

the revisions to the shared object attached. The revisions made by User 2 are synchronized with shared object 252 on web server 250. When the email is received at client 230, the revisions made by User 2 are automatically synchronized with the local shared object stored in cache 232. User 3 may then make further revisions to shared object 272 and reply to User 2 with the entire shared object or just the revisions to the shared object included as email attachment 270. The revisions made by User 3 are synchronized with shared object 252 on web server 250. The shared object at client 220 is also updated to include the revisions made by User 3.

[0023] In another example, User 1 may access a shared object either at home on client 200 or at work on client 210 through exchange server 240. Exchange servers are often utilized when access to an external server is not permitted or not available. Revision file 242 includes revisions to the shared object. Revision file 242 may be transferred between clients 200, 210 through a universal serial bus (USB) drive, an email application, or some other mechanism that allows revisions to be transferred back and forth. The revisions are applied to clients 200, 210 such that the local shared object stored in caches 202, 212 may be updated.

[0024] Exchange server 240 may have a restriction on the size of files it can handle (e.g., 2 megabytes maximum). User 1 may upload revision file 242 that includes any revisions to the shared object from client 200 to exchange server 240. Revision file 242 may be transferred from client 200 to client 210 in subsections when revision file 242 exceeds the size limitations of exchange server 240. The file protocol permits a request/fill process for transferring the subsections. In one embodiment, exchange server 240 is associated with an email application. Revisions made by another user (User 2) may be transferred from client 220 to client 210 through web server 250 or peer-to-peer network 260 and then transferred to client 200 through an email account issued to User 1. In another embodiment, client 200 periodically polls exchange server 240 for a current revision file.

[0025] In another example, peer-to-peer network 260 may be established between clients 210, 220 when the connection from clients 210, 220 to web server 250 is lost or when User 1 and User 2 prefer to communicate directly and synchronously in real time. User 1 and User 2 may prefer to communicate through peer-to-peer network 260 because object sharing over web server 250 may result in a lag between when revisions are made at client 210 and when the revisions are available at client 220. The lag may be a result of heavy server traffic. Peer-to-peer network 260 allows revisions to the shared object to be directly transferred between clients 210, 220 instead of through web server 250. In one embodiment, peer-to-peer network 260 is a transmission control protocol/internet protocol (TCP/IP) direct network. The TCP/IP direct network allows revisions to be stored and retrieved quickly.

[0026] Clients 210, 220 may each have a copy of shared object 252 locally stored in cache 212, 222 when web server 250 connectivity is interrupted. The peer group identifier associated with shared object 252 indicates that both User 1 and User 2 are accessing the shared object simultaneously. The users become aware of each other when they both access an established peer-to-peer network (e.g., both users are working on laptop computers on the same airplane). Peer-to-peer network 260 allows User 1 and User 2 to

simultaneously access revisions to shared object 264 on virtual server 262 and implement further revisions. The revisions are instantly replicated on clients 210, 220 such that User 1 and User 2 may actively collaborate on shared object 264. Peer-to-peer network 260 may be disabled when User 1 and User 2 are no longer in the same vicinity (e.g., each user returns to their respective offices) or when User 1 and User 2 no longer wish to communicate in real time. Shared object 252 may then be accessed from web server 250. The transition between accessing shared object revisions on peer-to-peer network 260 and web server 250 is automatic and seamless.

[0027] Clients 210, 220 may receive current revisions from both web server 250 and peer-to-peer network 260. Each revision made to the shared object is associated with a global unique identifier (GUID) and a time stamp. The time stamp identifies the time when the revision was made. Client 210 may modify shared object 252 on web server 250. Client 220 determines whether the local version of the shared object in cache 222 is current by comparing a GUID and a time stamp associated with the cached object to the GUID and the time stamp associated with shared object 252 on web server 250. If the current version is not stored locally, the latest revisions that have not been implemented in the cached object are loaded from web server 250 to client 220 and synchronized with the local file. Thus, the entire shared object need not be loaded to client 220 each time the local version of the shared object is updated.

[0028] In one embodiment, client 220 may determine from the GUID and the time stamp associated with the revision that the same modifications are available from peer-to-peer network 260. However, no action results because client 220 has already implemented the modifications. In another embodiment, client 220 may determine from the GUID and the time stamp associated with the revision that the same modifications are not available from peer-to-peer network 260. Thus, client 220 submits the revisions to peer-to-peer network 260 such that other users connected to peer-to-peer network 260 may synchronize with the current version of the shared object.

[0029] Client 220 may receive another revision from peer-to-peer network 260. The shared object in cache 222 is updated. Client 220 determines whether the current state of the shared object is also available on web server 250 using the GUID and the time stamp associated with the revision. If shared object 252 on web server 250 is not synchronized with the current state of the shared document, client 220 submits the latest revision to web server 250 such that shared object 252 may be updated.

[0030] Asynchronous communication may occur when a client revises the shared object while connected to the system through a physical server. Server limitations may cause shared object synchronizations to be delayed from the time the revisions are implemented by a user. In one embodiment, the client may be revising a locally cached version of the shared object while not connected to the system. Any revisions made by the client may be synchronized with the shared object when the client reconnects to the system through a server. The client may seamlessly transition between local access, synchronous and asynchronous communication such that the user is not aware that the mode of communication has changed. A user may change