that "the first message and the second message are email messages." Dr. Mowry admits that this claim element is not taught by *Riepling*, instead asserting it to be "conventional" because email "is so popular." (EX1102 ¶ 138.) Regardless of email's so-called "popularity" at the time of invention, *Riepling*'s alerts are sent only "within [*Riepling's*] content sharing service," ***not*** via email. (EX1107 ¶ 45.) This results in a situation such that the requesting, ***registered*** user will ***never*** receive an email response from a requestee, ***unregistered*** user. (EX1107 ¶¶ 21, 29, 30, 56.) *Riepling* makes no allowances or exceptions in this regard, and therefore, it is my opinion that *Riepling* does not teach or make obvious this element of claim 15.

114. In my opinion, Petitioner has therefore failed to demonstrate that claims 9, 10, 14, and 15 are rendered obvious over *Riepling*. Therefore, Ground 1 of the Petition fails.

7. For Ground 3, *Meisels* Does Not Cure *Riepling*'s Deficiencies

115. Ground 3 relies on both *Riepling* and *Meisels* for claims 9, 10, 14 and 15, but *Meisels* fails to disclose the missing elements noted above. Indeed, Petitioner does not even assert that *Meisels* discloses claim elements 9[b], 9[g], or 15 and purports to rely on *Meisels* only for elements 9[l] and 14[j]. (Pet. at 65–66.) Otherwise, Petitioner relies on its arguments in Ground 1 regarding all other required claim elements of independent claims 9 and 14 and dependent claims 10 and 14. (Pet. at 59–64.) As described above, it is my opinion that Petitioner's arguments fall short because *Riepling* does not teach or suggest every element of independent claims 9 and 14, or their dependent claims 10 and 15. Therefore, I believe that Ground 3 of the Petition fails.

116. I have already discussed claim element 9[l] above. *Meisels* "discloses an electronic message with an automatically inserted [] link to a document in place of the document attachment itself." (Paper 11 at 21.) But *Meisels*'s process is divorced entirely from any other interfaces or software processes. A user must manually create an email, manually confirm to upload an attachment to insert a link to the attachment, and manually send the email to its recipient. (e.g., EX1108 ¶¶ 3, 37, 59.) Then, selecting the link merely retrieves the attachment. (EX1108 ¶ 23 ("Using the link generally retrieves the content that is pointed to by the link.").)

117. In contrast, the claimed "second message" of claim element 9[l] is generated "***in response to the receipt of the indication of the selection of the seventh user interface element***" and "includes an eighth user interface element" such that the claimed apparatus will, "in response to the receipt of the indication of the selection of the eighth user interface element of the second message, ***cause, at the first node, display of at least one fourth interface with a reference to the at least one file for allowing access to the at least one file at the first node***." (claim elements 9[l] and 9[o].) Likewise, in claim element 14[j], the "claimed second message" of claim 14 element 14[j] is generated "***in response to the receipt of the message associated with the selection of the fifth user interface element"*** and "in response to the receipt of the message associated with the selection of the sixth user interface element of the second message, send a second communication over the at least one network that ***results in display, at the first node, of a reference to the at least one file***" and "allow access, at the first node, to the at least one file via the reference." (claim elements 14[m]–[n].) There is no indication in *Meisels* of generating a message "in response to" anything, let alone that of the claimed "user interface element[s]." There is also no indication in *Meisels* that selection of one of its inserted links can "cause . . . display of at least one fourth interface with a

reference to" one of its uploaded attachments "for allowing access" to that attachment.

118. In my opinion, *Riepling* does not make up for these deficiencies, and therefore, *Riepling*, alone or in combination with *Meisels*, does not teach or suggest every element of claims 9 and 14 does not render them obvious.

C. Grounds 3 and 4: Petitioner Has Failed to Show that Claims 9, 10, 14, and 15 Are Obvious Over *Riepling* and *Meisels*, and that Claims 11–13, 16, 22–24 Are Obvious over *Riepling*, *Manzano*, and *Meisels* under 35 U.S.C. § 103

119. Petitioner purports to rely on *Meisels* only for elements 9[l], 14[j], and 22[i]. (Pet. at 65–66.) Otherwise, Petitioner relies on its arguments in Grounds 1 and 2 regarding all other required claim elements of independent claims 9, 14, and 22. (Pet. at 59–64.) As described above with respect to Grounds 1 and 2, I do not believe that *Riepling*, alone or in combination with *Meisels* and/or *Manzano*, teaches or suggests every element of independent claims 9, 14, and 22, or their dependent claims 10–13, 15–16, 23, and 24. Therefore, I believe that Grounds 3 and 4 of the Petition fail.

120. I have already discussed claim element 9[l] above. *Meisels* "discloses an electronic message with an automatically inserted [] link to a document in place of the document attachment itself." (Paper 11 at 21.) But *Meisels*'s process is divorced entirely from any other interfaces or software processes. A user must

manually create an email, manually confirm to upload an attachment to insert a link to the attachment, and manually send the email to its recipient. (e.g., EX1108 ¶¶ 3, 37, 59.) Then, selecting the link merely retrieves the attachment. (EX1108 ¶ 23 ("Using the link generally retrieves the content that is pointed to by the link.").)

121. In contrast, the claimed "second message" of claim element 9[l] is generated "***in response to the receipt of the indication of the selection of the seventh user interface element***" and "includes an eighth user interface element" such that the claimed apparatus will, "in response to the receipt of the indication of the selection of the eighth user interface element of the second message, ***cause, at the first node, display of at least one fourth interface with a reference to the at least one file for allowing access to the at least one file at the first node***." (claim elements 9[l] and 9[o].) There is no indication in *Meisels* of generating a message "in response to" anything, let alone that of the claimed "user interface element[s]." There is also no indication in *Meisels* that selection of one of its inserted links can "cause . . . display of at least one fourth interface with a reference to" one of its uploaded attachments "for allowing access" to that attachment.

122. In my opinion, *Riepling* does not make up for these deficiencies, and therefore, *Riepling*, alone or in combination with *Meisels*, does not teach or suggest every element of claim 9 and does not render claim 9 obvious.

D. Grounds 2, 4, 5 and 6: Petitioner Has Failed to Show that Claims 11–13, 16, and 22–24 Are Obvious over any combination of *Riepling*, *Manzano*, *Meisels* and *Garcia* under 35 U.S.C. § 103

123. With regard to claims 11–13 and 16, and 22–24, it is my opinion that Petitioner has failed to show that the combination of *Riepling, Manzano, Meisels*, and *Garcia* makes obvious each and every element of the challenged claims.

1. These Grounds Fail for the Same Reasons as Grounds 1 and 3 with Respect to *Riepling* and *Meisels*

a. Claims 11–13 and 16

124. Petitioner purports to rely on *Manzano* only for elements of dependent claims 11–13 and 16, which rely on either independent claim 9 (claims 11–13) or independent claim 14 (claim 16) (Pet. at 49.) Likewise, Petitioner purports to rely on *Garcia* only for claim elements of dependent claims 11 and 16. (Pet. at 71–72.) Petitioner does not rely on either *Manzano* or *Garcia* for independent claims 9 or 14, which, as set forth above, are not obvious over *Riepling* alone or in combination with *Meisels*. Therefore, for the reasons set forth above with respect to Grounds 1 and 3, it is my opinion that Petitioner has not demonstrated that claims 11–13 and 16 are rendered obvious over *Riepling* alone or in combination with *Meisels* in further view of *Manzano*, or both *Manzano* and *Garcia*.

b. Claims 22–24

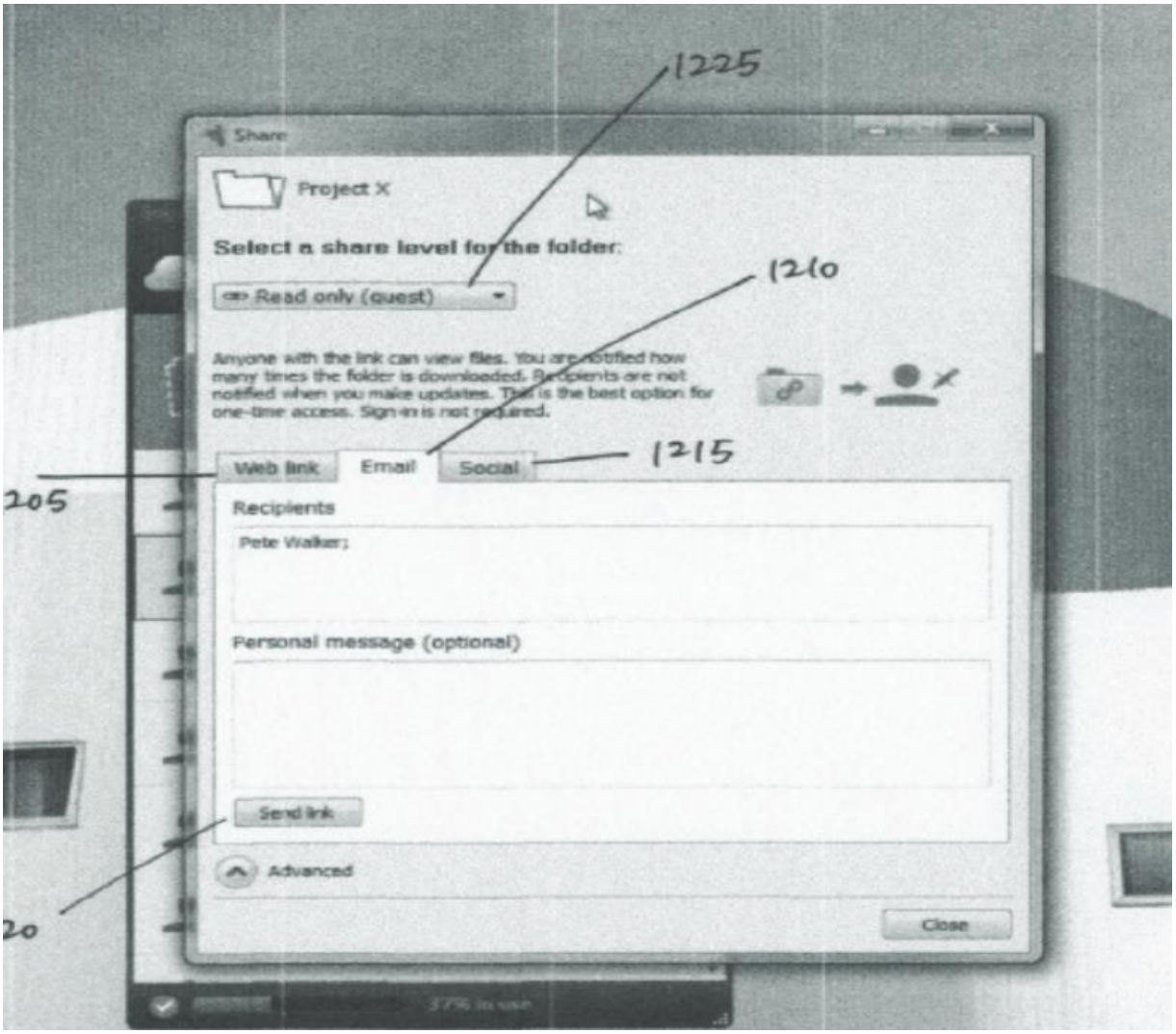
125. Petitioner purports to rely on *Manzano* and *Garcia* only for elements 22[h], 22[j], and 22[k]. (Pet. at 59–64, 71–72.) Otherwise, Petitioner relies on *Riepling* and Petitioner's arguments regarding independent claim 9 for independent claim 22. (Pet. at 59–64.) For example, with respect to claim element 22[a], Petitioner relies on its arguments for claim elements 9[a] and 9[b]. (Pet. at 59.) With respect to claim element 22[e], Petitioner relies on its arguments for claim elements 9[g] and 9[h]. (Pet. at 60.) With respect to claim element 22[i], Petitioner relies on its arguments for claim elements 9[l], 9[n], and other elements that also depend on the arguments for claim elements 9[l] and 9[n]. (Pet. at 63.) For at least the reasons described above with respect to Grounds 1 and 3, it is my opinion that *Riepling* does not teach or suggest all of these elements of claim 9, including for example, the "file description" aspects for element 9[b], 9[g], and/or 9[l] described above and neither *Meisels*, *Manzano* nor *Garcia* cure those deficiencies. Therefore, I opine that Grounds 2, 4, 5 and 6 fail for the same reasons.

2. Grounds 2, 4, 5 and 6 Also Fail Because No Reference Discloses the Creation of a Representation of the at Least one File in a Location Among One or More Folders on the File Explorer Interface as Claims

126. Claims 11–13, 16 and 22–24 require the capability to cause "a creation of a representation of the at least one file is caused, at the first node, in a location among one or more folders on a file explorer interface." Petitioner does not assert this is taught by *Riepling* and it is not taught by *Manzano*. *Manzano* discloses that a recipient client is instructed "to set up an electronic file folder into which the transferred file will be place." (EX1105 ¶ [0041].) It does not disclose a representation of a file being created in a location among one or more folders on a file explorer interface as recited in claims 11–13, 16 and 22–24. Indeed, *Manzano* only discloses transferring ***files*** into a specific folder that is created for the files. (*E.g.*, EX1105 ¶¶ [0041], [0052].) It does not disclose the creation of a representation in a location among one or more folders on the client's client-based file explorer application. (*Id.*)

127. Petitioner does not assert this limitation is taught by *Meisels* nor is it taught by *Garcia*. *Garcia* discloses that the recipient of a shared folder ***does download*** the ***contents*** of the shared folder at the node where the folder share invitation is received. That the user's ***other devices*** (e.g., different nodes) may show representations of the shared folder is beside the point. *Garcia* does not

disclose the creation of a representation at the node where the invitation to the shared folder is received. *Garcia* discloses "in one embodiment, when a user drags a folder to a particular contact, ***the folder*** or ***link to the folder*** is automatically sent (via email, text, or other means) to the contact," (EX1104, 11:16–19 (emphasis added)), and "when the contact clicks on the link to folder 115, the ***contents*** of the folder are shared with the contact," (EX1104 at 11:29–31 (emphasis added).) (*See also* EX1104 at 10:32–43 (web link to download).) A higher resolution image of Figure 12 available on PAIR is reproduced below and further demonstrates that the recipient of the folder share ***downloads*** the contents (i.e., files) in response:



(EX2006.) As shown above, the window shown in *Garcia* Figure 12 has "email" selected and states "Anyone with the link can view files. You are notified how many times the folder is downloaded." (*Id.*) At the bottom, there is a button stating "Send Link." (EX2006.) Thus, *Garcia* discloses that the recipient of the shared folder receives a link and downloads the ***contents*** of the folder (*i.e.*, the files) when the link is accessed. It does ***not*** disclose the creation of a

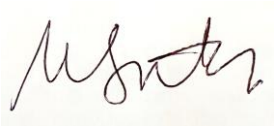
representation of the folder or any representation of the files at the receiving node. For this reason, it is my opinion that Petitioner has not shown that claims 11–13, 16, or 22–24 are unpatentable under any Ground.

128. Therefore, in my opinion, Petitioner's remaining Grounds fail for these additional reasons.

IX. RESERVATION OF RIGHTS

129. My opinions are based upon the information that I have considered to date. I reserve the right, however, to supplement my opinions in the future to respond to any arguments raised by Petitioner and to consider new information that becomes available to me.

130. I declare under the penalty of perjury that all statements made in this Declaration are true and correct.



Dr. Michael Smith

APPENDIX SLIP PAGE FOLLOWED BY PRIORITY CHART

Appendix: 1 Exemplary Written Description for Claims 9-16 and 22-24 of U.S. Patent No. 11,611,520 in Original Applications[1]

Claim 9	Support in Original Applications
An apparatus, comprising: at least one non-transitory memory storing instructions; and one or more processors in communication with the at least one non-transitory memory, wherein the one or more processors execute the instructions to:	**'906 application (EX2004)** [0007] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.") [0033] ("The execution environment is said to be the execution environment of the device and/or devices. An execution environment may be and/or may include a virtual execution environment including software components operating in a host execution environment. Exemplary devices included in and/or otherwise providing suitable execution environments for configuring according to the subject matter include personal computers, notebook computers, tablet computers, servers, portable electronic devices, handheld electronic devices, mobile devices, multiprocessor devices, distributed systems, consumer electronic devices, routers, communication servers . . . .")

[1] This claim chart shows exemplary support in the '635 and '906 applications. These examples are not limiting. Additional support may be found in the Original Applications. I reserve the right to supplement and modify my opinions and citations.

	[0039] ("Execution environment 102 may include software components stored in persistent secondary storage 108, in remote storage accessible via a network, and/or in a processor memory.") [0102] ("[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network") [0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0162] ("FIG. **7** includes a dataflow message **708** that illustrates a detect data object identification request dataflow that may be internal to execution environment **401** of second node **504** and/or may include interoperation with another node via a network.")

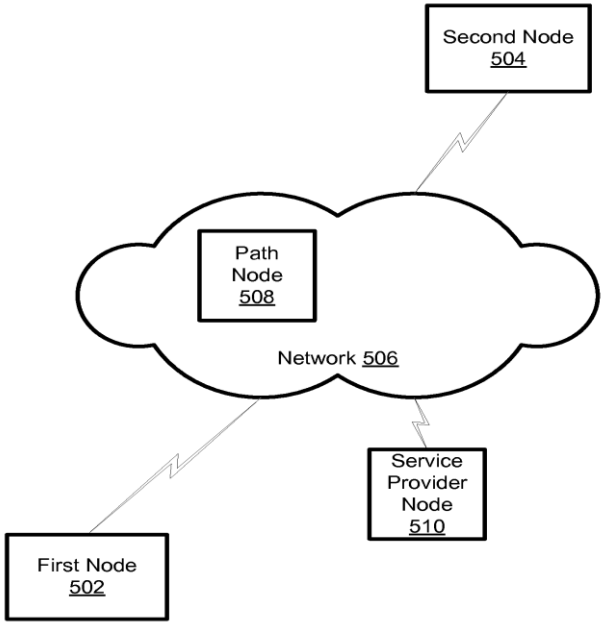
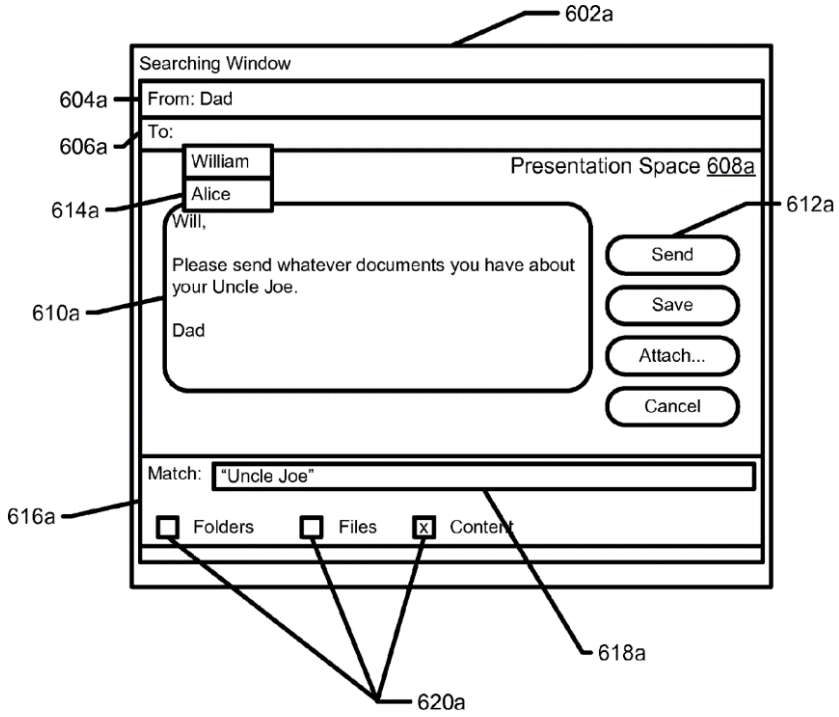


FIG. 5

See also FIGs. 1, 4, and 5; [0008]-[0011], [0033]-[0050] (description of FIG. 1 components, UIs generally, and web pages having UI elements); [0052], [0053], [0099]-[0101] (components of FIGs. 1 , 3A, 3B, and 4); [0103] (FIG. 5 description of nodes in a network); [0111], [0115]-[0117], [0171], [0209], [0210]; Original claims 1-22

’635 application (EX2002)
FIG. 1; [0193]

cause, at a first node, display of a first interface with a first user interface element and a second user interface element, the first interface for use in creating a file request requesting at least one file;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6A [0126] (“DOI UI element 616a illustrates a user interface for a keyword based schema. Match textbox UI element 618a illustrates a textbox allowing a user to enter a keyword expression.

	Various checkbox UI elements 620a allow a user to define a scope of a data object identification request by identifying whether a resulting data object identification request is to be applied to folders, files, and/or content of files in a file system and/or other data store of a receiving execution environment.") [0047] ("The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as 'UI elements'. More specifically, visual outputs of a user interface are referred to herein as 'visual interface elements'. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. . . .") [0050] ("For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application") [0111] ("FIG. 6A-G illustrates various communications agent windows 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. In FIG. 6a, a communications agent window 602a includes a contactor user interface (UI) element 604a for presenting an identifier of a communicant in the role of a contactor in a communication represented by the communications agent window 602a. A communications agent window 602a also includes a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) included in the communication. A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more UI controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.")

	[0117] (“Data may be sent in a communication in response to a communicant input. A contactor may provide an input corresponding to send UI element 612a in FIG. 6A. The input may be received by presentation controller component 417 and/or one or more UI element handlers 415 corresponding to send UI element 612a. In response to detecting the input, presentation controller component 417 may provide data to be sent in the communication to one or more content handler components 413 according to the content type(s) of the data to be sent.”) [0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”) [0125] (“FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415 may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether data object information and/or a data object identification request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing data object information and/or data object identification requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language

	(SOL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0175] ("In another aspect, a data object identification request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object identification request UI element and input from the user, query director component 412 may receive one or more additional data object matching criterion information and/or may receive input to modify a data object matching criterion identified in a received data object identification request. Query director component 412 may locate one or more data objects in response to receiving the additional criterion.") *See also* [0006]-[0011], [0020], [0100], [0111], [0115]-[0117]; Fig. 5, Fig. 6A (e.g. Match window 616a); [0125]-[0126]; [0184]; Original claims 1-22 **'635 application (EX2002)** [0049] ("The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as 'UI elements'. More specifically, visual outputs of a user interface are referred to herein as 'visual interface elements'. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. An application interface may include one or more of the elements listed. Those skilled in the art will understand that this list is not exhaustive. The terms 'visual representation', 'visual output', and 'visual interface element' are used interchangeably in this document. Other types of UI elements include audio outputs referred to as 'audio interface elements', tactile outputs referred to as 'tactile interface elements', and the like.") [0051] (""A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a

	display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”)

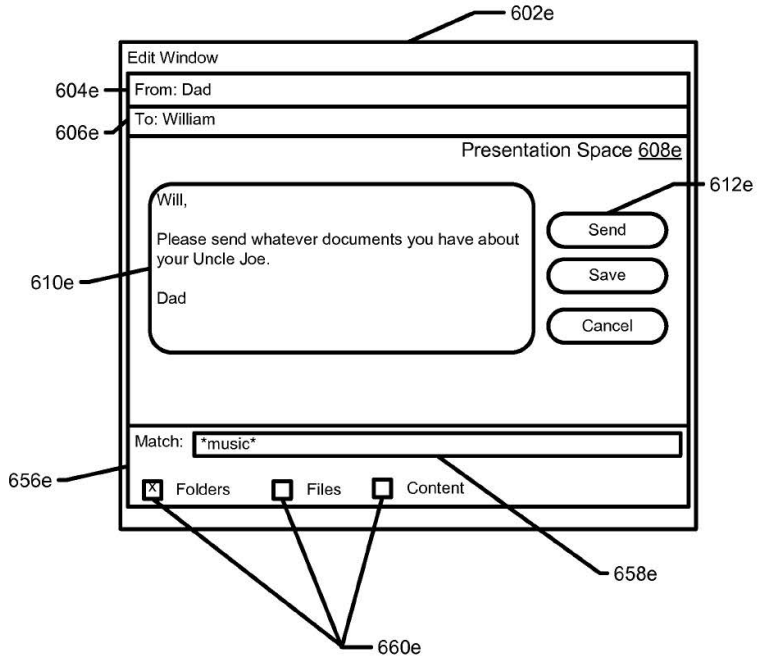
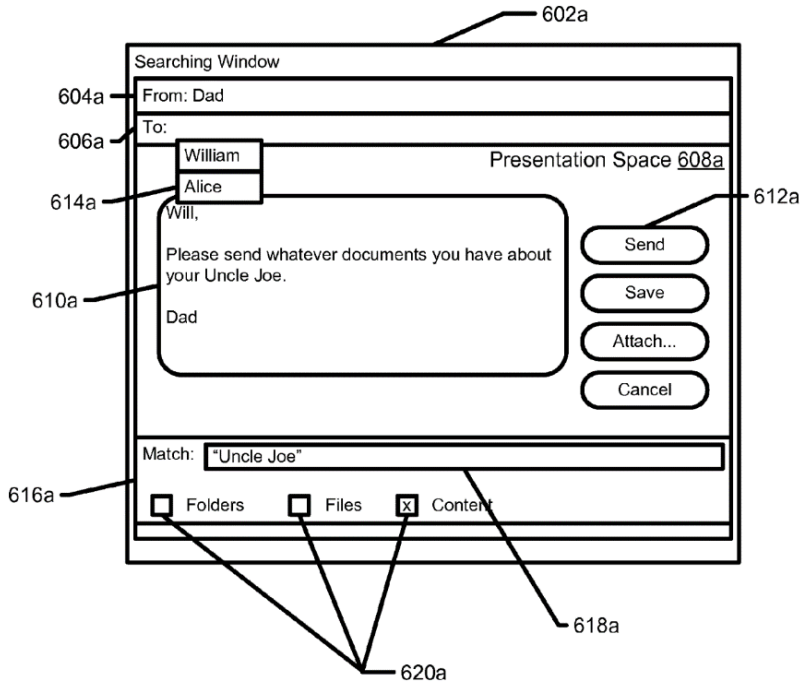


FIG. 6E

[0145] ("MI UI element 656e illustrates a user interface for a keyword based schema. Match textbox U I element 658e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema.")

	[0144] ("FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (Ml) UI element 656e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in Ml UI element 656e. MI U I element 656e and one or more U I elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0146] ("FIG. 7 includes data flow message that illustrates a create data object request (CIReq) data flow 714 that may be internal to execution environment 401 of first node 502 and/or may include a message received via network 506 by first node 502. With respect to FIG. 4, CIReq data flow 714 may illustrate a data flow to an IMC component 410 operating in execution environment 401 of first node 502. Match information may be identified to and/or otherwise provided to IMC component 410.") [0153] ("In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a group, a requester, a relationship including a requesting user and a responding user, a keyword , a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.") *See also* FIGs. 7, 8B, [0142]-[0143], [0147]-[0152], [0172], Original claim 12

receive, from the first node via at least one network, a file description via the first user interface element that includes a	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting.

textbox, utilizing the first interface, the file description describing the at least one file in connection with the file request;	**'906 application (EX2004)**  FIG. 6A [0126] ("DOI UI element 616a illustrates a user interface for a keyword based schema. Match textbox UI element 618a illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 620a allow a user to define a scope of a data object identification request by identifying whether a resulting data object identification request is to be applied to folders, files, and/or content of files in a file system and/or other data store of a receiving execution environment.")

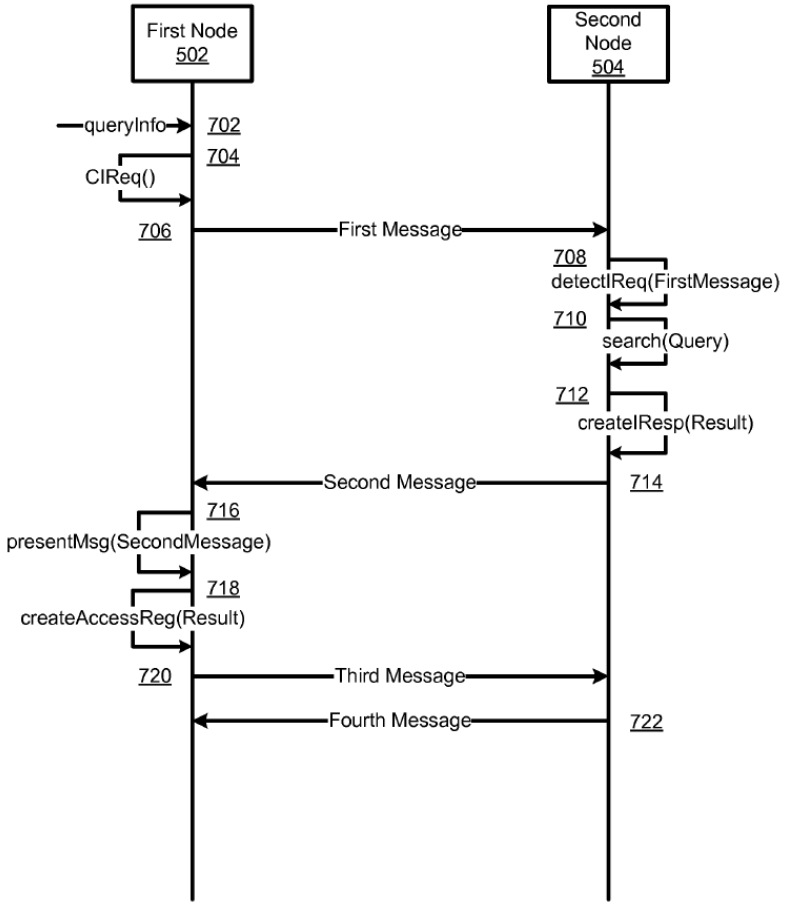


FIG. 7

[0123] ("FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node

	502. Data object information is identified to and/or otherwise received by query handler component 402.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0133] ("FIG. 8A illustrates an 'application/id-request' MIME type identifier that may be defined to identify a schema for an XML-based language for specifying id-request XML documents. FIG. 8A illustrates id-request document 806a. Id-request document 806a, as illustrated, includes criterion tag elements 808a corresponding to the form elements in the user interface illustrated in FIG. 6A. A criterion tag element 808a identifies a data object matching criterion name, such as 'query' indicating that the criterion is a query expression or a portion thereof. A 'type' attribute identifies a schema with the identifier "keyword" for the expression. The criterion tag element 808a identifies a value for a query 'Uncle Joe' in a match-expression attribute.") *See also* [0100], [0111]-[0117]; Fig. 6A (text box 618a); [0125]-[0126]; FIGS. 2A, 2B; FIG 8A; Original claims 1-22 **'635 application (EX2002)** [0049] ("The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as 'UI elements'. More specifically, visual outputs of a

	user interface are referred to herein as 'visual interface elements'. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. An application interface may include one or more of the elements listed. Those skilled in the art will understand that this list is not exhaustive. The terms 'visual representation', 'visual output', and 'visual interface element' are used interchangeably in this document. Other types of UI elements include audio outputs referred to as 'audio interface elements', tactile outputs referred to as 'tactile interface elements', and the like.") [0051] (""A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A "program entity" is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as "presentation information". Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described

herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.")

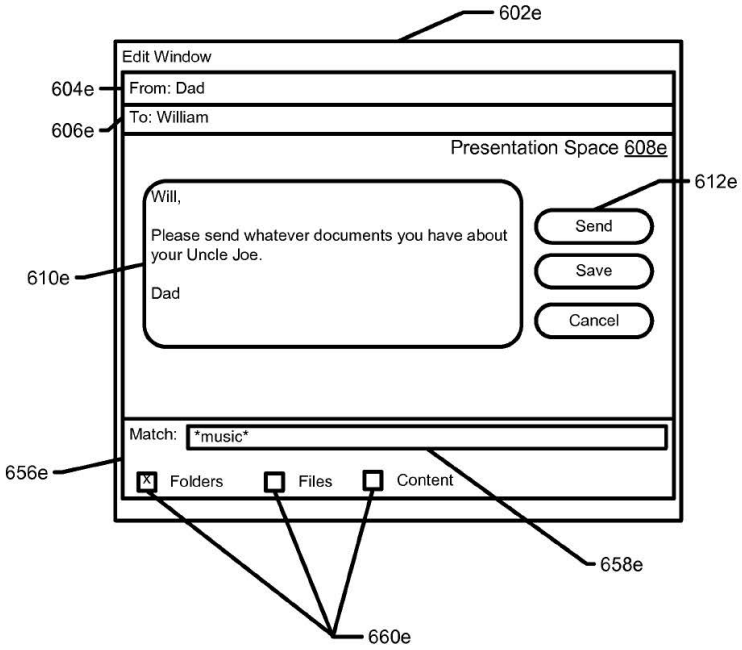


FIG. 6E

[0145] ("MI UI element 656e illustrates a user interface for a keyword based schema. Match textbox U I element 658e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or

	content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema.") [0144] ("FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (Ml) UI element 656e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in Ml UI element 656e. MI U I element 656e and one or more U I elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0146] ("FIG. 7 includes data flow message that illustrates a create data object request (CIReq) data flow 714 that may be internal to execution environment 401 of first node 502 and/or may include a message received via network 506 by first node 502. With respect to FIG. 4, CIReq data flow 714 may illustrate a data flow to an IMC component 410 operating in execution environment 401 of first node 502. Match information may be identified to and/or otherwise provided to IMC component 410.") [0153] ("In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a

	group, a requester, a relationship including a requesting user and a responding user, a keyword , a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.") *See also* FIGs. 7, 8B, [0147]-[0152], [0172], Original claim 12
receive, from the first node via the at least one network, an indication of at least one of a date or a time via the second user interface element, utilizing the first interface;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)**

	[0180] (“The methods illustrated in FIG. 2A-B may include additional aspects supported by various adaptations and/or analogs of the arrangement of components in FIG. 3A-B. In various aspects, a data object identification request may be based on a date, a time, a length of time . . . .”) [0093] (“In specifying a resource scope, one or more criterion may be specified. The terms "criterion" and "constraint" are used interchangeably herein, with respect to a resource scope. A criterion included in specifying a resource scope is referred to herein as a "scope criterion". A scope criterion may be expressed via any suitable manner, such as a file name, a wildcard expression, a regular expression, a data store location, a data base query, geospatial coordinates**,** a time, a date, a duration, a relationship to a user, and/or an address - too name a few examples. With respect to a tag, a scope criterion included in defining a resource scope differs from a tagging criterion for the tag in that the scope criterion included in defining a resource scope need not be met in order for a resource to be tagged with the tag. Further a resource scope may be used for purposes other than processing related to a tagged resource.”) [0095] (“A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. . . . Whether a criterion, in a scope descriptor is met or is not met, may depend on an evaluation of the scope criterion that is based on an identifiable entity. For a scope condition, such an entity is referred to herein as a "condition entity". More generally, for a scope criterion, such an entity is referred to herein as a "scoping entity". Exemplary entities include a user, a document, a measure of time, a location, and the like.”) [0122] (“the arrangement in FIG. 3A, includes query handler component 302 that is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion.”) [0175] (“In another aspect, a data object identification request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object

	identification request UI element and input from the user, query director component 412 may receive one or more additional data object matching criterion information and/or may receive input to modify a data object matching criterion identified in a received data object identification request. Query director component 412 may locate one or more data objects in response to receiving the additional criterion.") *See also* FIGs. 2A, 2B, [0092], [0094], [0123]-[0124], [0133], [0206]; Original claims 1-22 **<u>'635 Application (EX2002)</u>** [0151] ("FIG. 8B illustrates an "application/id-request" MIME type identifier that may be defined to identify a schema for an XML-based language for specifying id-request XML documents. FIG. 8B illustrates id-request document 806 b. Id-request document 806 b, as illustrated, includes criterion tag elements 808 b corresponding to the form elements in a user interface FIG. 6A. A criterion tag element 808 b identifies a matching criterion name, such as "query" indicating that the criterion is a query expression or a portion thereof. A "type" attribute identifies a schema with the identifier "keyword" for the expression. The criterion tag element 808 b identifies a value for a query "*music*" in a match-expression attribute. Another criterion tag element 808 b specifies a scope query specified in the first criterion tag element. FIG. 8B illustrates an "and" tag 810 b indicating that all the matching criteria must be met for identifying a data object. An "or" tag (not shown) may be defined by a schema for id-request documents. Other operator elements and operator precedence may be defined by the schema. Grouping elements for managing operator precedence, such as parenthesis element, may be defined by the schema.") [0152] ("FIG. 7 includes a data flow that illustrates a previous message 716 (with respect to first message 702) including a data object request. Previous message 716, as shown, may be sent via network 506 by execution environment 401 of first node 502 to execution environment 401 of second node 504. In an aspect, first message 702 may be sent in response to previous message 716.") [0153] ("In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a

	format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a group, a requester, a relationship including a requesting user and a responding user, a keyword, a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.") [0154] ("A descriptor handler component in an execution environment may receive data object information in response to a request from a communications agent in another execution environment. As described above, FIG. 7 illustrates previous message 716, including a data object request, received via network 506 by execution environment 401 of second node 504 from execution environment 401 of first node 502.") *See also* FIGs. 7, 8B
cause, at the first node, display of a second interface with a third user interface element and a fourth user interface element, the second interface for use in causing the file request to be sent;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)**

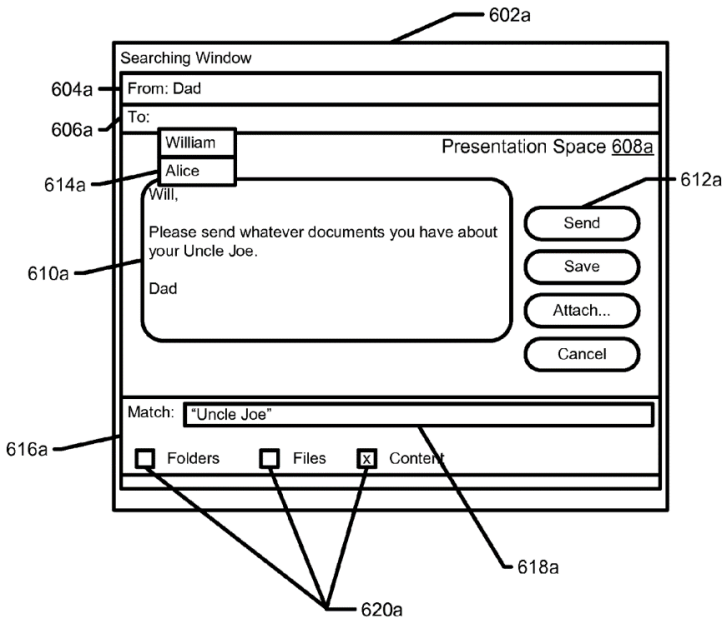


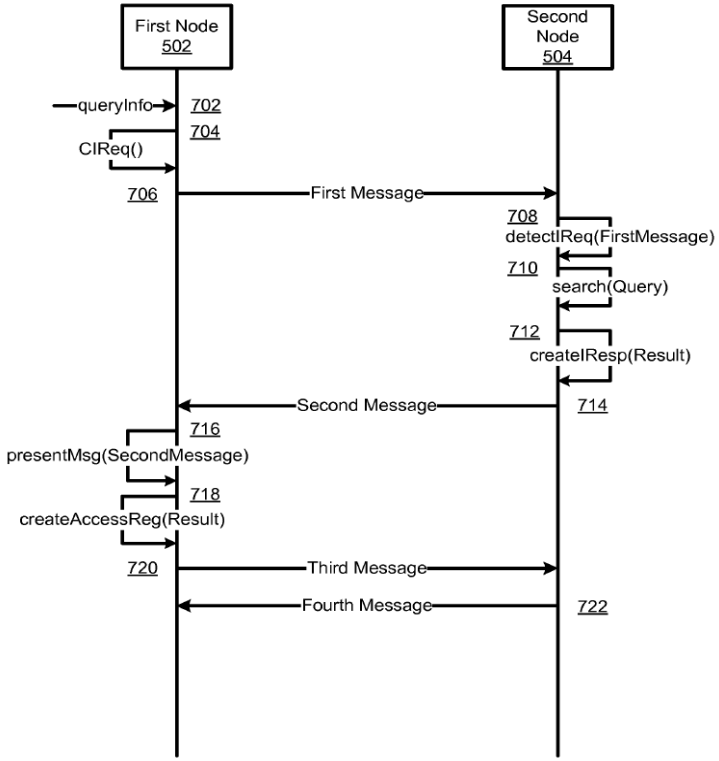
FIG. 6A

[0111] ("A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more U I controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.")

[0117] ("Data may be sent in a communication in response to a communicant input. A contactor may provide an input corresponding to send UI element 612a in FIG. 6A. The input may be received by presentation controller component 417 and/or one or more UI element handlers 415

	corresponding to send UI element 612a. In response to detecting the input, presentation controller component 417 may provide data to be sent in the communication to one or more content handler components 413 according to the content type(s) of the data to be sent.") *See also* [0127], Fig. 6A (Presentation Space 608a, contactee user interface element 606a, Send button user interface element 612a); FIG. 7, Original claims 1-22 **'635 application (EX2002)** [0082] ("FIGS. 6A-H illustrate various user interfaces, some of which include window UI elements 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. A window UI element 602 may include a contactor UI element 604 for presenting an identifier of a communicant in the role of a contactor in a communication represented by the window UI element 602. A window UI element 602 may also include a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) in the communication. A presentation space 608 may be included in each window UI element 602 for presenting a user message UI element 610 that my present a user message from one user to another. A presentation space 608 may also be provided for presenting one or more UI elements that correspond to user inputs for exchanging data in and/or otherwise managing a communication.") *See also* [0051], [0144]-[0145], FIG. 6E, FIG. 7
receive, from the first node via the at least one network, indicia associated with at least one email address via the third user interface element, utilizing the second interface, the at least one	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)**

email address for being used to send the file request;	[0111] (“FIG. 6A-G illustrates various communications agent windows 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. In FIG. 6a, a communications agent window 602a includes a contactor user interface (UI) element 604a for presenting an identifier of a communicant in the role of a contactor in a communication represented by the communications agent window 602a. A communications agent window 602a also includes a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) included in the communication. A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more UI controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.”) [0115] (“Sending data in a communication requires identifying a contactee. A contactee may be represented by a communications address in an address space of a communications protocol**.** In one aspect, information identifying a communications address may be received from a user/communicant of a communications agent in a user node. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 in FIG. 6A for first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 604. The user/communicant of first node 502 may enter a contactor alias, such as "Dad", . . . Communications agent 403 may identify a communication address associated with the contactor alias, "Dad". Other communications address(es), such as for one or more contactees, may be received similarly and/or in any suitable manner.”) [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message.”) [0131] (“In FIG. 8A, a portion of an email communication 800a is illustrated formatted as a multipart/mixed content type including data object identification request portion 802a.”)

	[0193] (“A data object identification request, a data object identification response, an access request, and/or an access response may be included in an email . . . .”)  FIG. 7 [0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.”)

	[0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”)

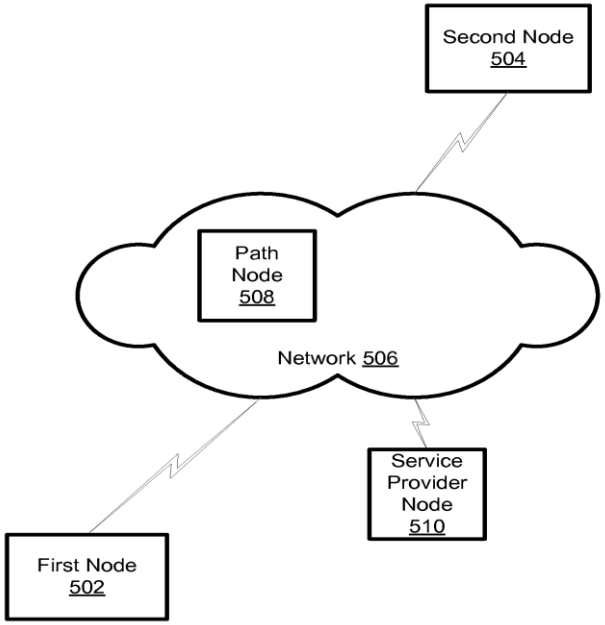
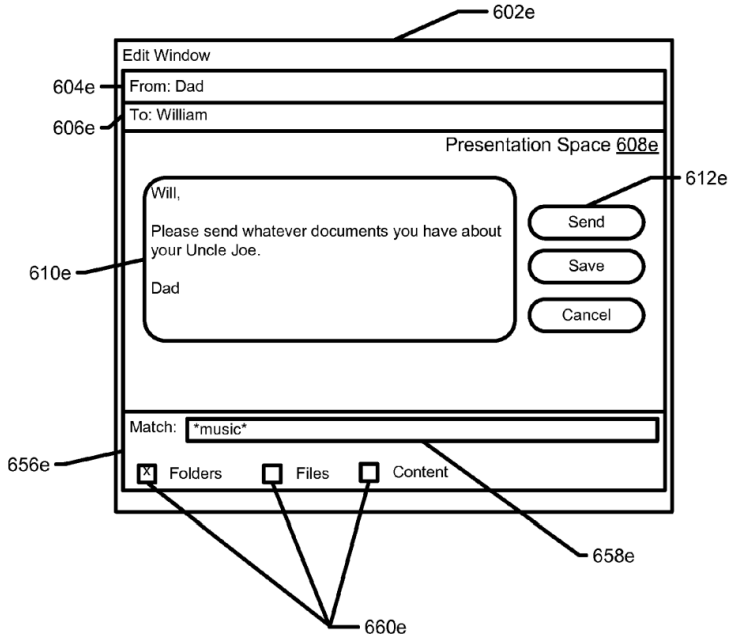


FIG. 5

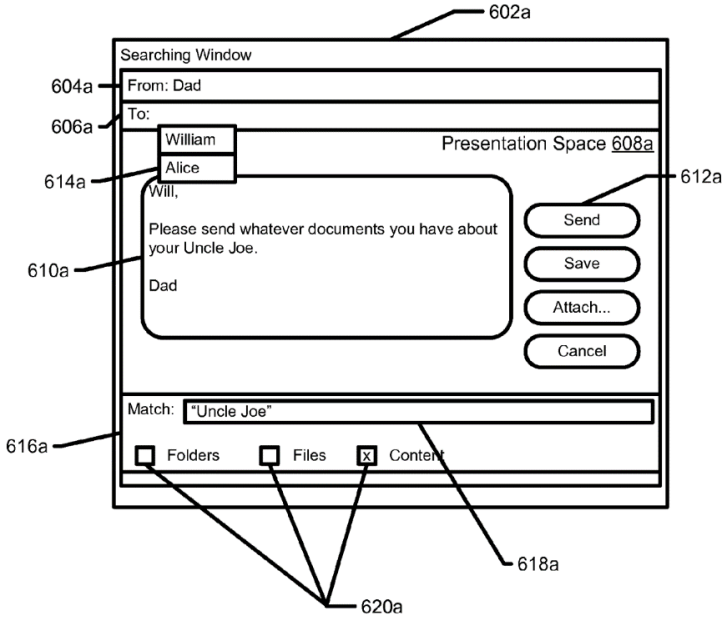
See also Fig. 6A (Presentation Space 608a, contactee user interface element 606a); [0125], [0206]; Original claims 1-22

	’635 application (EX2002)  FIG. 6E [0082] (“FIGS. 6A-H illustrate various user interfaces, some of which include window UI elements 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. A window UI element 602 may include a contactor UI element 604 for presenting an identifier of a communicant in the role of a contactor in a communication represented by the window UI element 602. A window UI element 602 may also include a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) in the communication. A presentation space 608 may be included in each window UI element 602 for presenting a user message UI element 610 that my present a user message from one user to another. A presentation space 608 may also be provided for presenting

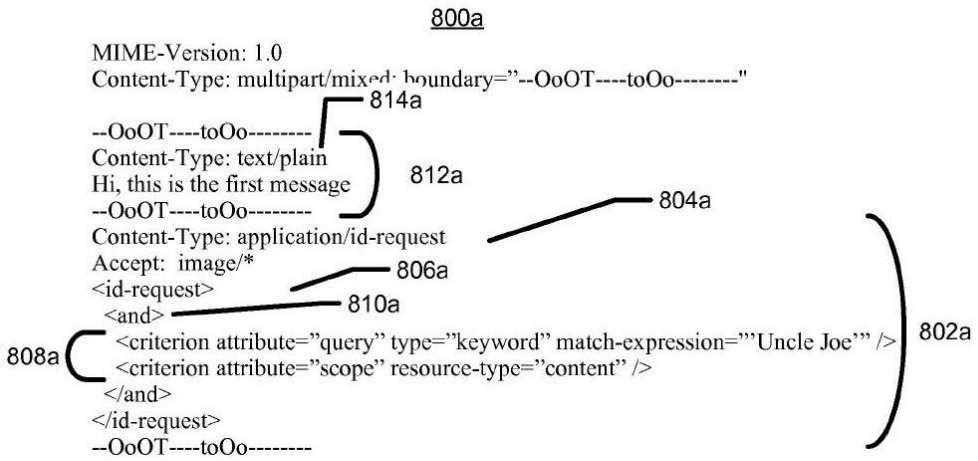
	one or more UI elements that correspond to user inputs for exchanging data in and/or otherwise managing a communication.") [0144] ("FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (MI) UI element 656 e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in MI UI element 656 e. MI UI element 656 e and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0145] ("MI UI element 656 e illustrates a user interface for a keyword based schema. Match textbox UI element 658 e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660 e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema." [0051] ("A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A "program entity" is an object included in and/or otherwise processed by an application or

	executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.”) [0066] (“The term “communicant alias” as used herein refers to an identifier of a communicant in a communication where the communicant alias is not a communications address included in an address space of a communications protocol for sending and/or receiving data in the communication.”) [0086] (“A contactee is identified for sending data in a communication. A contactee may be represented by a communications address in an address space of a communications protocol. In one aspect, information identifying a communications address may be received from a communicant of a communications agent in an execution environment. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 a in FIG. 6A for execution environment 401 of first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 415. The user/communicant of execution environment 401 of first node 502 may enter a contactor alias, such as “William”, via a keyboard and/or select a predefined communicant alias presented in a selection UI control element via a UI element handler component 415. The user input may be detected by input driver 421. Corresponding input information may be routed to presentation controller 417 by GUI subsystem 423. GUI subsystem 423 operates to send presentation information to a display device via a graphics subsystem 425. Communication agent 403 may

	identify a communication address associated with the contactor alias, “William”. Other communications address, such as for one or more contactees, may be received similarly and/or in any suitable manner. In an aspect, one or more contactor and/or contactee identifiers may be determined automatically.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0172] (“As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.”) *See also* [0142]-[0143], [0146]-[0153], FIGs 7, 8B.

receive, from the first node via the at least one network, an indication of a selection of the fourth user interface element, utilizing the second interface, where the receipt of the indication of the selection of the fourth user interface element indicates that the file request is to be caused to be sent;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6A [0111] (“A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more U I controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that

	may correspond to user input to send data in a communication to one or more identified contactees.") [0123] ("FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") *See also* FIG. 6A (send UI element 612a); [0111] (discussing 612a); [0117]; FIG. 7 (flowchart of request process); [0123]-[0124]; [0206]; Original claims 1-22 **'635 application (EX2002)** FIG. 6E, 7, 8B, [0142]-[0153]; *see also* [0073], [0172]

in response to the receipt of the indication of the selection of the fourth user interface element and based on the file description, the indication of the at least one of the date or the time, and the indicia associated with the at least one email address; generate a first message addressed to the at least one email address and including the file request, the first message indicating the at least one of the date or the time on which the file request is based, and further including a fifth user interface element and at least one sentence including the file description describing the at least one file in connection with the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  Fig. 8A [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message. ...”) [0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.”)

	[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0127] ("Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. Accordingly, a system for processing a data object identification request in a communication includes means for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. For example, the arrangement in FIG. 3A, includes identification request constructor component 304 that is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. FIG. 4 illustrates identification request constructor component 404 as an adaptation and/or analog of identification request constructor component 304 in FIG. 3A. One or more identification request constructor components 404 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, according to a first communications

	protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. In FIG. 4, identification request constructor component 404 is illustrated as a component of communications agent 403.") [0128] ("FIG. 7 includes dataflow message 704 that illustrates a data flow internal to execution environment 401 of first node 502 and/or message received via network 506 by execution environment 401 of first node 502. Dataflow message 704 illustrates a create data object identification request (CIReq) dataflow to an IRC component 404 operating in execution environment 401 of first node 502. Data object information may be identified to and/or otherwise provided to IRC component 404. A query handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide data object information to an IRC component 404.")

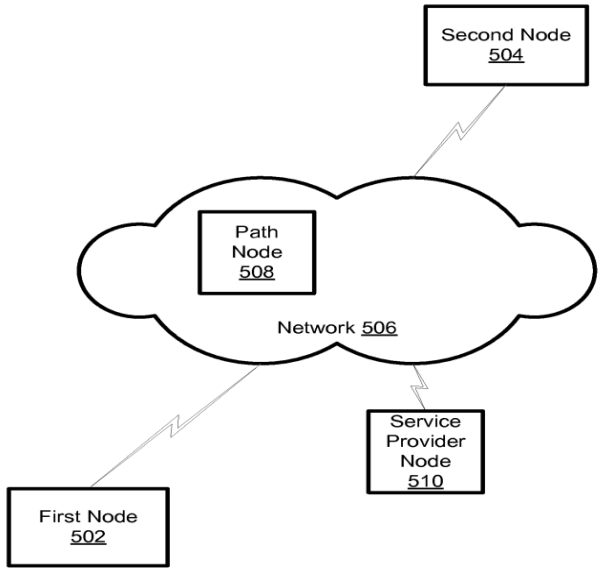
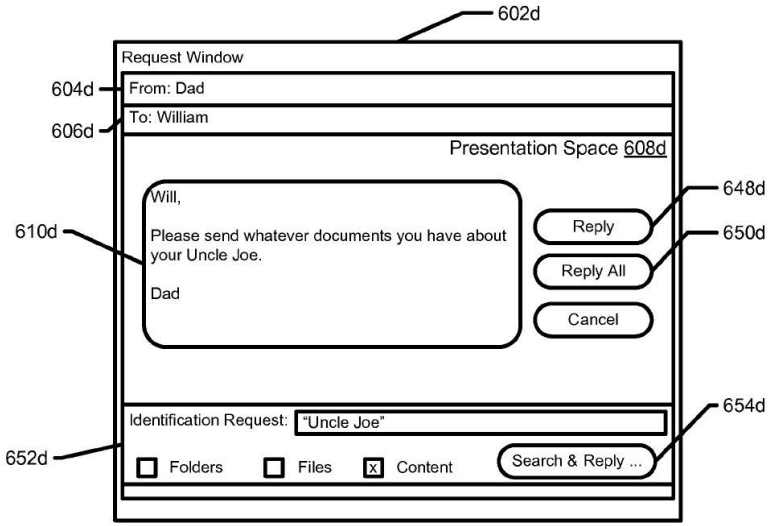


FIG. 5

See also [0092]-[0093], [0123]-[0124]; [0129]-[0135] (message creation and contents); [0175], [0180]; [0206], FIG. 2A, FIG. 6B, FIG. 6D, FIG. 7 (flowchart of request process); FIG. 8A; Original claims 1-22

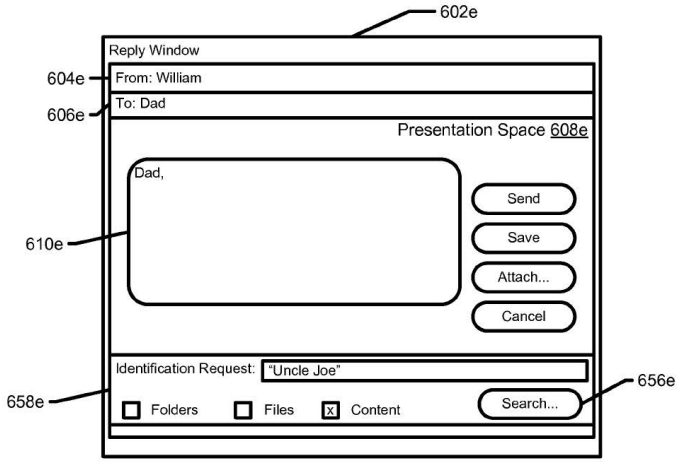
’635 application (EX2002)

FIG. 6E, 7, 8B, [0142]-[0153]; *see also* [0073], [0172]

send, to a second node via the at least one network, the first message;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0127] (“Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user…”)  FIG. 6D

	[0162] (“FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object identification request may include and/or otherwise identify a data object matching criterion identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.”) [0166] (“FIG. 6D illustrates a window, request window UI element 602d, presenting an exemplary representation of a message received from execution environment 401 of first node 502 in a communication with second node 504. As in FIG. 6A, request window UI element 602d includes a contactor UI element 604d, a contactee UI element 606d, a presentation space 608d of the request window UI element 602d that includes a user message UI element 610d including a text portion of the message provided by the communicant represented by execution environment 401 of first node 502. The various UI elements may be presented by one or more UI element handler components 415 in communications agent 403 operating in execution environment 401 of second node 504. A reply UI element 648d and a reply all UI element 648d are illustrated as exemplary UI elements in presentation space 608d presented to allow a user of execution environment 401 of second node 504 to respond and/or otherwise perform an operation based on the received message. FIG. 6D illustrates a data object identification request UI interface

	element 652d where the data object identification request identified in the message from execution environment 401 of first node 502 is represented to the communicant represented by the execution environment 401 of second node 504. In an aspect, the communicant is allowed to authorize the request by providing an input corresponding to a search and reply UI element 654d. In another aspect, the communicant may be allowed to modify the received data object identification request.") *See also* [0092]-[0093], [0123]-[0124]; [0128]-[0131]; [0133], [0134]-[0136], [0157], [0175], [0180]; FIGs. 2A, 2B, 3A, 4; FIG. 6D; FIG. 7 (flowchart of request process); FIG. 8A; Original claims 1-22 **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") *See also* FIG. 6E, 7, 8B, [0142]-[0173]; [0073]

receive, from the second node via the at least one network, an indication of a selection of the fifth user interface element of the first message, where the receipt of the indication of the selection of the fifth user interface element indicates that the at least one file is to be uploaded in response to the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6E [0151] (“ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.”) [0162] (“FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation

	between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object identification request may include and/or otherwise identify a data object matching criterion identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.") [0167] ("FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object

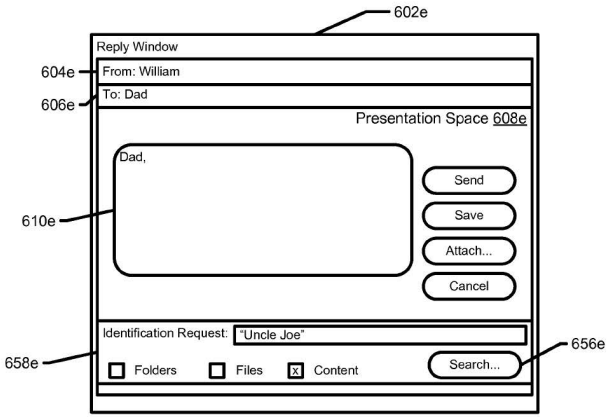
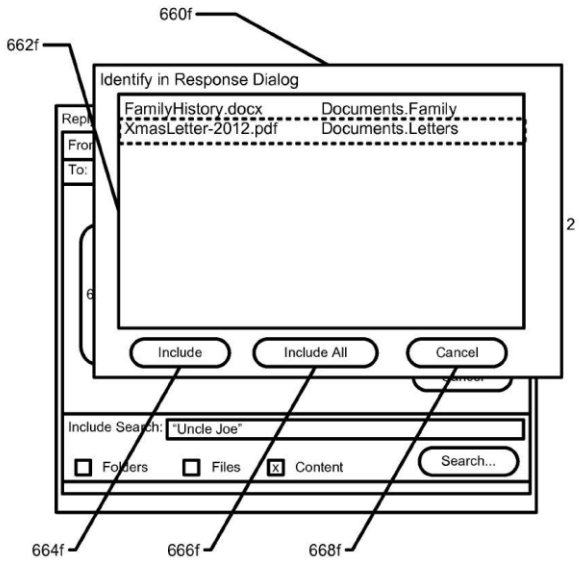
	identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0179] ("FIG. 8B illustrates an exemplary data object identification response described in detail above. Second message portion 802b illustrates a user message portion that may be included in second message 714. Note that a user message portion may be empty or absent in some aspects. Data object identification response portion 804b illustrates the data object identification response as a list of URls identifying data objects in a file system of execution environment 401 of second node 504. User Message portion 802b and data object identification response portion 804b may be provided by respective content handler components 413 to content manager component 409. Content manager component 409 may construct the content as illustrated in FIG. 8B and/or otherwise provide for encoding, translating, combining, and/or otherwise preparing the content for sending in second message 714.") [0181] ("In another aspect, a data object located and/or otherwise identified in response to receiving a data object identification request must meet match a query identified by the data object identification request. Alternatively or additionally, a data object identification request may include an instruction and/or input for generating a data object. Locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object identification request may identify the template. A query handler component 402 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to an access request, execution environment 401 of second node 504 may return the document.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.")

	[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") *See also* [0100], [0111], [0137] (discusses receiving a message from second node); FIG. 5, FIG. 6B (selecting element 628b to bring up window 630b); [0146]-[0148]; FIG. 2B, [0157]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0170], [0173]; [0206]; FIG. 7; Original claims 1-22.

	<u>’635 application (EX2002)</u> [0171]-[0172] (“Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.”) [0069] (“The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0112] (“FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be

	separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed

	Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.") *See also* [0157], FIGs. 8A, 8B; Original claim 14

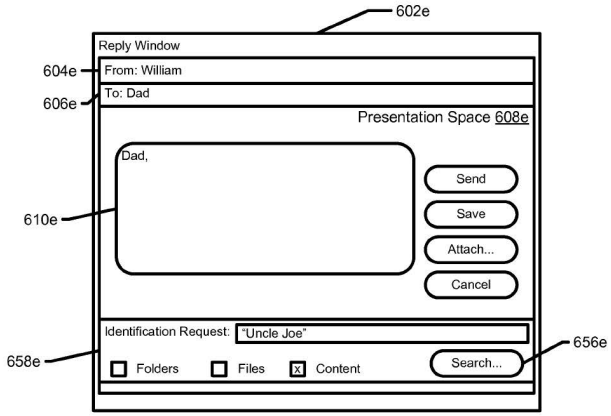
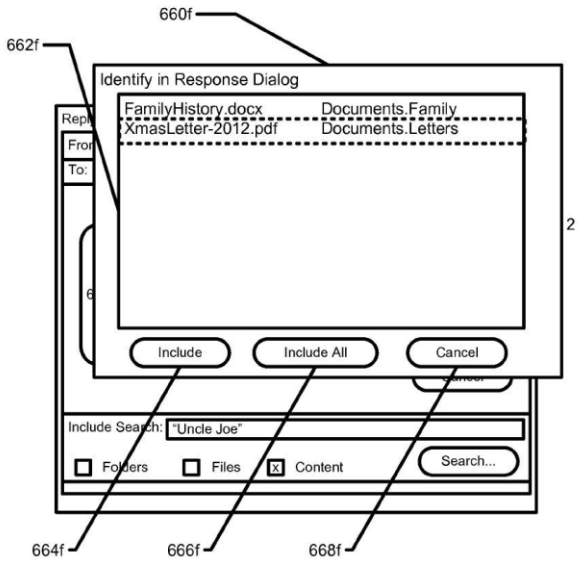
in response to the receipt of the indication of the selection of the fifth user interface element, cause, at the second node, display of a third interface with a sixth user interface element and a seventh user interface element, the third interface for use in uploading the at least one file in response to the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6E  FIG.6F [0167] (“FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG.

	6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0172] "(FIG. 6F, illustrates an identify in response dialog UI element 660f that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to processing the data object identification request to locate matching data objects. In an aspect, a user of communications agent 403 operating in execution environment 401 of second node 504 may provide input corresponding to search UI element 656e in reply window UI element 602e in FIG. 6E. A corresponding UI element handler component 415 may operate to invoke query director component 412 to perform and/or otherwise to provide for performing one or more operations to locate one or more data objects based on the data object identification request. Query director component 412 may interoperate with one or more UI element handlers 415, directly or indirectly via query handler component 402, in an aspect, to present the UI element 660f and its contents. Identify in response dialog UI element 660f includes results pane UI element 662f including representations of two data objects located and/or otherwise identified by query director component 412 based on the data object identification request. A user may select one or more data objects to identify in a data object identification response. An include UI element 664f may be presented, as illustrated, to allow a user to instruct communications agent 403 to identify one or more selected data objects in results pane U I element 662f in a data object identification response. An include all UI element 666f is

	illustrated to allow for user input instructing communications agent 403 to identify all data objects represented in the UI element 660f in a data object identification response in a communication with execution environment 401 of first node 502. A cancel UI element 668f allows a user to end presentation of dialog UI element 660f. A user may opt to identify no data objects represented in the dialog UI element 660f and located and/or otherwise identified by query director component 412 based on the data object identification request.") *See also* FIGS. 5, 6C, 6G; [0146]-[0148]; [0150]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0206], FIG. 7 **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") [0069] ("The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0073] ("As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a

	browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.")

	[0117] (“For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.”) [0118] (“Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.”) [0119] (“Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.”) [0130] (“FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the “Music” folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.”) *See also* [0157], FIGs. 8A, 8B; Original claim 14

receive, from the second node via the at least one network, an indication of a selection of the seventh user interface element, utilizing the third interface;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** *See* support cited above.  FIG. 6E  FIG.6F [0167] (“FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all

	of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0172] "(FIG. 6F, illustrates an identify in response dialog UI element 660f that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to processing the data object identification request to locate matching data objects. In an aspect, a user of communications agent 403 operating in execution environment 401 of second node 504 may provide input corresponding to search UI element 656e in reply window UI element 602e in FIG. 6E. A corresponding UI element handler component 415 may operate to invoke query director component 412 to perform and/or otherwise to provide for performing one or more operations to locate one or more data objects based on the data object identification request. Query director component 412 may interoperate with one or more UI element handlers 415, directly or indirectly via query handler component 402, in an aspect, to present the UI element 660f and its contents. Identify in response dialog UI element 660f includes results pane UI element 662f including representations of two data objects located and/or otherwise identified by query director component 412 based on the data object identification request. A user may select one or more data objects to identify in a data object identification response. An include UI element 664f may be presented, as illustrated, to allow a user to instruct communications agent 403 to identify one or more selected data objects in results

	pane U I element 662f in a data object identification response. An include all UI element 666f is illustrated to allow for user input instructing communications agent 403 to identify all data objects represented in the UI element 660f in a data object identification response in a communication with execution environment 401 of first node 502. A cancel UI element 668f allows a user to end presentation of dialog UI element 660f. A user may opt to identify no data objects represented in the dialog UI element 660f and located and/or otherwise identified by query director component 412 based on the data object identification request.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

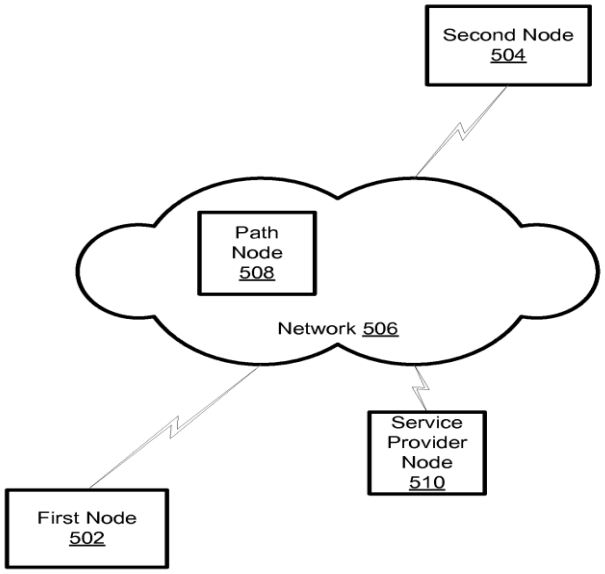


FIG. 5

See also FIGS. 6C, 6G; [0146]-[0148]; [0150]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0206]

’635 application (EX2002)

[0069] (“The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.”)

	[0073] (“As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0112] (“FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.”) [0116] (“In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage

	servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a

	Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.") [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") *See also* [0157], FIGs. 8A, 8B; Original claim 14

in response to the receipt of the indication of the selection of the seventh user interface element, generate a second message that includes an eighth user interface element, and that does not include a file attachment with the second message, the second message indicating that the file request has been responded to;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0008] (“Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.”)

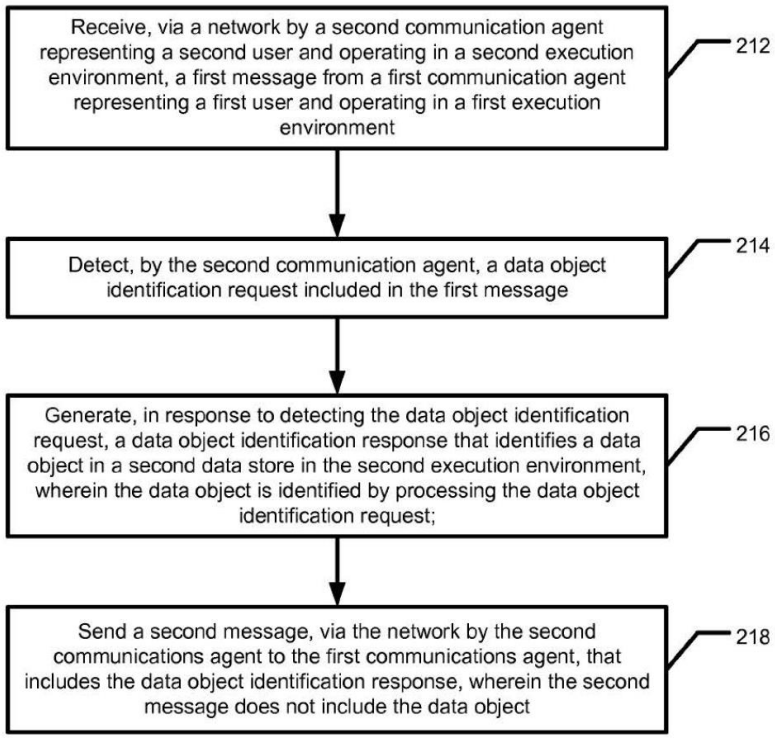

Receive, via a network by a second communication agent representing a second user and operating in a second execution environment, a first message from a first communication agent representing a first user and operating in a first execution environment
212
Detect, by the second communication agent, a data object identification request included in the first message
214
Generate, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request;
216
Send a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object
218

FIG. 2B

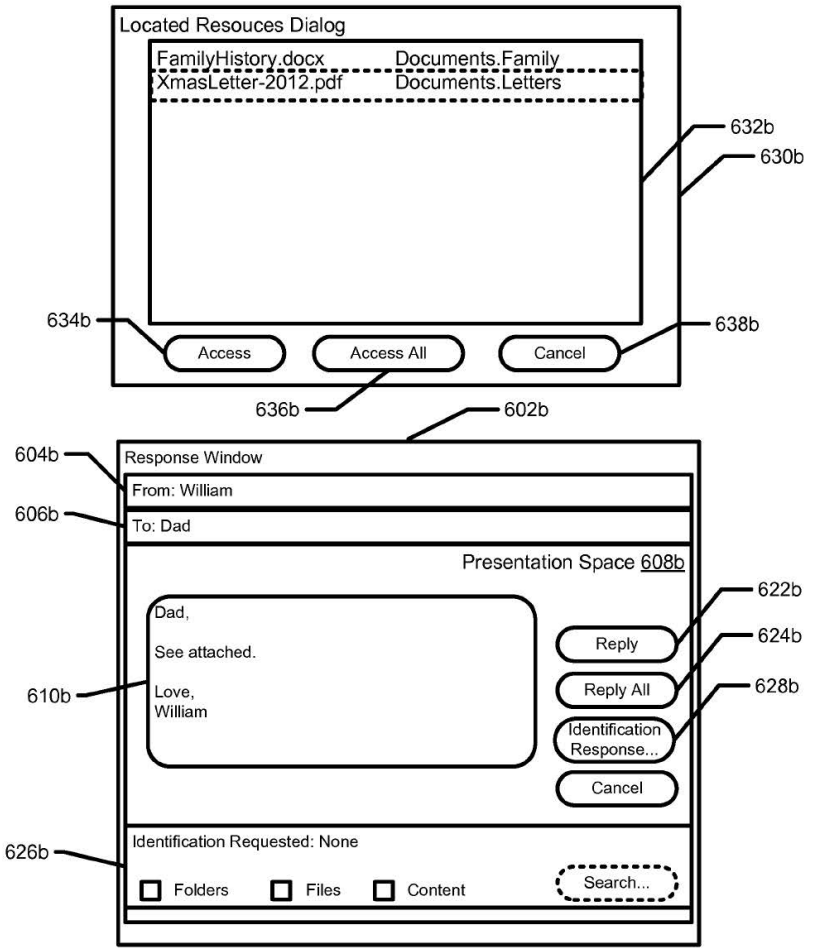

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
638b
Access
Access All
Cancel
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
Dad,
See attached.
624b
610b
Love,
William
Reply
Reply All
628b
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

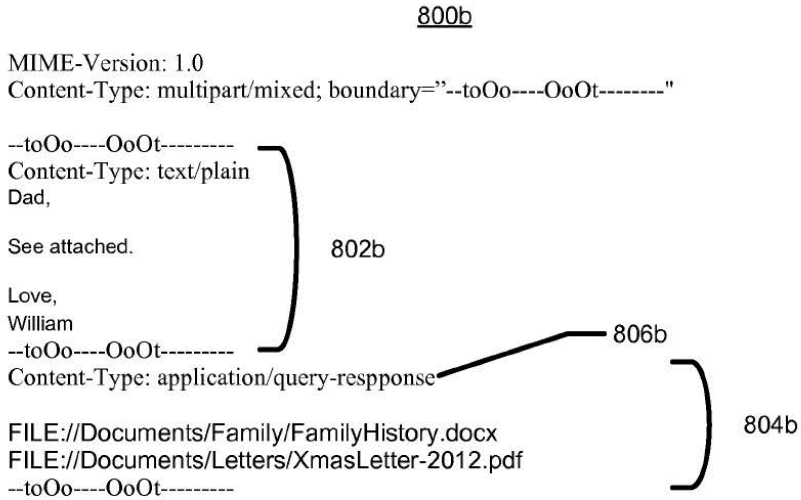


FIG. 8B

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

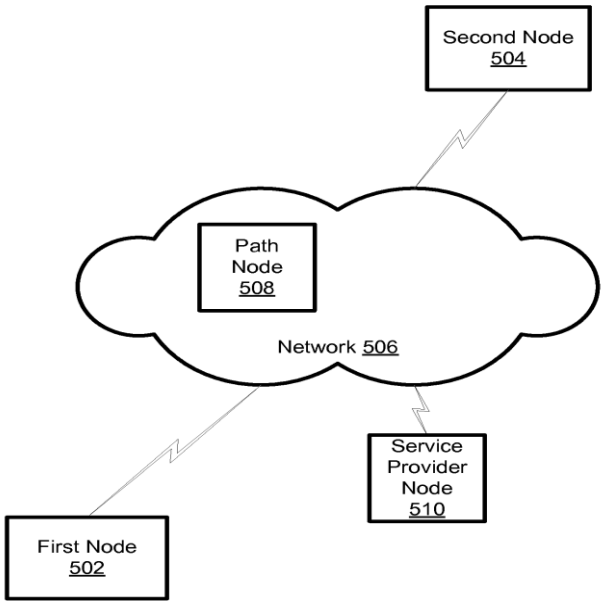

Second Node
504
Path
Node
508
Network 506
Service
Provider
Node
510
First Node
502

FIG. 5

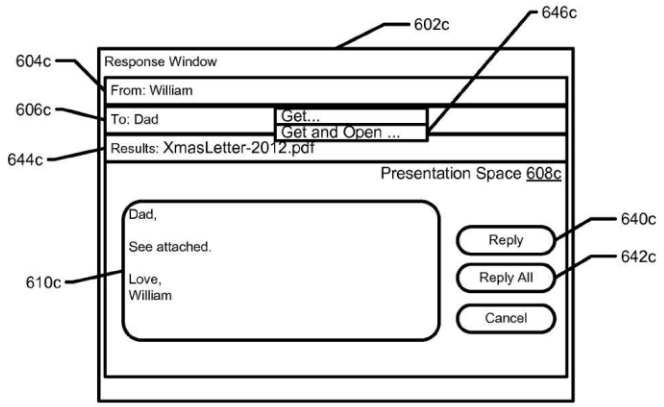


FIG. 6C

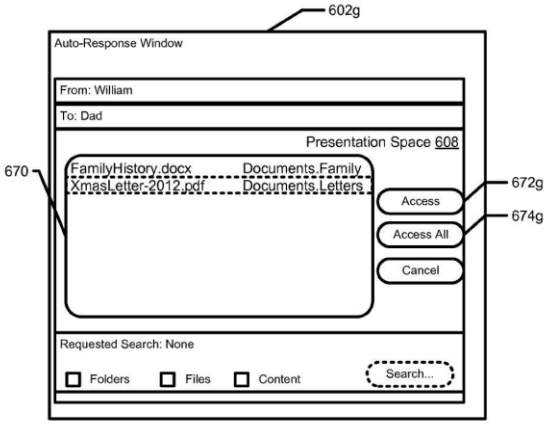
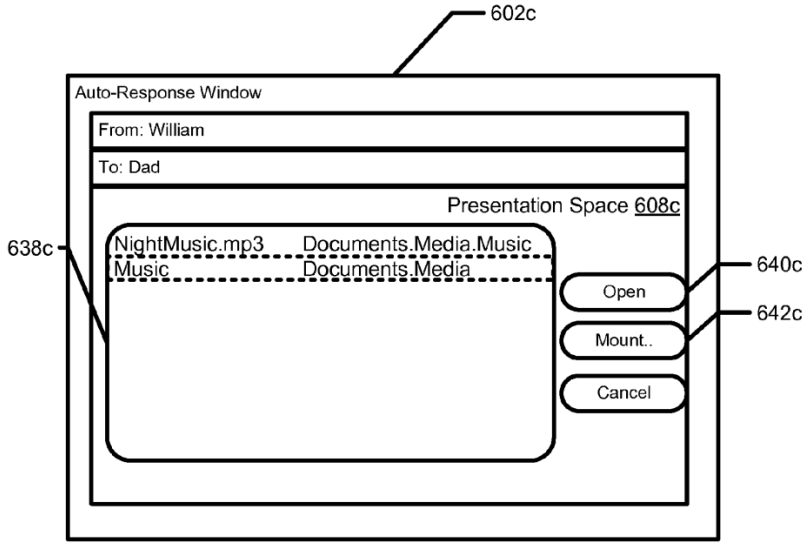


FIG. 6G

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather

	than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") *See also* [0007], [0009]-[0011], [0150]; [0137] (discusses receiving a message from second node, "wherein the data object is not received in the second message"); FIG. 8B, [0141]-[0142]; FIG. 6C, [0150]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 2B, [0170]; [0171]-[0173]; [0176]; [0179]; [0200]; [0206]. **'635 application (EX2002)** 800a MIME-Version: 1.0 Content-Type: multipart/mixed; boundary="--toOo----OoOt--------" --toOo----OoOt--------- Content-Type: text/plain Dad, See attached. Love, William --toOo----OoOt--------- Content-Type: application/mount-desscirptor smb://SecondNode/Music http://firstuser@secondnode.tld/users/seconduser/documents/media/music/ ftp://SecondNode/music/ --toOo----OoOt--------- 802a 806a 808a 810a 804a 812a Fig. 8A

	[0096] (“FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user readable message portion 802 a and a mount descriptor portion 804 a described in more detail below. The received communication may include a message addressed to a user represented by communications agent 403 in execution environment 401 of first node 502.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the “Music” found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the “Music” folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the “Music” folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the “Music” folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.”) [0168] (“In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an

	instruction and/or input for generating a data object, and locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.")  FIG. 6C [0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a

	communication without a user message. Rather than a UI element for displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.")

send, to the first node via the at least one network, the second message;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0008] (“Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.”)

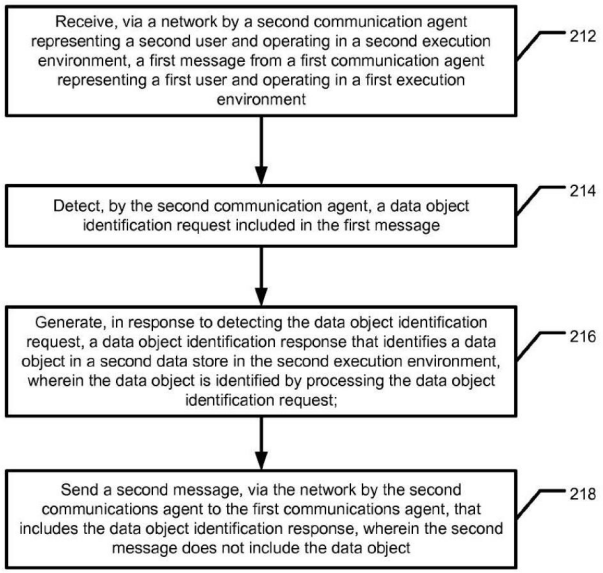

Receive, via a network by a second communication agent representing a second user and operating in a second execution environment, a first message from a first communication agent representing a first user and operating in a first execution environment
212
Detect, by the second communication agent, a data object identification request included in the first message
214
Generate, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request;
216
Send a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object
218

FIG. 2B

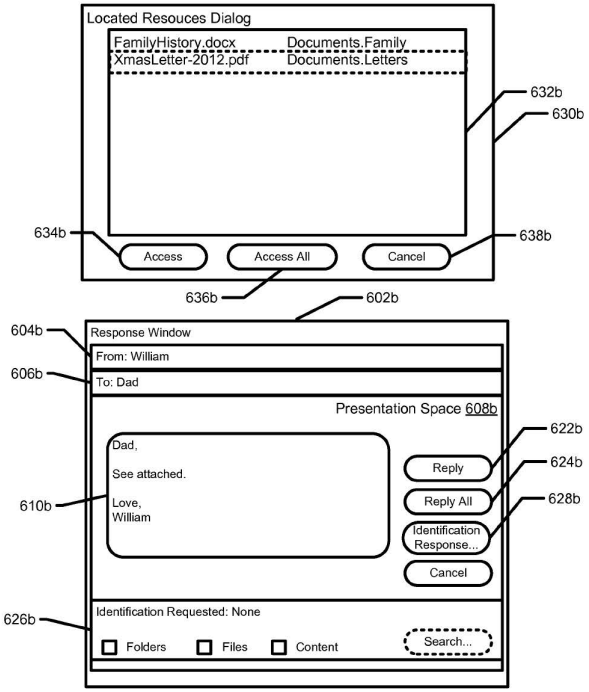

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
638b
Access
Access All
Cancel
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
624b
628b
Dad,
See attached.
610b
Love,
William
Reply
Reply All
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

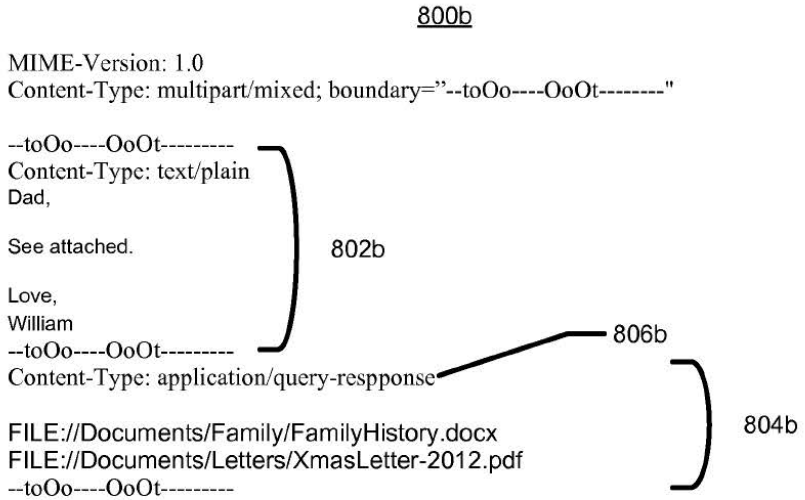


FIG. 8B

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

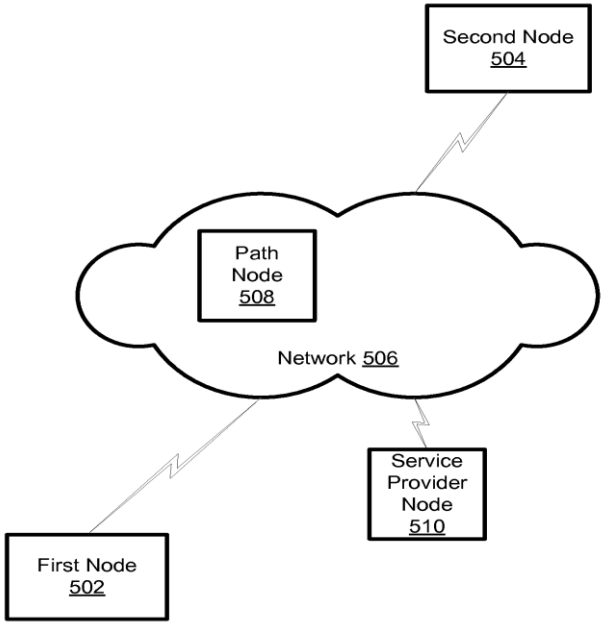

Second Node
504
Path
Node
508
Network 506
Service
Provider
Node
510
First Node
502

FIG. 5

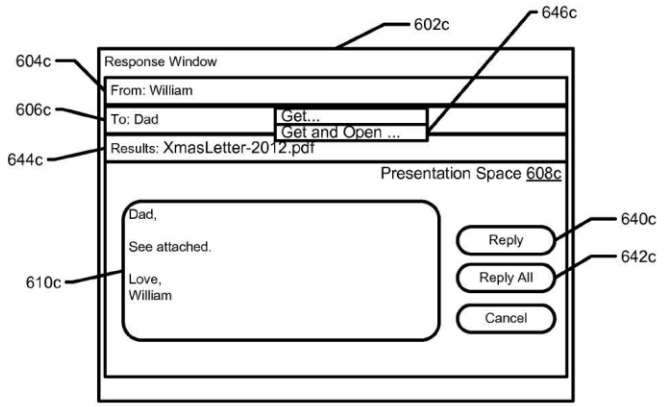


FIG. 6C

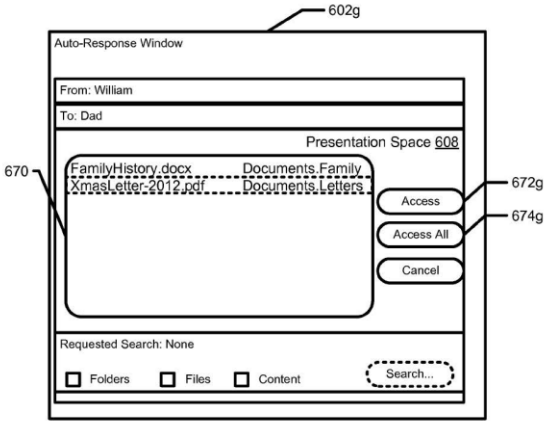


FIG. 6G

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather

	than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.”)

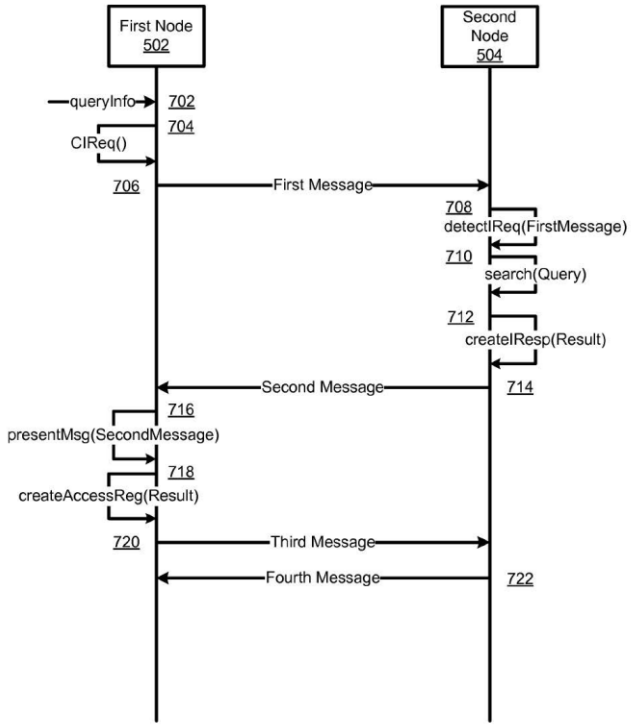
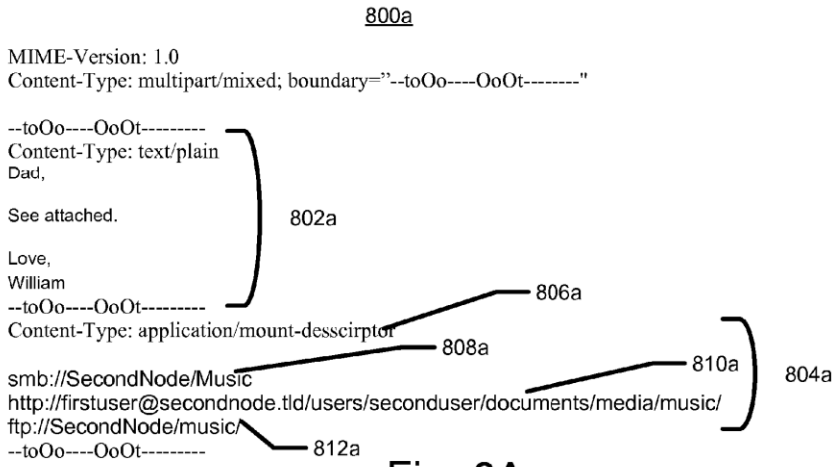


FIG. 7

See also [0150]; FIGS. 6C, 6G; FIGS. 2A, 2B; [0137] (discusses receiving a message from second node, "wherein the data object is not received in the second message"); FIG. 7, [0138]; [0139]-[0140] (data communications); [0141]; [0151]; [0176]; [0177]

	’635 application (EX2002)  Fig. 8A [0096] (“FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user readable message portion 802 a and a mount descriptor portion 804 a described in more detail below. The received communication may include a message addressed to a user represented by communications agent 403 in execution environment 401 of first node 502.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the “Music” found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the “Music” folder. Mount service component 408 in execution environment 401 of

	first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0168] ("In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an instruction and/or input for generating a data object, and locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.")

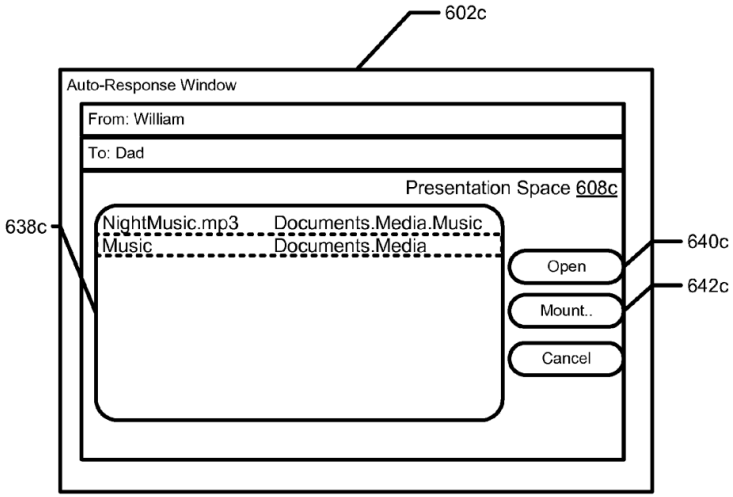


FIG. 6C

[0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a communication without a user message. Rather than a UI element for displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.")

See also FIG. 7

receive, from the first node via the at least one network, an indication of a selection of the eighth user interface element of the second message; in response to the receipt of the indication of the selection of the eighth user interface element of the second message, cause, at the first node, display of at least one fourth interface with a reference to the at least one file for allowing access to the at least one file at the first node.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0146] (“FIG. 68 illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes a user message UI element 610b, analogous to similar UI elements described with respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409 in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.”) [0147] (“Response window UI element 602b in FIG. 68 includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user

	detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect, one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **68,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution environment 401 of first node 502 to request one or more data objects identified in the data object identification response.")

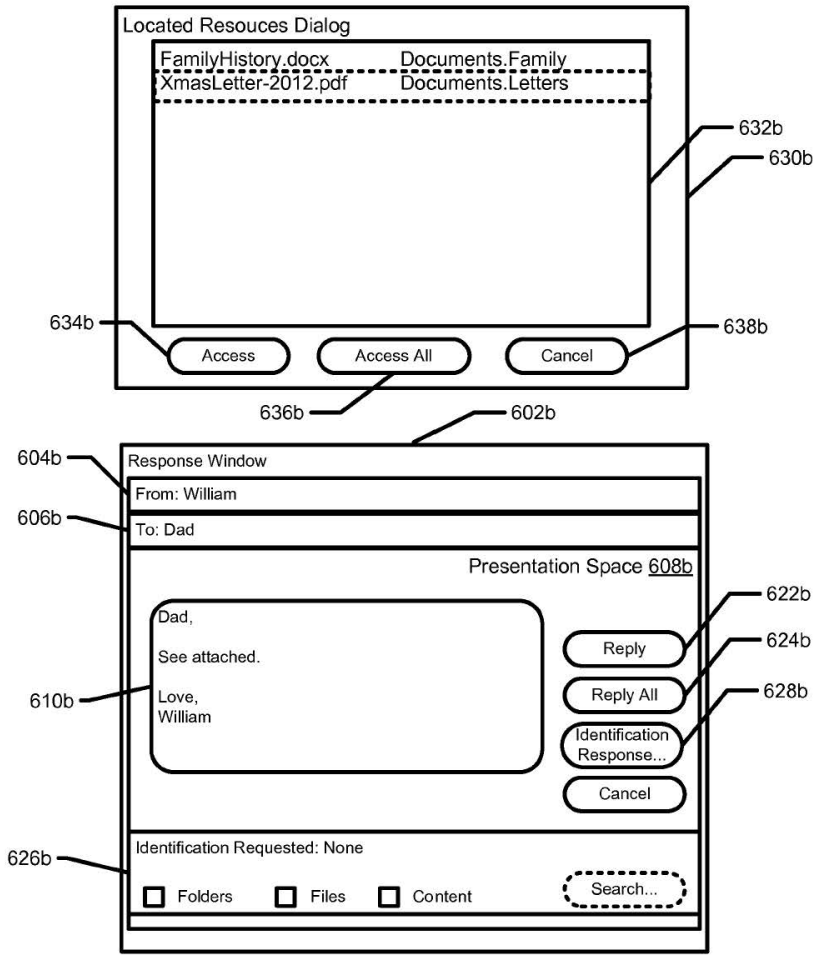

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
Access
Access All
Cancel
638b
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
Dad,
See attached.
Love,
William
624b
610b
Reply
Reply All
628b
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

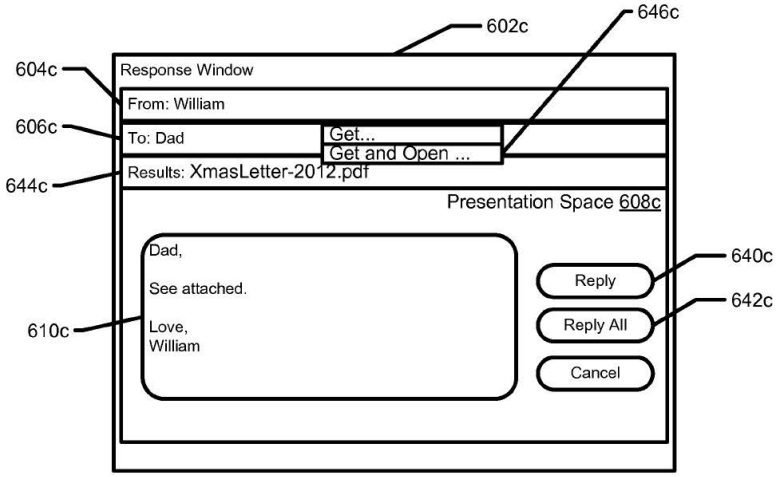


FIG. 6C

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one

	or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

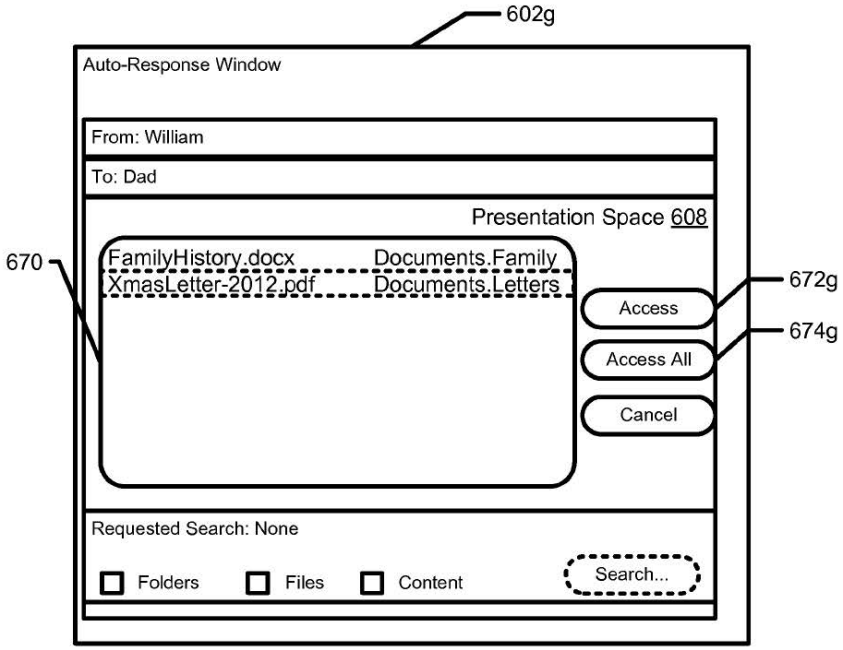

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
Access
Access All
Cancel
672g
674g
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

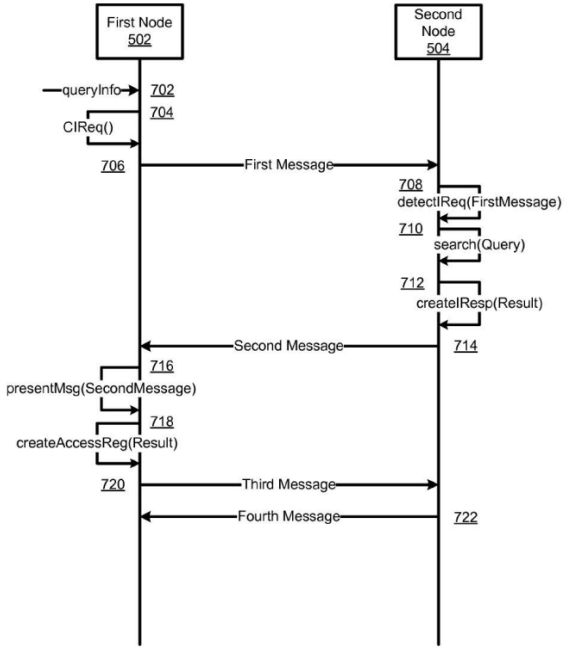


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A.

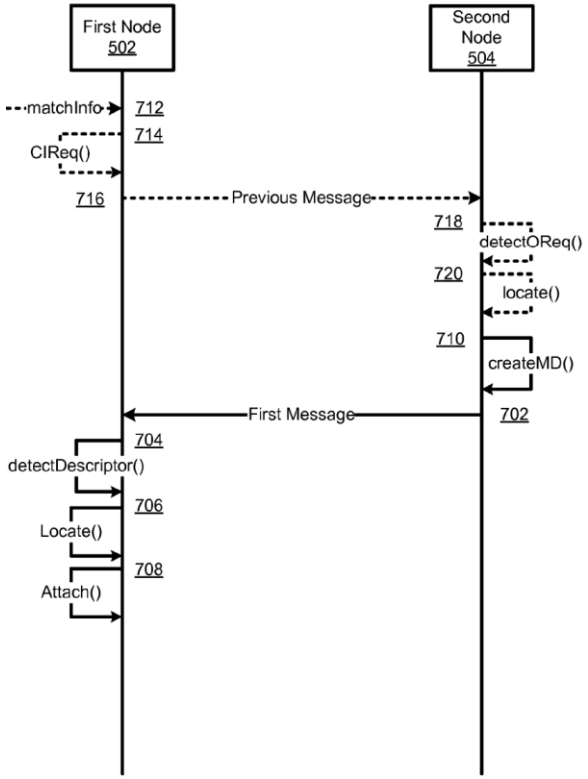
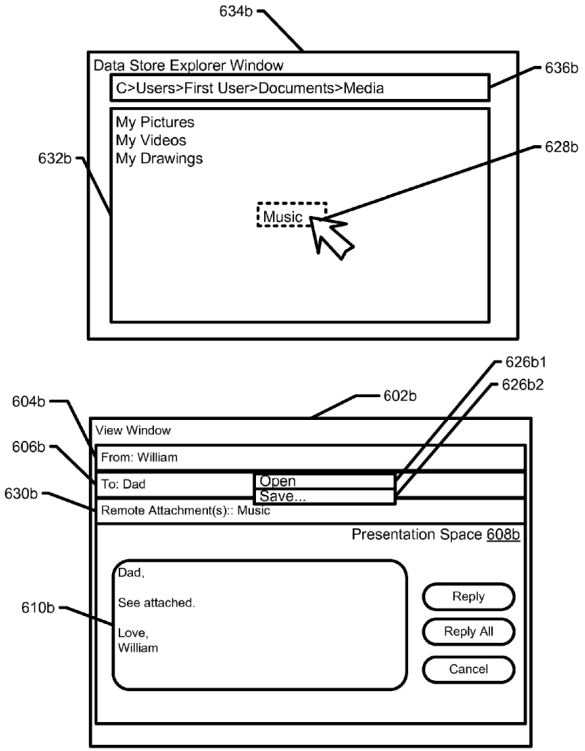
	One or more access request constructor components 408 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") [0144] ("FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.") *See generally* [0150]-[0151], [0142], [179]; [0143]-[0144] (sending "message" to server to access file(s)); FIG. 7 (message 720); [0149]; [0153], FIG. 2A; [0155]; [0201]; [0206] **'635 application (EX2002)** [0106] ("In response to detecting the mount descriptor in first message 702, execution environment 401 of first node 502 may present a view window UI element 602 b in FIG. 6B including data received in the first message 702 to a user of execution environment 401 of first node 502 as described with respect to FIG. 6A. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" menu UI element 626 b 1, identified in a mount descriptor received from execution environment 401 of second node 504. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to

	attach and/or otherwise mount a representation of a data object, in response to an input corresponding to “Save . . . ” menu UI element 626 b 2, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.”) [0130] (“FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a”) [0152] (“FIG. 7 includes a data flow that illustrates a previous message 716 (with respect to first message 702) including a data object request. Previous message 716, is shown, may be sent via network 506 by execution environment 401 of first node 502 to execution environment 401 of second node 504. In an aspect, first message 702 may be sent in response to previous message 716.”) *See also* [0112], [0115]-[0119], [0142]-[0173], FIGs. 2A, 2B, 7,

Claim 10	
The apparatus of claim 9, wherein the apparatus is configured such that the at least one of the date or the time of the indication of the at least one of the date or the time is a criterion, and the indication of the at least one file is received based on whether the criterion is met.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0007] (“Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.”) [0093] (“In specifying a resource scope, one or more criterion may be specified. The terms "criterion" and "constraint" are used interchangeably herein, with respect to a resource scope. A criterion included in specifying a resource scope is referred to herein as a "scope criterion". A scope criterion may be expressed via any suitable manner, such as a file name, a wildcard expression, a regular expression, a data store location, a data base query, geospatial coordinates**,** a time, a date, a duration, a relationship to a user, and/or an address - too name a few examples. With respect to a tag, a scope criterion included in defining a resource scope differs from a tagging criterion for the tag in that the scope criterion included in defining a resource scope need not be met in order for a resource to be tagged with the tag. Further a resource scope may be used for purposes other than processing related to a tagged resource.”)

	[0095] (“A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. . . . Whether a criterion, in a scope descriptor is met or is not met, may depend on an evaluation of the scope criterion that is based on an identifiable entity. For a scope condition, such an entity is referred to herein as a "condition entity". More generally, for a scope criterion, such an entity is referred to herein as a "scoping entity". Exemplary entities include a user, a document, a measure of time, a location, and the like.”) [0125] (“FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415 may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether data object information and/or a data object identification request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing data object information and/or data object identification requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SOL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.”) *See also* [0092], [0096]-[0098], [0133], [0175], [0180] (“additional matching criteria.”) **’635 application (EX2002)**

	[0167] (“In another aspect, a data object request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object request UI element and input from the user, descriptor handler component 404 may receive one or more additional matching criterion information and/or may receive input to modify a matching criterion identified in a received data object request. Descriptor handler component 404 may locate one or more data objects in response to receiving the additional criterion.”)

Claims 11 and 12	
11. The apparatus of claim 9, wherein the apparatus is configured such that, in response to the receipt of the indication of the selection of the seventh user interface element, a creation of a representation of the at least one file is caused, at the first node, in a location among one or more folders on a file explorer interface that is not a web page and utilizes a client-based file explorer application, such that the at least one file is accessible via both the second message and the file explorer interface. 12. The apparatus of claim 11, wherein the apparatus is configured such that the second message is sent without including the at least one file as an attachment of the second message, and a storage at the first node does not store the at least one file when the creation of the representation of the at least one file is caused, both for avoiding the at least one file from having to be communicated to and stored at the first node	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'635 application (EX2002)**  FIG. 7  FIG. 6B

until the at least one file is requested by a user at the first node using the representation or the reference.	[0008] ("Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment. The method further includes detecting the mount descriptor. The method still further includes determining a first location in a first data store in the first execution environment. The method additionally includes creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.") [0009] ("Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment; detecting the mount descriptor; determining a first location in a first data store in the first execution environment; and creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store.") [0010] ("Further, a system for sharing a data object in a data store via a communication is described. The system includes a content manager component for receiving , by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second

	data store in the second execution environment. The system further includes a descriptor handler component for detecting the mount descriptor. The system still further includes a data store navigator component for determining a first location in a first data store in the first execution environment. The system additionally includes a mount service component for, creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the content manager component, the descriptor handler component, the data store navigator component, and the mount service component includes an instruction that is executed by the processor during operation of the system.") [0011] ("Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The method further includes creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The method still further includes including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The method additionally includes sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.") [0012] ("Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent; creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first

	communications agent that represents a first user; including, by the second communications agent, the mount descriptor in a first message, addressed to the first user; and sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store." [0013] ("Further, a system for sharing a data object in a data store via a communication is described. The system includes a descriptor handler component for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The system further includes a descriptor generator component for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The system still further includes a content manager component for including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The system additionally includes a com-out component for, sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the descriptor handler component, the descriptor generator component, the content manager component, and the com-out component includes an instruction that is executed by the processor during operation of the system.") [0098] ("FIG. 7 includes a data flow that illustrates a detect descriptor data flow 704 that may be internal to execution environment 401 of first node 502 and/or that may include a message received via network 506 by first node 502. In an aspect, detect descriptor data flow 704 may illustrate a data flow to descriptor handler component 404 operating in execution environment 401 of first node 502. A mount descriptor may be identified to and/or otherwise provided to descriptor handler component 404. Content handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide a mount descriptor to a descriptor handler component 404.")

	[0109] ("…a system for sharing a data object in a data store via a communication includes means for creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store…") [0111] ("Once a location is identified, in a data store of an execution environment, for attaching and/or otherwise mounting a representation of a data object in a data store in another execution environment; any of a variety means available to the execution environment mounting the data object may be utilized.") [0112] ("…A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502…") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder.

	The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0069] ("…Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.")

	[0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user…") [0181] ("A data object request and/or a mount descriptor or respective portions thereof may include at least a portion of a keyword expression, a regular expression, expression including a Boolean operator, an expression including a precedence information, and a structured query language statement." [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0041] ("Execution environment 102 may include software components stored in persistent secondary storage 108, in remote storage accessible via a network, and/or in a processor memory. FIG. 1 illustrates execution environment 102 including an operating system 120, one or more applications 122, and other program code and/or data components illustrated by other libraries and subsystems 124. In an aspect, some or all software components may be stored in locations accessible to processor 104 in a shared memory address space shared by the software components. The software components accessed via the shared memory address space are stored in a shared processor memory defined by the shared memory address space. In another aspect, a first software component may be stored in one or more locations accessed by processor 104 in a

	first address space and a second software component may be stored in one or more locations accessed by processor 104 in a second address space. The first software component is stored in a first processor memory defined by the first address space and the second software component is stored in a second processor memory defined by the second address space.") [0053] ("As used herein, the terms "program" and "executable" refer to any data representation that may be translated into a set of machine code instructions and may optionally include associated program data. The terms are used interchangeably herein. Program representations other than machine code include object code, byte code, and source code. Object code includes a set of instructions and/or data elements that either are prepared for linking prior to loading or are loaded into an execution environment. When in an execution environment, object code may include references resolved by a linker and/or may include one or more unresolved references. The context in which this term is used will make clear the state of the object code when it is relevant. This definition can include machine code and virtual machine code, such as Java™ byte code. As used herein, the terms "application", and "service" may be realized in one or more executables and/or in one or more hardware components. The terms are used interchangeably herein.") [0072] ("The arrangement of components illustrated in FIG. 4 may be adapted to operate in a number of execution environments to perform the method illustrated in FIG. 2A and/or the method illustrated in FIG. 2B. FIG. 4 is a block diagram illustrating the components of FIG. 3A, FIG. 3B, and/or analogs of the components of FIG. 3A and FIG. 3B, respectively adapted to operate in an execution environment 401 that includes and/or otherwise is provided by one or more nodes. FIG. 1 illustrates key components of an exemplary device that may at least partially provide and/or otherwise be included in an execution environment. The components illustrated in FIG. 4 may be included in or otherwise combined with the components of FIG. 1 to create a variety of arrangements of components according to the subject matter described herein.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a Samba client (not shown) included in and/or associated with file system 427, which in an aspect may be

	and/or may be included in the data store of execution environment 401 of first node 502. Samba URL 808 a may include user identification and authentication information to the Samba client if needed. Alternatively, the Samba client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the “Music” folder may be mounted as a drive in execution environment 401 of first node 502.) [0051]-[0052] (“A “user interface (UI) element handler” component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like. [] A representation of a program entity may be stored and/or otherwise maintained in a presentation space. As used in this document, the term “presentation space” refers to a storage region allocated and/or otherwise provided for storing presentation information, which may include audio, visual, tactile, and/or other sensory data for presentation by and/or on an output device. For example, a buffer for storing an image and/or text string may be a presentation space as sensory information for a user. A presentation space may be physically and/or logically contiguous or non-contiguous. A presentation space may have a virtual as well as a physical representation. A presentation space may include a storage location in a processor memory, secondary storage, a memory of an output adapter

	device, and/or a storage medium of an output device. A screen of a display, for example, is a presentation space.") [0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. In FIG. 4, descriptor generator component 414 is illustrated as a component of communications agent 403.") *See also* FIGs. 2A, 2B, 3A, 4, 6A, 6B, 6C, Original Claims 1, 8, 9, 11, 19, 20. **'906 Application (EX2004)** [0007] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.") [0008] ("Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying

	a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.") [0009] ("Methods and systems are described for processing a data object identification request in a communication. In one aspect, the method includes receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment. The method further includes detecting, by the second communications agent, a data object identification request included in the first message. The method still further includes generating, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request. The method additionally includes sending a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.")

	[0010] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment; detecting, by the second communications agent, a data object identification request included in the first message; generating, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request; and sending a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object.") [0011] ("Further, a system for processing a data object identification request in a communication is described. The system includes a content manager component for receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment. The system further includes a query handler component for detecting, by the second communications agent, a data object identification request included in the [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") *See also* FIGs. 2A, 2B, 6G,

Claim 14	
An apparatus, comprising: at least one non-transitory memory storing instructions; and one or more processors in communication with the at least one non-transitory memory, wherein the one or more processors execute the instructions to:	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)** [0007] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.") [0033] ("The execution environment is said to be the execution environment of the device and/or devices. An execution environment may be and/or may include a virtual execution environment including software components operating in a host execution environment. Exemplary devices included in and/or otherwise providing suitable execution environments for configuring according to the subject matter include personal computers, notebook computers, tablet computers, servers, portable electronic devices, handheld electronic devices, mobile devices, multiprocessor devices, distributed systems, consumer electronic devices, routers, communication servers . . . .") [0036] ("FIG. 1 illustrates a virtual processor memory 118 spanning at least part of physical processor memory 106 and may span at least part of persistent secondary storage 108. Virtual

	memory addresses in a memory address space may be mapped to physical memory addresses identifying locations in physical processor memory 106.") [0039] ("Execution environment 102 may include software components stored in persistent secondary storage 108, in remote storage accessible via a network, and/or in a processor memory.") [0102] ("[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network") [0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0162] ("FIG. **7** includes a dataflow message **708** that illustrates a detect data object identification request dataflow that may be internal to execution environment **401** of second node **504** and/or may include interoperation with another node via a network.")

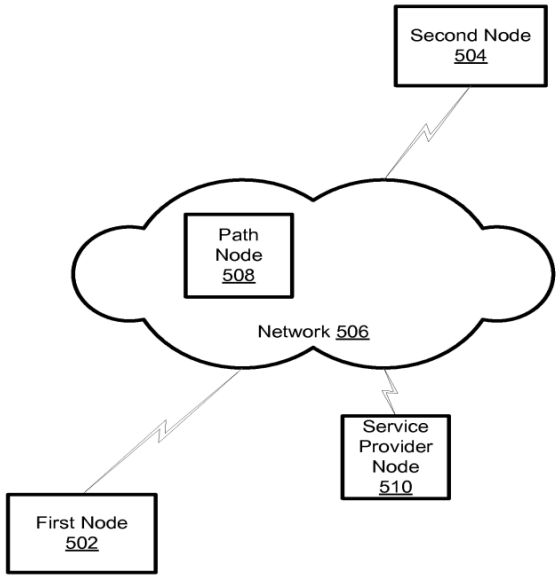


FIG. 5

See also FIGs. 1, 4, and 5; [0008]-[0011], [0033]-[0050] (description of FIG. 1 components, UIs generally, and web pages having UI elements); [0052], [0053], [0099]-[0101] (components of FIGs. 1 , 3A, 3B, and 4); [0102] ("arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network."); [0103] (FIG. 5 description of nodes in a network); [0111], [0115]-[0117], [0171], [0209], [0210]; Original claims 1-22

'635 application (EX2002)
FIG. 1; [0193]

display, at a first node, of a first user interface element, a second user interface element, and a third user interface element, the first user interface element, the second user interface element, and the third user interface element for use in creating a file request requesting one or more desired files;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0050] (“For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application”)

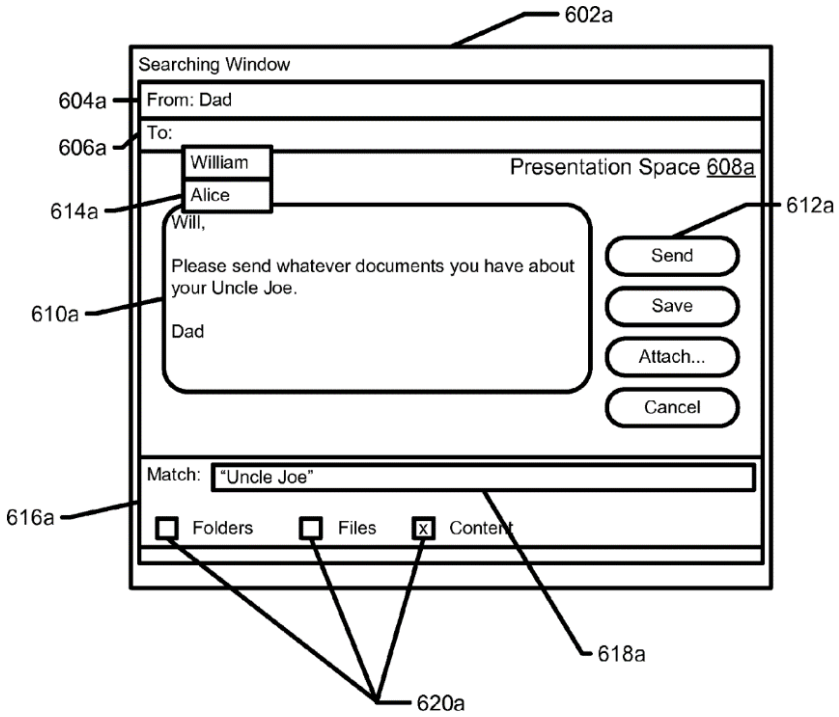


FIG. 6A

[0111] ("FIG. 6A-G illustrates various communications agent windows 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. In FIG. 6a, a communications agent window 602a includes a contactor user interface (UI) element 604a for presenting an identifier of a communicant in the role of a contactor in a communication represented by the communications agent window 602a. A communications agent window 602a also includes a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) included in the

	communication. A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more UI controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.") [0117] ("Data may be sent in a communication in response to a communicant input. A contactor may provide an input corresponding to send UI element 612a in FIG. 6A. The input may be received by presentation controller component 417 and/or one or more UI element handlers 415 corresponding to send UI element 612a. In response to detecting the input, presentation controller component 417 may provide data to be sent in the communication to one or more content handler components 413 according to the content type(s) of the data to be sent.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0125] ("FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415

	may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403.") [0126] ("DOI UI element 616a illustrates a user interface for a keyword based schema. Match textbox UI element 618a illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 620a allow a user to define a scope of a data object identification request by identifying whether a resulting data object identification request is to be applied to folders, files, and/or content of files in a file system and/or other data store of a receiving execution environment.") [0047] ("The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as 'UI elements'. More specifically, visual outputs of a user interface are referred to herein as 'visual interface elements'. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. . . .") [0175] ("In another aspect, a data object identification request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object identification request UI element and input from the user, query director component 412 may receive one or more additional data object matching criterion information and/or may receive input to modify a data object matching criterion identified in a received data object identification request. Query director component 412 may locate one or more data objects in response to receiving the additional criterion.") *See also* [0006]-[0011], [0020], [0100], [0111], [0115]-[0117]; Fig. 6A (e.g. Match window 616a); [0125]-[0126]; [0184];Original claims 1-22

	<u>’635 application (EX2002)</u> [0049] (“The user-detectable outputs of a user interface are generically referred to herein as “user interface elements” or abbreviated as ‘UI elements’. More specifically, visual outputs of a user interface are referred to herein as ‘visual interface elements’. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. An application interface may include one or more of the elements listed. Those skilled in the art will understand that this list is not exhaustive. The terms ‘visual representation’, ‘visual output’, and ‘visual interface element’ are used interchangeably in this document. Other types of UI elements include audio outputs referred to as ‘audio interface elements’, tactile outputs referred to as ‘tactile interface elements’, and the like.”) [0051] (““A “user interface (UI) element handler” component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.”)

[0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.")

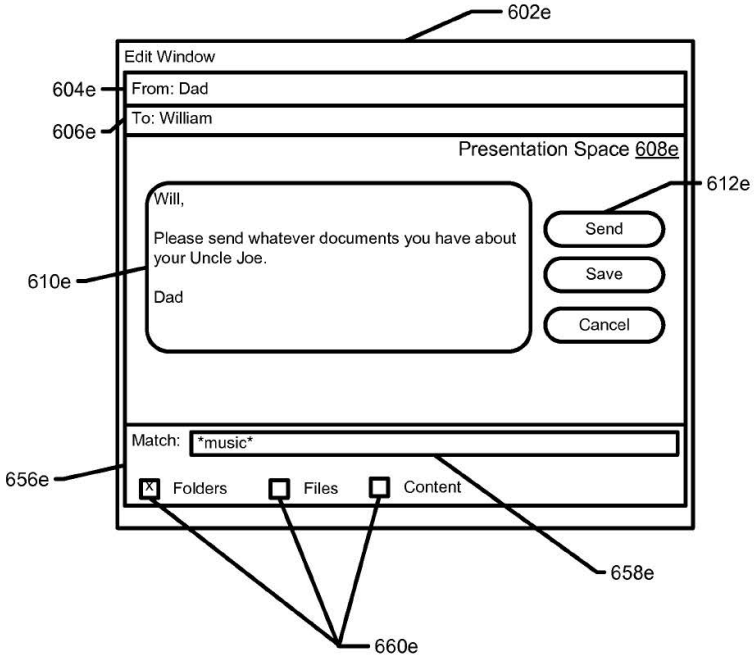
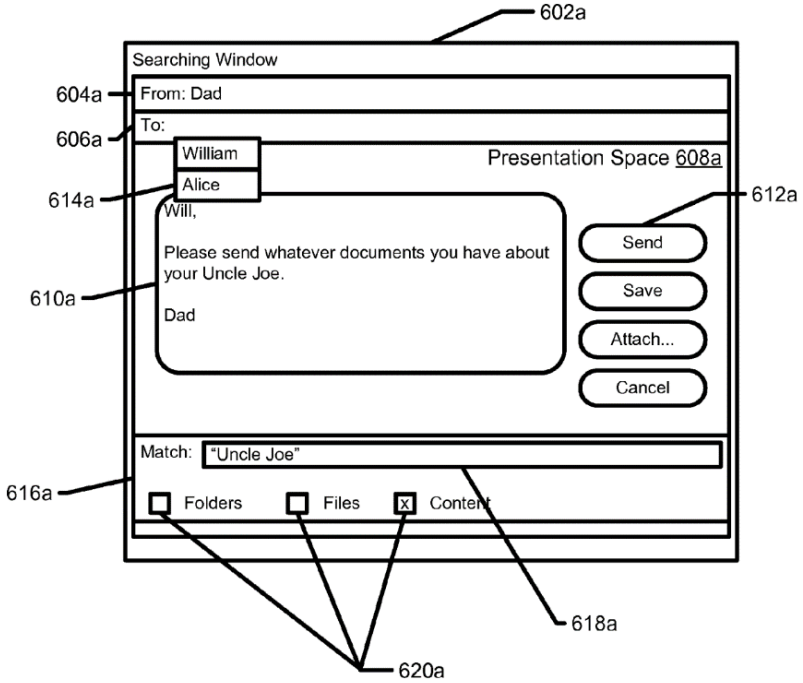


FIG. 6E

	[0145] (“MI UI element 656e illustrates a user interface for a keyword based schema. Match textbox U I element 658e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema.”) [0144] (“FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (Ml) UI element 656e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in Ml UI element 656e. MI U I element 656e and one or more U I elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.”) [0146] (“FIG. 7 includes data flow message that illustrates a create data object request (CIReq) data flow 714 that may be internal to execution environment 401 of first node 502 and/or may include a message received via network 506 by first node 502. With respect to FIG. 4, CIReq data flow 714 may illustrate a data flow to an IMC component 410 operating in execution environment 401 of first node 502. Match information may be identified to and/or otherwise provided to IMC component 410.”)

	[0153] (“In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a group, a requester, a relationship including a requesting user and a responding user, a keyword , a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.”) *See also* FIGs. 7, 8B, [0142]-[0143], [0147]-[0152], [0172], Original claim 12

receive, from the first node via at least one network, text associated with one or more desired files via the first user interface element that includes a textbox, the text describing the one or more desired files in connection with the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6A [0126] (“DOI UI element 616a illustrates a user interface for a keyword based schema. Match textbox UI element 618a illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 620a allow a user to define a scope of a data object

identification request by identifying whether a resulting data object identification request is to be applied to folders, files, and/or content of files in a file system and/or other data store of a receiving execution environment.")

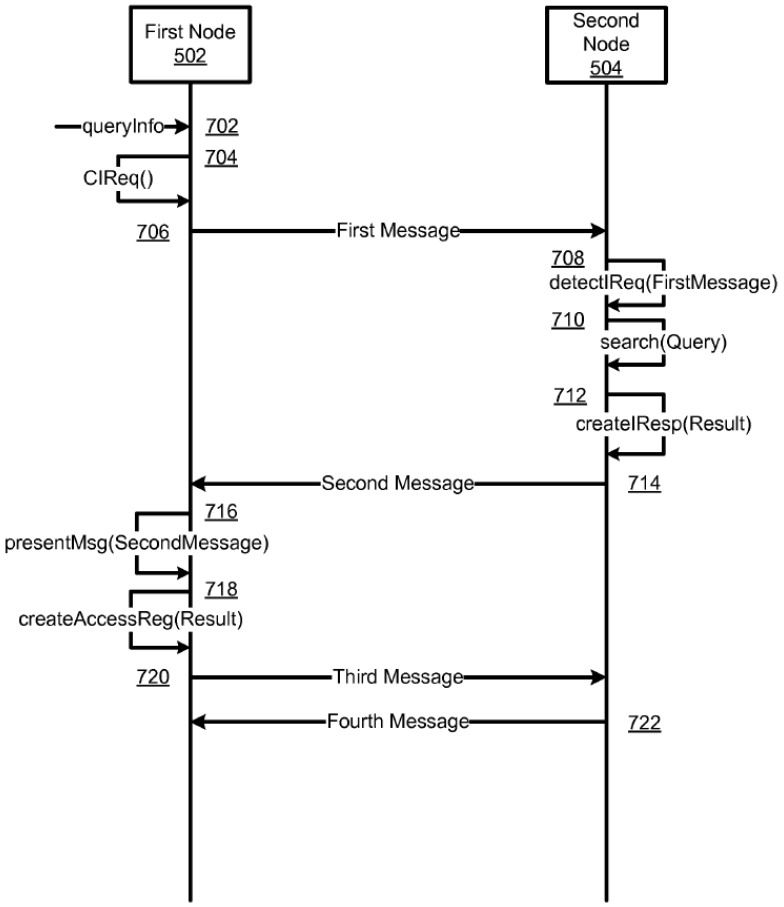
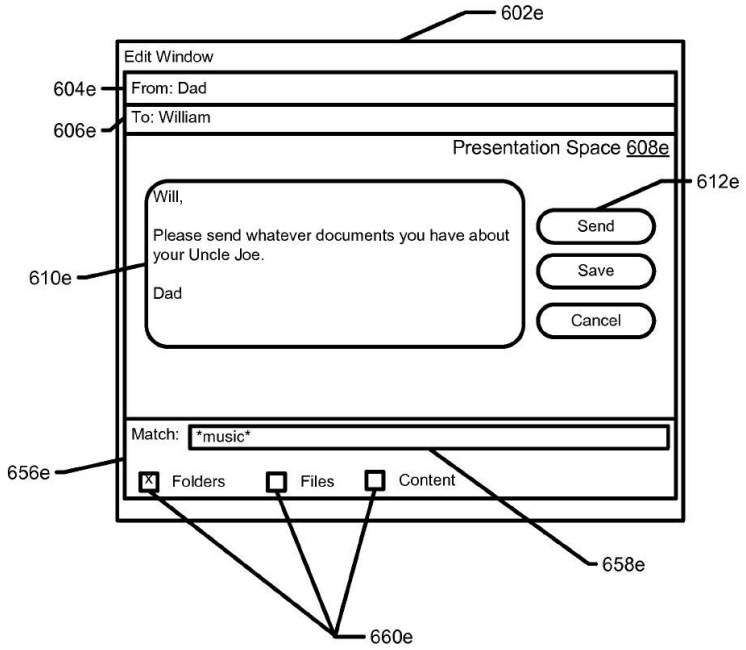


FIG. 7

	[0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.”) [0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”) [0133] (“FIG. 8A illustrates an ‘application/id-request’ MIME type identifier that may be defined to identify a schema for an XML-based language for specifying id-request XML documents. FIG. 8A illustrates id-request document 806a. Id-request document 806a, as illustrated, includes criterion tag elements 808a corresponding to the form elements in the user interface illustrated in FIG. 6A. A criterion tag element 808a identifies a data object matching criterion name, such as ‘query’ indicating that the criterion is a query expression or a portion thereof. A ‘type’ attribute identifies a schema with the identifier "keyword" for the expression. The criterion tag element 808a identifies a value for a query ‘Uncle Joe’ in a match-expression attribute.”) *See also* [0100], [0111]-[0117]; Fig. 6A (text box 618a); [0125]-[0126]; FIGS. 2A, 2B; FIG 8A; Original claims 1-22

	<u>’635 application (EX2002)</u> [0049] (“The user-detectable outputs of a user interface are generically referred to herein as “user interface elements” or abbreviated as ‘UI elements’. More specifically, visual outputs of a user interface are referred to herein as ‘visual interface elements’. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. An application interface may include one or more of the elements listed. Those skilled in the art will understand that this list is not exhaustive. The terms ‘visual representation’, ‘visual output’, and ‘visual interface element’ are used interchangeably in this document. Other types of UI elements include audio outputs referred to as ‘audio interface elements’, tactile outputs referred to as ‘tactile interface elements’, and the like.”) [0051] (““A “user interface (UI) element handler” component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual

	interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.")  FIG. 6E

	[0145] (“MI UI element 656e illustrates a user interface for a keyword based schema. Match textbox U I element 658e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema.”) [0144] (“FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (Ml) UI element 656e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in Ml UI element 656e. MI U I element 656e and one or more U I elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.”) [0146] (“FIG. 7 includes data flow message that illustrates a create data object request (CIReq) data flow 714 that may be internal to execution environment 401 of first node 502 and/or may include a message received via network 506 by first node 502. With respect to FIG. 4, CIReq data flow 714 may illustrate a data flow to an IMC component 410 operating in execution environment 401 of first node 502. Match information may be identified to and/or otherwise provided to IMC component 410.”)

	[0153] (“In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a group, a requester, a relationship including a requesting user and a responding user, a keyword , a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.”) *See also* FIGs. 7, 8B, [0147]-[0152], [0172], Original claim 12

receive, from the first node via the at least one network, an object associated with at least one email address via the second user interface element, the at least one email address for being used to send the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0111] (“FIG. 6A-G illustrates various communications agent windows 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. In FIG. 6a, a communications agent window 602a includes a contactor user interface (UI) element 604a for presenting an identifier of a communicant in the role of a contactor in a communication represented by the communications agent window 602a. A communications agent window 602a also includes a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) included in the communication. A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more UI controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.”) [0115] (“Sending data in a communication requires identifying a contactee. A contactee may be represented by a communications address in an address space of a communications protocol**.** In one aspect, information identifying a communications address may be received from a user/communicant of a communications agent in a user node. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 in FIG. 6A for first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 604. The user/communicant of first node 502 may enter a contactor alias, such as "Dad", . . . Communications agent 403 may identify a communication address associated with the contactor alias, "Dad". Other communications address(es), such as for one or more contactees, may be received similarly and/or in any suitable manner.”)

	[0117] (“Data may be sent in a communication in response to a communicant input. A contactor may provide an input corresponding to send UI element 612a in FIG. 6A. The input may be received by presentation controller component 417 and/or one or more UI element handlers 415 corresponding to send UI element 612a. In response to detecting the input, presentation controller component 417 may provide data to be sent in the communication to one or more content handler components 413 according to the content type(s) of the data to be sent.”) [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message.”) [0131] (“In FIG. 8A, a portion of an email communication 800a is illustrated formatted as a multipart/mixed content type including data object identification request portion 802a.”) [0193] (“A data object identification request, a data object identification response, an access request, and/or an access response may be included in an email . . . .”)

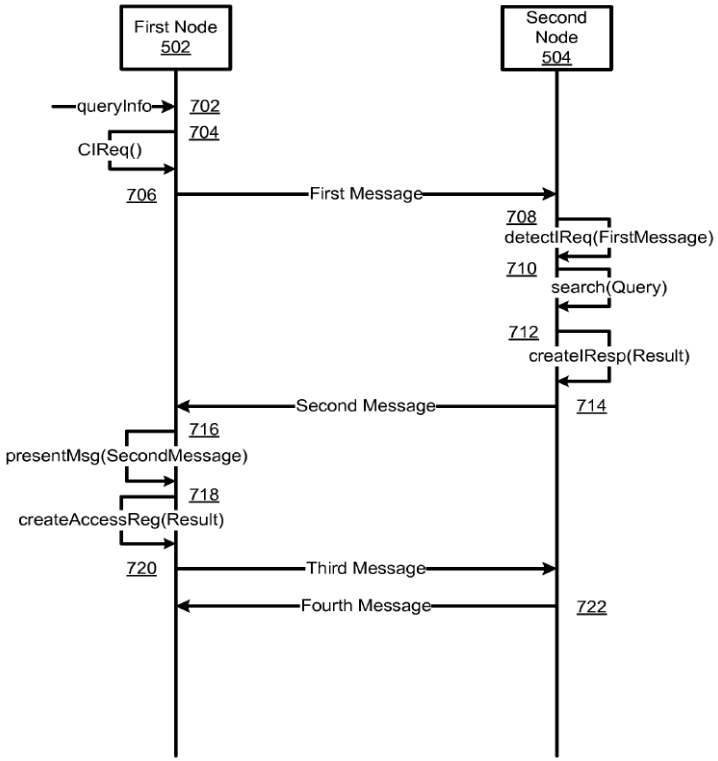
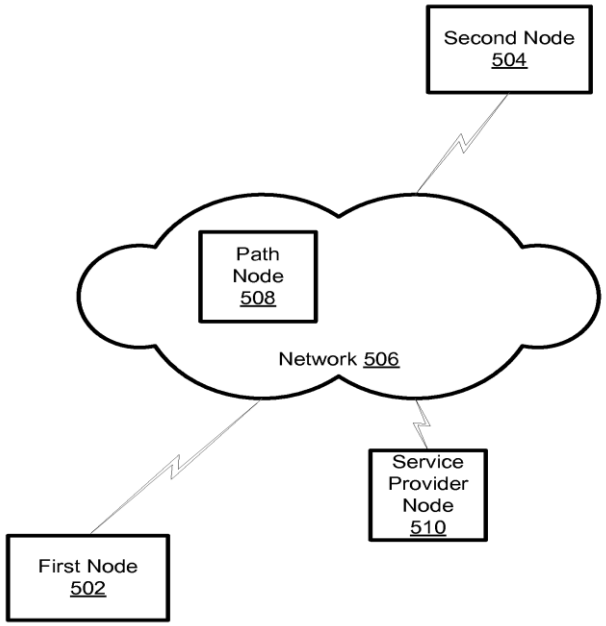
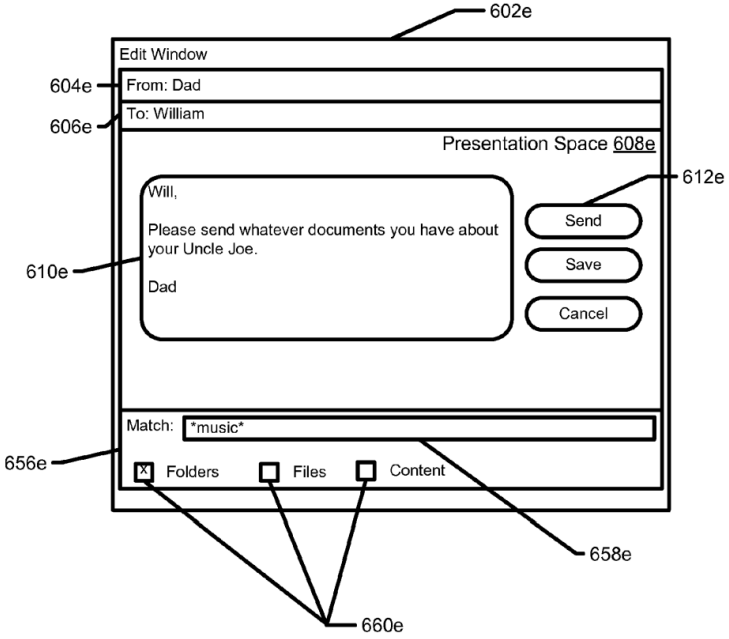


FIG. 7

[0123] ("FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.")

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object

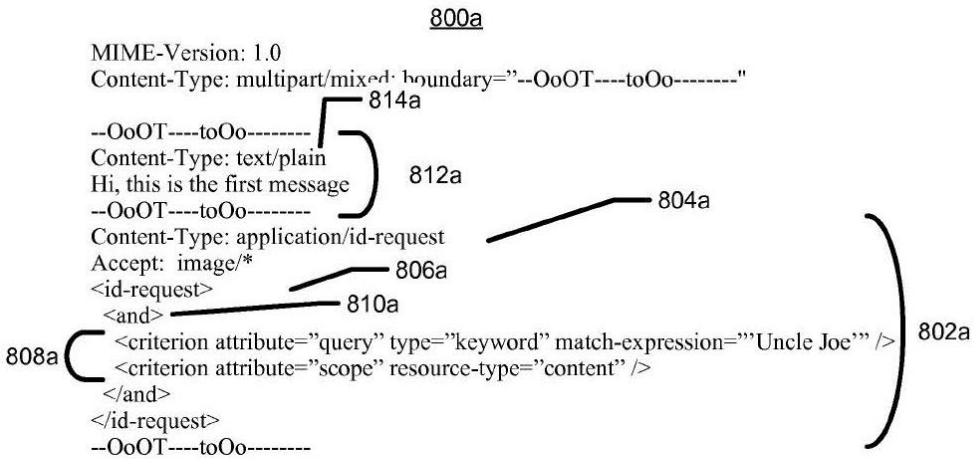
	information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")  FIG. 5

	See also Fig. 6A (Presentation Space 608a, contactee user interface element 606a); [0125], [0206]; Original claims 1-22 **<u>'635 application (EX2002)</u>**  FIG. 6E [0082] ("FIGS. 6A-H illustrate various user interfaces, some of which include window UI elements 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. A window UI element 602 may include a contactor UI element 604 for presenting an identifier of a communicant in the role of a contactor in a communication represented by the window UI element 602. A window UI element 602 may also include a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in

	the role of contactee(s) in the communication. A presentation space 608 may be included in each window UI element 602 for presenting a user message UI element 610 that my present a user message from one user to another. A presentation space 608 may also be provided for presenting one or more UI elements that correspond to user inputs for exchanging data in and/or otherwise managing a communication.") [0144] ("FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (MI) UI element 656 e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in MI UI element 656 e. MI UI element 656 e and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0145] ("MI UI element 656 e illustrates a user interface for a keyword based schema. Match textbox UI element 658 e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660 e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema."

	[0051] (“A “user interface (UI) element handler” component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.”) [0066] (“The term “communicant alias” as used herein refers to an identifier of a communicant in a communication where the communicant alias is not a communications address included in an address space of a communications protocol for sending and/or receiving data in the communication.”) [0086] (“A contactee is identified for sending data in a communication. A contactee may be represented by a communications address in an address space of a communications protocol. In one aspect, information identifying a communications address may be received from a communicant of a communications agent in an execution environment. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 a in FIG. 6A for execution environment 401 of first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 415. The user/communicant of execution environment 401 of first node 502 may enter a contactor alias, such as “William”, via a keyboard and/or select a predefined communicant alias presented

	in a selection UI control element via a UI element handler component 415. The user input may be detected by input driver 421. Corresponding input information may be routed to presentation controller 417 by GUI subsystem 423. GUI subsystem 423 operates to send presentation information to a display device via a graphics subsystem 425. Communication agent 403 may identify a communication address associated with the contactor alias, "William". Other communications address, such as for one or more contactees, may be received similarly and/or in any suitable manner. In an aspect, one or more contactor and/or contactee identifiers may be determined automatically.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0172] ("As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") *See also* [0051], [0142]-[0143], [0146]-[0153], FIGs 7, 8B.

based on the text associated with the one or more desired files and the object associated with the at least one email address; generate a first message addressed to the at least one email address and including the file request, the first message including a fourth user interface element and the text associated with the one or more desired files describing the one or more desired files in connection with the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  Fig. 8A [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message. ...”) [0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.”)

	[0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”) [0127] (“Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. Accordingly, a system for processing a data object identification request in a communication includes means for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. For example, the arrangement in FIG. 3A, includes identification request constructor component 304 that is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. FIG. 4 illustrates identification request constructor component 404 as an adaptation and/or analog of identification request constructor component 304 in FIG. 3A. One or more identification request constructor components 404 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, according to a first

	communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. In FIG. 4, identification request constructor component 404 is illustrated as a component of communications agent 403.") [0128] ("FIG. 7 includes dataflow message 704 that illustrates a data flow internal to execution environment 401 of first node 502 and/or message received via network 506 by execution environment 401 of first node 502. Dataflow message 704 illustrates a create data object identification request (CIReq) dataflow to an IRC component 404 operating in execution environment 401 of first node 502. Data object information may be identified to and/or otherwise provided to IRC component 404. A query handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide data object information to an IRC component 404.")

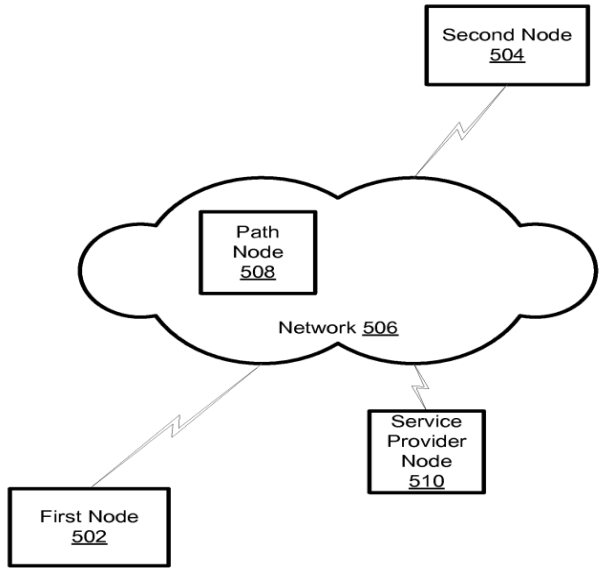
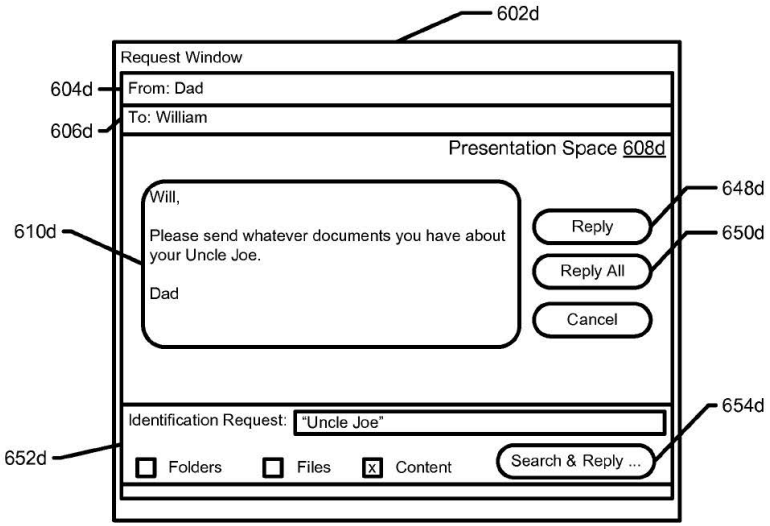


FIG. 5

See also [0092]-[0093], [0123]-[0124]; [0129]-[0135] (message creation and contents); [0175], [0180]; [0206], FIG. 2A, FIG. 6B, FIG. 6D, FIG. 7 (flowchart of request process); FIG. 8A; Original claims 1-22

’635 application (EX2002)

FIG. 6E, 7, 8B, [0142]-[0153]; *see also* [0073], [0172]

send, to a second node via the at least one network, the first message;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0127] (“Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user…”)  FIG. 6D

	[0162] (“FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object identification request may include and/or otherwise identify a data object matching criterion identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.”) [0166] (“FIG. 6D illustrates a window, request window UI element 602d, presenting an exemplary representation of a message received from execution environment 401 of first node 502 in a communication with second node 504. As in FIG. 6A, request window UI element 602d includes a contactor UI element 604d, a contactee UI element 606d, a presentation space 608d of the request window UI element 602d that includes a user message UI element 610d including a text portion of the message provided by the communicant represented by execution environment 401 of first node 502. The various UI elements may be presented by one or more UI element handler components 415 in communications agent 403 operating in execution environment 401 of second node 504. A reply UI element 648d and a reply all UI element 648d are illustrated as exemplary UI elements in presentation space 608d presented to allow a user of execution environment 401 of second node 504 to respond and/or otherwise perform an operation based on the received message. FIG. 6D illustrates a data object identification request

	UI interface element 652d where the data object identification request identified in the message from execution environment 401 of first node 502 is represented to the communicant represented by the execution environment 401 of second node 504. In an aspect, the communicant is allowed to authorize the request by providing an input corresponding to a search and reply UI element 654d. In another aspect, the communicant may be allowed to modify the received data object identification request.") *See also* [0092]-[0093], [0123]-[0124]; [0128]-[0131]; [0133], [0134]-[0136], [0157], [0175], [0180]; FIGs. 2A, 2B, 3A, 4; FIG. 6D; FIG. 7 (flowchart of request process); FIG. 8A; Original claims 1-22 **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") *See also* FIG. 6E, 7, 8B, [0142]-[0173]; [0073]

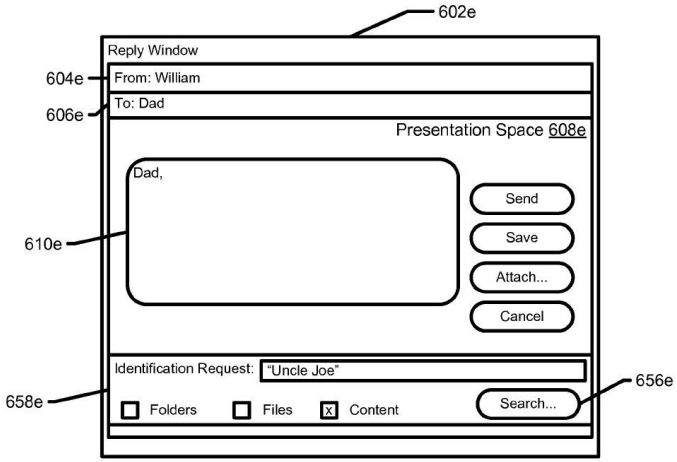
receive, from the second node via the at least one network, a message associated with a selection of the fourth user interface element of the first message for indicating that at least one file is to be uploaded in response to the file request;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6E [0151] (“ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.”) [0162] (“FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to

	FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object identification request may include and/or otherwise identify a data object matching criterion identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.") [0167] ("FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message

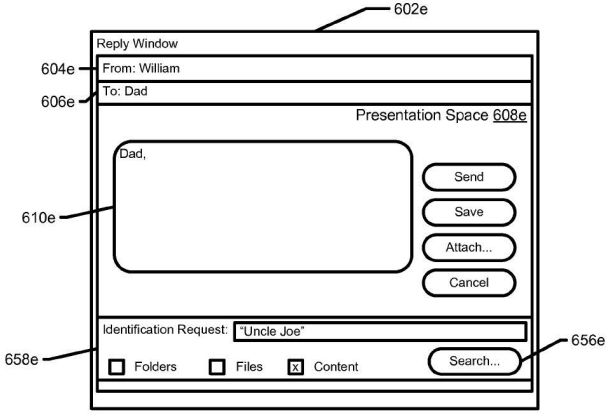
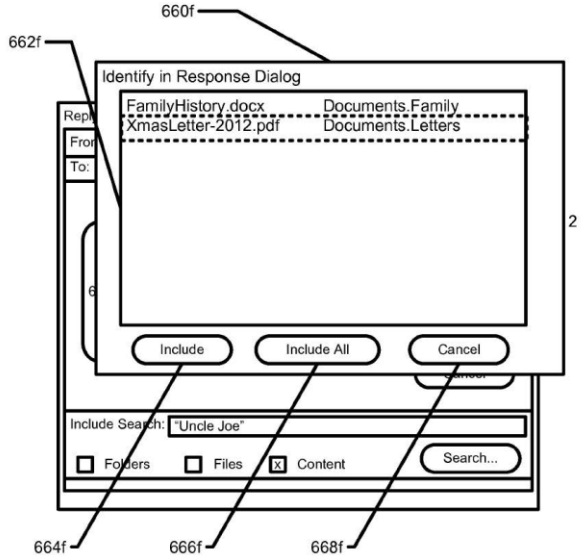
	710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0179] ("FIG. 8B illustrates an exemplary data object identification response described in detail above. Second message portion 802b illustrates a user message portion that may be included in second message 714. Note that a user message portion may be empty or absent in some aspects. Data object identification response portion 804b illustrates the data object identification response as a list of URls identifying data objects in a file system of execution environment 401 of second node 504. User Message portion 802b and data object identification response portion 804b may be provided by respective content handler components 413 to content manager component 409. Content manager component 409 may construct the content as illustrated in FIG. 8B and/or otherwise provide for encoding, translating, combining, and/or otherwise preparing the content for sending in second message 714.") [0181] ("In another aspect, a data object located and/or otherwise identified in response to receiving a data object identification request must meet match a query identified by the data object identification request. Alternatively or additionally, a data object identification request may include an instruction and/or input for generating a data object. Locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object identification request may identify the template. A query handler component 402 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to an access request, execution environment 401 of second node 504 may return the document.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.")

	[0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”) [0201] (“With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.”) *See also* [0100], [0111], [0137] (discusses receiving a message from second node); FIG. 5, FIG. 6B (selecting element 628b to bring up window 630b); [0146]-[0148]; FIG. 2B, [0157]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0170], [0173]; [0206]; FIG. 7; Original claims 1-22.

	<u>’635 application (EX2002)</u> [0171]-[0172] (“Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.”) [0069] (“The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0112] (“FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be

	separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed

	Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.") *See also* [0157], FIGs. 8A, 8B; Original claim 14

in response to the receipt of the message associated with the selection of the fourth user interface element, send a first communication over the at least one network that results in display, at the second node, of a fifth user interface element for use in uploading the at least one file in response to the file request; receive, from the second node via the at least one network, a message associated with a selection of at least one file for being used to upload the at least one file in response to the file request; receive, from the second node via the at least one network, a message associated with a selection of the fifth user interface element;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  FIG. 6E  FIG.6F [0167] (“FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all

	of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0172] "(FIG. 6F, illustrates an identify in response dialog UI element 660f that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to processing the data object identification request to locate matching data objects. In an aspect, a user of communications agent 403 operating in execution environment 401 of second node 504 may provide input corresponding to search UI element 656e in reply window UI element 602e in FIG. 6E. A corresponding UI element handler component 415 may operate to invoke query director component 412 to perform and/or otherwise to provide for performing one or more operations to locate one or more data objects based on the data object identification request. Query director component 412 may interoperate with one or more UI element handlers 415, directly or indirectly via query handler component 402, in an aspect, to present the UI element 660f and its contents. Identify in response dialog UI element 660f includes results pane UI element 662f including representations of two data objects located and/or otherwise identified by query director component 412 based on the data object identification request. A user may select one or more data objects to identify in a data object identification response. An include UI element 664f may be presented, as illustrated, to allow a user to instruct communications agent 403 to identify one or more selected data objects in

	results pane U I element 662f in a data object identification response. An include all UI element 666f is illustrated to allow for user input instructing communications agent 403 to identify all data objects represented in the UI element 660f in a data object identification response in a communication with execution environment 401 of first node 502. A cancel UI element 668f allows a user to end presentation of dialog UI element 660f. A user may opt to identify no data objects represented in the dialog UI element 660f and located and/or otherwise identified by query director component 412 based on the data object identification request.") *See also* FIGS. 5, 6C, 6G; [0146]-[0148]; [0150]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0206], FIG. 7 **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") [0069] ("The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0073] ("As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a

	browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0112] (“FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.”) [0116] (“In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.”)

	[0117] (“For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.”) [0118] (“Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.”) [0119] (“Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.”) [0130] (“FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the “Music” folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.”) *See also* [0157], FIGs. 8A, 8B; Original claim 14

in response to the receipt of the message associated with the selection of the fifth user interface element, generate a second message that includes a sixth user interface element, and that does not include a file attachment with the second message, the second message indicating that the file request has been responded to;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0008] (“Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.”)

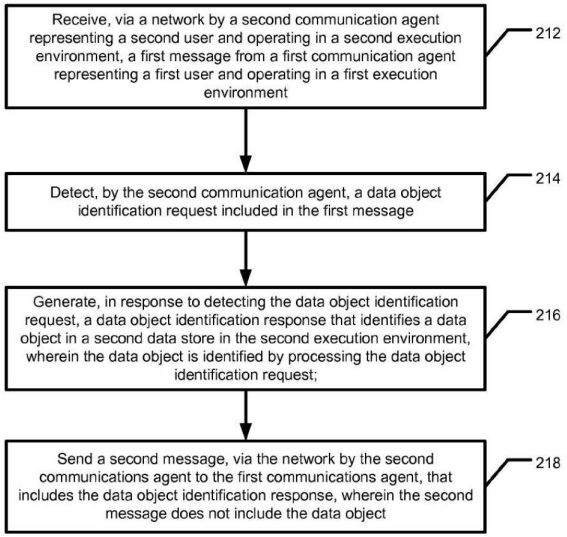

Receive, via a network by a second communication agent representing a second user and operating in a second execution environment, a first message from a first communication agent representing a first user and operating in a first execution environment
212
Detect, by the second communication agent, a data object identification request included in the first message
214
Generate, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request;
216
Send a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object
218

FIG. 2B

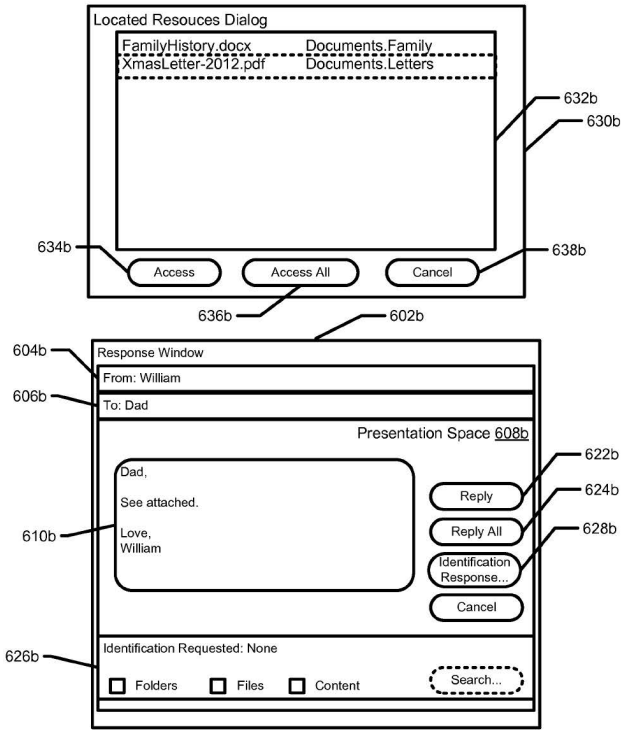

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
Access
Access All
Cancel
638b
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
Dad,
See attached.
Reply
624b
Reply All
610b
Love,
William
628b
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

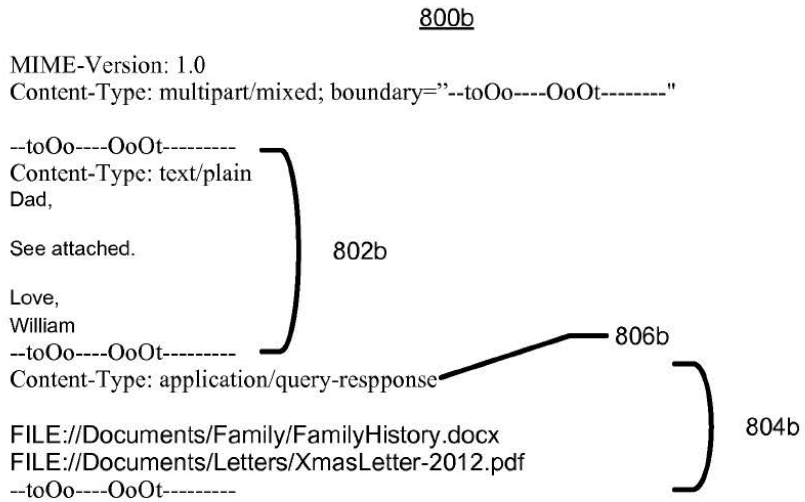


FIG. 8B

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

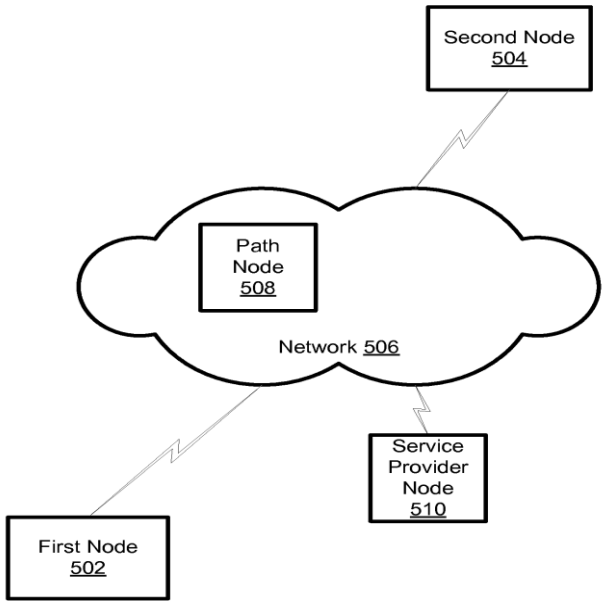

Second Node
504
Path
Node
508
Network 506
Service
Provider
Node
510
First Node
502

FIG. 5

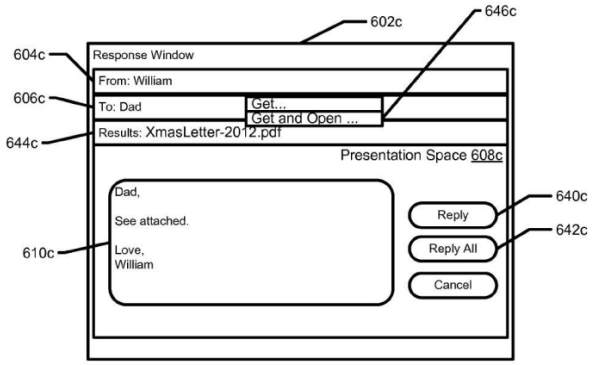


FIG. 6C

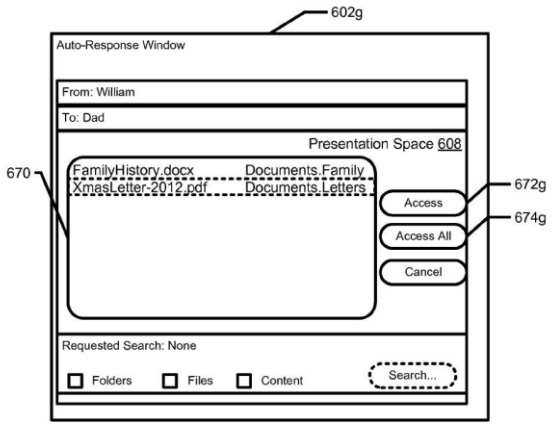
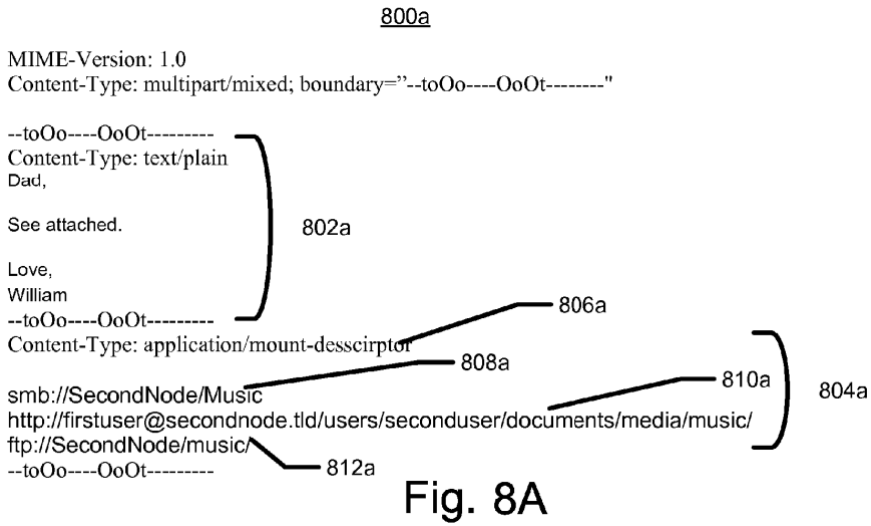
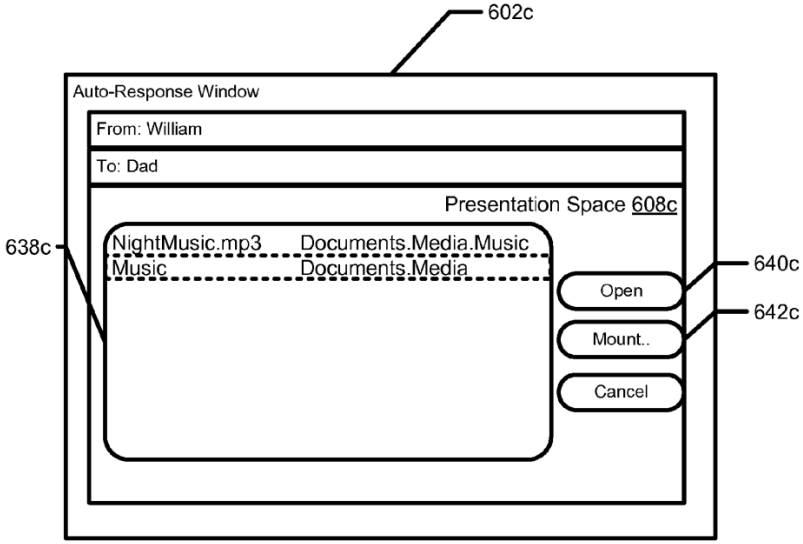


FIG. 6G

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by

	execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") *See also* [0007], [0009]-[0011], [0150]; [0137] (discusses receiving a message from second node, "wherein the data object is not received in the second message"); FIG. 8B, [0141]-[0142]; FIG. 6C, [0150]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 2B, [0170]; [0171]-[0173]; [0176]; [0179]; [0200]; [0206]. **'635 application (EX2002)**  Fig. 8A

	[0096] (“FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user readable message portion 802 a and a mount descriptor portion 804 a described in more detail below. The received communication may include a message addressed to a user represented by communications agent 403 in execution environment 401 of first node 502.”) [0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the “Music” found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the “Music” folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the “Music” folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the “Music” folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.”) [0168] (“In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an instruction and/or input for generating a data object, and locating the data object may include

	generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.")  FIG. 6C [0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a communication without a user message. Rather than a UI element for

	displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.") *See also* [0007]-[0013]; Original claims.

send, to the first node via the at least one network, the second message;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0008] (“Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.”)

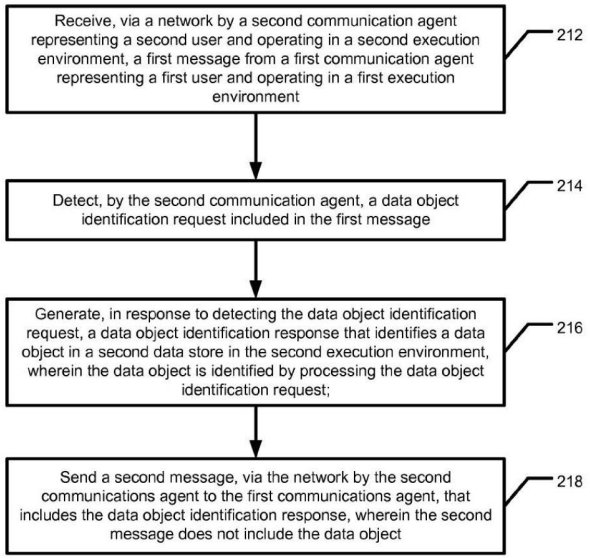

Receive, via a network by a second communication agent representing a second user and operating in a second execution environment, a first message from a first communication agent representing a first user and operating in a first execution environment
212
Detect, by the second communication agent, a data object identification request included in the first message
214
Generate, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request;
216
Send a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object
218

FIG. 2B

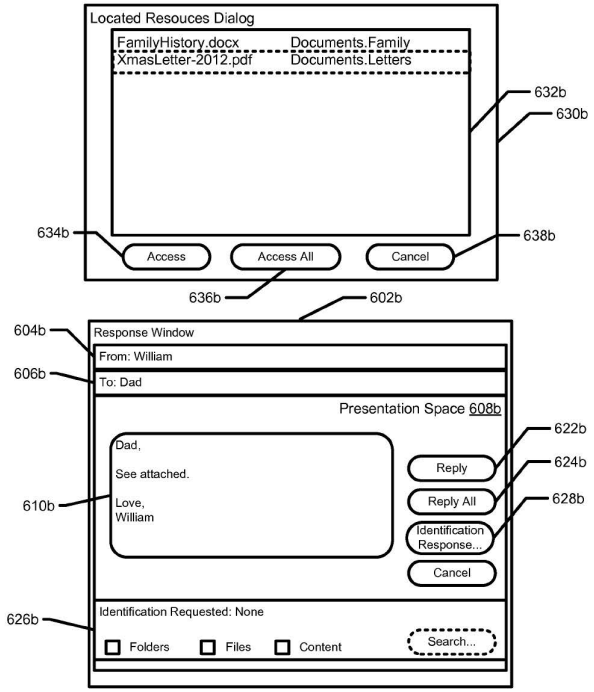

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
638b
Access
Access All
Cancel
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
624b
628b
Dad,
See attached.
610b
Love,
William
Reply
Reply All
Identification Response...
Cancel
Identification Requested: None
626b
Folders
Files
Content
Search...

FIG. 6B

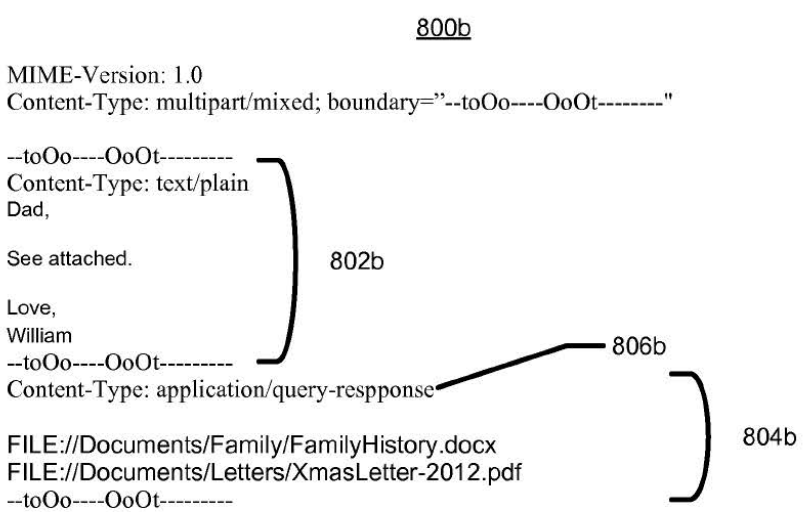


FIG. 8B

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

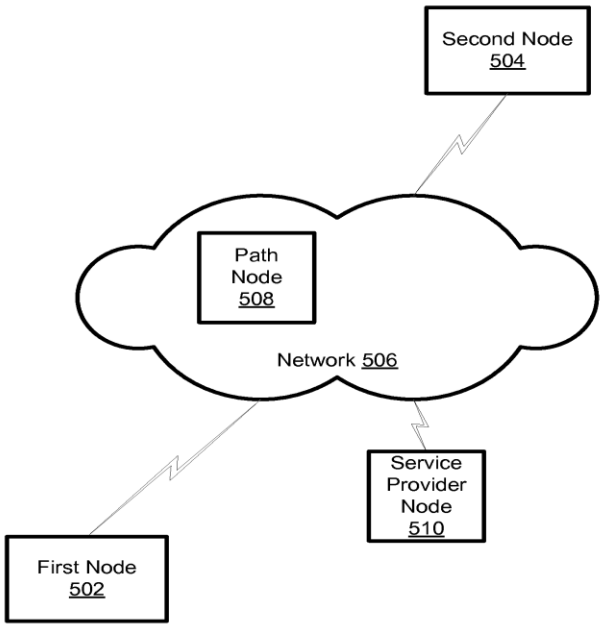

Second Node
504
Path
Node
508
Network 506
Service
Provider
Node
510
First Node
502

FIG. 5

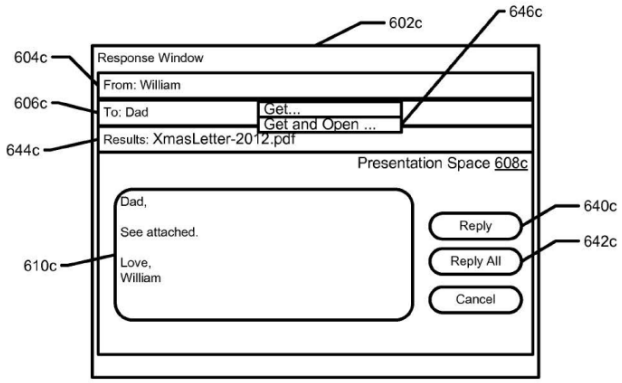


FIG. 6C

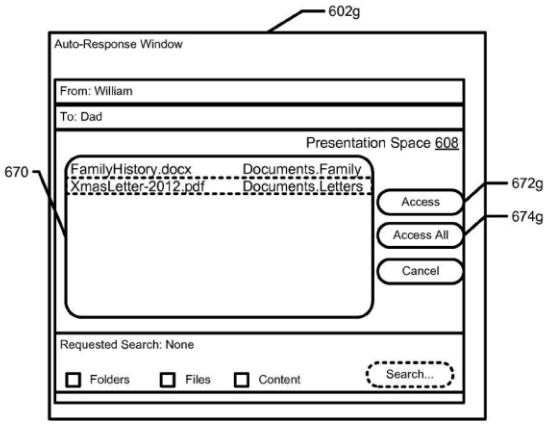


FIG. 6G

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element

	670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.”)

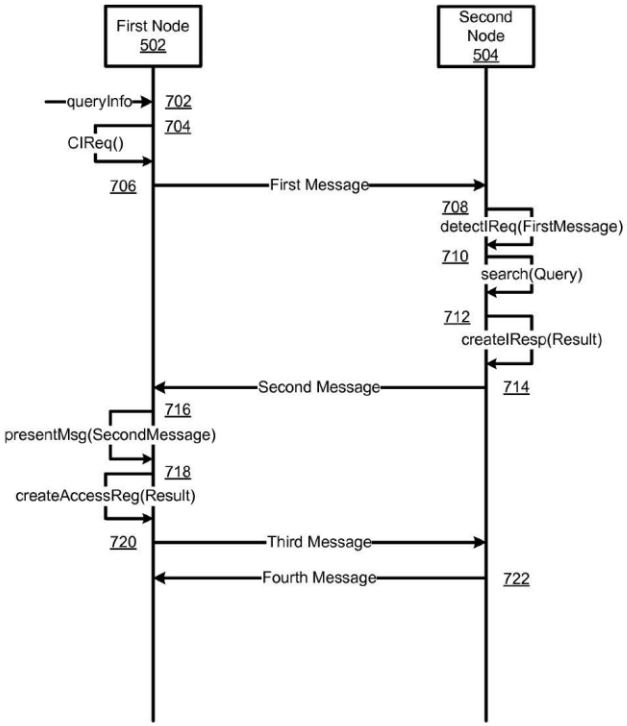


FIG. 7

See also [0150]; FIGS. 6C, 6G; FIGS. 2A, 2B; [0137] (discusses receiving a message from second node, "wherein the data object is not received in the second message"); FIG. 7, [0138]; [0139]-[0140] (data communications); [0141]; [0151]; [0176]; [0177]

'635 application (EX2002)

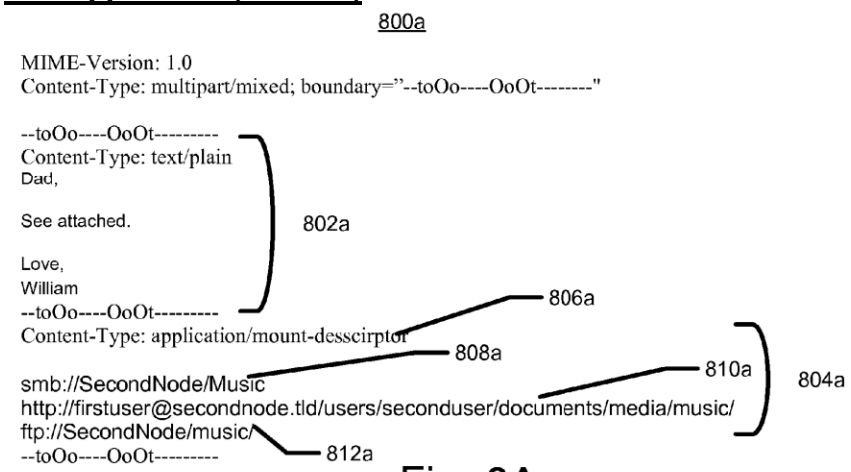


Fig. 8A

[0096] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user readable message portion 802 a and a mount descriptor portion 804 a described in more detail below. The received communication may include a message addressed to a user represented by communications agent 403 in execution environment 401 of first node 502.")

[0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol

	(HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the “Music” folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the “Music” folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the “Music” folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.”) [0168] (“In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an instruction and/or input for generating a data object, and locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.”)

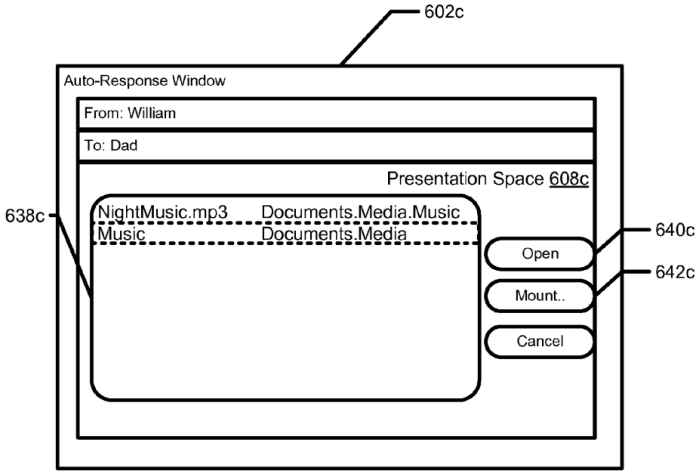


FIG. 6C

[0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a communication without a user message. Rather than a UI element for displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.")

See also FIG. 7

receive, from the first node via the at least one network, a message associated with a selection of the sixth user interface element of the second message;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0146] (“FIG. 68 illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes a user message UI element 610b, analogous to similar UI elements described with respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409 in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.”) [0147] (“Response window UI element 602b in FIG. 68 includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect,

	one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **68,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution environment 401 of first node 502 to request one or more data objects identified in the data object identification response.")

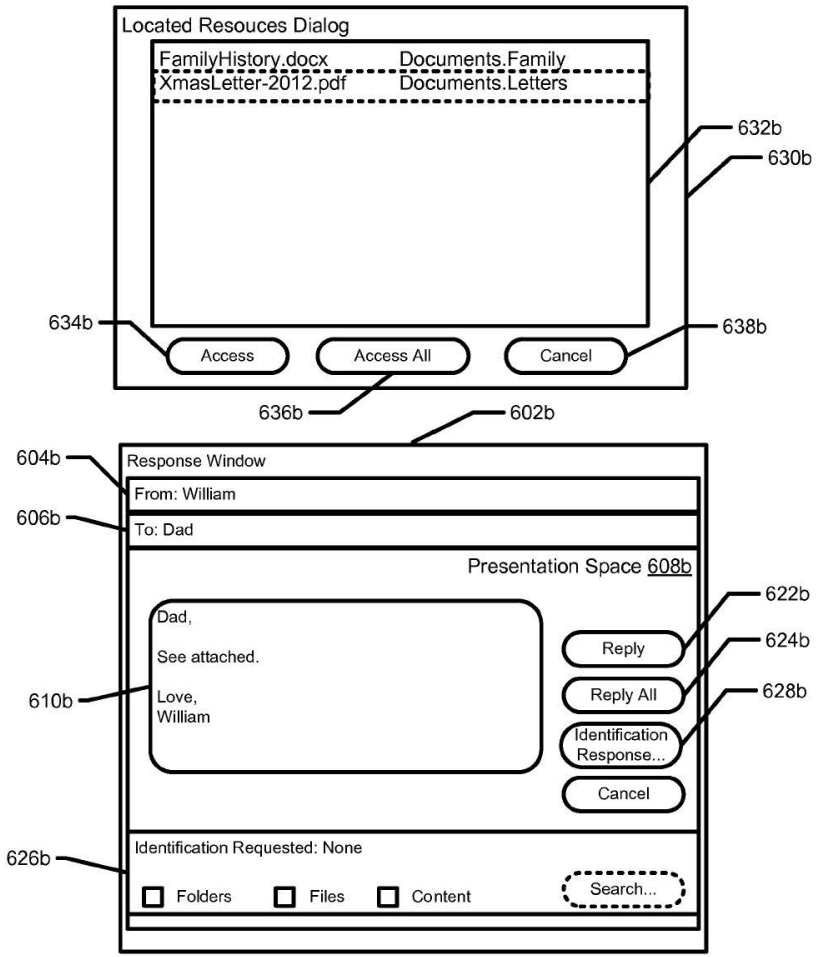

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
Access
Access All
Cancel
638b
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
Dad,
See attached.
Love,
William
624b
610b
Reply
Reply All
628b
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

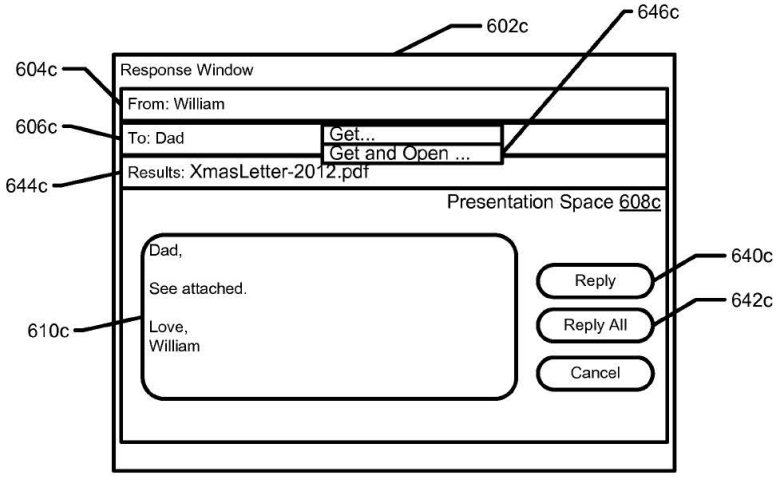


FIG. 6C

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response.

	FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

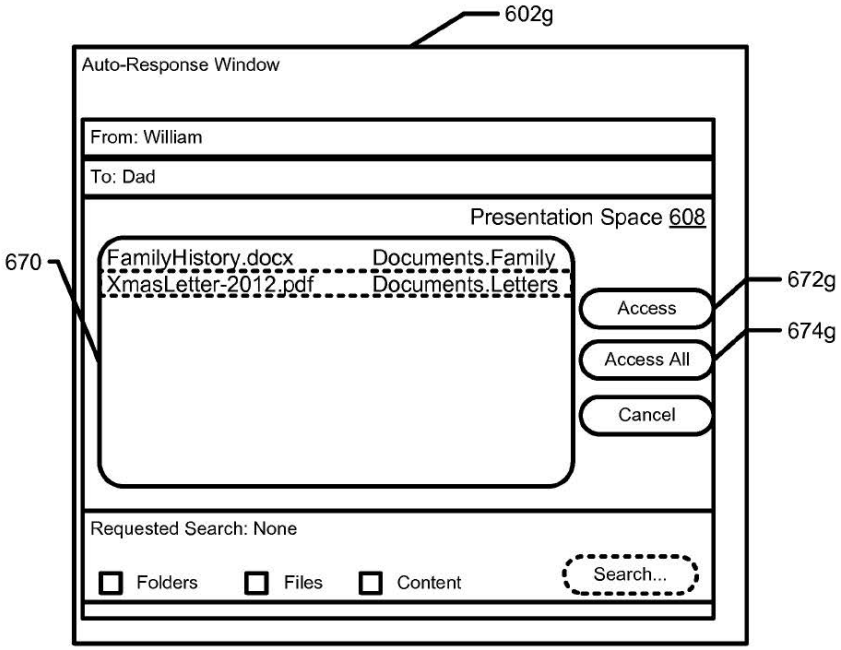

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
Access
Access All
Cancel
672g
674g
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

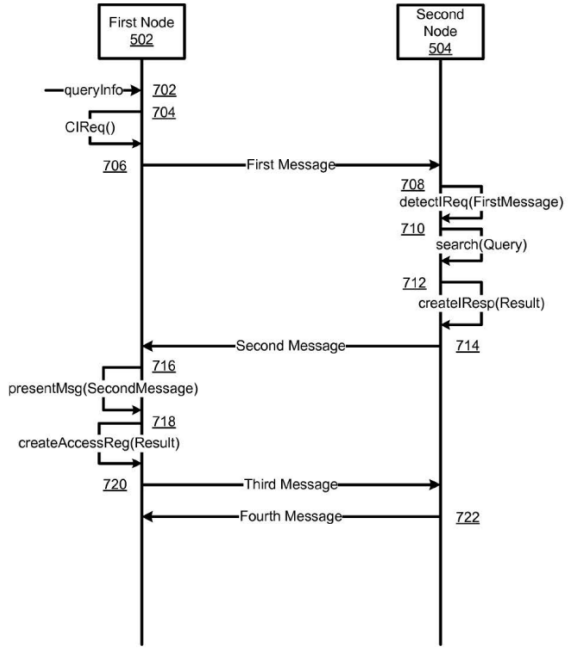


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A. One or more access request constructor components 408 operate in an execution environment 401. The

	system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") [0144] ("FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.") *See generally* [0150]-[0151], [0142], [179]; [0143]-[0144] (sending "message" to server to access file(s)); FIG. 7 (message 720); [0149]; [0153], FIG. 2A; [0155]; [0201]; [0206] **'635 application (EX2002)** [0106] ("In response to detecting the mount descriptor in first message 702, execution environment 401 of first node 502 may present a view window UI element 602 b in FIG. 6B including data received in the first message 702 to a user of execution environment 401 of first node 502 as described with respect to FIG. 6A. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" menu UI element 626 b 1, identified in a mount descriptor received from execution environment 401 of second node 504. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, in response to an

	input corresponding to “Save . . . ” menu UI element 626 b 2, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.”) [0130] (“FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a”) [0152] (“FIG. 7 includes a data flow that illustrates a previous message 716 (with respect to first message 702) including a data object request. Previous message 716, is shown, may be sent via network 506 by execution environment 401 of first node 502 to execution environment 401 of second node 504. In an aspect, first message 702 may be sent in response to previous message 716.”) *See also* [0112], [0115]-[0119], [0142]-[0173], FIGs. 2A, 2B, 7.

in response to the receipt of the message associated with the selection of the sixth user interface element of the second message, send a second communication over the at least one network that results in display, at the first node, of a reference to the at least one file; and allow access, at the first node, to the at least one file via the reference.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0144] (“FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0102] (“[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network”)

	[0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0146] ("FIG. 6B illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes a user message UI element 610b, analogous to similar UI elements described with

	respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409 in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.") [0147] ("Response window UI element 602b in FIG. 6B includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect, one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **6B,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution

	environment 401 of first node 502 to request one or more data objects identified in the data object identification response.") [0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

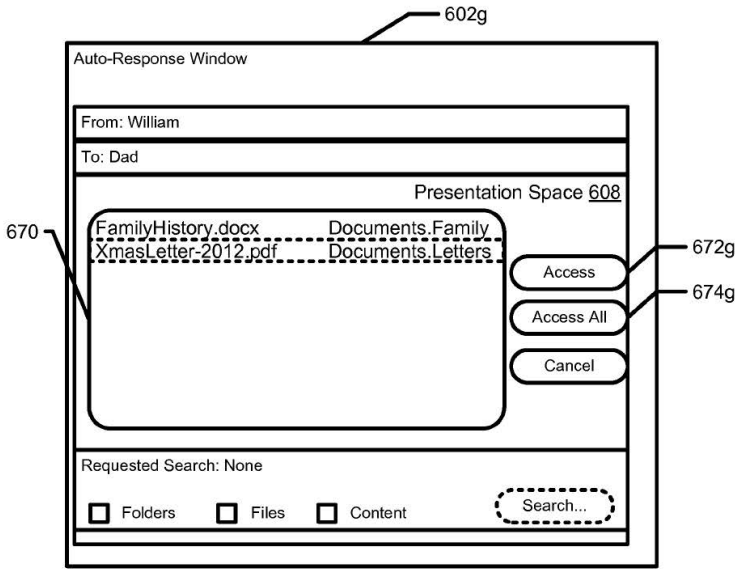

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
672g
Access
674g
Access All
Cancel
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

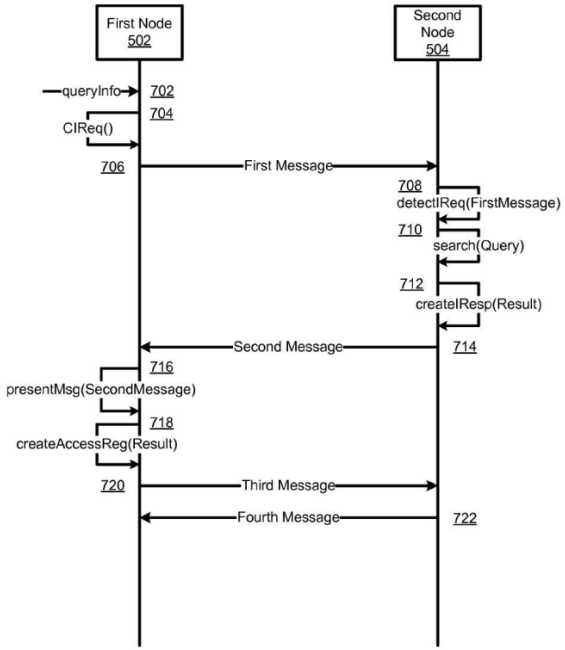


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A. One or more access request constructor components 408 operate in an execution environment 401. The

	system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") *See also* [0007]-[0011], FIGs. 2A, 2B, 5, 6G, **'635 application (EX2002)** [0112] ("…A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502…") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music

	folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0069] ("…Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.")

	[0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user…") [0181] ("A data object request and/or a mount descriptor or respective portions thereof may include at least a portion of a keyword expression, a regular expression, expression including a Boolean operator, an expression including a precedence information, and a structured query language statement." [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0152] ("FIG. 7 includes a data flow that illustrates a previous message 716 (with respect to first message 702) including a data object request. Previous message 716, is shown, may be sent via network 506 by execution environment 401 of first node 502 to execution environment 401 of second node 504. In an aspect, first message 702 may be sent in response to previous message 716.")

Claim 15	
The apparatus of claim 14, wherein the apparatus is configured such that: the first message and the second message are email messages; the object includes the at least one email address or an alias associated with the at least one email address;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)** [0115] ("Sending data in a communication requires identifying a contactee. A contactee may be represented by a communications address in an address space of a communications protocol**.** In one aspect, information identifying a communications address may be received from a user/communicant of a communications agent in a user node. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 in FIG. 6A for first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 604. The user/communicant of first node 502 may enter a contactor alias, such as "Dad", . . . Communications agent 403 may identify a communication address associated with the contactor alias, "Dad". Other communications address(es), such as for one or more contactees, may be received similarly and/or in any suitable manner.") [0118] ("The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message.") [0131] ("In FIG. 8A, a portion of an email communication 800a is illustrated formatted as a multipart/mixed content type including data object identification request portion 802a.") [0193] ("A data object identification request, a data object identification response, an access request, and/or an access response may be included in an email . . . .") *See also* FIG. 2A, 2B, 5, 7, 8A, 8B.

	<u>'635 application (EX2002)</u> [0082] ("FIGS. 6A-H illustrate various user interfaces, some of which include window UI elements 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. A window UI element 602 may include a contactor UI element 604 for presenting an identifier of a communicant in the role of a contactor in a communication represented by the window UI element 602. A window UI element 602 may also include a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) in the communication. A presentation space 608 may be included in each window UI element 602 for presenting a user message UI element 610 that my present a user message from one user to another. A presentation space 608 may also be provided for presenting one or more UI elements that correspond to user inputs for exchanging data in and/or otherwise managing a communication.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. 8A illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.") *See also* FIGs. 7, 8A, 8B, [0076] (email clients), [0087] ("An email and/or instant message may use a store and forward model of delivery."), [0089], [0095] (email), [0096] (email); [0122], [0149], [0179].

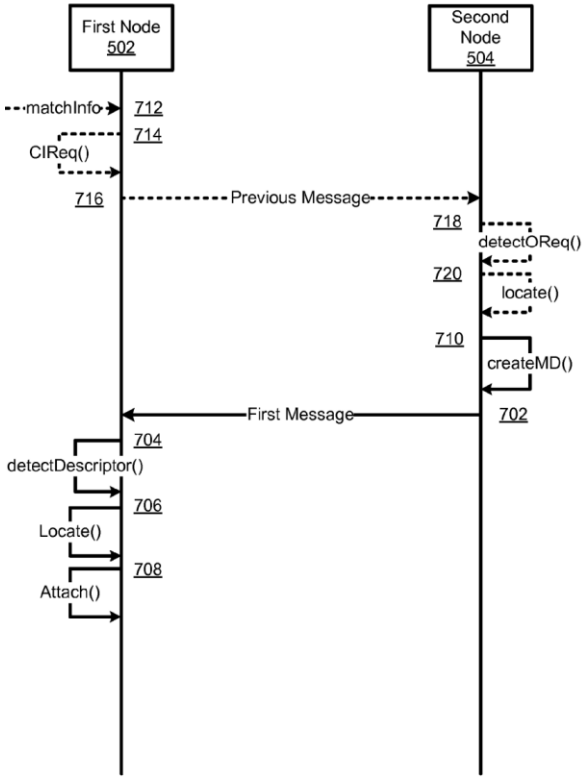
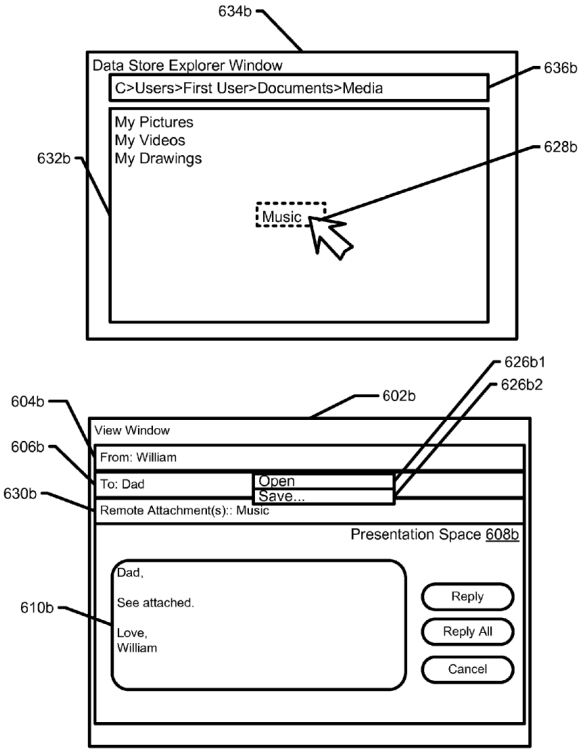
the second message is at least partially pre-written and is automatically generated without user involvement after the receipt of the message associated with the selection of the fifth user interface element; and	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0072] (“The arrangement of components illustrated in FIG. 4 may be adapted to operate in a number of execution environments to perform the method illustrated in FIG. 2A and/or the method illustrated in FIG. 2B. FIG. 4 is a block diagram illustrating the components of FIG. 3A, FIG. 3B, and/or analogs of the components of FIG. 3A and FIG. 3B, respectively adapted to operate in an execution environment 401 that includes and/or otherwise is provided by one or more nodes. FIG. 1 illustrates key components of an exemplary device that may at least partially provide and/or otherwise be included in an execution environment. The components illustrated in FIG. 4 may be included in or otherwise combined with the components of FIG. 1 to create a variety of arrangements of components according to the subject matter described herein.”) [0150] (“In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment

	401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0173] ("Data objects located and/or otherwise identified by query director component 412 may be identified to IRB component 414 to construct and/or otherwise create a data object identification response identifying the data objects. As described above, in an aspect, a user may be allowed to select which data objects, if any, are to be identified in a data object identification response from one or more data objects located and/or otherwise identified by query director component 412 based on the data object identification response. In another aspect, communications agent 403 in FIG. 4 may invoke query director component 412 automatically to locate one or more data objects based on a data object matching criterion identified in a data object identification request and in response to detecting a data object identification request in a communication. In another aspect, communications agent 403 may invoke query director component 412 automatically to locate one or more data objects based on a data object identification request in a communication in response to detecting an indication to present, via a output device, some or all of a user message received in a message in the communication. Query director component 412 may be invoked and/or may operate based on the data object identification request and/or data object matching criterion identified to locate a data object prior to, during, and/or after other content received in the communication is processed.") [0200] ("In light of the description provided above, a data object identification response may be generated automatically in response to detecting a corresponding data object identification request by an execution environment 401 of first node 502. Further, the automatically generated data object identification response, may be sent to the node that sent the corresponding data object identification request, automatically in response to the generating of the data object identification response.")

	See also [0174]; [0196] **’635 application (EX2002)** [0182] (“With respect to the method illustrated in FIG. 28, the method as described above may include locating and/or otherwise identifying a data object automatically in response to receiving a data object request that matches and/or otherwise identifies the data object. Further, a mount descriptor may be created and/or otherwise generated automatically, in response to locating and/or otherwise identifying the data object.”) *See also* FIG. 2B, 7.
the message associated with the at least one file is conditionally received based on whether a date-related criterion or a time-related criterion is met.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0007] (“Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.”)

	[0093] (“In specifying a resource scope, one or more criterion may be specified. The terms "criterion" and "constraint" are used interchangeably herein, with respect to a resource scope. A criterion included in specifying a resource scope is referred to herein as a "scope criterion". A scope criterion may be expressed via any suitable manner, such as a file name, a wildcard expression, a regular expression, a data store location, a data base query, geospatial coordinates**,** a time, a date, a duration, a relationship to a user, and/or an address - too name a few examples. With respect to a tag, a scope criterion included in defining a resource scope differs from a tagging criterion for the tag in that the scope criterion included in defining a resource scope need not be met in order for a resource to be tagged with the tag. Further a resource scope may be used for purposes other than processing related to a tagged resource.”) [0095] (“A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. . . . Whether a criterion, in a scope descriptor is met or is not met, may depend on an evaluation of the scope criterion that is based on an identifiable entity. For a scope condition, such an entity is referred to herein as a "condition entity". More generally, for a scope criterion, such an entity is referred to herein as a "scoping entity". Exemplary entities include a user, a document, a measure of time, a location, and the like.”) [0125] (“FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415 may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether data object information and/or a data object identification request is valid.

	Those skilled in the art will understand that numerous languages and/or schemas for providing data object information and/or data object identification requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SOL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") *See also* [0092], [0133], [0175], [0180] ("additional matching criteria.") **'635 application (EX2002)** [0167] ("In another aspect, a data object request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object request UI element and input from the user, descriptor handler component 404 may receive one or more additional matching criterion information and/or may receive input to modify a matching criterion identified in a received data object request. Descriptor handler component 404 may locate one or more data objects in response to receiving the additional criterion.")

Claim 16	
The apparatus of claim 15, wherein the apparatus is configured such that, in response to the receipt of the indication of the selection of the fifth user interface element, a creation of a representation of the at least one file is caused, at the first node, in a location among one or more folders on a file explorer interface that is not a web page and is displayed utilizing a client-based file explorer application, such that the at least one file is accessible via both the second message and the file explorer interface, wherein the apparatus is further configured such that the second message is sent without including the at least one file as an attachment of the second message, and a storage at the first node does not store the at least one file when the creation of the representation of the at least one file is caused, both for avoiding the at least one file from having to be communicated to and stored at the first node until the at least one file is requested by a user at	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'635 application (EX2002)**  FIG. 7  FIG. 6B

the first node using the representation or the reference.	[0008] (“Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment. The method further includes detecting the mount descriptor. The method still further includes determining a first location in a first data store in the first execution environment. The method additionally includes creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.”) [0009] (“Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment; detecting the mount descriptor; determining a first location in a first data store in the first execution environment; and creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store.”) [0010] (“Further, a system for sharing a data object in a data store via a communication is described. The system includes a content manager component for receiving , by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by

	the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment. The system further includes a descriptor handler component for detecting the mount descriptor. The system still further includes a data store navigator component for determining a first location in a first data store in the first execution environment. The system additionally includes a mount service component for, creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the content manager component, the descriptor handler component, the data store navigator component, and the mount service component includes an instruction that is executed by the processor during operation of the system.") [0011] ("Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The method further includes creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The method still further includes including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The method additionally includes sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.") [0012] ("Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent; creating a mount descriptor for

	accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user; including, by the second communications agent, the mount descriptor in a first message, addressed to the first user; and sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store." [0013] ("Further, a system for sharing a data object in a data store via a communication is described. The system includes a descriptor handler component for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The system further includes a descriptor generator component for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The system still further includes a content manager component for including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The system additionally includes a com-out component for, sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the descriptor handler component, the descriptor generator component, the content manager component, and the com-out component includes an instruction that is executed by the processor during operation of the system.") [0098] ("FIG. 7 includes a data flow that illustrates a detect descriptor data flow 704 that may be internal to execution environment 401 of first node 502 and/or that may include a message received via network 506 by first node 502. In an aspect, detect descriptor data flow 704 may illustrate a data flow to descriptor handler component 404 operating in execution environment 401 of first node 502. A mount descriptor may be identified to and/or otherwise provided to descriptor handler component 404. Content handler component 402, operating as described

	with respect to block 202 of FIG. 2A, may identify and/or otherwise provide a mount descriptor to a descriptor handler component 404.") [0109] ("…a system for sharing a data object in a data store via a communication includes means for creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store…") [0111] ("Once a location is identified, in a data store of an execution environment, for attaching and/or otherwise mounting a representation of a data object in a data store in another execution environment; any of a variety means available to the execution environment mounting the data object may be utilized.") [0112] ("…A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502…") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502.

	The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0069] ("…Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.")

	[0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user…") [0181] ("A data object request and/or a mount descriptor or respective portions thereof may include at least a portion of a keyword expression, a regular expression, expression including a Boolean operator, an expression including a precedence information, and a structured query language statement." [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0041] ("Execution environment 102 may include software components stored in persistent secondary storage 108, in remote storage accessible via a network, and/or in a processor memory. FIG. 1 illustrates execution environment 102 including an operating system 120, one or more applications 122, and other program code and/or data components illustrated by other libraries and subsystems 124. In an aspect, some or all software components may be stored in locations accessible to processor 104 in a shared memory address space shared by the software components. The software components accessed via the shared memory address space are stored in a shared processor memory defined by the shared memory address space. In another

	aspect, a first software component may be stored in one or more locations accessed by processor 104 in a first address space and a second software component may be stored in one or more locations accessed by processor 104 in a second address space. The first software component is stored in a first processor memory defined by the first address space and the second software component is stored in a second processor memory defined by the second address space.") [0053] ("As used herein, the terms "program" and "executable" refer to any data representation that may be translated into a set of machine code instructions and may optionally include associated program data. The terms are used interchangeably herein. Program representations other than machine code include object code, byte code, and source code. Object code includes a set of instructions and/or data elements that either are prepared for linking prior to loading or are loaded into an execution environment. When in an execution environment, object code may include references resolved by a linker and/or may include one or more unresolved references. The context in which this term is used will make clear the state of the object code when it is relevant. This definition can include machine code and virtual machine code, such as Java™ byte code. As used herein, the terms "application", and "service" may be realized in one or more executables and/or in one or more hardware components. The terms are used interchangeably herein.") [0072] ("The arrangement of components illustrated in FIG. 4 may be adapted to operate in a number of execution environments to perform the method illustrated in FIG. 2A and/or the method illustrated in FIG. 2B. FIG. 4 is a block diagram illustrating the components of FIG. 3A, FIG. 3B, and/or analogs of the components of FIG. 3A and FIG. 3B, respectively adapted to operate in an execution environment 401 that includes and/or otherwise is provided by one or more nodes. FIG. 1 illustrates key components of an exemplary device that may at least partially provide and/or otherwise be included in an execution environment. The components illustrated in FIG. 4 may be included in or otherwise combined with the components of FIG. 1 to create a variety of arrangements of components according to the subject matter described herein.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba

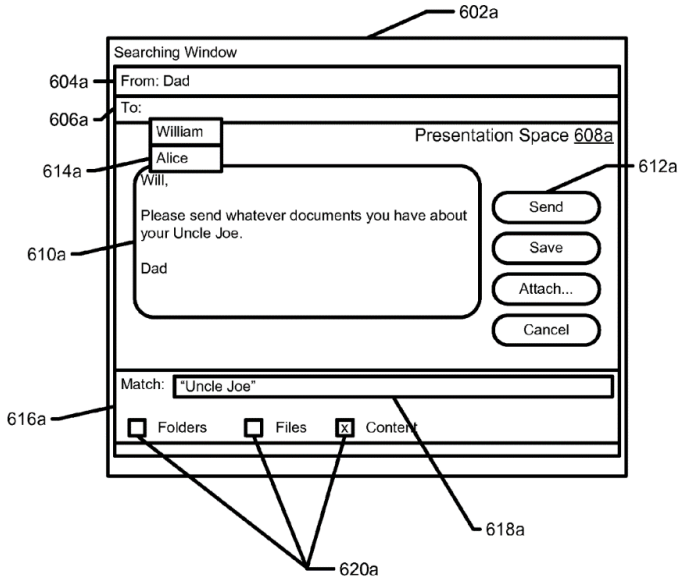
	URL 808 a identifies a protocol and a location for accessing the “Music” folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a Samba client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. Samba URL 808 a may include user identification and authentication information to the Samba client if needed. Alternatively, the Samba client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the “Music” folder may be mounted as a drive in execution environment 401 of first node 502.) [0051]-[0052] (“A “user interface (UI) element handler” component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A “program entity” is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as “presentation information”. Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like. [] A representation of a program entity may be stored and/or otherwise maintained in a presentation space. As used in this document, the term “presentation space” refers to a storage region allocated and/or otherwise provided for storing presentation information, which may include audio, visual, tactile, and/or other sensory data for presentation by and/or on an output device. For example, a buffer for storing an image and/or text string may be a presentation space as sensory information for a user. A presentation space may be physically and/or logically contiguous or non-contiguous. A presentation space

	may have a virtual as well as a physical representation. A presentation space may include a storage location in a processor memory, secondary storage, a memory of an output adapter device, and/or a storage medium of an output device. A screen of a display, for example, is a presentation space.") [0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. In FIG. 4, descriptor generator component 414 is illustrated as a component of communications agent 403.") *See also* [0009]-[001FIGs. 2A, 2B, 3A, 4, 6A, 6B, 6C, 7, Original Claims 1, 8, 9, 11, 19, 20. **'906 Application (EX2004)** [0007] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.") [0008] ("Further, a system for processing a data object identification request in a

	communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.") [0009] ("Methods and systems are described for processing a data object identification request in a communication. In one aspect, the method includes receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment. The method further includes detecting, by the second communications agent, a data object identification request included in the first message. The method still further includes generating, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request. The method additionally includes sending a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object. Performing at least one the

	preceding actions comprising the method includes execution of an instruction by a processor.") [0010] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment; detecting, by the second communications agent, a data object identification request included in the first message; generating, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request; and sending a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object.") [0011] ("Further, a system for processing a data object identification request in a communication is described. The system includes a content manager component for receiving, via a network by a second communications agent representing a second user and operating in a second execution environment, a first message from a first communications agent representing a first user and operating in a first execution environment. The system further includes a query handler component for detecting, by the second communications agent, a data object identification request included in the [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect,

	a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node.") *See also* [0007]-[0011], FIGs. 2A, 2B, 6G, 7

Claim 22	
A method, comprising: causing, at a first node, display of: a first user interface element that includes a textbox, for collecting a file request description for at least one file, a second user interface element, for enabling collection of an indication of at least one of a date or a time, a third user interface element, for collecting at least one object associated with at least one email address, the at least one object associated with at least one email address being the at least one email address or an alias associated with the at least one email address, and a fourth user interface element, for detecting an indication of a selection thereof to cause an initiation	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)** [0050] ("For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application")  FIG. 6A

of a request for the at least one file;	[0111] (“FIG. 6A-G illustrates various communications agent windows 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. In FIG. 6a, a communications agent window 602a includes a contactor user interface (UI) element 604a for presenting an identifier of a communicant in the role of a contactor in a communication represented by the communications agent window 602a. A communications agent window 602a also includes a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) included in the communication. A presentation space 608 is provided by each communications agent window 602a for presenting a user message UI element 610 for presenting a message addressed to one or more contactees identified in a contactee UI element 606a. A presentation space 608a may also be provided for presenting one or more UI controls for exchanging data in and/or otherwise managing a communication. A send UI element 612a illustrates an exemplary UI element that may correspond to user input to send data in a communication to one or more identified contactees.”) [0115] (“Sending data in a communication requires identifying a contactee. A contactee may be represented by a communications address in an address space of a communications protocol**.** In one aspect, information identifying a communications address may be received from a user/communicant of a communications agent in a user node. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 in FIG. 6A for first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 604. The user/communicant of first node 502 may enter a contactor alias, such as "Dad", . . . Communications agent 403 may identify a communication address associated with the contactor alias, "Dad". Other communications address(es), such as for one or more contactees, may be received similarly and/or in any suitable manner.”) [0117] (“Data may be sent in a communication in response to a communicant input. A contactor may provide an input corresponding to send UI element 612a in FIG. 6A. The input may be received by presentation controller component 417 and/or one or more UI element handlers 415 corresponding to send UI element 612a. In response to detecting the input, presentation controller component 417 may provide data to be sent in the communication to one or more content handler components 413 according to the content type(s) of the data to be sent.”)

	[0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”) [0125] (“FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415 may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403.”) [0126] (“DOI UI element 616a illustrates a user interface for a keyword based schema. Match textbox UI element 618a illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 620a allow a user to define a scope of a data object identification request by identifying whether a resulting data object identification request is to be applied to folders, files, and/or content of files in a file system and/or other data store of a receiving execution environment.”) [0047] (“The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as ‘UI elements’. More specifically, visual outputs of a user interface are referred to herein as ‘visual interface elements’. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include

	windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. . . .") [0175] ("In another aspect, a data object identification request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object identification request UI element and input from the user, query director component 412 may receive one or more additional data object matching criterion information and/or may receive input to modify a data object matching criterion identified in a received data object identification request. Query director component 412 may locate one or more data objects in response to receiving the additional criterion.") *See also* [0006]-[0011], [0047], [0020], [0100], [0111], [0115]-[0117]; Fig. 6A (e.g. Match window 616a); [0125]-[0126]; [0184];Original claims 1-22 **'635 application (EX2002)** [0049] ("The user-detectable outputs of a user interface are generically referred to herein as "user interface elements" or abbreviated as 'UI elements'. More specifically, visual outputs of a user interface are referred to herein as 'visual interface elements'. A visual interface element may be a visual output of a graphical user interface (GUI). Exemplary visual interface elements include windows, textboxes, sliders, list boxes, drop-down lists, spinners, various types of menus, toolbars, ribbons, combo boxes, tree views, grid views, navigation tabs, scrollbars, labels, tooltips, text in various fonts, balloons, dialog boxes, and various types of button controls including check boxes and radio buttons. An application interface may include one or more of the elements listed. Those skilled in the art will understand that this list is not exhaustive. The terms 'visual representation', 'visual output', and 'visual interface element' are used interchangeably in this document. Other types of UI elements include audio outputs referred to as 'audio interface elements', tactile outputs referred to as 'tactile interface elements', and the like.") [0051] (""A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-

	detectable representation of the program entity by an output device, such as a display. A "program entity" is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as "presentation information". Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0082] ("FIGS. 6A-H illustrate various user interfaces, some of which include window UI elements 602 presentable in a presentation space of a display device, such as output device 130 in FIG. 1. A window UI element 602 may include a contactor UI element 604 for presenting an identifier of a communicant in the role of a contactor in a communication represented by the window UI element 602. A window UI element 602 may also include a contactee UI element 606 for presenting one or more contactee identifier(s) identifying one or more communicants in the role of contactee(s) in the communication. A presentation space 608 may be included in each window UI element 602 for presenting a user message UI element 610 that my present a user message from one user to another.

	A presentation space 608 may also be provided for presenting one or more UI elements that correspond to user inputs for exchanging data in and/or otherwise managing a communication.") [0086] ("A contactee is identified for sending data in a communication. A contactee may be represented by a communications address in an address space of a communications protocol. In one aspect, information identifying a communications address may be received from a communicant of a communications agent in an execution environment. In FIG. 4, presentation controller 417 and/or a UI element handler 415, presenting and/or managing interaction with contactor UI element 604 a in FIG. 6A for execution environment 401 of first node 502, may receive a contactor alias in response to a user input corresponding to UI element handler 415. The user/communicant of execution environment 401 of first node 502 may enter a contactor alias, such as "William", via a keyboard and/or select a predefined communicant alias presented in a selection UI control element via a UI element handler component 415. The user input may be detected by input driver 421. Corresponding input information may be routed to presentation controller 417 by GUI subsystem 423. GUI subsystem 423 operates to send presentation information to a display device via a graphics subsystem 425. Communication agent 403 may identify a communication address associated with the contactor alias, "William". Other communications address, such as for one or more contactees, may be received similarly and/or in any suitable manner. In an aspect, one or more contactor and/or contactee identifiers may be determined automatically.")

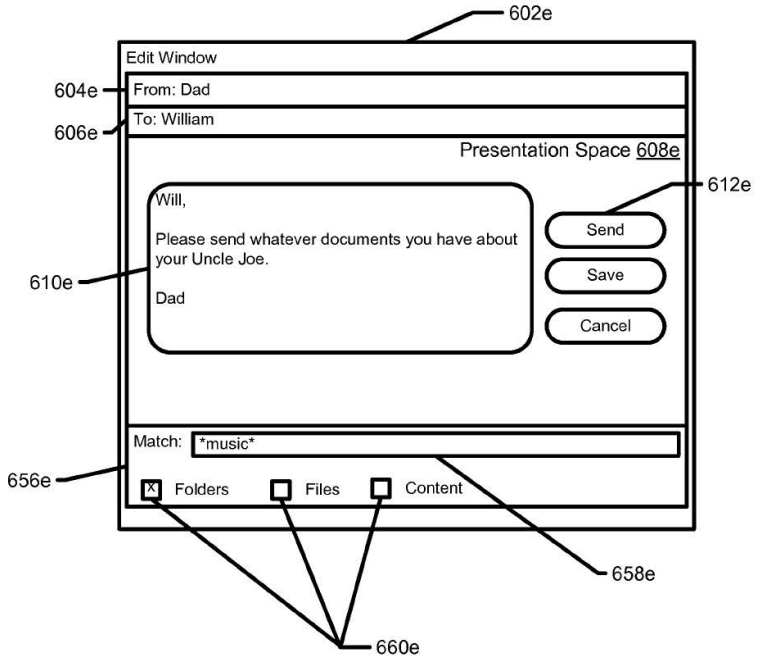
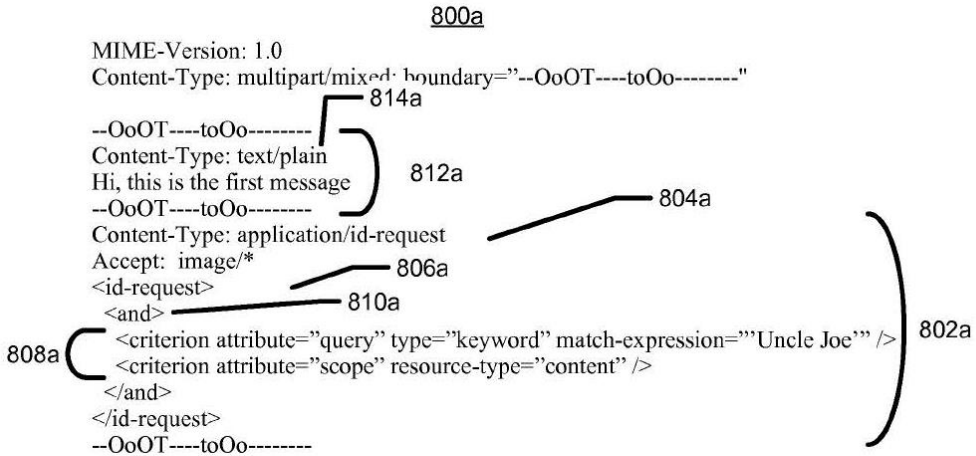


FIG. 6E

[0145] ("MI UI element 656e illustrates a user interface for a keyword based schema. Match textbox U I element 658e illustrates a textbox allowing a user to enter a keyword expression. Various checkbox UI elements 660e allow a user to define a scope of a data object request by identifying whether a resulting data object request is to be applied to folders, files, and/or content of files in a data store. A communications agent 403 may support more than one data object request schema and/or corresponding matching information schema. A communications agent may provide a user interface to receive valid matching information for each respective schema.")

[0144] ("FIG. 6E illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler

	component 415 may operate to present a user interface element to allow a user to identify matching information. A matching information (Ml) UI element 656e is illustrated in FIG. 6E as an example. The same or different UI element handler component 415 may present various other UI elements included in Ml UI element 656e. MI U I element 656e and one or more U I elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid matching information and/or data object requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether matching information and/or a data object request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing matching information and/or data object requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SQL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.") [0146] ("FIG. 7 includes data flow message that illustrates a create data object request (CIReq) data flow 714 that may be internal to execution environment 401 of first node 502 and/or may include a message received via network 506 by first node 502. With respect to FIG. 4, CIReq data flow 714 may illustrate a data flow to an IMC component 410 operating in execution environment 401 of first node 502. Match information may be identified to and/or otherwise provided to IMC component 410.") [0153] ("In various aspects, a data object request may be based on a date, a time, a length of time, a file type, a database record key, content of the data object, a content type identifier, a format rule, a vocabulary, a role of a user, a security attribute, a location in a data store such as a file system, an attribute of an identified data object, a size, a task, a transaction, a state, a user, a group, a requester, a relationship including a requesting user and a responding user, a keyword , a tag, a folder, and/or a path portion of a data object identifier, to name a few examples.") *See also* FIGs. 7, 8B, [0142]-[0143], [0147]-[0152], [0172], Original claim 12

causing generation of a first email, based on the file request description for the at least one file, the at least one object associated with the at least one email address, and the detection of the indication of the selection of the fourth user interface element to cause the initiation of the request for the at least one file,	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  Fig. 8A [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message. ...”) [0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node

	502. Data object information is identified to and/or otherwise received by query handler component 402.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0127] ("Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. Accordingly, a system for processing a data object identification request in a communication includes means for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. For example, the arrangement in FIG. 3A, includes identification request constructor component 304 that is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. FIG. 4 illustrates identification request constructor component 404 as an adaptation and/or analog of identification request constructor component 304 in FIG. 3A. One or more identification request constructor components 404 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more

	processors that when executed is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. In FIG. 4, identification request constructor component 404 is illustrated as a component of communications agent 403.") [0128] ("FIG. 7 includes dataflow message 704 that illustrates a data flow internal to execution environment 401 of first node 502 and/or message received via network 506 by execution environment 401 of first node 502. Dataflow message 704 illustrates a create data object identification request (CIReq) dataflow to an IRC component 404 operating in execution environment 401 of first node 502. Data object information may be identified to and/or otherwise provided to IRC component 404. A query handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide data object information to an IRC component 404.")

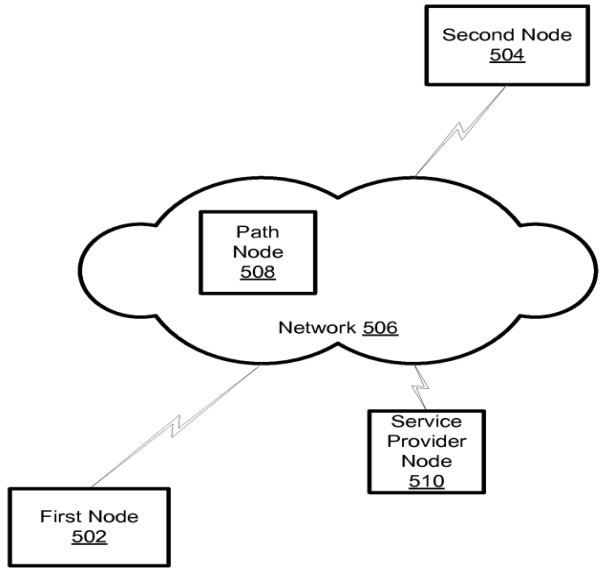
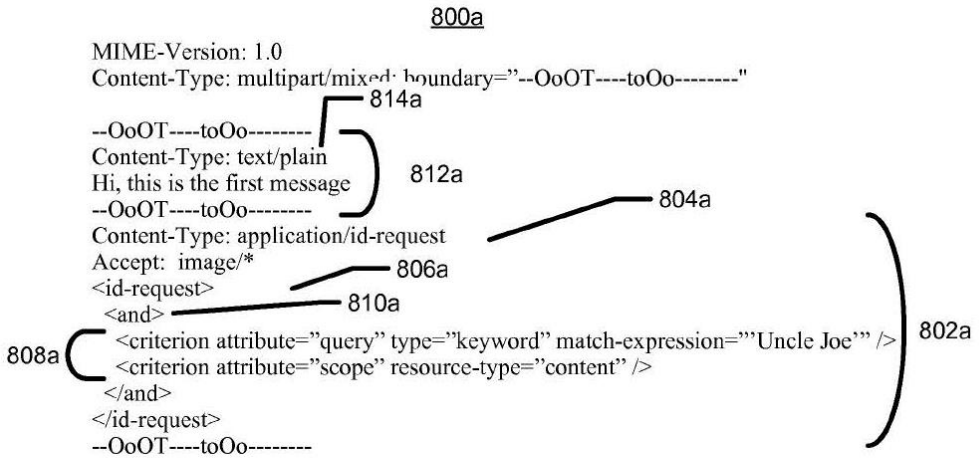


FIG. 5

See also [0092]-[0093], [0123]-[0124]; [0129]-[0135] (message creation and contents); [0175], [0180]; [0206], FIG. 2A, FIG. 6B, FIG. 6D, FIG. 7 (flowchart of request process); FIG. 8A; Original claims 1-22

’635 application (EX2002)

See FIG. 6E, 7, 8B, [0142]-[0153]; *see also* [0073], [0172]

where the first email includes: at least one sentence including the file request description for the at least one file, and	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**  Fig. 8A [0118] (“The one or more content handler components 413 may encode, format, and/or otherwise transform the data for sending in a message, such as an email message. ...”) [0123] (“FIG. 7 includes dataflow message 702 that illustrates a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. Dataflow message 702 illustrates a data object information dataflow to query handler component 402 operating in execution environment 401 of first node 502. Data object information is identified to and/or otherwise received by query handler component 402.”)

	[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0127] ("Returning to FIG. 2A, a block 204 illustrates that the method further includes sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. Accordingly, a system for processing a data object identification request in a communication includes means for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. For example, the arrangement in FIG. 3A, includes identification request constructor component 304 that is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. FIG. 4 illustrates identification request constructor component 404 as an adaptation and/or analog of identification request constructor component 304 in FIG. 3A. One or more identification request constructor components 404 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object

	identification request based on the data object matching criterion, wherein the first message is addressed to the second user. In FIG. 4, identification request constructor component 404 is illustrated as a component of communications agent 403.") [0128] ("FIG. 7 includes dataflow message 704 that illustrates a data flow internal to execution environment 401 of first node 502 and/or message received via network 506 by execution environment 401 of first node 502. Dataflow message 704 illustrates a create data object identification request (CIReq) dataflow to an IRC component 404 operating in execution environment 401 of first node 502. Data object information may be identified to and/or otherwise provided to IRC component 404. A query handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide data object information to an IRC component 404.") *See also* [0100], [0111]-[0117]; Fig. 6A (text box 618a); [0125]-[0126]; [0129]-[0135] (message creation and contents); FIGS. 2A, 2B; FIG 8A; Original claims 1-22 **'635 application (EX2002)** *See also* FIG. 6E, 7, 8B, [0142]-[0153]; see also [0073], [0172]
a fifth user interface element for detecting an indication of a selection thereof;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting.

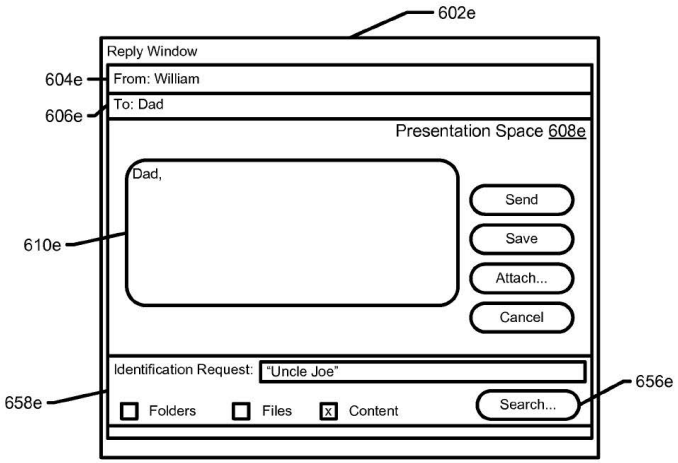


FIG. 6E

[0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

[0162] ("FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object identification request may include and/or otherwise identify a data object matching criterion

	identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.") [0167] ("FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.") [0179] ("FIG. 8B illustrates an exemplary data object identification response described in detail above. Second message portion 802b illustrates a user message portion that may be included in

	second message 714. Note that a user message portion may be empty or absent in some aspects. Data object identification response portion 804b illustrates the data object identification response as a list of URls identifying data objects in a file system of execution environment 401 of second node 504. User Message portion 802b and data object identification response portion 804b may be provided by respective content handler components 413 to content manager component 409. Content manager component 409 may construct the content as illustrated in FIG. 8B and/or otherwise provide for encoding, translating, combining, and/or otherwise preparing the content for sending in second message 714.") [0181] ("In another aspect, a data object located and/or otherwise identified in response to receiving a data object identification request must meet match a query identified by the data object identification request. Alternatively or additionally, a data object identification request may include an instruction and/or input for generating a data object. Locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object identification request may identify the template. A query handler component 402 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to an access request, execution environment 401 of second node 504 may return the document.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the

	client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") *See also* [0100], [0111], [0137] (discusses receiving a message from second node); FIG. 5, FIG. 6B (selecting element 628b to bring up window 630b); [0146]-[0148]; FIG. 2B, [0157]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0170], [0173]; [0206]; FIG. 7; Original claims 1-22. **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") [0069] ("The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service

	including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0073] ("As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be

	synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG.

	8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the “Music” folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.”) *See also* [0157], FIGs. 8A, 8B; Original claim 14
based on the detection of the indication of the selection of the fifth user interface element of the first email, causing, at the second node, display of: a sixth user interface element, for collecting an indication of at least one file, and	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**

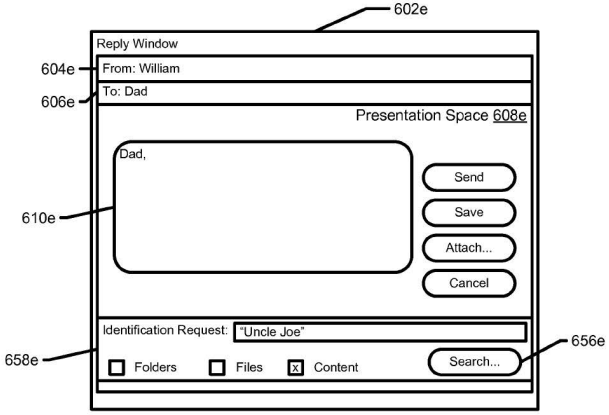


FIG. 6E

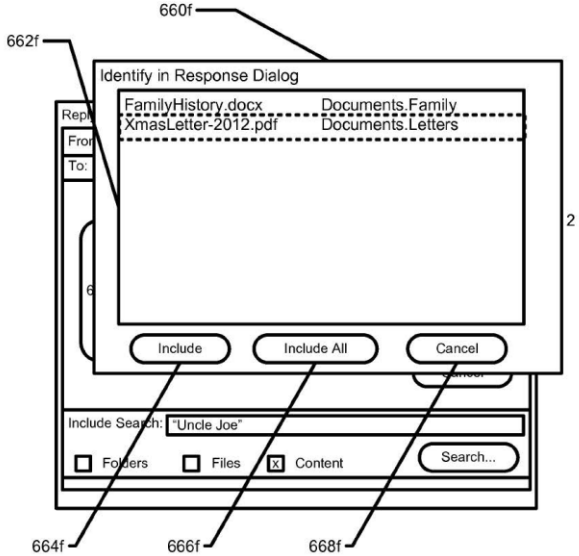


FIG.6F

[0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

[0162] ("FIG. 7 includes a dataflow message 708 that illustrates a detect data object identification request dataflow that may be internal to execution environment 401 of second node 504 and/or may include interoperation with another node via a network. With respect to FIG. 4, dataflow message 708 may correspond to a data exchange and/or other interoperation between content handler component 409 and query handler component 402 to detect and/or otherwise process the data object identification request received in first message 706 by execution environment 401 of second node 504. FIG. 7 further includes a dataflow message 710 that illustrates a search message for locating one or more data objects based on the data object identification request. The data object

	identification request may include and/or otherwise identify a data object matching criterion identified by data object information identified by the communicant represented by execution environment 401 of first node 502. The one or more data objects may be located and/or otherwise identified by determining a data object or data objects that match the data object matching criterion. The data object(s) may be in a data store accessible to execution environment 401 of second node 504, such as one or more files in a file system of execution environment 401 of second node 504. Dataflow message 710 may be internal to execution environment 401 of second node 504 and/or may be a message sent to a remote device, such as database server and/or a network attached storage device providing storage locations for some or all of a file system of execution environment 401 of second node 504.") [0167] ("FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.") [0171] ("FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.")

	[0172] "(FIG. 6F, illustrates an identify in response dialog UI element 660f that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to processing the data object identification request to locate matching data objects. In an aspect, a user of communications agent 403 operating in execution environment 401 of second node 504 may provide input corresponding to search UI element 656e in reply window UI element 602e in FIG. 6E. A corresponding UI element handler component 415 may operate to invoke query director component 412 to perform and/or otherwise to provide for performing one or more operations to locate one or more data objects based on the data object identification request. Query director component 412 may interoperate with one or more UI element handlers 415, directly or indirectly via query handler component 402, in an aspect, to present the UI element 660f and its contents. Identify in response dialog UI element 660f includes results pane UI element 662f including representations of two data objects located and/or otherwise identified by query director component 412 based on the data object identification request. A user may select one or more data objects to identify in a data object identification response. An include UI element 664f may be presented, as illustrated, to allow a user to instruct communications agent 403 to identify one or more selected data objects in results pane U I element 662f in a data object identification response. An include all UI element 666f is illustrated to allow for user input instructing communications agent 403 to identify all data objects represented in the UI element 660f in a data object identification response in a communication with execution environment 401 of first node 502. A cancel UI element 668f allows a user to end presentation of dialog UI element 660f. A user may opt to identify no data objects represented in the dialog UI element 660f and located and/or otherwise identified by query director component 412 based on the data object identification request.") [0179] ("FIG. 8B illustrates an exemplary data object identification response described in detail above. Second message portion 802b illustrates a user message portion that may be included in second message 714. Note that a user message portion may be empty or absent in some aspects. Data object identification response portion 804b illustrates the data object identification response as a list of URls identifying data objects in a file system of execution environment 401 of second node 504. User Message portion 802b and data object identification response portion 804b may be provided by respective content handler components 413 to content manager component 409. Content manager component 409 may construct the content as illustrated in FIG. 8B and/or

	otherwise provide for encoding, translating, combining, and/or otherwise preparing the content for sending in second message 714.") [0181] ("In another aspect, a data object located and/or otherwise identified in response to receiving a data object identification request must meet match a query identified by the data object identification request. Alternatively or additionally, a data object identification request may include an instruction and/or input for generating a data object. Locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object identification request may identify the template. A query handler component 402 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to an access request, execution environment 401 of second node 504 may return the document.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a

	representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") *See also* FIGS. 5, 6C, 6G; [0146]-[0148]; [0150]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0206], FIG. 7 **'635 application (EX2002)** [0171]-[0172] ("Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.") [0069] ("The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0073] ("As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that

	arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an

	application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.")

	See also [0157], FIGs. 8A, 8B; Original claim 14
a seventh user interface element, for detecting an indication of a selection thereof;	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)**

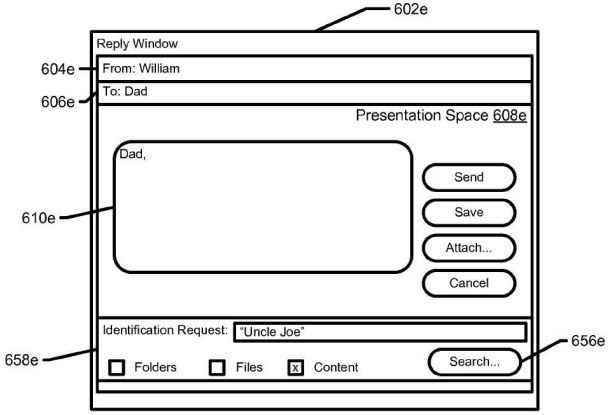


FIG. 6E

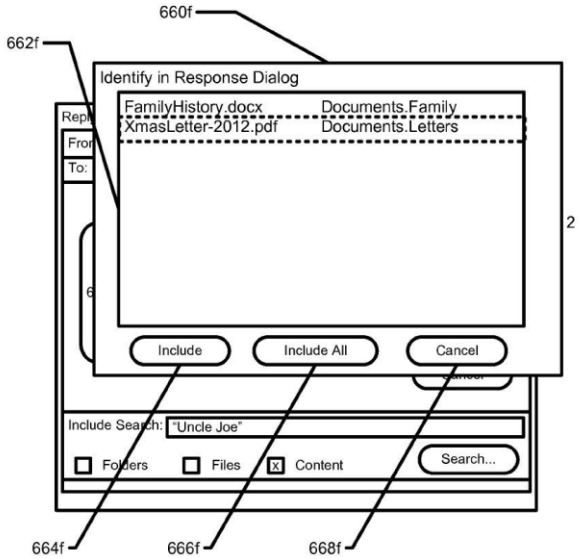


FIG.6F

[0167] ("FIG. 6E, illustrates a reply window UI element 602e that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to an user input corresponding to reply UI element 648d detected by a corresponding UI element handler component 415. In an aspect, a user of communications agent 403 operating in in execution environment 401 of second node 504 may provide input authorizing some or all of the processing of a data object identification request. Reply window UI element 602e in FIG. 6E includes a search UI element 656e for receiving user input authorizing and/or otherwise instructing execution environment 401 of second node 504 to locate data objects that match the data object identification request, illustrated in a data object identification requested UI element 658e.")

	[0171] (“FIG. 7 includes dataflow message 712 that illustrates a create data object identification response data flow to process results returned from search dataflow 710. In FIG. 4, dataflow message 712 in FIG. 7 may represent data exchanged between query handler component 402 and query director component 412 to process results returned from a search operation performed based on the data object identification request, as illustrated by dataflow message 710. Processing the results may include creating and/or otherwise constructing a data object identification response to the data object identification request identified in first message 706. Dataflow message 712 may be internal to execution environment 401 of second node 504 and/or may include interoperation by execution environment 401 of second node 504 via a network with another node.”) [0172] “(FIG. 6F, illustrates an identify in response dialog UI element 660f that, in an aspect, may be presented by communications agent 403 in execution environment 401 of second node 504 in response to processing the data object identification request to locate matching data objects. In an aspect, a user of communications agent 403 operating in execution environment 401 of second node 504 may provide input corresponding to search UI element 656e in reply window UI element 602e in FIG. 6E. A corresponding UI element handler component 415 may operate to invoke query director component 412 to perform and/or otherwise to provide for performing one or more operations to locate one or more data objects based on the data object identification request. Query director component 412 may interoperate with one or more UI element handlers 415, directly or indirectly via query handler component 402, in an aspect, to present the UI element 660f and its contents. Identify in response dialog UI element 660f includes results pane UI element 662f including representations of two data objects located and/or otherwise identified by query director component 412 based on the data object identification request. A user may select one or more data objects to identify in a data object identification response. An include UI element 664f may be presented, as illustrated, to allow a user to instruct communications agent 403 to identify one or more selected data objects in results pane U I element 662f in a data object identification response. An include all UI element 666f is illustrated to allow for user input instructing communications agent 403 to identify all data objects represented in the UI element 660f in a data object identification response in a communication with execution environment 401 of first node 502. A cancel UI element 668f allows a user to end presentation of dialog UI element 660f. A user may opt to identify no data objects represented in the dialog UI element 660f and located and/or otherwise identified by query director component 412 based on the data object identification request.”)

[0124] (“In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.”)

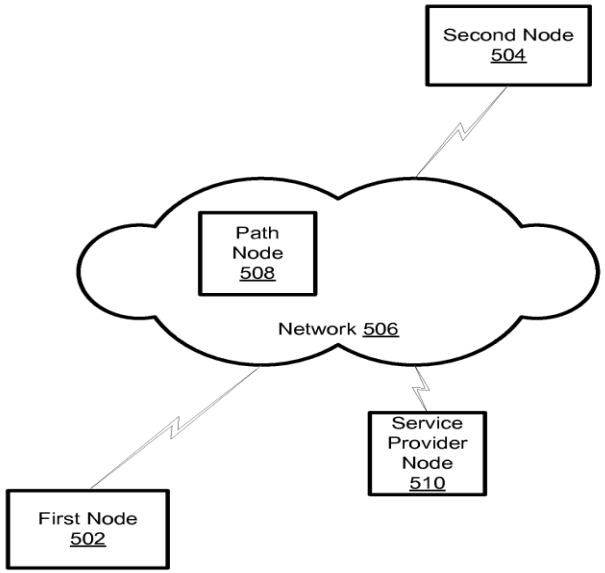


FIG. 5

	See also FIGS. 6C, 6G; [0146]-[0148]; [0150]; [0162]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 6F, [0172]; [0206] **’635 application (EX2002)** [0171]-[0172] (“Exemplary data objects that may be requested via a data object request include a file, a program component, a data base record, video data, audio data, markup language, binary data, text data, an output of a service. Requested data objects may be pre-existing, volatile, and/or generated in response to the request. [] As described above, receiving match information may include receiving a message, identifying the match information, via network from a node that received match information from the user. For example, match information may be received from a browser by a web server.”) [0069] (“The term "data object" as used herein refers to an entity, identifiable within a data store, for storing and/or accessing data. Exemplary data stores include file systems, directory service including DNS and LDAP, and data bases including relational databases and hierarchical databases, to name a few examples. Exemplary data objects include files and folders in a file system. Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in Fig. 3A as well as the various adaptations and analogs of the arrangement in Fig. 38 illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in Fig. 2A and the method illustrated in Fig. 28 may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0112] (“FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808a identifies a protocol and a location for accessing the "Music" folder.”)

	[0115] (“As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502.”) [0116] (“In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.”) [0117] (“For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.”)

	[0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount description.") [0130] ("FIG. 8A includes an exemplary content portion 800a of an email message. Content portion 800a includes a user message portion 802a and a mount descriptor portion 804a described in more detail above. FIG. SA illustrates an "application/mountdescriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804a including two identifiers for the "Music" folder. A first identifier 808a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804a. An FTP URL 812a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804a.") *See also* [0157], FIGs. 8A, 8B; Original claim 14
receiving, from the second node and at at least one server, a signal for causing creation of a first representation of the at least	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)**

one file, in a location among one or more folders, that is stored at the at least one server and that is displayable via at least one web page;	[0144] ("FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0102] ("[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network") [0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

	[0201] (“With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.”) [0185] (“As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.”) [0146] (“FIG. 6B illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes a user message UI element 610b, analogous to similar UI elements described with respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409

	in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.") [0147] ("Response window UI element 602b in FIG. 6B includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect, one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **6B,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution environment 401 of first node 502 to request one or more data objects identified in the data object identification response.") [0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu

	item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

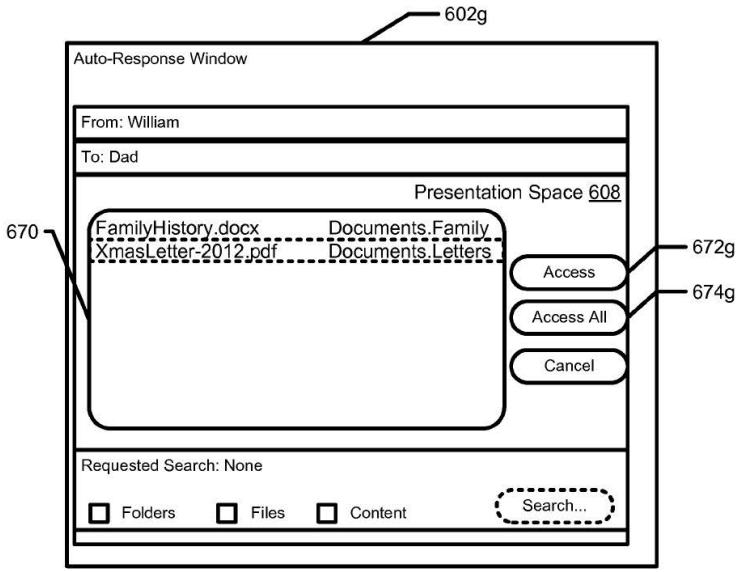

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
Access
Access All
Cancel
672g
674g
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

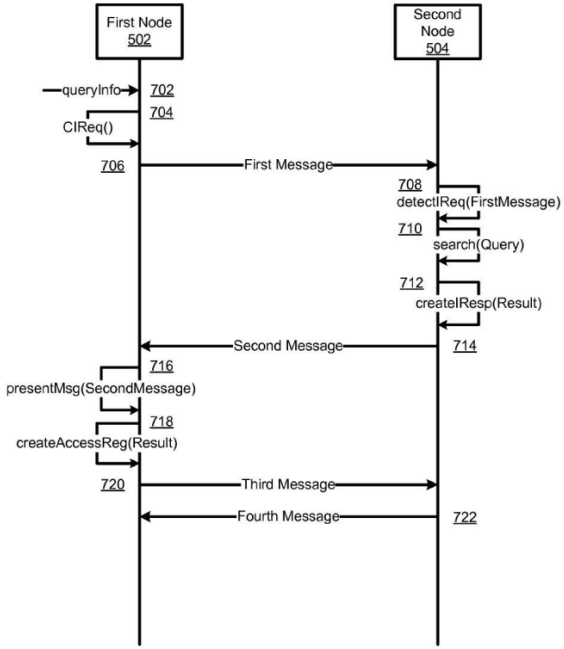
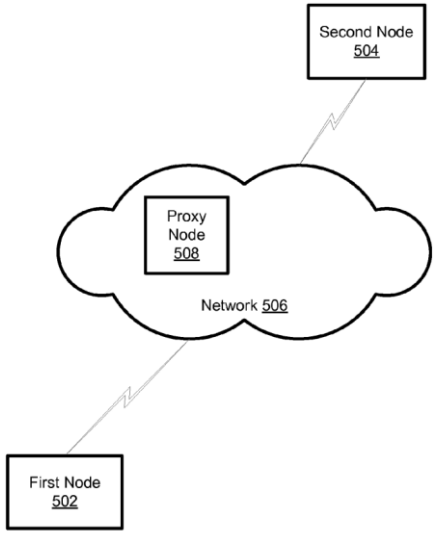
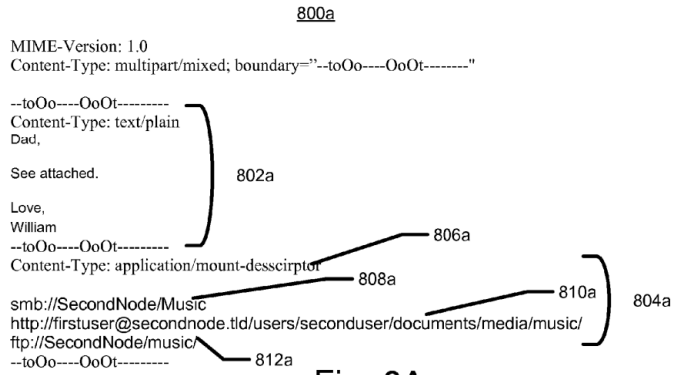


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A. One or more access request constructor components 408 operate in an execution environment 401. The system for processing a data object identification request in a communication includes one or more processors and logic encoded in

	one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") *See also* [0007]-[0011], FIGs. 2A, 2B, 5, 6G, **'635 application (EX2002)**  FIG. 5  Fig. 8A

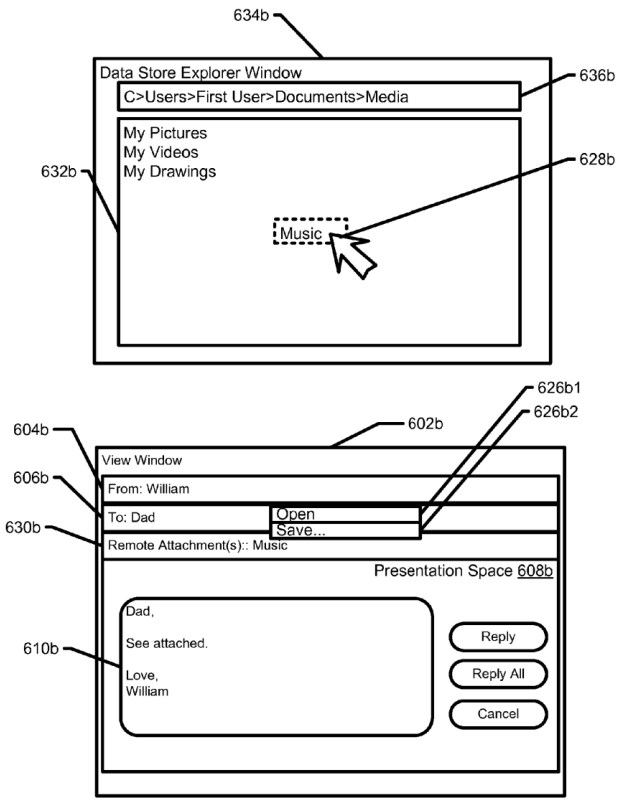


FIG. 6B

[0103] ("In an aspect, determining a location in a data store for creating a representation of a data object may include presenting a navigation UI element for navigating the data store. The location may be determined in response to detecting a user input identifying the location. View window UI element 602 a in FIG. 6A includes a mount UI element 618 a for receiving user input to display a navigation window UI element 620 a for identifying a location in a data store of execution environment 401 of first node 502 to mount and/or otherwise attach the remote "Music" folder identified in the mount descriptor. In response to an input directed to mount UI element 618 a, a corresponding UI element handler 415 may interoperate with data store navigator component 406

	to present navigation window UI element 620 a that includes a presentation space 622 a for presenting a user detectable representation of one or more locations in a data store of execution environment 401 of first node 502.") [0106] ("In response to detecting the mount descriptor in first message 702, execution environment 401 of first node 502 may present a view window UI element 602 b in FIG. 6B including data received in the first message 702 to a user of execution environment 401 of first node 502 as described with respect to FIG. 6A. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" menu UI element 626 b 1, identified in a mount descriptor received from execution environment 401 of second node 504. Context menu 626 b provides UI elements to allow the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, in response to an input corresponding to "Save . . . " menu UI element 626 b 2, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.") [0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a communication without a user message. Rather than a UI element for displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.")

	[0111] ("Once a location is identified, in a data store of an execution environment, for attaching and/or otherwise mounting a representation of a data object in a data store in another execution environment; any of a variety means available to the execution environment mounting the data object may be utilized.") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116]-[0119] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle

	problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly. For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated. Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system. Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount descriptor.") [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a

	FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0051] ("A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a display. A "program entity" is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as "presentation information". Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like.") *See also* [0035] & [0069]

causing generation of a second email, based on the detection of the indication of the selection of the seventh user interface element,	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **<u>'906 application (EX2004)</u>** [0008] ("Further, a system for processing a data object identification request in a communication is described. The system includes a query handler component for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion. The system further includes an identification request constructor component for sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user. The system still further includes a result handler component for receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message. The system also includes a processor, wherein at least one of the query handler component, the identification request constructor component, and the result handler component includes an instruction that is executed by the processor during operation of the system.")

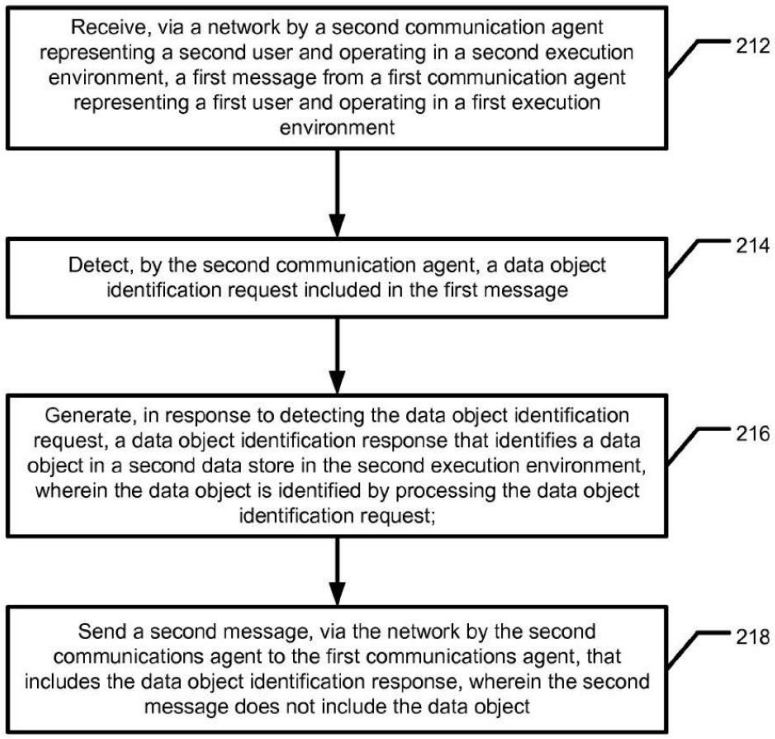

Receive, via a network by a second communication agent representing a second user and operating in a second execution environment, a first message from a first communication agent representing a first user and operating in a first execution environment
212
Detect, by the second communication agent, a data object identification request included in the first message
214
Generate, in response to detecting the data object identification request, a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the data object is identified by processing the data object identification request;
216
Send a second message, via the network by the second communications agent to the first communications agent, that includes the data object identification response, wherein the second message does not include the data object
218

FIG. 2B

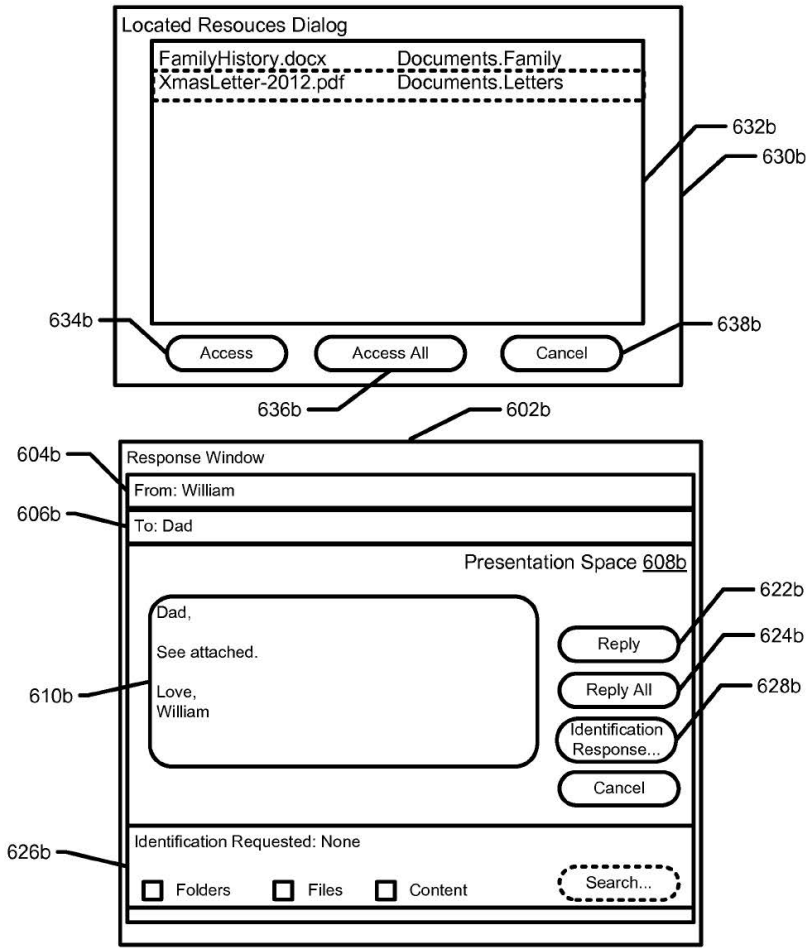

Located Resouces Dialog
FamilyHistory.docx
Documents.Family
XmasLetter-2012.pdf
Documents.Letters
632b
630b
634b
Access
Access All
Cancel
638b
636b
602b
604b
Response Window
From: William
606b
To: Dad
Presentation Space 608b
622b
Dad,
See attached.
Reply
624b
610b
Love,
William
Reply All
628b
Identification Response...
Cancel
626b
Identification Requested: None
Folders
Files
Content
Search...

FIG. 6B

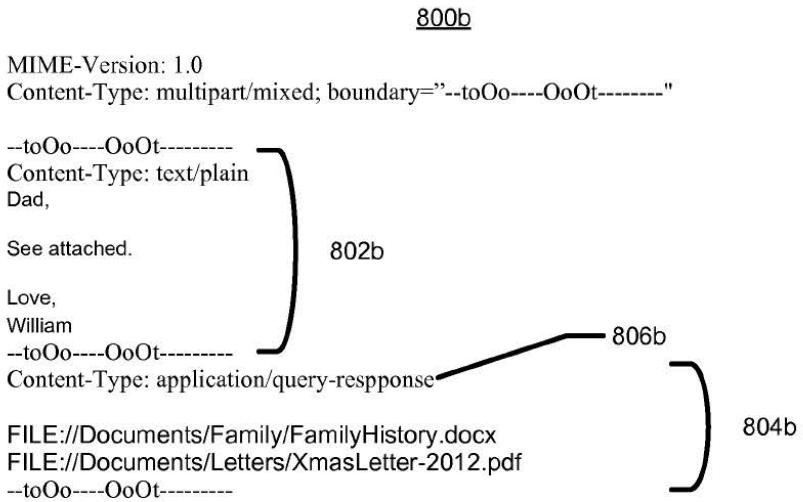


FIG. 8B

[0124] ("In various aspects, a query handler component 402 may receive data object information from a user via one or more UI element handler components 415. The data object information may be received, in response to a user input detected by an input device of an execution environment 401 that includes the query handler component 402. Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.")

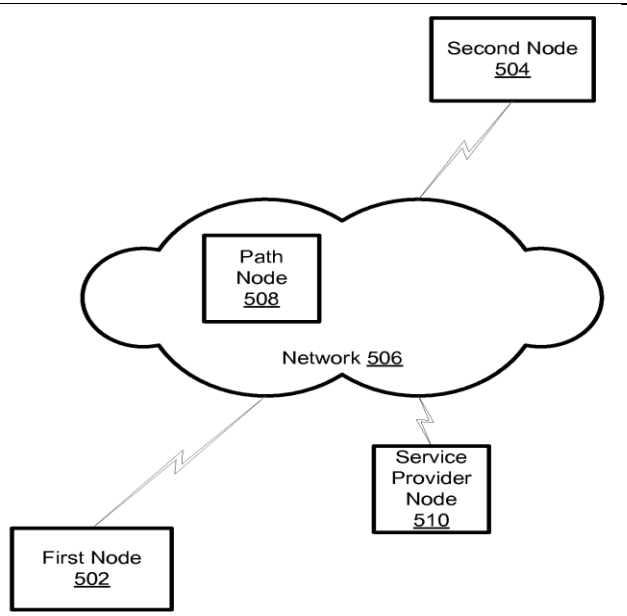

Second Node
504
Path
Node
508
Network 506
Service
Provider
Node
510
First Node
502

FIG. 5

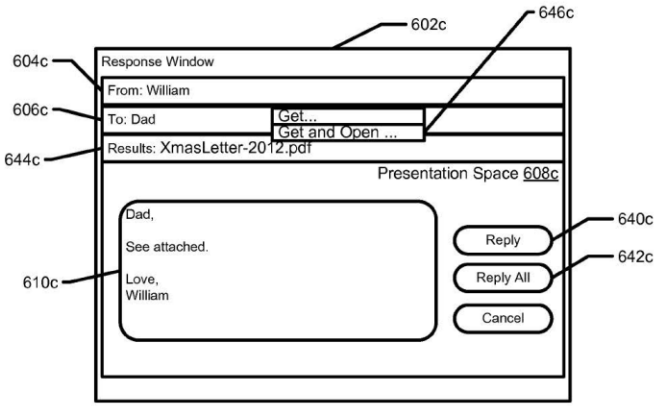


FIG. 6C

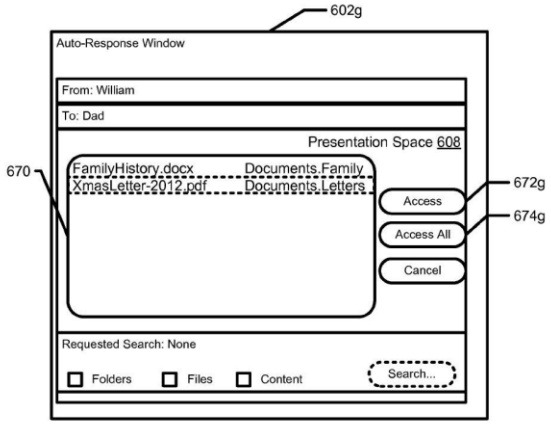


FIG. 6G

[0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a

user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.")

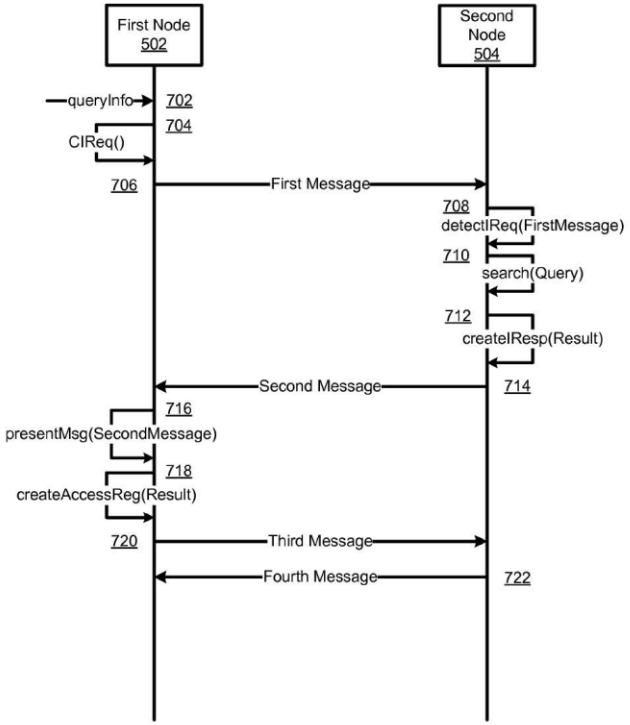
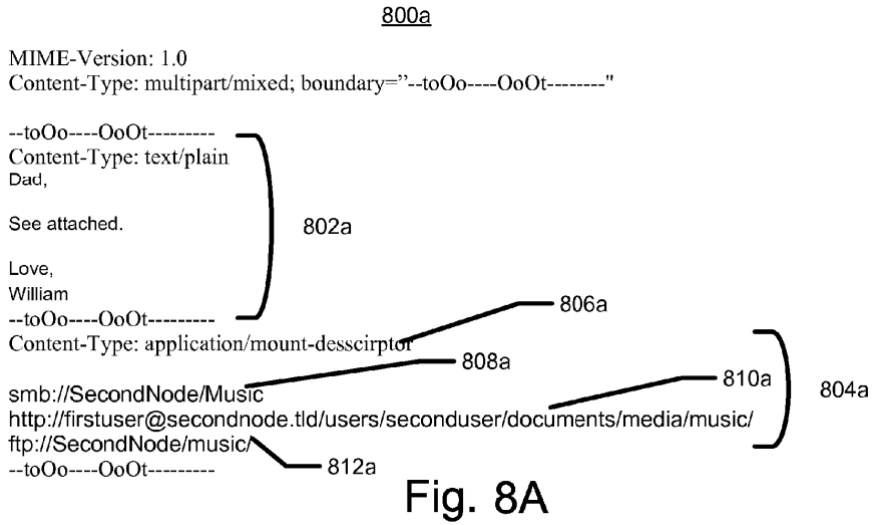
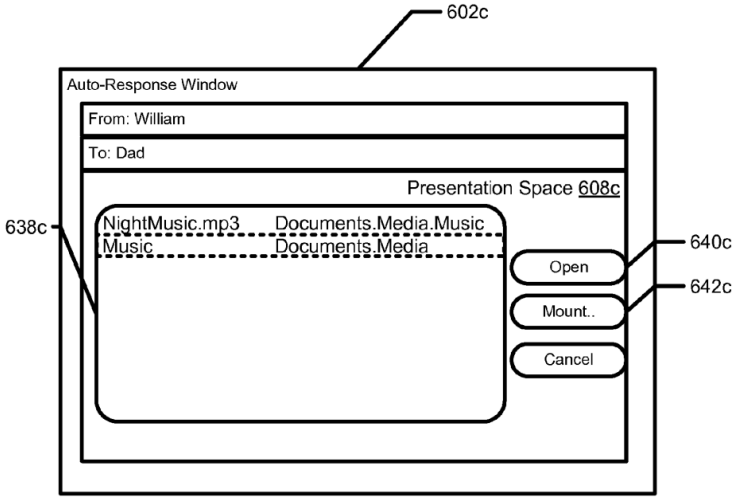


FIG. 7

	See also [0007], [0009]-[0011], [0150]; [0137] (discusses receiving a message from second node, "wherein the data object is not received in the second message"); FIG. 5, FIG. 8B, [0141]-[0142]; FIG. 6C, [0150]; FIG. 6D, [0166]; FIG. 6E, [0167]; FIG. 2B, [0170]; [0171]-[0173]; [0176]; [0179]; [0200]; [0206]. **'635 application (EX2002)**  Fig. 8A [0096] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user readable message portion 802 a and a mount descriptor portion 804 a described in more detail below. The received communication may include a message addressed to a user represented by communications agent 403 in execution environment 401 of first node 502.") [0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a communication without a user message. Rather than a UI element for displaying a user message, a

	remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.")

	[0168] ("In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an instruction and/or input for generating a data object, and locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.")  FIG. 6C [0108] ("FIG. 6C illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to mount and/or otherwise attach a representation of a data object identified in a mount descriptor to a location in a data store of execution environment 401 of first node 502. Auto-response window 602 c may be presented in response to a mount descriptor received in a

	communication without a user message. Rather than a UI element for displaying a user message, a remote data object pane UI element 638 c may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 in a mount descriptor received by execution environment 401 of first node 502. UI elements in presentation space 608 c are illustrated that allow a user to instruct execution environment 401 of first node 502 to retrieve a data object, in response to an input corresponding to "Open" UI element 640 c, identified in a mount descriptor received from execution environment 401 of second node 504. Mount UI element 642 c allows the user to instruct execution environment 401 of first node 502 to attach and/or otherwise mount a representation of a data object, identified in a mount descriptor at a location in the data store of execution environment 401 of first node 502.") *See also* [0007]-[0013], [0115]-[0119], FIG. 7, FIGs 2A, 2B.
where the second email: includes an eighth user interface element for detecting an indication of a selection thereof, does not include a file attachment, for permitting avoidance of the at least one file from being communicated to and stored at the first node via the second email, is at least partially pre-written, and is automatically caused to be received without requiring further user involvement after the detection of the indication	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)** [007]-[0011] (does not include file attachment) [0072] ("The arrangement of components illustrated in FIG. 4 may be adapted to operate in a number of execution environments to perform the method illustrated in FIG. 2A and/or the method illustrated in FIG. 2B. FIG. 4 is a block diagram illustrating the components of FIG. 3A, FIG. 3B, and/or analogs of the components of FIG. 3A and FIG. 3B, respectively adapted to operate in an execution environment 401 that includes and/or otherwise is provided by one or more nodes. FIG. 1 illustrates key components of an exemplary device that may at least partially provide and/or otherwise be included in an execution environment. The components illustrated in FIG. 4 may be included in or otherwise combined with the components of FIG. 1 to create a variety of arrangements of components according to the subject matter described herein.") [0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c

of the selection of the seventh user interface element;	provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0173] ("Data objects located and/or otherwise identified by query director component 412 may be identified to IRB component 414 to construct and/or otherwise create a data object identification response identifying the data objects. As described above, in an aspect, a user may be allowed to select which data objects, if any, are to be identified in a data object identification response from one or more data objects located and/or otherwise identified by query director component 412 based on the data object identification response. In another aspect, communications agent 403 in FIG. 4 may invoke query director component 412 automatically to locate one or more data objects based on a data object matching criterion identified in a data object identification request and in response to detecting a data object identification request in a communication. In another aspect, communications agent 403 may invoke query director component 412 automatically to locate one or more data objects based on a data object identification request in a communication in response to detecting an indication to present, via a

	output device, some or all of a user message received in a message in the communication. Query director component 412 may be invoked and/or may operate based on the data object identification request and/or data object matching criterion identified to locate a data object prior to, during, and/or after other content received in the communication is processed.") [0200] ("In light of the description provided above, a data object identification response may be generated automatically in response to detecting a corresponding data object identification request by an execution environment 401 of first node 502. Further, the automatically generated data object identification response, may be sent to the node that sent the corresponding data object identification request, automatically in response to the generating of the data object identification response.") *See also* [0174]; [0196] **'635 application (EX2002)** [0182] ("With respect to the method illustrated in FIG. 28, the method as described above may include locating and/or otherwise identifying a data object automatically in response to receiving a data object request that matches and/or otherwise identifies the data object. Further, a mount descriptor may be created and/or otherwise generated automatically, in response to locating and/or otherwise identifying the data object.") *See also* FIG. 2B, 7.
causing, at the first node, receipt of the at least one web page, that results in display, at the first node and via the at least one web page, of a file structure that permits access to the first representation of the at least one file that is	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'906 application (EX2004)** [0144] ("FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an

stored at the at least one server;	output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.") [0073] ("As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.") [0102] ("[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network") [0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output

	device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0146] ("FIG. 6B illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes a user message UI element 610b, analogous to similar UI elements described with respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409 in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.")

	[0147] ("Response window UI element 602b in FIG. 6B includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect, one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **6B,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution environment 401 of first node 502 to request one or more data objects identified in the data object identification response.") [0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user

	interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

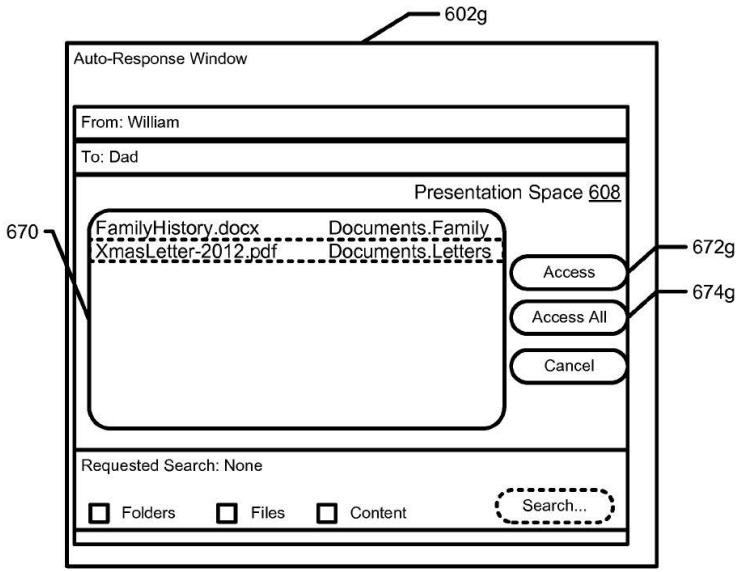

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
Access
Access All
Cancel
672g
674g
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

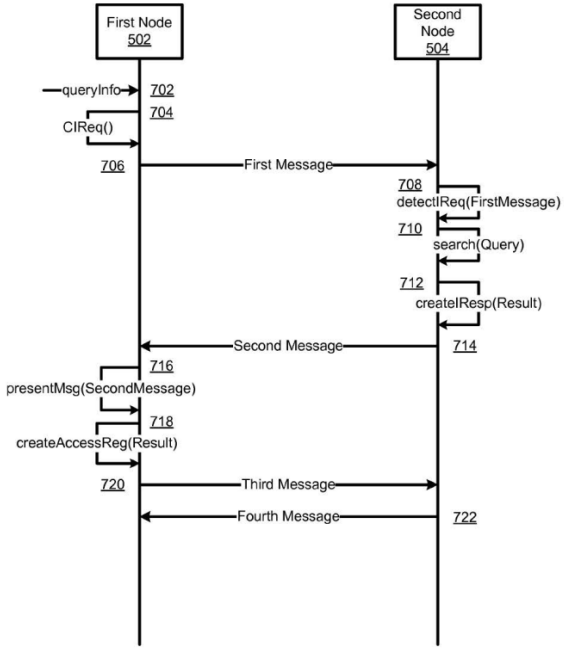


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A. One or more access request constructor components 408 operate in an execution environment 401. The system for processing a data object

	identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") *See also* [0007]-[0011], FIGs. 2A, 2B, 5, 6G, **'635 application (EX2002)** [0102] ("FIG. 6A illustrates an attached mounts UI element 616a identifying, based on the mount descriptor in first message 702, a "Music" folder as a mountable folder from a data store of execution environment 401 of second node 504.") [0112] ("…A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502…") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music"

	folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0168] ("In another aspect, a data object located and/or otherwise identified by data object information in response to receiving a data object request must match a query identified by the data object request. Alternatively or additionally, a data object request may include an instruction and/or input for generating a data object, and locating the data object may include generating the data object. Generating a data object may include creating the data object and/or may include modifying and/or otherwise transforming an existing data object. For example, execution environment 401 of second node 504 may include a template stored in a file system. A data object request may identify the template. A descriptor handler component 404 may operate to identify a document or other data object that may be created, is being created, and/or has been created based on the template. In response to a data object request, execution environment 401 of second node 504 may return a mount descriptor that identifies the created and/or modified data object.")

	[0069] ("…Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user…") [0181] ("A data object request and/or a mount descriptor or respective portions thereof may include at least a portion of a keyword expression, a regular expression, expression including a Boolean operator, an expression including a precedence information, and a structured query language statement." [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0152] ("FIG. 7 includes a data flow that illustrates a previous message 716 (with respect to first message 702) including a data object request. Previous message 716, is shown, may be sent via network 506 by execution environment 401 of first node 502 to execution environment 401 of second node 504. In an aspect, first message 702 may be sent in response to previous message 716.")

	See also [0051]-[0052], [0072], [0112], [0117]-[0119], [0130]
causing, at the first node, receipt of code for storage at the first node and cooperation with a file explorer interface of a client-based file explorer application, for being utilized to: based on the detection of the indication of the selection of the seventh user interface element, cause creation of a second representation of the at least one file, in a location among one or more folders, that is stored at the first node and that is displayable via the file explorer interface of the client-based file explorer application.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **’906 application (EX2004)** [0144] (“FIG. 7 includes dataflow message 716 that illustrates a present message that may be internal to execution environment 401 of first node 502 and/or may be a message sent to a remote device, such as a node including a browser, to send presentation information to present, via an output device, a representation of the data object identification response received in second message 714. FIG. 7 also includes dataflow message 718 that illustrates a create access request message that may be a data flow internal to execution environment 401 of first node 502 and/or a message received via network 506 by execution environment 401 of first node 502. In the dataflow message 718 an ARC component 408 operating in execution environment 401 of first node 502 may be invoked to operate based on result information. Result information may be identified to and/or otherwise by ARC component 408 as part of and/or in response to being invoked.”) [0073] (“As stated, the various adaptations and analogs of the arrangement in FIG. 3A as well as the various adaptations and analogs of the arrangement in FIG. 3B illustrated and described herein are not exhaustive. For example, those skilled in the art will see based on the description herein that arrangements of components for performing the method illustrated in FIG. 2A and the method illustrated in FIG. 2B may each be distributed across more than one node and/or execution environment. For example, such arrangements may operate at least partially in a browser in a one execution environment and at least partially in a server in another execution environment interoperating via a network.”) [0102] (“[A]rrangements of components for performing the method illustrated in FIG. **2A** and the method illustrated in FIG. **28** may each be distributed across more than one node and/or execution

	environment. For example, such an arrangements may operate at least partially in a browser in a one node and at least partially in a server in another node interoperating via a network") [0124] ("Alternatively or additionally, data object information may be received by a query handler component in a message received via a network. For example, a user may identify data object information to a browser operating in a client node. The browser may send the data object information via a network communicatively coupling the client node to a web service provider node. The web service provider node may include a query handler component that may operate to receive the data object information identified in the message from the browser.") [0201] ("With respect to the method illustrated in FIG. 2A, the third message may be sent automatically in response to detecting the data object identification response. In another aspect, a representation of a received data object identification response may be presented, via an output device, to a user. The representation may identify one or more data objects located and/or otherwise identified based on the data object identification request. The representation may be presented automatically, in response to detecting the data object identification response. Further, a user input may be detected that corresponds to and/or otherwise targets the representation. The user input may be processed as an indication to retrieve the data object. As a result an access request may be created or otherwise generated that identifies the data object. The access request may be sent in a message via the network to retrieve the data object. The message may be sent to a node that sent the data object identification response and/or to another node . . . The access request may include a universal data object identifier (URI) that identifies the data object.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") [0146] ("FIG. 6B illustrates a response window UI element 602b as an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502 to present data received in a communication from execution environment 401 of second node 504. FIG. 68 illustrates a contactor UI element 604b, a contactee UI element 606b, a presentation space 608b of the response window UI element 602b. Presentation space 608 includes

	a user message UI element 610b, analogous to similar UI elements described with respect to FIG. 6A. The various UI elements may be presented by one or more UI element handler components 415 in the communications agent 402 in execution environment 401 of first node 502. A reply UI element 622b and a reply all UI element 624b illustrate UI elements in presentation space 608b presented to allow a user of the execution environment 401 of first node 502 to respond and/or otherwise instruct execution environment 401 of first node 502 to perform an operation based on the message received in the communication. FIG. 68 illustrates a data object identification requested query UI interface element 626b where the presented term "None" may be defined to indicate no data object identification request was detected by the content manager component 409 in execution environment 401 of first node 502 in the communication received by execution environment 401 of first node 502 from second node 504 via network 506.") [0147] ("Response window UI element 602b in FIG. 6B includes a data object identification response UI element 628b for receiving user input to display a representation of one or more data objects located and/or otherwise identified by the execution environment **401** of second node **504** and identified in the data object identification response. One or more of the data objects is not included in the message received in the communication. In response to an input directed to data object identification response UI element **628b,** a corresponding UI element handler **415** may interoperate with result handler component **406** to present a located data resources dialog UI element **630b** that includes a presentation space **632b** for presenting a user detectable representation of some or all of the data objects identified in the received data object identification response from the execution environment **401** of second node **504.** In an aspect, one or more data objects located and/or otherwise identified by execution environment **401** of second node **504** may be included in a communication as attachments. In a further aspect, located and/or otherwise identified data objects not included in the message may be identified as described with respect to located and/or otherwise identified data objects dialog UI element **630b** and/or in any other suitable manner. In yet another aspect, files and/or other data entities identified in the data object identification response may be illustrated in a manner similar to attachments rather than in a separate UI element as in FIG. **6B,** and accessed from first node **504** via network **506** as described below.") [0148] ("FIG. 6B illustrates an access UI element 634b and an access all UI element 636b allowing a user to provide user input(s) to instruct communications agent 403 in execution environment 401

	of first node 502 to request one or more data objects identified in the data object identification response.") [0150] ("In response to receiving second message 714, execution environment 401 of first node 502 may present, in a manner described above, data received in the message to a user of execution environment 401 of first node 502 as described with respect to FIG. 6C. Context menu 646c provides UI elements to allow the user to instruct execution environment 401 of first node 502 to retrieve a data object identified in a data object identification response received from execution environment 401 of second node 504. In response to a user input corresponding to a context menu item included in context menu 646c a corresponding UI element handler component 415 may invoke ARC component 408 to construct an access request to retrieve one or more data objects identified in the data object identification response. FIG. 6G illustrates another exemplary user interface that execution environment 401 of first node 502 may present to the user to allow the user to instruct execution environment 401 of first node 502 to construct and send a request. Auto-response window 602g may be presented in response to a data object identification response received in a communication without a user message. Rather than a UI element for displaying a user message, a results pane UI element 670g may be presented that identifies data objects located and/or otherwise identified by execution environment 401 of second node 504 that may be retrieved by execution environment 401 of first node 502. In an aspect, execution environment 401 of second node 504 may process a data object identification request and automatically send a data object identification response without involving the user of execution environment 401 of second node 504. The data object identification response may be received without a user message as FIG. 6G illustrates or may be received with a prewritten and/or automatically generated message. UI elements 672g-674g may be presented by execution environment 401 of first node 502 to allow the user of execution environment 401 of first node 502 to request the retrieval of one or more data objects identified in the data object identification response.") [0151] ("ARC component 408 may be invoked to construct an access request to retrieve one or more data objects identified in the data object identification response. An access request may be based on a URL, such as an HTTP and/or FTP URL that identifies and locates a data object.")

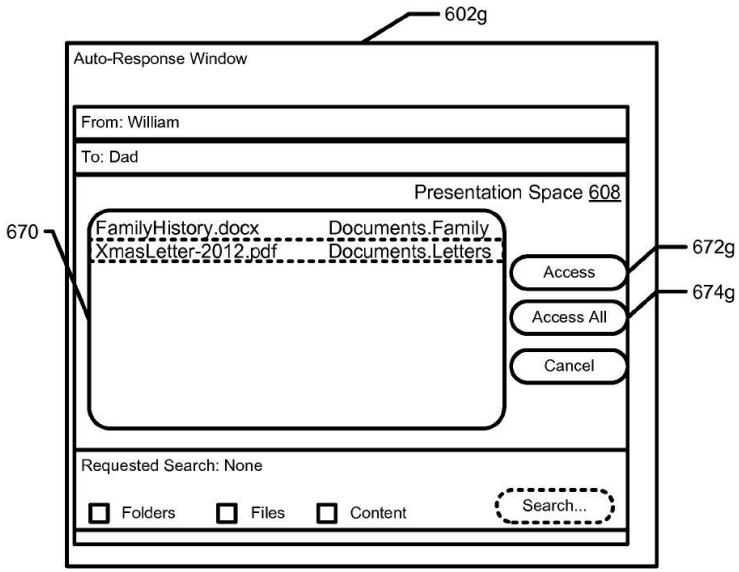

602g
Auto-Response Window
From: William
To: Dad
Presentation Space 608
670
FamilyHistory.docx Documents.Family
XmasLetter-2012.pdf Documents.Letters
Access
Access All
Cancel
672g
674g
Requested Search: None
Folders
Files
Content
Search...

FIG. 6G

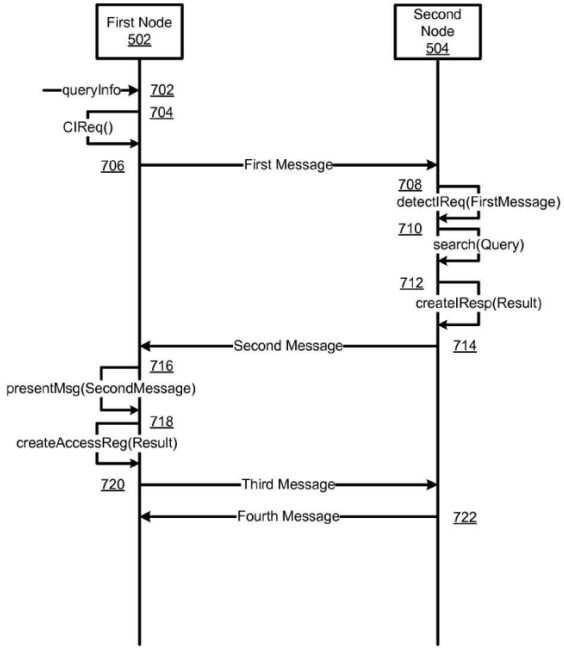


FIG. 7

[0143] ("Returning to FIG. 2A, a block 208 illustrates that the method yet further includes sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. Accordingly a system for processing a data object identification request in a communication includes means for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. For example, the arrangement in FIG. 3A, includes access request constructor component 313 that is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. FIG. 4 illustrates access request constructor component 408 as an adaptation and/or analog of access request constructor component 308 in FIG. 3A. One or more access request constructor components 408 operate in an execution environment 401. The system for processing a data object

	identification request in a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for sending, based on the data object identification response, a third message, via the network by the first execution environment, to access the data object. In FIG. 4, a access request constructor component 408 is illustrated as a component of result handler component 406.") [0185] ("As described above, receiving data object information may include receiving, via network from a node that received the data object information from the user, a message that identifies the data object information. For example, data object information may be received from a browser by a web server.") *See also* [0007]-[0011], FIGs. 2A, 2B, 5, 6G, **'635 application (EX2002)**

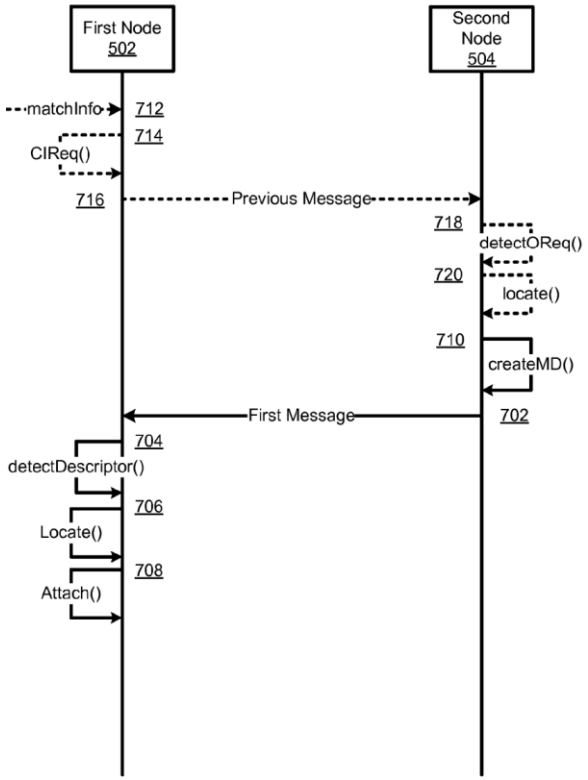


FIG. 7

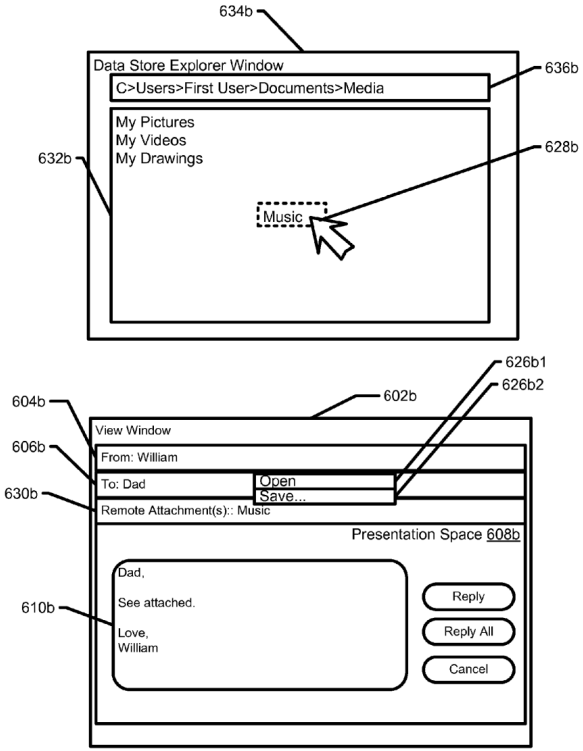


FIG. 6B

[0008] ("Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment. The method further includes detecting the mount descriptor. The method

	still further includes determining a first location in a first data store in the first execution environment. The method additionally includes creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.") [0009] ("Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment; detecting the mount descriptor; determining a first location in a first data store in the first execution environment; and creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store.") [0010] ("Further, a system for sharing a data object in a data store via a communication is described. The system includes a content manager component for receiving , by a first communications agent in a first execution environment according to a first communication protocol via a network from a second communications agent in a second execution environment, a first message, in a communication and addressed to a first user represented by the first communications agent, that includes a mount descriptor that identifies a data object in a second data store in the second execution environment. The system further includes a descriptor handler component for detecting the mount descriptor. The system still further includes a data store navigator component for determining a first location in a first data store in the first execution environment. The system additionally includes a mount service component for, creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the content manager component, the descriptor handler component, the data store

	navigator component, and the mount service component includes an instruction that is executed by the processor during operation of the system.") [0011] ("Methods and systems are described for sharing a data object in a data store via a communication. In one aspect, the method includes receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The method further includes creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The method still further includes including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The method additionally includes sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. Performing at least one the preceding actions comprising the method includes execution of an instruction by a processor.") [0012] ("Also, a system for sharing a data object in a data store via a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent; creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user; including, by the second communications agent, the mount descriptor in a first message, addressed to the first user; and sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store."

	[0013] ("Further, a system for sharing a data object in a data store via a communication is described. The system includes a descriptor handler component for receiving, by a second communications agent that represents a second user, data object information that identifies a data object in a second data store in a second execution environment that includes the second communications agent. The system further includes a descriptor generator component for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. The system still further includes a content manager component for including, by the second communications agent, the mount descriptor in a first message, addressed to the first user. The system additionally includes a com-out component for, sending, via a communications protocol via the network by the second communications agent, the first message for delivery to the first communications agent to create, based on the mount descriptor, a representation, of the data object in the first data storage, that when accessed includes accessing the data object from the second data store. The system also includes a processor, wherein at least one of the descriptor handler component, the descriptor generator component, the content manager component, and the com-out component includes an instruction that is executed by the processor during operation of the system.") [0098] ("FIG. 7 includes a data flow that illustrates a detect descriptor data flow 704 that may be internal to execution environment 401 of first node 502 and/or that may include a message received via network 506 by first node 502. In an aspect, detect descriptor data flow 704 may illustrate a data flow to descriptor handler component 404 operating in execution environment 401 of first node 502. A mount descriptor may be identified to and/or otherwise provided to descriptor handler component 404. Content handler component 402, operating as described with respect to block 202 of FIG. 2A, may identify and/or otherwise provide a mount descriptor to a descriptor handler component 404.") [0109] ("…a system for sharing a data object in a data store via a communication includes means for creating, based on the mount descriptor, a representation of the data object at the location, wherein accessing the representation includes accessing the data object from the second data store…")

	[0111] ("Once a location is identified, in a data store of an execution environment, for attaching and/or otherwise mounting a representation of a data object in a data store in another execution environment; any of a variety means available to the execution environment mounting the data object may be utilized.") [0112] ("…A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502…") [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may

	interoperate with an XtreeemFS client to create a representation of a data object identified in a mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0069] ("…Accessing a data object includes one or more of accessing data in the data object and accessing metadata, maintained by the data store, about the data object.") [0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user…") [0181] ("A data object request and/or a mount descriptor or respective portions thereof may include at least a portion of a keyword expression, a regular expression, expression including a Boolean

	operator, an expression including a precedence information, and a structured query language statement." [0130] ("FIG. 8A includes an exemplary content portion 800 a of an email message. Content portion 800 a includes a user message portion 802 a and a mount descriptor portion 804 a described in more detail above. FIG. 8A illustrates an "application/mount-descriptor" MIME type identifier that may be defined to identify a schema for content including one or more mount descriptors. FIG. 8A illustrates a mount descriptor 804 a including two identifiers for the "Music" folder. A first identifier 808 a illustrates a Samba URL for accessing the "Music" folder via a Samba file sharing system. The second identifier 810 a illustrates a WebDAV HTTP URL that may allow a receiving node to access the identified data object via a WebDAV server operating in and/or available to an execution environment sending the mount descriptor 804 a. An FTP URL 812 a illustrates a file transfer protocol (FTP) that may allow a receiving node to access the identified data object via a FTP server operating in and/or available to an execution environment sending the mount descriptor 804 a") [0041] ("Execution environment 102 may include software components stored in persistent secondary storage 108, in remote storage accessible via a network, and/or in a processor memory. FIG. 1 illustrates execution environment 102 including an operating system 120, one or more applications 122, and other program code and/or data components illustrated by other libraries and subsystems 124. In an aspect, some or all software components may be stored in locations accessible to processor 104 in a shared memory address space shared by the software components. The software components accessed via the shared memory address space are stored in a shared processor memory defined by the shared memory address space. In another aspect, a first software component may be stored in one or more locations accessed by processor 104 in a first address space and a second software component may be stored in one or more locations accessed by processor 104 in a second address space. The first software component is stored in a first processor memory defined by the first address space and the second software component is stored in a second processor memory defined by the second address space.") [0053] ("As used herein, the terms "program" and "executable" refer to any data representation that may be translated into a set of machine code instructions and may optionally include associated program data. The terms are used interchangeably herein. Program representations other than

	machine code include object code, byte code, and source code. Object code includes a set of instructions and/or data elements that either are prepared for linking prior to loading or are loaded into an execution environment. When in an execution environment, object code may include references resolved by a linker and/or may include one or more unresolved references. The context in which this term is used will make clear the state of the object code when it is relevant. This definition can include machine code and virtual machine code, such as Java™ byte code. As used herein, the terms "application", and "service" may be realized in one or more executables and/or in one or more hardware components. The terms are used interchangeably herein.") [0072] ("The arrangement of components illustrated in FIG. 4 may be adapted to operate in a number of execution environments to perform the method illustrated in FIG. 2A and/or the method illustrated in FIG. 2B. FIG. 4 is a block diagram illustrating the components of FIG. 3A, FIG. 3B, and/or analogs of the components of FIG. 3A and FIG. 3B, respectively adapted to operate in an execution environment 401 that includes and/or otherwise is provided by one or more nodes. FIG. 1 illustrates key components of an exemplary device that may at least partially provide and/or otherwise be included in an execution environment. The components illustrated in FIG. 4 may be included in or otherwise combined with the components of FIG. 1 to create a variety of arrangements of components according to the subject matter described herein.") [0112] ("FIG. 8A illustrates a mount descriptor identifying alternative URLs for accessing the "Music" folder in the data store in execution environment 401 of second node 504. A Samba URL 808 a identifies a protocol and a location for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a Samba client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. Samba URL 808 a may include user identification and authentication information to the Samba client if needed. Alternatively, the Samba client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a drive in execution environment 401 of first node 502.) [0051]-[0052] ("A "user interface (UI) element handler" component, as the term is used herein, includes a component of configured to send information representing a program entity for presenting a user-detectable representation of the program entity by an output device, such as a

	display. A "program entity" is an object included in and/or otherwise processed by an application or executable. The user-detectable representation is presented based on the sent information. Information that represents a program entity for presenting a user detectable representation of the program entity by an output device is referred to herein as "presentation information". Presentation information may include and/or may otherwise identify data in one or more formats. Exemplary formats include image formats such as raw pixel data, JPEG, video formats such as MP4, markup language data such as hypertext markup language (HTML) and other XML-based markup, a bit map, and/or instructions such as those defined by various script languages, byte code, and/or machine code. For example, a web page received by a user agent from a remote application provider may include HTML, ECMAScript, and/or byte code for presenting one or more UI elements included in a user interface of the remote application. Components configured to send information representing one or more program entities for presenting particular types of output by particular types of output devices include visual interface element handler components, audio interface element handler components, tactile interface element handler components, and the like. [] A representation of a program entity may be stored and/or otherwise maintained in a presentation space. As used in this document, the term "presentation space" refers to a storage region allocated and/or otherwise provided for storing presentation information, which may include audio, visual, tactile, and/or other sensory data for presentation by and/or on an output device. For example, a buffer for storing an image and/or text string may be a presentation space as sensory information for a user. A presentation space may be physically and/or logically contiguous or non-contiguous. A presentation space may have a virtual as well as a physical representation. A presentation space may include a storage location in a processor memory, secondary storage, a memory of an output adapter device, and/or a storage medium of an output device. A screen of a display, for example, is a presentation space.") [0124] ("…The system for sharing a data object in a data store via a communication includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for creating a mount descriptor for accessing the data object from the second data store by a first data store in a first execution environment including a first communications agent that represents a first user. In FIG. 4, descriptor generator component 414 is illustrated as a component of communications agent 403.") *See also* FIGs. 2A, 2B, 3A, 4, 6A, 6B, 6C, Original Claims 1, 8, 9, 11, 19, 20.

Claim 23	
The method of claim 22, wherein, at a time of sending, to the at least one server, of the signal, one or more representations for the one or more folders that is stored at the at least one server already exist in the location among the one or more folders that is stored at the first node.	Note: In addition to the exemplary excerpts cited in the above and/or below rows along with all Figures and the description thereof (which are incorporated herein by reference), see the excerpts below, which are illustrative and not limiting. **'635 application (EX2002)** [0115] ("As an alternative and/or in case execution environment 401 of first node 502 does not include a Samba client and/or a compatible analog, a mount descriptor may specify one or more alternatives in addition to or instead Samba. For example execution environments may be separated and/or otherwise blocked by one or more firewalls, Mount Service 408 may operate to mount the "Music" found via another identifier in mount descriptor 804 a. Web Distributed Authoring and Versioning (WebDAV) is an extension of the Hypertext Transfer Protocol (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on World Wide Web servers. A WebDAV URL 810 a identifies HTTP as an access protocol and identifies a location in execution environment 401 of second node 504 for accessing the "Music" folder. Mount service component 408 in execution environment 401 of first node 502 may interoperate with a WebDAV client (not shown) included in and/or associated with file system 427, which in an aspect may be and/or may be included in the data store of execution environment 401 of first node 502. WebDAV URL 810 a may include user identification and authentication information to the WebDAV client if needed. Alternatively, the WebDAV client and/or mount service 408 may receive such information from the user and/or from a pre-stored profile. In an aspect, the "Music" folder may be mounted as a folder in, for example, a pre-existing Music folder in execution environment 401 of first node 502. The two music folders may be presented as a merged entity. In another aspect, the "Music" folder identified in the mount descriptor may be mounted at a location in the pre-existing Music folder. The mounted folder may be renamed to distinguish it from a folder with the same name and/or to identify its source by user and/or by node, for example. Alternatives to WebDAV include the file transfer protocol FTP.") [0116] ("In yet another aspect, a representation of a data object a may include and/or may otherwise be based on a replica of at least a portion of the data object. The replica may be synchronized with other replicas and/or an original data object. A mount service 408 may interoperate with an XtreeemFS client to create a representation of a data object identified in a

	mount descriptor at location in a data store. XtreemFS replicates file data across multiple storage servers, which can be distributed worldwide. The replication algorithm is designed to handle problems and failure scenarios that can occur in distributed systems including message loss, network partitions, server crashes, and the like. When a replicated file is opened, XtreemFS automatically elects a primary replica for that file which processes all file operations. If the primary fails, one of the backup replicas will automatically take over after a short failover period. To an application accessing a file via XtreemFS a regular file and/or folder and a replicated representation behave the same or similarly.") [0117] ("For read-only access, replicas of an original folder and/or file may be created and distributed to various replication servers to enhance performance. Through replication a caching infrastructure may be built in order to reduce latency and bandwidth consumption. Partial replicas may be supported, in addition to or instead of full replicas. File, folder, and/or other data object metadata may also be replicated.") [0118] ("Thus, as describes above, a representation of a remote data object may be created based on a mount descriptor that includes information for accessing the data object based on at least one of Samba, content management interoperability services (CMIS), Web Distributed Authoring and Versioning (WedDAV), Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and an identified network file system.") [0119] ("Other distributed file systems that may be accessed to mount a representation of a data object identified in a mount descriptor at a location in a data store. The Andrew File System (AFS), the Distributed File System (DFS), and the Network File System (NFS). Atom Publishing Protocol (AtomPub), file transfer protocol (FTP), RSS, SSH FTP, and content management interoperability services (CMIS) are other means for mounting a data object identified in a mount descriptor.")_

Claim 24	
The method claim 22, wherein the access to the at least one file is allowed at the first node based on whether a date-related criterion or a time-related criterion is met in connection with a receipt of the at least one file.	**'906 application (EX2004)** [0007] ("Also, a system for processing a data object identification request in a communication is described that includes one or more processors and logic encoded in one or more tangible media for execution by the one or more processors that when executed is operable for receiving, by a first communications agent in a first execution environment from a first user represented by the first communications agent, data object information identifying a data object matching criterion; sending, according to a first communications protocol via a network in a communication to a second communications agent in a second execution environment representing a second user, a first message including a data object identification request based on the data object matching criterion, wherein the first message is addressed to the second user; and receiving, by the first communications agent in response to sending the data object identification request in the first message, a second message that includes a data object identification response that identifies a data object in a second data store in the second execution environment, wherein the second message is addressed to the first user and the data object is not received in the second message.") [0093] ("In specifying a resource scope, one or more criterion may be specified. The terms "criterion" and "constraint" are used interchangeably herein, with respect to a resource scope. A criterion included in specifying a resource scope is referred to herein as a "scope criterion". A scope criterion may be expressed via any suitable manner, such as a file name, a wildcard expression, a regular expression, a data store location, a data base query, geospatial coordinates**,** a time, a date, a duration, a relationship to a user, and/or an address - too name a few examples. With respect to a tag, a scope criterion included in defining a resource scope differs from a tagging criterion for the tag in that the scope criterion included in defining a resource scope need not be met in order for a resource to be tagged with the tag. Further a resource scope may be used for purposes other than processing related to a tagged resource.")

	[0095] (“A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. A scope criterion in the role of a scope condition in a scope descriptor is said to be "met" when it indicates the scope descriptor is active, as is said to be "not met" otherwise. . . . Whether a criterion, in a scope descriptor is met or is not met, may depend on an evaluation of the scope criterion that is based on an identifiable entity. For a scope condition, such an entity is referred to herein as a "condition entity". More generally, for a scope criterion, such an entity is referred to herein as a "scoping entity". Exemplary entities include a user, a document, a measure of time, a location, and the like.”) [0125] (“FIG. 6A illustrates an exemplary user interface that may be presented by communications agent 403 operating in execution environment 401 of first node 502. A UI element handler component 415 may operate to present a user interface element to allow a user to identify data object information. A data object information (DOI) UI element 616a is illustrated in FIG. 6A as an example. The same or different UI element handler component 415 may present various other UI elements included in DOI UI element 616a. DOI UI element 616a and one or more UI elements it includes may be presented by one or more corresponding UI element handler component(s) 415 that may operate based on a schema that defines valid data object information and/or data object identification requests for including in a communication via a communications agent 403. The schema defines one or more rules and/or a vocabulary that defines whether data object information and/or a data object identification request is valid. Those skilled in the art will understand that numerous languages and/or schemas for providing data object information and/or data object identification requests currently exist including relatively simple key word based queries to relatively more complicated regular expression languages and data base query languages, such as various versions of structured query language (SOL). New schemas are and will be created, and are, thus, considered within the scope of the subject matter described herein.”)

	See also [0092], [0096]-[0098], [0133], [0175], [0180] ("additional matching criteria.") **<u>'635 application (EX2002)</u>** [0167] ("In another aspect, a data object request UI element may be presented to receive one or more additional matching criteria from a user. Based on the data object request UI element and input from the user, descriptor handler component 404 may receive one or more additional matching criterion information and/or may receive input to modify a matching criterion identified in a received data object request. Descriptor handler component 404 may locate one or more data objects in response to receiving the additional criterion.")