

Corresponding conflict pages may also be associated with tabs that are distinct from tab 440. For example, the conflict page tabs may be indented with respect to tab 440 or collapsed below tab 440. The conflict page that includes unmerged conflicts associated with a particular user may be identified by a tab that is distinct from the other tabs such that the user's attention is drawn to the conflict pages generated by that user. The user may then select the tab to navigate to the corresponding conflict page.

[0052] The selected conflict page may be displayed alongside the corresponding master page of the shared object such that the user may reconcile and merge any conflicting revisions in view of the merged revisions. Conflict page 450 is associated with master page 400. Conflict page 450 resembles master page 400 except that any conflicting content containers are highlighted to indicate that the conflict has not been resolved. For example, content container 460 is presented with a highlighted background to draw the user's attention to the unmerged conflict. Content container 460 may have been deleted and synchronized with the shared object before a user revised data in content container 460 thereby creating a conflict. The user may select content container 460 to merge the revision on master page 400.

[0053] In one embodiment, a conflict page is associated with one particular user. Thus, more than one conflict page may be associated with master page 400 when more than one user makes revisions on a page that cannot be merged. All conflicting revisions are stored for all users who are authorized to access the shared object. The user who accesses the conflict page presumably is most concerned with the conflicts generated by that user such that the user may reconcile those conflicts. For example, User 1 is presented with his corresponding conflict page when he selects the conflict indicator. The user may also view a conflict page associated with another user. For example, User 1 may select tab 470 to navigate to a conflict page associated with User 2.

[0054] Many conflict pages associated with one master page of the shared object may accumulate over time. During that time period, the user may have synchronized several revisions with the master version of the shared object located on a server while ignoring any corresponding conflict pages. Thus, the older conflict pages that the user did not reconcile are presumably no longer relevant. In one embodiment, any conflict pages identified as irrelevant may be purged after a predetermined time period has elapsed and the user has synchronized revisions of the page during that time period. For example, the three most recent conflict pages associated with any master page are preserved while any other associated conflict pages are purged after a month from creation.

[0055] In one embodiment, conflict pages are not created during real time communication because conflicts may occur more frequently than during asynchronous communication. Instead, users may collaboratively revise the same content container. Any conflicts may be immediately disposed of since all users can quickly determine if their revisions have been implemented. Alternatively, a user is notified that another user is revising a particular content container. The user may be encouraged to revise a different content container until the other user's revisions are complete.

[0056] FIG. 5 illustrates a block diagram of a system for synchronizing multiple user revisions to a shared object. The

system includes clients 500, 510, 540, 550 and servers 520, 530. Client 500 is coupled to servers 520, 530. Client 510 is coupled to server 520. Clients 540, 550 are coupled to server 530. Client 540 includes store 542 and child store 544. Server 520 includes store 522 and child stores 524, 526. Server 530 includes store 532. Store 532 includes substores 534, 536.

[0057] Stores 522, 532, child stores 524, 526, and substores 534, 536 may store revisions associated with a shared object. Store 522, 532, child stores 524, 526 and substores 534, 536 are hierarchical. For example, store 522 may be associated with an entire shared notebook document. Child store 524 may be associated with a section of the shared notebook document. Child store 526 may be associated with a page of the shared notebook document. In one embodiment, only the most recent version of the top-level shared object is included in store 522. Store 532 may store an entire top-level shared object. Substores 534, 536 are associated with portions of the shared object. For example, substore 534 may be associated with a section of the shared object, and substore 536 may be associated with a different section of the shared object.

[0058] An application may load a shared object from server 520 or server 530 to client 500 without a current version of a particular content container of the shared object. For example, client 500 requests a shared object from store 522. The most recent available version of the shared object is presented at client 500. The most recent available version of the shared object may not correspond to the current version of the shared object because data of the most recent revision is not available in the corresponding child store 526. A request tag is assigned to child store 526 to indicate that client 500 requires the most recent revision data to update the shared object. Child store 526 may also be assigned a time stamp that identifies the time and date when client 500 requested the revision data from child store 526. Child store may also be assigned a GUID that identifies the client that requested the data (e.g., client 500). The request tag, time stamp, and GUID are used to inform client 500 when another client accesses child store 526. For example, client 510 may access child store 526 with the most current revision data. Thus, client 500 is informed that the most current revision data of the shared object is available in child store 526.

[0059] Client 500 may be a user's home computer and client 540 may be a user's work computer. Server 530 may be an exchange server that transfers a revision file between clients 500, 540. The revision file may be used to update a shared object stored on clients 500, 550. In one embodiment, client 500 is restricted from handling files larger than a predetermined size (e.g., 2 megabytes). For example, client 500 may include an email application that limits the size of email messages that may be received. Store 542 includes revisions associated with a top-level shared object. Child store 544 includes revisions associated with a content container of the shared object.

[0060] Client 540 may poll server 530 to determine whether another client has submitted a data revision request. Client 540 may satisfy the request when the latest version of the requested data revision is available in store 542 or child store 544. Client 540 may transfer the entire requested revision to client 500 if the size of the revision file is less