US 20190146783A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2019/0146783 A1**

GINCHEREAU et al. (43) **Pub. Date: May 16, 2019**

(54) **COLLABORATIVE SOFTWARE DEVELOPMENT WITH HETEROGENEOUS DEVELOPMENT TOOLS**

(71) Applicant: **Microsoft Technology Licensing, LLC**, Redmond, WA (US)

(72) Inventors: **Jason Earl GINCHEREAU**, Redmond, WA (US); **Kesavan SHANMUGAM**, Redmond, WA (US); **Charles Eric LANTZ**, Burnsville, MN (US); **Jonathan Preston CARTER**, Seattle, WA (US); **Simon CALVERT**, Sammamish, WA (US); **Daniel LEBU**, Redmond, WA (US); **Anthony VAN DER HOORN**, Portland, OR (US); **Rodrigo Andres Varas SILVA**, Sammamish, WA (US); **Alexandre PANOV**, Duvall, WA (US); **German David Obando CHACON**, Kirkland, WA (US); **Srivatsn NARAYANAN**, Bothell, WA (US); **Oleg SOLOMKA**, Chicago, IL (US); **David Coimbra KHOURSHID**, Orlando, FL (US); **Erich GAMMA**, Gutenswil (CH); **Johannes RIEKEN**, Zurich (CH)

(21) Appl. No.: **15/879,256**

(22) Filed: **Jan. 24, 2018**

Related U.S. Application Data

(60) Provisional application No. 62/585,988, filed on Nov. 14, 2017.

Publication Classification

(51) **Int. Cl.**

G06F 8/71 (2006.01)

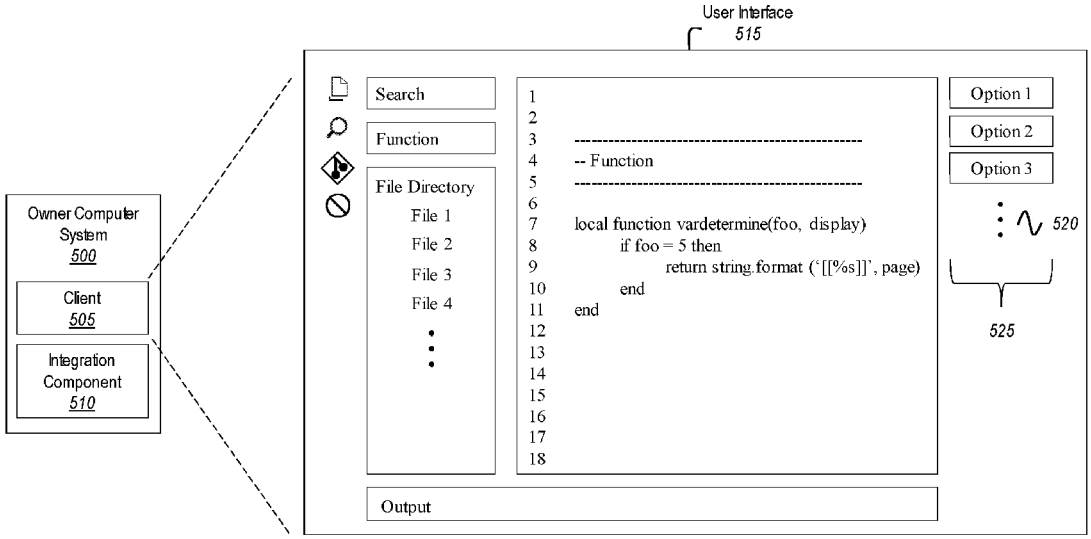
G06F 21/62 (2006.01)

(52) **U.S. Cl.**

CPC ***G06F 8/71*** (2013.01); *G06F 3/0482* (2013.01); ***G06F 21/6218*** (2013.01)

(57) **ABSTRACT**

A collaboration session is provided in which an owner computer system and a participant computer system are both members. Within this session, the collaborators are provided access to a multi-file workspace that is stored locally on the owner computer system. Initially, a set of development tools are identified. These tools are hosted by the owner computer system and are able to operate on the workspace's files. After the tools are identified, they are made accessible to the participant computer system. Later, a request is received from the participant computer system. In some instances, the request is directed to a particular file within the multi-file workspace and is generated using one of the development tools. In this manner, the collaboration session enables the owner computer system's development tools to become accessible to the participant computer system.



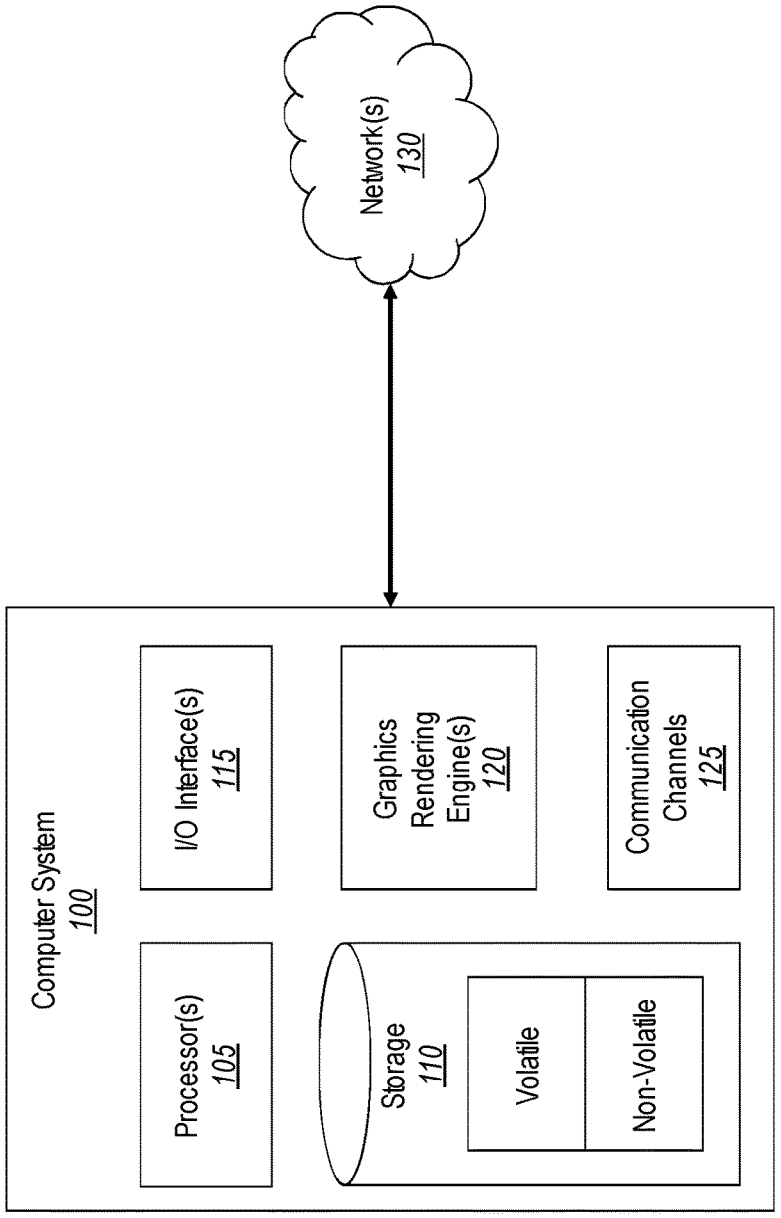

Network(s)
130
Computer System
100
I/O Interface(s)
115
Graphics
Rendering
Engine(s)
120
Communication
Channels
125
Processor(s)
105
Storage
110
Volatile
Non-Volatile

FIG. 1

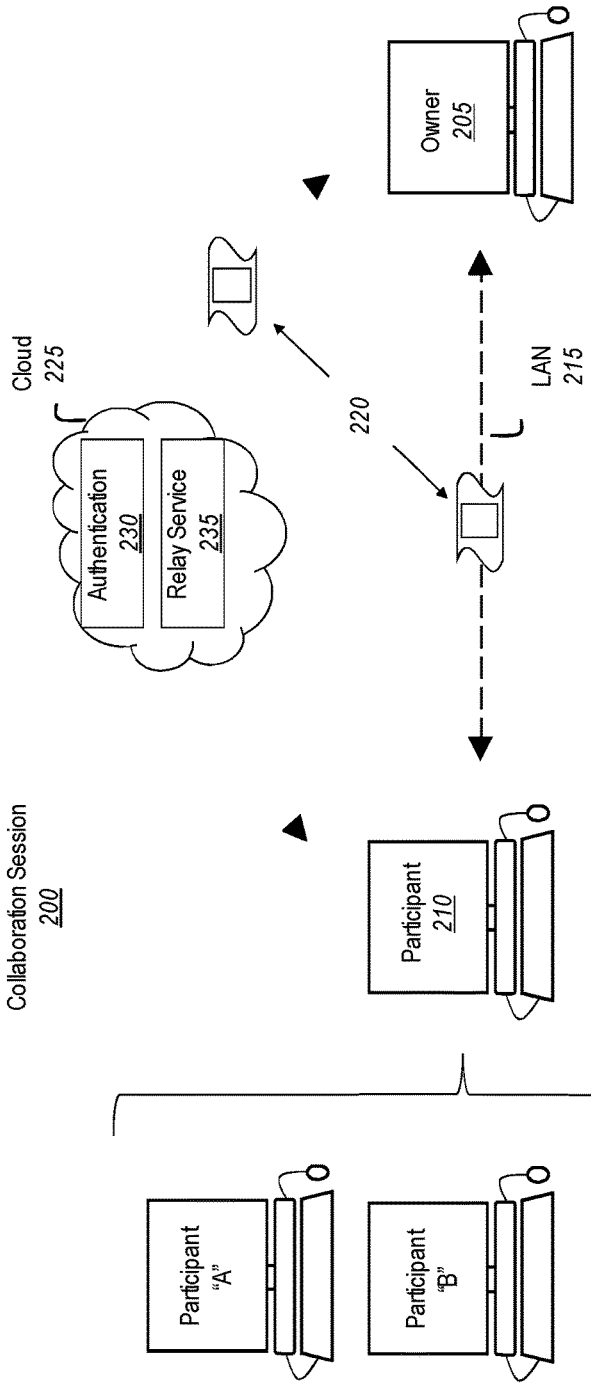

Collaboration Session
200
Participant
210
Participant
"A"
Participant
"B"
Participant
"C"
220
Cloud
225
Authentication
230
Relay Service
235
LAN
215
Owner
205

FIG. 2

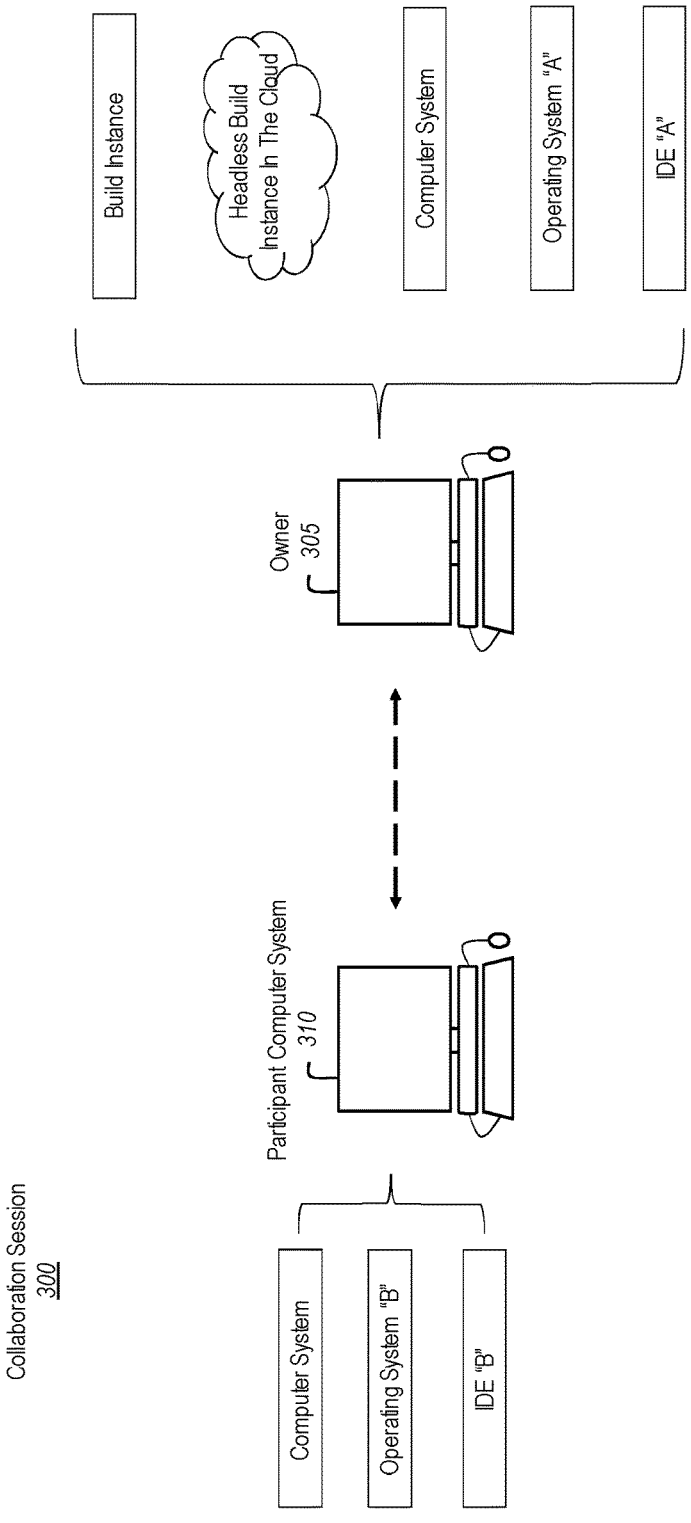

Collaboration Session
300
Participant Computer System
310
Computer System
Operating System "B"
IDE "B"
Owner
305
Build Instance
Headless Build
Instance In The Cloud
Computer System
Operating System "A"
IDE "A"

FIG. 3

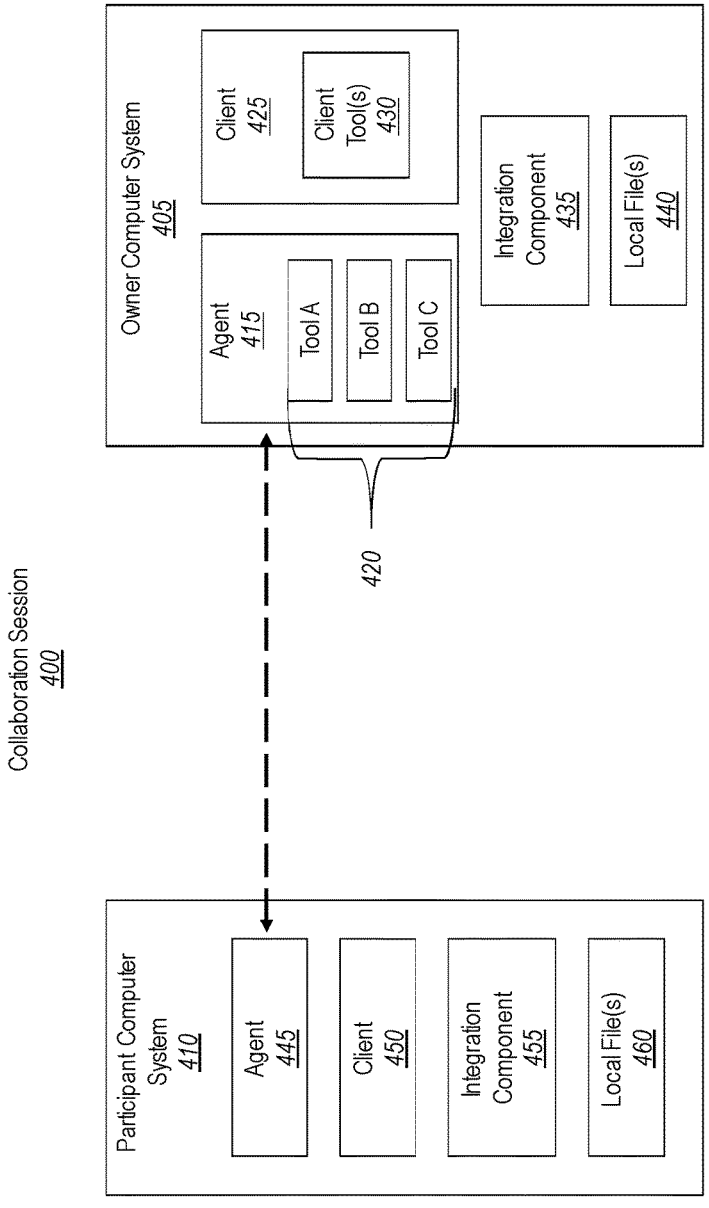

Collaboration Session
400
Owner Computer System
405
Agent
415
Tool A
Tool B
Tool C
Client
425
Client
Tool(s)
430
Integration
Component
435
Local File(s)
440
420
Participant Computer
System
410
Agent
445
Client
450
Integration
Component
455
Local File(s)
460

FIG. 4

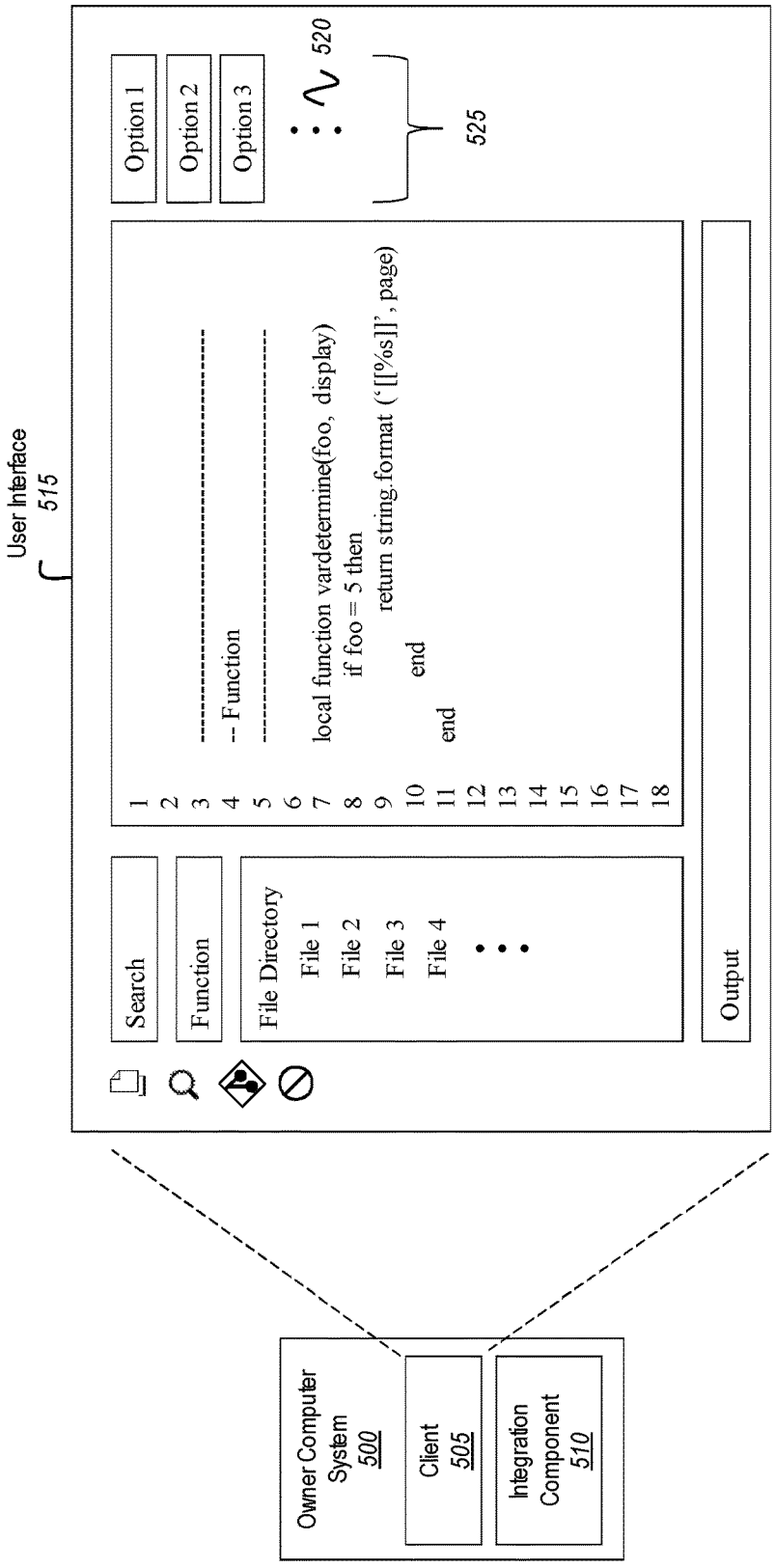

Owner Computer System 500
Client 505
Integration Component 510
User Interface 515
Search
Function
File Directory
File 1
File 2
File 3
File 4
Output
Option 1
Option 2
Option 3
520
525
-- Function
local function vardetermine(foo, display)
if foo = 5 then
return string.format ('[[%s]]', page)
end
end

FIG. 5

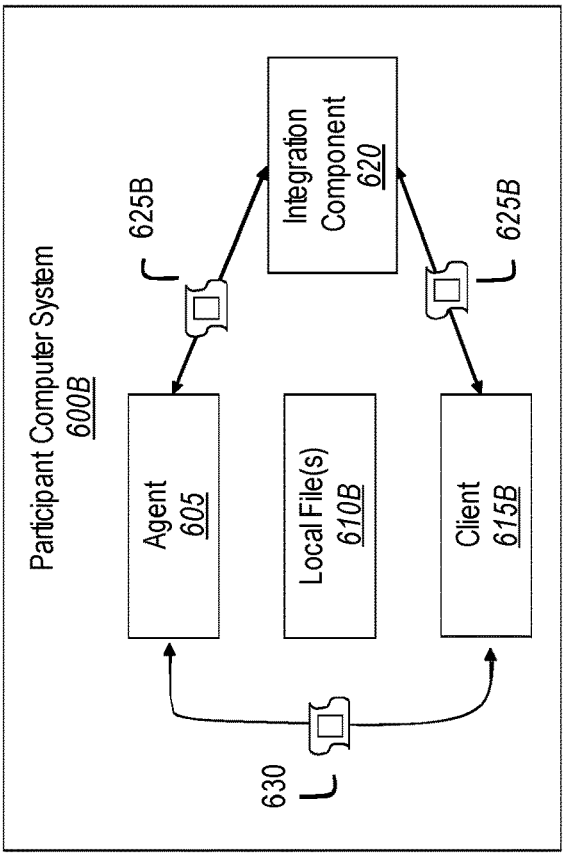

Participant Computer System
600B
Agent
605
Local File(s)
610B
Client
615B
Integration Component
620
625B
625B
630

FIG. 6B

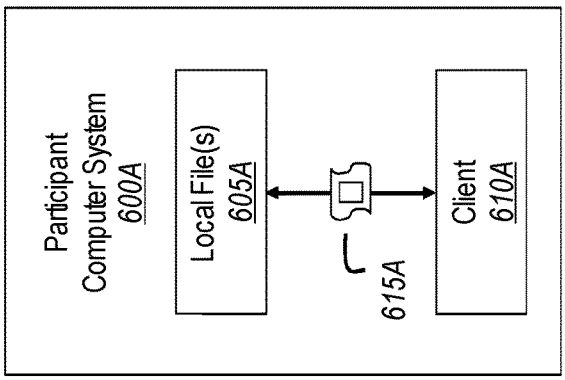

Participant Computer System
600A
Local File(s)
605A
Client
610A
615A

FIG. 6A

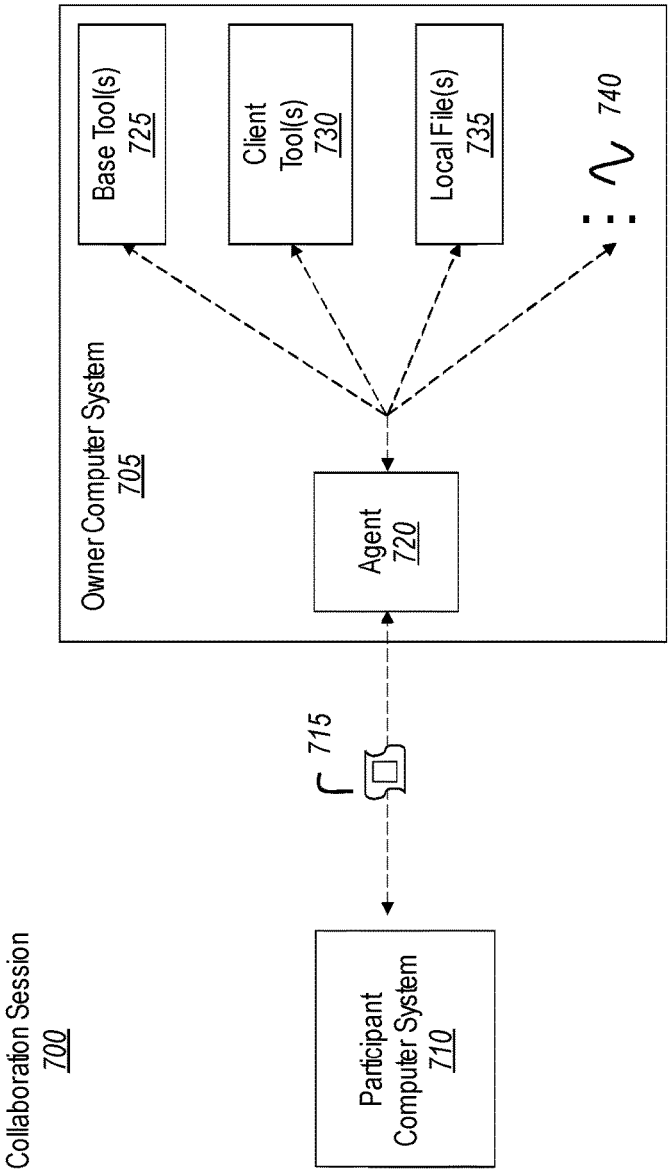

Collaboration Session
700
Participant Computer System
710
715
Owner Computer System
705
Agent
720
Base Tool(s)
725
Client Tool(s)
730
Local File(s)
735
740

FIG. 7

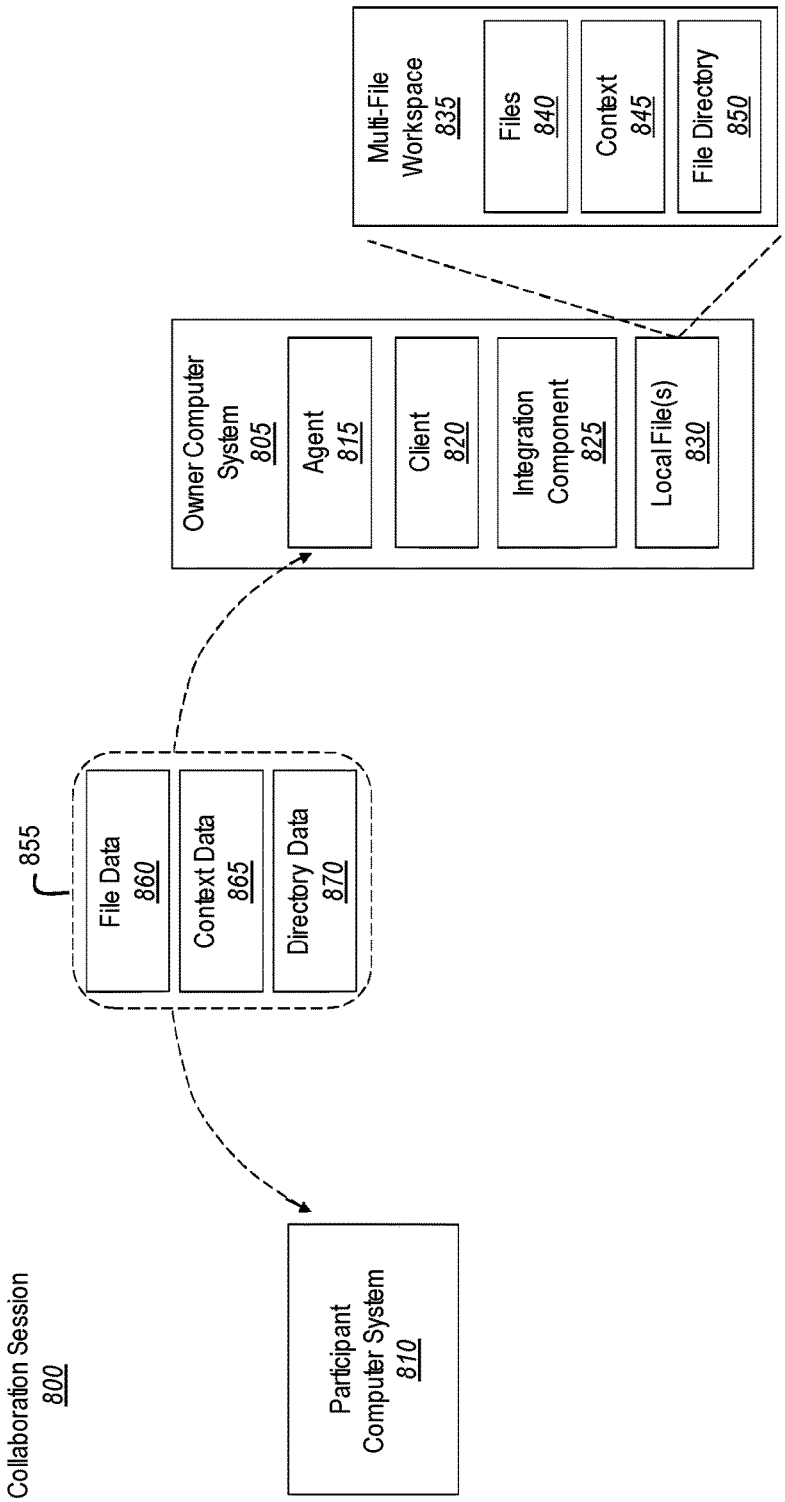

Collaboration Session
800
Participant Computer System
810
855
File Data
860
Context Data
865
Directory Data
870
Owner Computer System
805
Agent
815
Client
820
Integration Component
825
Local File(s)
830
Multi-File Workspace
835
Files
840
Context
845
File Directory
850

FIG. 8

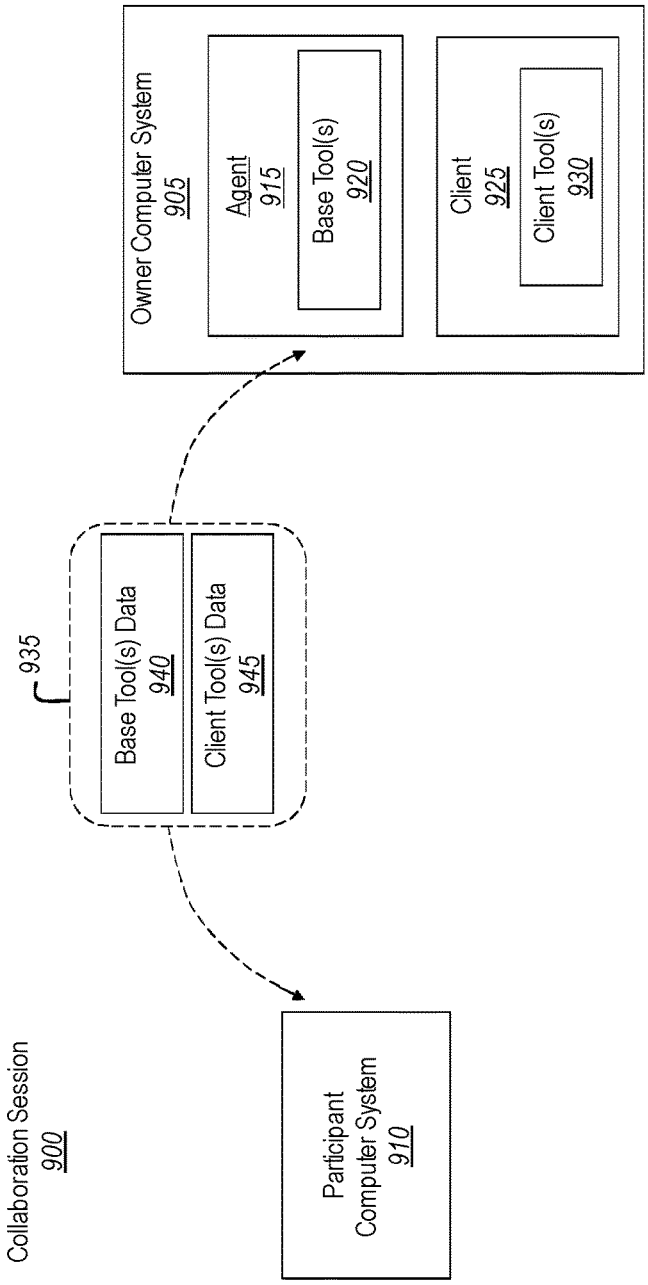

Collaboration Session
900
Participant Computer System
910
935
Base Tool(s) Data
940
Client Tool(s) Data
945
Owner Computer System
905
Agent
915
Base Tool(s)
920
Client
925
Client Tool(s)
930

FIG. 9

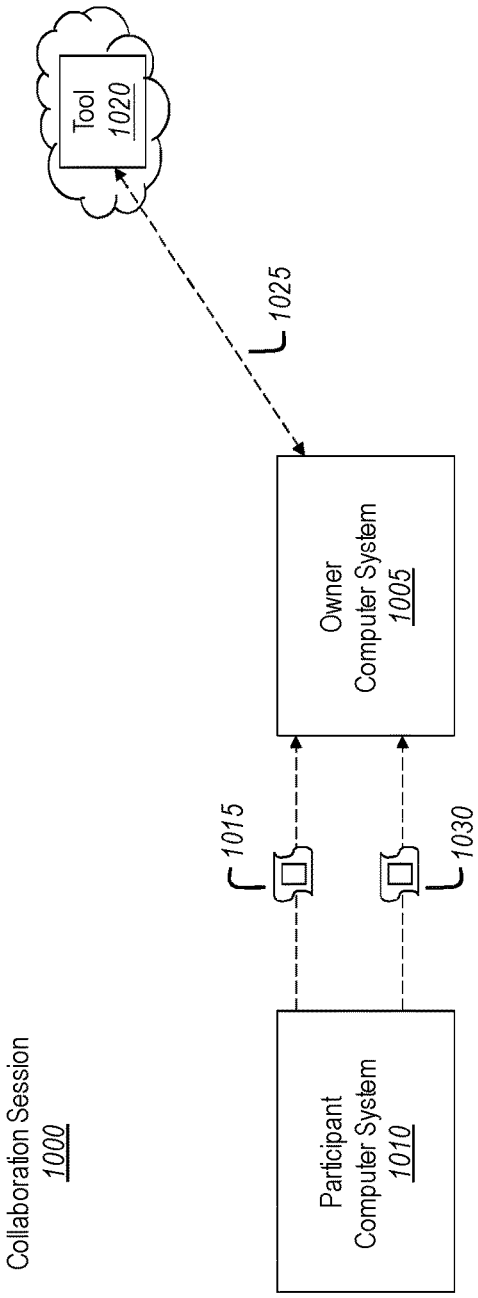

Tool
1020
1025
Owner
Computer System
1005
1015
1030
Participant
Computer System
1010
Collaboration Session
1000

FIG. 10

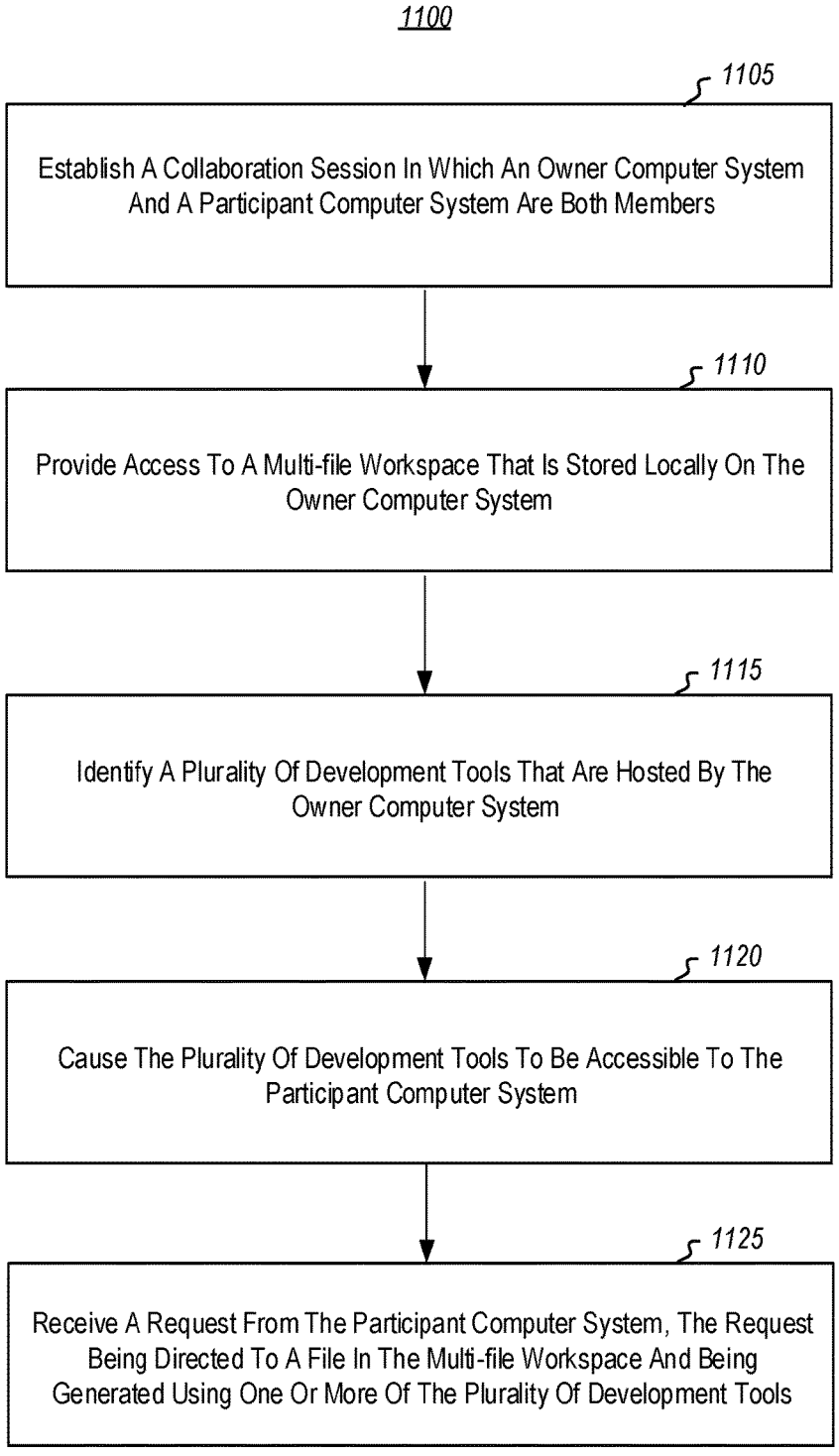

1100
1105
Establish A Collaboration Session In Which An Owner Computer System And A Participant Computer System Are Both Members
1110
Provide Access To A Multi-file Workspace That Is Stored Locally On The Owner Computer System
1115
Identify A Plurality Of Development Tools That Are Hosted By The Owner Computer System
1120
Cause The Plurality Of Development Tools To Be Accessible To The Participant Computer System
1125
Receive A Request From The Participant Computer System, The Request Being Directed To A File In The Multi-file Workspace And Being Generated Using One Or More Of The Plurality Of Development Tools

FIG. 11

COLLABORATIVE SOFTWARE DEVELOPMENT WITH HETEROGENEOUS DEVELOPMENT TOOLS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Application Serial No. 62/585,988, filed on Nov. 14, 2017 and entitled "MULTI-FILE REAL-TIME COLLABORATION ENVIRONMENT," the entirety of which is incorporated herein by reference.

BACKGROUND

[0002] Computers and computing systems have impacted nearly every aspect of modern-day living. For instance, computers are generally involved in work, recreation, healthcare, transportation, and so forth.

[0003] A computer operates by executing a set of executable instructions (i.e. code). These instructions were created in a development environment by a developer in order to create an application. In many instances, the developer will generate many different versions of the code in an effort to improve the computer's operations and to remove any bugs that are present in the code. Once the instructions are compiled, interpreted, and/or built, an underlying computer system executes the instructions to provide the application's functionalities.

[0004] Different tools have been created to assist a developer in writing, editing, testing, and debugging an application's executable instructions. Some of these tools include program code text editors, source code editors, debuggers, and integrated development environments (IDEs), just to name a few. The process of generating and debugging code can be improved through the participation of additional developers. For example, by working together in a team, team members are able to jointly collaborate to review and improve a project.

[0005] The subject matter claimed herein is not limited to embodiments that solve any disadvantages or that operate only in environments such as those described above. Rather, this background is provided to illustrate only one example technology area where some embodiments described herein may be practiced.

BRIEF SUMMARY

[0006] At least some of the embodiments described herein relate to establishing a collaboration session so that a set of development tools may become accessible, in real-time, to multiple collaborators. For example, a collaboration session in which an owner computer system and a participant computer system are both members is established. Within this collaboration session, the collaborators (e.g., the owner and participant computer systems) are provided access to a multi-file workspace that is stored locally on the owner computer system.

[0007] Here, a set of development tools is identified. These tools are hosted by the owner computer system (i.e. the tools reside on the owner computer system and are managed by the owner computer system). Further, these tools are configured to operate on the files included within the multi-file workspace. After the tools are identified, the participant computer system is then given access to the tools so that the participant computer system can perform operations on the multi-file workspace while using these tools. Later, a request is received from the participant computer system. This request is directed to a particular file within the multi-file workspace and is generated using one of the development tools. In this manner, the collaboration session enables the participant computer system to use the owner computer system's development tools to perform operations on the owner computer system's multi-file workspace.

[0008] Therefore, as indicated above, the present embodiments provide many significant advantages which will be discussed in more detail later. As a brief introduction, however, the embodiments improve how a multi-file workspace is edited because multiple entities are able to work on the same workspace as opposed to having to individually download the workspace. Further, the entities are able to continue to use a preferred development tool. In this manner, the embodiments not only improve the editing process, but they also improve the underlying functionality of a computer system.

[0009] This Summary is not intended to identify key or essential features of the claimed subject matter nor is it intended to be used as an aid in determining the scope of the claimed subject matter. Instead, this Summary is provided to introduce a selection of concepts in a simplified form. These concepts are more fully described below in the Detailed Description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] In order to describe the manner in which the above-recited and other advantages and features can be obtained, a more particular description of various embodiments will be rendered by reference to the appended drawings. Understanding that these drawings depict only sample embodiments and are not therefore to be considered limiting, the embodiments will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

[0011] FIG. **1** illustrates an overall view of a computer system.

[0012] FIG. **2** provides an example depiction of how a collaboration session may be initiated.

[0013] FIG. **3** illustrates the various forms and characteristics that a collaborator may have.

[0014] FIG. **4** illustrates an example architecture that may be used to facilitate a collaboration session.

[0015] FIG. **5** shows how an integration component is able to configure a client application's user interface so that it includes collaboration options.

[0016] FIG. **6**A illustrates how a client application is able to access the local files on a participant computer system.

[0017] FIG. **6**B illustrates how a client application's communications can be intercepted and re-routed.

[0018] FIG. **7** demonstrates how an owner computer system is able to process requests submitted by a participant computer system.

[0019] FIG. **8** demonstrates how a multi-file workspace can become accessible to a participant computer system.

[0020] FIG. **9** demonstrates how the tools of an owner computer system can become accessible to a participant computer system.

[0021] FIG. **10** demonstrates how an owner computer system is able to acquire new tools.

[0022] FIG. **11** illustrates an example method for providing a collaboration session in which multiple collaborators are able to jointly work on a multi-file workspace.

DETAILED DESCRIPTION

[0023] The disclosed embodiments are able to provide a collaboration session in which a set of development tools become accessible, in real-time, to each collaborator who is participating in the session. For example, a collaboration session in which an owner computer system and a participant computer system are both members is established. Within this collaboration session, the collaborators (e.g., the owner and participant computer systems) are provided access to a multi-file workspace that is stored locally on the owner computer system.

[0024] Here, a set of development tools is identified. These tools are hosted by the owner computer system and are configured to operate on the files included within the multi-file workspace. After the tools are identified, the participant computer system is then given access to the tools so that the participant computer system can perform operations on the multi-file workspace while using these tools. Later, a request is received from the participant computer system. This request is directed to a particular file within the multi-file workspace and is generated using one of the development tools. In this manner, the collaboration session enables the participant computer system to use the owner computer system's development tools to perform operations on the owner computer system's multi-file workspace.

[0025] An advantage of the disclosed embodiments is that they allow an "owner collaborator" (or simply "owner") to remain in control of his/her multi-file workspace even when outside collaborators (also called hereinafter "participant collaborators" or simply "participant(s)") are joined with the owner in a "collaborative session." Here, the term "owner" can refer to either a "human owner" or an "owner computer system." Similarly, the term "participant" can refer to either a "human participant" or a "participant computer system." In contrast, an "owner computer system" and a "participant computer system" refer only to a computer system and do not include a human operator. Additionally, "collaborator" refers to any entity (e.g., an owner or a participant) that has joined a collaboration session while "collaborators" refers to some or all of the entities in the collaboration session (e.g., the owner and all of the participants).

[0026] As also used herein, a "multi-file workspace" is an assembly of multiple files that operate collectively by interacting with each other. As an example, a code development project may include multiple files of source code that, when executed, operate together to perform complex functions. Thus, a code development project may be considered a multi-file workspace. Other examples of a multi-file workspace include, but are not limited to, text files and/or word processing files (e.g., where the files are linked via a table of contents or some other linking unit), or any other content in which multiple sources of data are joined together. Yet another non-limiting example of a multi-file workspace is a wiki-based workspace that is configured to receive edits and/or markdowns from multiple entities. Accordingly, from this disclosure, it will be appreciated that the embodiments are able to operate with regard to any kind of multi-file workspace. For the sake of brevity, and by way of example only, the remaining portion of this disclosure will focus on a multi-file workspace that includes multiple files of source code.

[0027] Here, it is also worthwhile to note that a "collaboration session," as used herein, is an occurrence in which an owner computer system is joined with one or more participant computer systems in order to jointly collaborate on a multi-file workspace. During this collaboration session, the participant computer systems are provided access to a multi-file workspace that is stored locally on the owner computer system. Further, providing this access also includes causing the multi-file workspace's context to be accessible to the participant computer systems. In this manner, the participants need not download the multi-file workspace. Instead, the participant computer systems act as headless units because editing and other operations may be performed on the owner computer system as opposed to occurring on each of the participant computer systems.

[0028] Of note, collaboration sessions may be initiated for a broad variety of reasons. For example, in some instances, a collaboration session may be established so that the participants can assist the owner in performing a certain function. For instance, if the collaboration involves debugging, the owner might be the primary person tasked with generating the code, whereas the participants may be other developers who are helping the owner debug that code. In a learning environment, the owner may be a student, and the participant may be a teacher. In an alternative learning environment, a professor may be the owner and his/her students may be the participants. In such a scenario, the professor is able to guide his/her students in demonstrating how the workspace operates. In this context, the owner is able to retain administrative control over his/her multi-file workspace.

[0029] Yet another example scenario includes a coding interview process in which the interviewer sets up a workspace environment (e.g., a coding environment). Here, the interviewer is the owner and the interviewee is the participant. In another example scenario, an owner need not be a human on one end of the system. Instead, an owner computer system may be a build server that has no human owner. In this scenario, as will be discussed in more detail later, a human participant, who is using a participant computer system, is able to join a remote codebase (i.e. an "owner" build instance) for the purpose of collaborating in a debugging scenario. Of note, in situations where the owner is a build instance, the disclosed embodiments enable one (or perhaps more) of the participants to assume control of the multi-file workspace. Relatedly, a participant is also able to assume ownership for specific changes to the multi-file workspace.

[0030] Having just described some of the situations in which the embodiments may be practiced, the disclosure will now introduce some of the technical benefits that are provided herein. For example, the disclosed embodiments may be implemented to overcome many of the technical difficulties and computational expenses associated with jointly controlling and collaborating on a multi-file workspace. To illustrate, conventional methods for debugging an application often involve each collaborator installing a workspace's global environment/context and then applying the same data (or steps) in order to reproduce the exact issues that led to finding a bug. Such a process consumes a

significant amount of time, computing resources, and manpower. As used herein, "context" refers to the state of a multi-file workspace.

[0031] In contrast, the disclosed embodiments provide significant advantages because they enable multiple computers to connect to a single computer, which is running a workspace's environmental logic (e.g., services) and which is maintaining a global context of the workspace, to thereby allow the collaborators to jointly collaborate on the multi-file workspace (as opposed to having multiple different workspaces operating on multiple different computers). These operations result in a significant increase in the overall efficiency of the collaborating computer systems.

[0032] Another advantage of the disclosed embodiments is that because only a single workspace is being operated on, the collaborators' operations are synchronized with each other. For example, when multiple collaborators (e.g., code developers) are joined together in a collaborative session, each collaborator is made aware (in real-time) of at least some, and potentially all, of the actions of all of the other collaborators because each collaborator is working on the same workspace. In other words, each collaborator's individual action is "synced" with the actions of the other collaborators.

[0033] Having just described various advantages and high-level attributes of some of the disclosed embodiments, the disclosure will now focus on FIG. **1** which presents an introductory discussion of an example computer system. Following that discussion, the disclosure will focus on FIGS. **2** through **10**. In particular, these Figures illustrate various architectures and supporting illustrations for providing a collaboration session according to the disclosed principles. Finally, the disclosure will focus on FIG. **11**, which presents an example method in which the development tools of one computer system are able to be used by a different computer system.

Example Computer System

[0034] As illustrated in FIG. **1**, in its most basic configuration, a computer system **100** includes various different components. For example, FIG. **1** shows that computer system **100** includes at least one hardware processing unit **105** (aka "processor"), storage **110**, input/output (I/O) interfaces **115**, graphics rendering engines **120**, and various communication channels **125**.

[0035] The storage **110** may be physical system memory, which may be volatile, non-volatile, or some combination of the two. Accordingly, the storage **115** may be referred to as a "hardware storage device" on which computer-executable instructions are stored. The term "memory" may also be used herein to refer to non-volatile mass storage such as physical storage media. If the computer system **100** is distributed, the processing, memory, and/or storage capability may be distributed as well. As used herein, the term "executable module," "executable component," or even "component" can refer to software objects, routines, or methods that may be executed on the computer system **100**. The different components, modules, engines, and services described herein may be implemented as objects or processors that execute on the computer system **100** (e.g., as separate threads).

[0036] The disclosed embodiments may comprise or utilize a special-purpose or general-purpose computer including computer hardware, such as, for example, one or more processors (such as hardware processing unit **105**) and system memory (such as storage **110**), as discussed in greater detail below. Embodiments also include physical and other computer-readable media for carrying or storing computer-executable instructions and/or data structures. Such computer-readable media can be any available media that can be accessed by a general-purpose or special-purpose computer system. Computer-readable media that store computer-executable instructions in the form of data are physical computer storage media. Computer-readable media that carry computer-executable instructions are transmission media. Thus, by way of example and not limitation, the current embodiments can comprise at least two distinctly different kinds of computer-readable media: computer storage media and transmission media.

[0037] Computer storage media are hardware/physical storage devices, such as RAM, ROM, EEPROM, CD-ROM, solid state drives (SSDs) that are based on RAM, Flash memory, phase-change memory (PCM), or other types of memory, or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium that can be used to store desired program code means in the form of computer-executable instructions, data, or data structures and that can be accessed by a general-purpose or special-purpose computer.

[0038] The computer system **100** may also be connected (via a wired or wireless connection) to external sensors (e.g., data acquisition devices). Further, the computer system **100** may also be connected through one or more wired or wireless network(s) **130** to remote systems(s) that are configured to perform any of the processing described with regard to computer system **100**.

[0039] The graphics rendering engine **120** is configured, with the processor(s) **105**, to render one or more objects on a display of the computer system **100**. As a result, a user is able to view the results of the various functionalities of the computer system **100**.

[0040] A "network," like the network(s) **130** shown in FIG. **1**, is defined as one or more data links and/or data switches that enable the transport of electronic data between computer systems, modules, and/or other electronic devices. When information is transferred, or provided, over a network (either hardwired, wireless, or a combination of hardwired and wireless) to a computer, the computer properly views the connection as a transmission medium. As illustrated the computer system **100** includes one or more communication channels **125** that are used to communicate with the network(s) **130**. Transmissions media include a network that can be used to carry data or desired program code means in the form of computer-executable instructions or in the form of data structures. Further, these computer-executable instructions can be accessed by a general-purpose or special-purpose computer. Combinations of the above should also be included within the scope of computer-readable media.

[0041] Upon reaching various computer system components, program code means in the form of computer-executable instructions or data structures can be transferred automatically from transmission media to computer storage media (or vice versa). For example, computer-executable instructions or data structures received over a network or data link can be buffered in RAM within a network interface module (e.g., a network interface card or "NIC") and then eventually transferred to computer system RAM and/or to

less volatile computer storage media at a computer system. Thus, it should be understood that computer storage media can be included in computer system components that also (or even primarily) utilize transmission media.

[0042] Computer-executable (or computer-interpretable) instructions comprise, for example, instructions that cause a general-purpose computer, special-purpose computer, or special-purpose processing device to perform a certain function or group of functions. The computer-executable instructions may be, for example, binaries, intermediate format instructions such as assembly language, or even source code. Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the described features or acts described above. Rather, the described features and acts are disclosed as example forms of implementing the claims.

[0043] Those skilled in the art will appreciate that the embodiments may be practiced in network computing environments with many types of computer system configurations, including personal computers, desktop computers, laptop computers, message processors, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, mobile telephones, PDAs, tablets, pagers, routers, switches, cloud-based machines and infrastructures, and the like. The embodiments may also be practiced in distributed system environments where local and remote computer systems that are linked (either by hardwired data links, wireless data links, or by a combination of hardwired and wireless data links) through a network each perform tasks (e.g. cloud computing, cloud services and the like). In a distributed system environment, program modules may be located in both local and remote memory storage devices.

[0044] Additionally or alternatively, the functionality described herein can be performed, at least in part, by one or more hardware logic components. For example, and without limitation, illustrative types of hardware logic components that can be used include Field-Programmable Gate Arrays (FPGAs), Program-Specific or Application-Specific Integrated Circuits (ASICs), Program-Specific Standard Products (ASSPs), System-On-A-Chip Systems (SOCs), Complex Programmable Logic Devices (CPLDs), Central Processing Units (CPUs), and other types of programmable hardware.

[0045] As discussed above, computer systems are able to provide a broad variety of different functions. According to the principles disclosed herein, the embodiments are able to provide a multi-file real-time collaboration environment. Accordingly, attention will now be directed to FIGS. **2** through **10**, which figures present various architectures and supporting illustrations for making the development tools of one computer available for use to another computer system.

Collaboration Session

[0046] FIG. **2** illustrates a collaboration session **200** in which an owner computer system **205** and a participant computer system **210** are both members. Notably, both the owner computer system **205** and the participant computer system **210** may include all of the features and functionalities that were discussed in relation to the computer system **100** of FIG. **1**. Accordingly, when reference is made to a "computer system," it will be appreciated that such a system may include the features of computer system **100**.

[0047] Here, it will be appreciated that there may be any number of participant computer systems in the collaboration session **200**. For instance, FIG. **2** shows that Participant "A," Participant "B," and/or Participant "C" may be included in the collaboration session **200**. As a result, the owner computer system **205** may be a first member in the collaboration session **200**, Participant "A" may be a second member in the collaboration session **200**, Participant "B" may be a third member in the collaboration session, and so on with potentially no limit. Accordingly, FIG. **2** is for example purposes only and should not be considered limiting. Furthermore, the remaining portion of this disclosure focuses on collaboration sessions that depict only a single participant; however, it should be understood that the principles may be practiced with any number of participants.

[0048] Returning to FIG. **2**, the disclosed embodiments establish the collaboration session **200** so that the participant computer system **210** is able to gain access to a multi-file workspace stored on the owner computer system **205**. In this manner, the participant computer system **210** can operate on the remote workspace as if the remote workspace were local. For instance, a human participant can independently view, edit, and otherwise operate on the remote workspace. By creating this type of collaboration session, all of the collaborators (e.g., the owner and participant(s)) are all able to collaborate on a single multi-file workspace. Accordingly, the disclosed embodiments provide many efficiencies over traditional collaboration processes.

[0049] Here, the collaboration session **200** may be initiated in a variety of ways. In some embodiments, the collaborators (e.g., the owner computer system **205** and the participant computer system **210**) are able to communicate with each other over a local area network (LAN) connection **215**. When this type of connection is available, then the collaboration session **200** may be initiated by sending a request **220** over the LAN connection **215**. In this manner, the collaborators are able to communicate directly in a peer-to-peer manner.

[0050] Of note, in some instances, the owner computer system **205** may desire that the participant computer system **210** be authenticated prior to entering the collaboration session **200**. As such, the embodiments are able to use the cloud **225** to provide authentication services. For example, some embodiments provide an authentication service **230** in the cloud **225**. The participant computer system **210** can use this authentication service **230** to authenticate itself to the owner computer system **205**. After the authentication is complete, then the collaboration session **200** can be initiated and the owner computer system **205** and the authenticated participant computer system **210** can jointly work on a multi-file workspace.

[0051] In other embodiments, the collaboration session **200** is initiated by sending the request **220** through a relay service **235** operating in the cloud **225**. Here, this relay service **235** is able to connect computers that are on different native networks. Accordingly, the embodiments are able to use various services residing in the cloud **225** in order to initiate the collaboration session **200**.

[0052] Other embodiments use a hybrid approach to initiating the collaboration session **200**. For instance, if some participant computer systems are located on the same LAN as the owner computer system **205**, then the request **220** can

be sent to those participant computer systems using the LAN. Additionally, if some participant computer systems are not using the same LAN as the owner computer system **205**, then the request **220** can be transmitted using the relay service **235** in the cloud **225**. Accordingly, the disclosed embodiments are able to use a variety of methods for initiating the collaboration session **200**. Here, it is worthwhile to note that the disclosed embodiments are able to intelligently select whichever process is most efficient to initiate the collaboration session **200**. On a related note, the collaboration session **200** is able to continue to use the respective network connections to maintain the communications between the collaborators.

[0053] Ending the collaboration session **200** may be performed in a variety of ways. In some embodiments, the collaboration session **200** ends through an action of one of the collaborators. For example, one of the collaborators may select an "End Collaboration" option.

[0054] In another embodiment, the collaboration session may end upon expiration of a time-out period. For example, the owner may have configured a policy controlling how long the collaboration session will remain active. Upon expiration of that period, the participants' connection to the collaboration session will be terminated. Additionally, the owner may set a policy indicating a time-out period associated with the shutting down of a client (e.g., an IDE). For example, the owner may set a time-out period to cause the collaboration session to terminate after a predetermined period of time has elapsed after the client was shut down. Such a feature is beneficial because it provides the collaborators adequate time to save any changes or to provide documentation within the code.

[0055] Alternatively, other embodiments are configured to end the collaboration session when the owner computer system shuts down and/or the owner logs off of the owner computer system. In yet another alternative embodiment, the collaboration session may continue to run even after the owner computer system has shut down, gone to sleep, or been logged off. As an example, suppose the human owner decided to perform a quick reboot or had to quickly attend a meeting. Here, because the collaboration session is still active (even though the owner computer system may not be active), the human owner will be able to quickly resume her work when she returns from her activities. An advantage realized by this embodiment is that if any configuration options are adjusted, then those adjustments can be persisted and will be in place for the remaining duration of the collaboration session. In this manner, the collaborators will be able to quickly resume working at the locations where they left off, and they can continue to operate using the same configuration options.

[0056] Other embodiments include audit tracking features that record each collaborators' actions (e.g., their edits). In this manner, these embodiments provide an audit trail that can be reviewed and analyzed at a later time, if the need or desire arises. Once the collaboration session ends based on the principles discussed above, then the audit tracking may also end.

[0057] Similar to FIG. **2**, FIG. **3** illustrates another collaboration session **300**. Here, it will be appreciated that collaboration session **300** is analogous to the collaboration session **200** of FIG. **3**. In the scenario presented in FIG. **3**, the collaboration session **300** includes an owner **305** and a participant computer system **310**.

[0058] In particular, FIG. **3** shows some of the various characteristics that the owner **305** and the participant computer system **310** may have. As an example, the owner **305** may not be a computer system with a human operator. Instead, it may be a build instance of an application, as shown in FIG. **3**. On a related note, the owner **305** may be a headless build instance that is residing in the cloud. In such a scenario, then the various participants are able to operate on the codebase on which the build instance is based.

[0059] Alternatively, the owner **305** may be a computer system (as shown in FIG. **3**) that is using a first type of operating system (e.g., Operating System "A"). In some situations, a human owner will operate the owner **305**. Furthermore, the human owner is able to perform work on the multi-file workspace through the use of a client application that is residing on the owner **305**. As used herein, a "client application" may be any type of application that enables the owner **305** to operate on the multi-file workspace. In situations where the multi-file workspace is a collection of text files, a client application may be a text editor, a word processing program, or any other program suitable to operate on the text files. In situations where the multi-file workspace is a collection of source code files, a client application may be a source code editor, an integrated development environment (IDE), or any other program suitable to operate on source code. Here, it will be appreciated that these client applications are provided permissions to access the multi-file workspace residing on the owner **305**. Although FIG. **3** shows that the owner **305** is able to operate a client application that is in the form of an IDE (e.g., IDE "A"), it will be appreciated that any type of client application may be used, not just an IDE.

[0060] Turning now to the participant computer system **310**, the participant computer system **310** may also be a computer system that is using an operating system (e.g., Operating System "B"). Here it is worthwhile to note that Operating System "B" may be different than Operating System "A." As a result, the owner **305** and the participant computer system **310** need not use the same type of operating system in order to be joined together in the collaboration session **300** to work on the multi-file workspace. Relatedly, the participant computer system **310** need not use the same type of client application (e.g., IDE "B") as the owner **305**. Therefore, according to the principles disclosed herein, a participant is able to use his/her preferred operating system and client application regardless of the type of operating system and/or client application that the owner (e.g., owner **305**) is using. Accordingly, the embodiments are operating system agnostic and client application agnostic.

[0061] Up to this point, the disclosure has presented some of the ways in which a collaboration session may be initiated and some of the characteristics of the collaborators. In light of that background, attention will now be directed to FIG. **4**, which presents some architectural components that may be used to realize the benefits of the disclosed principles.

[0062] FIG. **4** illustrates a collaboration session **400** in which an owner computer system **405** and a participant computer system **410** are members. Here, the owner computer system **405** is a computer system that includes a collaboration agent **415**. As illustrated, this collaboration agent **415** includes a set of base tools **420** (e.g., Tool A, Tool B, and Tool C). Although FIG. **4** shows the collaboration agent **415** as including only three base tools, it will be appreciated that the collaboration agent **415** may include any

number of base tools. Additional details on the base tools **420** will be presented momentarily.

[0063] The owner computer system **405** also includes a client application **425**. As discussed earlier, a client application (e.g., client application **425**) may be a text editor, word processing editor, source code editor, IDE, or any other type of application that enables a user to operate on a multi-file workspace. In light of that understanding, client application **425** may include a client tool **430**. Similar to the above disclosure, although the client application **425** is shown as including only a single tool, it will be appreciated that the client application **425** may have any number of tools. Relatedly, the owner computer system **405** may have any number of client applications installed thereon. As an example, the client application **425** may be an integrated development environment (IDE) that has permissions to access the multi-file workspace. Further, this IDE may manage/host a set of client development tools. In this manner, the IDE can be used to work on the multi-file workspace.

[0064] Here, it will be appreciated that a base tool (e.g., Tool A, Tool B, or Tool C) may be a service or other type of function/tool that is generally common across many or all of the different types of client applications. For example, in the context of code editing, the base tools **420** may include a code completion tool/service, a code debugging service (e.g., a source code error checking tool), a code highlighting service, a code navigation service, a code colorization service (e.g., syntax highlighting in which different colors are applied to the syntax depending on what category a syntax term belongs to), a code refactoring service (e.g., restructuring code without altering its behavior), a code hinting service (e.g., code completion), a source code search tool, a source code control tool, and/or a lightbulb service (e.g., an icon service that provides an expanded display of options).

[0065] Additional services and tools include, but are not limited to, providing member lists, parameter information, symbol services, source code transpilation (e.g., changing the source code so it reads in a different coding language), hover features, smart typing tools/abilities, and quick code information. Relatedly, the client tool **430** may be a tool that is specific to a particular client application and that is not generally common among different client applications. As an example, the client tool **420** may be a tool or service specific to a particular type of IDE.

[0066] According to the principles disclosed herein, the owner computer system **405** is able to make these set of development tools (i.e. both the set of base tools **420** and the client tool **430**) accessible to the participant computer system **410**. Because these tools reside on the owner computer system **405**, the tools have access to the entire context of the multi-file workspace. By making the tools accessible to the participant computer system **410**, a human participant is able to use the tools in light of the entire context of the multi-file workspace. In this manner, the collaborators are able to operate on the multi-file workspace using a set of tools that understand the workspace's entire context.

[0067] Returning to FIG. **4**, the owner computer system **405** also includes an integration component **435** and a set of local files **440**. In some embodiments, the multi-file workspace is included within the set of local files **440** on the owner computer system **405**.

[0068] As discussed earlier, it is often desirable to enable a team of developers to jointly work on a project. According to the principles discussed here, the collaboration session **400** may be initiated so as to enable one or more participants (e.g., participant computer system **410**) to join the owner computer system **405** in collaborating on a multi-file workspace that is stored locally on the owner computer system **405** (e.g., perhaps in the local files **440**).

[0069] To achieve these benefits, the disclosed embodiments cause the integration component **435** to attach, or rather "hook," to the client application **425** in a lightweight manner. For example, the integration component **435** may be a plugin or other type of client extension that hooks into the client application **425** to perform "redirection," "rerouting," and customization operations. For example, the integration component **435** (which is on the owner computer system **405**) is configured to add additional functionalities to the client application **425**. Of note, these additional functionalities are at least directed to establishing and maintaining the collaboration session **400**.

[0070] To illustrate, FIG. **5** shows an owner computer system **500**, which is analogous to the owner computer system **405** of FIG. **4**. In FIG. **5**, the owner computer system **500** includes a client application **505** and an integration component **510**. The client application **505** is also shown as including a user interface **515**. After the integration component **510** hooks itself onto the client application **505**, then the integration component **510** is able to expand upon the abilities of the client application **505**.

[0071] For example, in some embodiments, the integration component **510** will alter the user interface **515** so that it includes additional features related to a collaboration session. To illustrate, a set of new collaboration options **525** have been added to the user interface **515** as a result of the integration component **510** attaching itself to the client application **505**. The set of new collaboration options **525** include Option 1, Option 2, and Option 3. The ellipses **520** demonstrates that any number of options may be added to the user interface **515**. Some of the set of collaboration options **525** may include, but are not limited to, (1) an option to initiate a collaboration session, (2) an option to terminate a collaboration session, and/or (3) an option to acquire information about a particular participant (e.g., the participant's avatar may be displayed and, when the avatar is selected, identifying information about the participant may be displayed).

[0072] Another option is a "pin" to participant's position option. As discussed, the embodiments enable a participant to work on a multi-file workspace that is stored locally on an owner computer system. Included with this functionality is the ability of the participant to independently navigate to areas within the multi-file workspace where the owner computer system is not currently operating (or even in locations where the owner computer system is operating). Furthermore, the embodiments also enable the participant to independently edit files. In light of these abilities, an owner may desire to learn where a participant is currently navigating/operating within his/her multi-file workspace.

[0073] By selecting the pin to participant's position option (e.g., the options can be selectable buttons), the embodiments automatically navigate the owner to the same location as a participant within the workspace. If there are multiple participants, then the owner may initially be prompted to select a particular participant. As an example, suppose an

owner is editing File **1** shown in the user interface **515** in FIG. **5**. At the same time, a participant may be independently editing File **2**. By clicking the pin to participant's position option, the owner can be automatically navigated to File **2**, and in particular to the exact location where the participant is editing File **2**. Therefore, although the embodiments enable participants to independently navigate and edit the files within the workspace, the embodiments also enable the owner to be automatically directed to the locations within the workspace where the participants are working. In some embodiments, this feature is also provided to each of the participants. Therefore, in these embodiments, each collaborator is able to follow the actions of the other collaborators.

[0074] Another option that may be provided among the set of new collaboration options **525** is the option to adjust the access controls of the participants. For example, the owner may be provided with an option to adjust a policy so that participants are allowed to navigate to or edit only certain files. Yet another option is related to an audit feature in which the actions of the participants are recorded and are made available for viewing by the owner. Accordingly, the integration component **510** is able to interact with the client application **505** to enhance the owner's control over the participants in a collaboration session.

[0075] Returning to FIG. **4**, attention will now be directed to the participant computer system **410**. Here, the participant computer system **410** is shown as including a collaboration agent **445**, a client application **450**, an integration component **455**, and local files **460**. Here, the collaboration agent **445** communicates with the collaboration agent **415** to provide the participant computer system **410** access to the multi-file workspace residing on the owner computer system **405**. Additional details on this interaction will be presented later. At this point, it is worthwhile to note that the client application **450** also provides a user interface to the participant so that the participant is able to view (and therefore work on) the multi-file workspace.

[0076] Similar to the integration component **435**, the integration component **455** also attaches itself to the client application **455**. The functionalities of this integration component **455** will now be discussed in relation to FIGS. **6**A and **6**B.

[0077] FIG. **6**A shows a participant computer system **600**A that includes a set of local files **605**A, which are analogous to the local files **460**, and a client application **610**A. In this scenario, the participant computer system **600**A does not have an integration component. Accordingly, when the participant computer system **600**A is thusly configured, the client application **610**A is able to submit requests **615**A to the set of local files **605**A. In this manner, the client application **610**A operates on files that are stored locally on the participant computer system **600**A.

[0078] To enable a participant computer system to operate on a remote multi-file workspace in an operating system agnostic and client application agnostic manner, the participant computer system uses a collaboration agent and an integration component. These features are shown in FIG. **6**B. For example, the participant computer system **600**B of FIG. **6**B includes a collaboration agent **605**, a set of local files **610**B, a client application **615**B, and an integration component **620**. After attaching itself to the client application **615**B, the integration component **620** is able to intercept requests **625**B that are issued by the client application **615**B. Normally, these requests **625**B would be fulfilled using the set of local files **610**B. Now, instead of the requests **625**B being fulfilled using the information from the set of local files **610**B, the integration component **620** intercepts those requests **625**B and routes the intercepted requests **625**B to the collaboration agent **605**. Once the requests **625**B are received, the collaboration agent **605** then routes the intercepted requests **625**B to a collaboration agent residing on the owner computer system (e.g., collaboration agent **415** in FIG. **4**).

[0079] Turning briefly to FIG. **7**, FIG. **7** shows how an owner-side collaboration agent handles requests that are received from a participant-side collaboration agent. Here, the collaboration session **700** includes an owner computer system **705** and a participant computer system **710**. The owner computer system **705** includes a collaboration agent **720**, a set of base tools **725**, a set of client tools **730**, and a set of local files **735**. The ellipses **740** demonstrates that the owner computer system **705** may have additional features.

[0080] In this scenario, a participant-side collaboration agent is able to receive an intercepted request as described in connection with FIG. **6**B. This request is shown as request **715** in FIG. **7**. Here, the participant-side collaboration agent transmits the request **715** to the collaboration agent **720**. After receiving the request **715**, the collaboration agent **720** then processes the request **715**. In some instances, processing the request **715** includes making the set of base tools **725** accessible to the participant computer system **710**. Relatedly, processing the request **715** may include making the set of client tools **730** accessible to the participant computer system **710**. In other instances, processing the request **715** may include making the set of local files **735** accessible to the participant computer system **710**. In this manner, a multi-file workspace residing on the owner computer system **705** may be made accessible to the participant computer system **710**. In some instances, processing the request **715** includes making edits to the files in the multi-file workspace. Edits include, but are not limited to, changing text within the file, adding a new file, deleting an existing file, or any other file editing operations.

[0081] Here, it is worthwhile to note that the participant computer system **710** is not downloading the multi-file workspace. Instead, it is being given access to the workspace through the use of its collaboration agent, its integration component, and the owner-side collaboration agent. In this manner, the participant computer system **710** is able to reach across and perform work on the owner computer system **705**'s multi-file workspace. After the request **715** is processed by the owner computer system **705**, the collaboration agent **720** then transmits a response back to the participant computer system **710**. In particular, the collaboration agent **720** transmits the response back to the participant-side collaboration agent.

[0082] Returning to FIG. **6**B, the collaboration agent **605** will then receive any responses generated by an owner computer system. Once a response is received, then some embodiments will cause the response to be transmitted back to the client application **615**B via the integration component **620**. In other embodiments, however, the collaboration agent **605** is able to directly transmit the response (e.g., response **630**) to the client application **615**B. In this manner, the participant is able to see the results of his/her processed requests.

[0083] Here, an example will be helpful. Suppose an owner establishes a collaboration session in which a par-

ticipant is a member. The owner has asked the participant to assist him/her in debugging source code. The owner begins by debugging a first file while the participant begins by debugging a second file. Of note, both of these files are included in the multi-file workspace and both are stored on the owner's computer system. In this example, the second file is displayed on the participant's computer system even though the file's contents are actually stored only on the owner's computer system.

[0084] Additionally, the participant is able to independently view and edit the contents of the second file even though the owner is currently working on the first file. In this manner, multiple collaborators are able to jointly work on a single multi- file workspace. In some instances, the owner and the participant will be working on the same file. When such a scenario is present, then the owner will be able to see (in real-time) the changes made by the participant, and the participant will be able to see (in-real time) the changes made by the owner. Accordingly, the changes made by the collaborators are made in real-time and are synchronized with each other. In light of this discussion, it will be appreciated that the participant is given the illusion that he/she is working on a local workspace whereas, in actuality, the workspace is not local.

[0085] By following these principles, the disclosed embodiments are able to provide a participant computer system access to a multi-file workspace that is stored on the owner computer system. Furthermore, a human participant is able to view that workspace and to edit that workspace. This viewing and editing can be performed independently of any viewing and editing that an owner may be performing on the multi-file workspace. In this manner, a participant no longer needs to replicate a workspace's context in order to work on that workspace. Additionally, the participant is able to use his/her preferred client application, even if that client application is different from the owner's client application. Even further, the participant is able to use the owner's set of tools, which tools understand the entire context of the multi-file workspace.

[0086] Attention will now be directed to FIG. **8**, which provides additional details for enabling a participant to collaborate on a multi-file workspace. Here, the collaboration session **800** includes an owner computer system **805** and a participant computer system **810**. The owner computer system **805** includes a collaboration agent **815**, a client application **820**, an integration component **825**, and a set of local files **830**. The set of local files **830** includes a multi-file workspace **835**. Here, this multi- file workspace **835** includes a set of files **840**, a context **845** of the multi-file workspace **835**, and a file directory **850**.

[0087] When the collaboration agent **815** receives a request from the participant computer system **810** according to the manner just described, the collaboration agent **815** is able to process the request and return a response **855** to the participant computer system **810**. As shown, this response **855** may include file data **860** (i.e. data concerning the set of files **840**), context data **865** (i.e. data concerning the context **845**), or directory data **870** (i.e. data concerning the file directory **850**). In some instances, this data is metadata while in other instances this data enables the participant computer system **810** to display the multi-file workspace and receive edits directed to the multi-file workspace. In this manner, providing the participant computer system **810** access to the multi-file workspace includes providing access to the workspace's file directory, context, and files. From this information, the multi-file workspace **835** can be displayed on the participant computer system **810**, and the participant will be able to operate on that multi-file workspace.

[0088] From the above disclosure, it will be appreciated that the owner computer system **805** is transmitting sufficient information (e.g., metadata and other information) so that the participant computer system **810** is able to understand what is included within the multi-file workspace **835**. Furthermore, the participant computer system **810** is able to receive enough information so that a visualization of the multi-file workspace **835** may be rendered on a user interface of the participant computer system **810**. In this manner, the participant computer system **810** is acting as a headless entity because a majority of the operations are actually being performed on the owner computer system **805**.

[0089] For example, the participant computer system **810** submits viewing and editing requests to the owner computer system **805**. The owner computer system **805** then processes those requests and returns results back to the participant computer system **810**. As such, the participant computer system **810** is provided the illusion that it is working on a local workspace, but in actuality the workspace is not local and the operations on the workspace are being performed by the owner computer system **805**.

[0090] FIG. **9** shows some additional operations that may be performed. Here, the collaboration session **900** includes an owner computer system **905** and a participant computer system **910**. The owner computer system **905** includes a collaboration agent **915**. As discussed, this collaboration agent **915** is able to maintain a set of base tools **920**. The owner computer system **905** also includes a client application **925**, which client application **925** is able to maintain a set of client tools **930**.

[0091] According to the principles discussed herein, the embodiments are also able to make the set of base tools **920** and the set of client tools **930** accessible to the participant computer system **910**. For example, the participant computer system **910** is able to use the tools that are residing on the owner computer system **905** in order to perform operations on the owner computer system **905**'s multi-file workspace. Therefore, not only is the participant computer system **910** able to view and edit the multi-file workspace on the owner computer system **905**, but it is also able to make use of the tools that are on the owner computer system **905**. The participant computer system **910** is able to receive information **935** that includes base tool data **940** and/or client tool data **945**. In this manner, the participant computer system **910** is able make use of the owner computer system **905**'s development tools.

[0092] In some instances, an owner computer system may not initially have a particular tool that the participant computer system desires to use. FIG. **10** shows that the embodiments are able to address such a situation. Here, the collaboration session **1000** includes an owner computer system **1005** and a participant computer system **1010**. The participant computer system **1010** is able to perform operations on the multi-file workspace because it uses the tools that reside on the owner computer system **1005**. So long as the tool is supported by the owner computer system **1005**, then the human participant is able to use his/her preferred tools to operate on the multi-file workspace. In some instances, however, the owner computer system **1005** may not initially

support a participant's preferred tool. When such a situation occurs, then the processes of FIG. **10** may be performed.

[0093] Here, the participant computer system **1010** submits a tool request **1015** to the owner computer system **1005**. This tool request **1015** may indicate that a particular tool is not currently included in the owner computer system **1005**'s set of development tools (e.g., the set of base tools or the set of client tools). Of note, the set of development tools may also include various other services. Accordingly, the tool request **1015** may also indicate that the participant computer system **1010** is requesting that the owner computer system **1005** acquire that particular tool.

[0094] In response to the tool request **1015**, the owner computer system **1005** is able to query the Internet, a cloud repository, a server, or some other databank in order to find that tool (e.g., tool **1020**). Once found, then the owner computer system **1005** is able to download **1025** that tool **1020**. After downloading **1025** the particular tool **1020**, then the owner computer system **1005** is able to make that particular tool accessible to the participant computer system **1010** during subsequent requests **1030** (e.g., subsequent operations). In this manner, the owner computer system **1005** enables the participant computer system **1010** to use the newly acquired tool (which resides on the owner computer system **1005**) while the participant computer system **1010** performs operations on the multi-file workspace.

[0095] As another example, the owner computer system **1005** is able to make a set of development tools accessible to the participant computer system **1010**. To do that, in some instances, the owner computer system **1005** will determine whether a particular development tool that the participant computer system **1010** is attempting to use is supported by the owner computer system **1005**. Upon a condition in which that particular tool is supported by the owner computer system **1005**, then the owner computer system **1005** will successfully process requests in which the particular development tool is being used.

Example Method(s)

[0096] The following discussion now refers to a number of methods and method acts that may be performed. Although the method acts may be discussed in a certain order or illustrated in a flow chart as occurring in a particular order, no particular ordering is required unless specifically stated, or required because an act is dependent on another act being completed prior to the act being performed. These methods are implemented by one or more processors of a computer system (e.g., the computer system **100** of FIG. **1**). By way of example, a computer system includes one or more computer-readable hardware storage media that store computer-executable code. This computer-executable code is executable by the one or more processors to cause the computer system to perform these methods.

[0097] FIG. **11** shows an example method **1100** of providing a collaboration session in which collaborators are provided real-time access to a multi-file workspace. Initially, method **1100** includes an act of establishing a collaboration session in which an owner computer system and a participant computer system are both members (act **1105**). This act may be performed by the collaboration agent **415** shown in FIG. **4**. The collaboration is initiated by following the principles that were discussed earlier.

[0098] Then, method **1100** includes an act of providing access to a multi-file workspace that is stored locally on the owner computer system (act **1110**). Here, this access is provided to both the owner computer system (because the workspace is stored locally on the owner computer system) and the participant computer system (through the use of the collaboration agents on the owner and participant computer systems). Similar to the above, this act may also be performed, at least in part, by the collaboration agent **415**.

[0099] Next, method **1100** is shown as including an act of identifying a set of development tools that are hosted by the owner computer system (act **1115**). In some instances, some (or all) of the tools are hosted and/or managed by the collaboration agent while in other instances some (or all) of the tools are hosted and/or managed by a client application (e.g., an IDE). In some instances, the tools managed by the collaboration agent and the tools managed by the client application are both made available to the participant computer system. This act may also be performed by the collaboration agent **415**, and the development tools may relate to the set of base tools **420** and/or the set of client tools **430** shown in FIG. **4**.

[0100] Method **1100** also includes an act of causing the set of development tools to become accessible to the participant computer system (act **1120**). This act may be performed by the collaboration agent **415**.

[0101] Finally, method **1100** includes an act of receiving a request from the participant computer system. Here, this request is directed to a particular file in the multi-file workspace. Of note, the request was generated using one or more of the development tools. Here, this act may also be performed by the collaboration agent **415**. Therefore, as a result of the collaboration session, the owner computer system's development tools are made accessible to the participant computer system.

[0102] Accordingly, the disclosed embodiments enable collaborators in a collaboration session to gain access to a multi-file workspace that is stored on an owner computer system. By following the principles disclosed herein, the collaborators are able to independently navigate and edit the files in the workspace. Further, the collaborators are able to make use of the owner computer system's development tools, which tools understand the workspace's entire context. In this manner, the collaborators need not download the workspace or its context/environment. As a result, significant advantages are realized because all edits and optimizations may be performed on a single workspace.

[0103] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. Alcages which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

1. An owner computer system comprising:

one or more processors; and

one or more computer-readable hardware storage devices having stored thereon computer-executable instructions that are structured to be executable by the one or more processors to thereby cause the owner computer system to:

establish a collaboration session in which the owner computer system and a participant computer system are both members;

within the collaboration session, provide access to a multi-file workspace that is stored locally on the owner computer system, the access being provided to both the owner computer system and the participant computer system;

identify a plurality of development tools that are hosted by the owner computer system, wherein each development tool in the plurality is configured to operate on one or more files included within the multi-file workspace, each of the one or more files stored locally on the owner computer system;

cause the plurality of development tools to be accessible to the participant computer system such that when one or more of the plurality of development tools causes a change to one or more files included within the multi-file workspace, the change is made to the one or more files stored locally on the owner computer system; and

receive a request from the participant computer system, the request being directed to a file in the multi-file workspace and being generated using one or more of the plurality of development tools,

whereby, as a result of the collaboration session, the owner computer system's plurality of development tools are made accessible to the participant computer system and changes made to one or more files included within the multi-file workspace using one or more of the plurality of development tools are made to the one or more files stored locally on the owner computer system.

2. The owner computer system of claim **1**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a source code search tool.

3. The owner computer system of claim **1**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a source code navigation tool.

4. The owner computer system of claim **1**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a source code error checking tool.

5. The owner computer system of claim **1**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a code completion tool.

6. The owner computer system of claim **1**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a source code control tool.

7. The owner computer system of claim **1**, wherein the plurality of development tools includes a smart typing tool.

8. The owner computer system of claim **1**, wherein providing the access to the multi-file workspace includes providing access to a file directory of the multi-file workspace.

9. The owner computer system of claim **1**, wherein a client application is operating on the owner computer system, the client application being an integrated development environment, the client application including permissions to access the multi-file workspace, and wherein at least some of the plurality of development tools are hosted by the integrated development environment.

10. The owner computer system of claim **9**, wherein execution of the computer-executable instructions further causes the owner computer system to:

receive a tool request from the participant computer system, the tool request indicating that (1) a particular tool is not currently included among the plurality of development tools hosted by the owner computer system and (2) the participant computer system is requesting the owner computer system to obtain the particular tool;

in response to the tool request, download the particular tool;

after downloading the particular tool, make the particular tool accessible to the participant computer system,

whereby the owner computer system enables the participant computer system to use the particular tool while the participant computer system performs operations on the multi-file workspace stored on the owner computer system.

11. The owner computer system of claim **1**, wherein causing the plurality of development tools to be accessible to the participant computer system includes:

determining whether a particular development tool that the participant computer system is attempting to use is supported by the owner computer system; and

upon a condition in which the particular development tool is supported by the owner computer system, process a subsequent request in which the particular development tool was used.

12. The owner computer system of claim **1**, wherein a collaboration agent residing on the owner computer system includes a set of base development tools, and wherein the set of base development tools are also made accessible to the participant computer system.

13. The owner computer system of claim **1**, wherein providing the access to the multi-file workspace includes causing a context of the multi-file workspace to be accessible to the participant computer system.

14. A method for providing a set of developments tools to multiple collaborators joined together in a collaboration session, the method being implemented by one or more processors of an owner computer system, the method comprising:

establishing a collaboration session in which the owner computer system and a participant computer system are both members;

within the collaboration session, providing access to a multi-file workspace that is stored locally on the owner computer system, the access being provided to both the owner computer system and the participant computer system;

identifying a plurality of development tools that are hosted by the owner computer system, wherein each development tool in the plurality is configured to operate on one or more files included within the multi-file workspace, each of the one or more files stored locally on the owner computer system;

causing the plurality of development tools to be accessible to the participant computer system such that when one or more of the plurality of development tools causes a change to one or more files included within the multi-file workspace, the change is made to the one or more files stored locally on the owner computer system; and

receiving a request from the participant computer system, the request being directed to a file in the multi-file workspace and being generated using one or more of the plurality of development tools,

whereby, as a result of the collaboration session, the owner computer system's plurality of development tools are made accessible to the participant computer system and changes made to one or more files included within the multi-file workspace using one or more of the plurality of development tools are made to the one or more files stored locally on the owner computer system.

15. The method of claim **14**, wherein the plurality of development tools includes a plurality of services.

16. The method of claim **14**, wherein a client application is operating on the owner computer system, the client application being an integrated development environment, and wherein at least some of the plurality of development tools are hosted by the integrated development environment.

17. The method of claim **16**, wherein a collaboration agent residing on the owner computer system hosts at least some of the plurality of development tools.

18. The method of claim **14**, wherein providing the access to the multi-file workspace includes providing access to a file directory of the multi-file workspace and a context of the multi-file workspace.

19. One or more hardware storage devices having stored thereon computer-executable instructions that are structured to be executable by one or more processors of an owner computer system to thereby cause the owner computer system to:

establish a collaboration session in which the owner computer system and a participant computer system are both members;

within the collaboration session, provide access to a multi-file workspace that is stored locally on the owner computer system, the access being provided to both the owner computer system and the participant computer system;

identify a plurality of development tools that are hosted by the owner computer system, wherein each development tool in the plurality is configured to operate on one or more files included within the multi-file workspace, each of the one or more files stored locally on the owner computer system;

cause the plurality of development tools to be accessible to the participant computer system such that when one or more of the plurality of development tools causes a change to one or more files included within the multi-file workspace, the change is made to the one or more files stored locally on the owner computer system; and

receive a request from the participant computer system, the request being directed to a file in the multi-file workspace and being generated using one or more of the plurality of development tools,

whereby, as a result of the collaboration session, the owner computer system's plurality of development tools are made accessible to the participant computer system and changes made to one or more files included within the multi-file workspace using one or more of the plurality of development tools are made to the one or more files stored locally on the owner computer system.

20. The one or more hardware storage devices of claim **19**, wherein the multi-file workspace includes multiple files of source code, and wherein the plurality of development tools includes a source code navigation tool, a source code error checking tool, and a code completion tool.

* * * * *