

Re: Homegrown synchronization

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.replication/2007-01/msg00019.html>

- *From:* "David W. Fenton" <XXXusenet@xxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 02 Jan 2007 21:39:05 -0600
-

"rdemyan" <rdemyan@xxxxxxxxxxxx> wrote in
<news:1167757655.154686.125340@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>:

David W. Fenton wrote:

"rdemyan" <rdemyan@xxxxxxxxxxxx> wrote in
<news:1167694158.878560.213810@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>:

David W. Fenton wrote:

"rdemyan" <rdemyan@xxxxxxxxxxxx>
wrote in
<news:1167608261.069673.213770@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>:

[]

2) Updates to the remote
local PC are now done by
simply
replacing the entire backend
file with an update from the
server.

I don't understand this. How do you not lose
updates this way?
How do you trigger the application of the
updates to the
server back-end you've just pushed up to the
server? I would
do it this way:

If the updating of the production backend is working correctly,
then I don't think it is so much a question of losing data.

Re: Homegrown synchronization

Example: WAN User A changes something in a table. When they shutdown MyApp, the CreateExportUpdate code is invoked. It creates an UpdatedDB and copies it to the server Inbox and then moves the local UpdatedDB (just copied to the server) to the Archive Outbox on WAN User A's harddrive. Now User A has shutdown.

SyncApp for a LAN User F finds the update and applies it to the production backend. It then creates a zip file of the production backend (okay, I may change the timing on this method, but that's how it works now). That new zip file is put in the server OUTBOX. When WAN User A logs in again, MyApp finds the new zip file in the server OUTBOX, downloads it, kills the local backend and unzips the new backend.

Now if WAN User B has also sent an update that got applied to production backend prior to WAN User A logging back in again, then a new zip file was created of the production backend that contains both WAN User A and WAN User B's updates. When A logs in again, he gets a fresh backend (via zipped file) containing both updates.

What if A logs off and, realizing she has forgotten something, logs on again, before the server has had time to apply the changes that were just uploaded? What if, for some reason, there's no synch app running on the server's LAN (everyone shut off their computers)?

I just don't see why you don't just apply the same code you're using and download the server back end and then update the local file accordingly. Or, overwrite the data file and run the file that was just uploaded against the newly downloaded file. If it's already got the updates, no problem. If it doesn't, it will now have them.

I think you can argue that this can be a timing issue. But I would counter that as long as WAN User A's update got applied (gets applied) to the production backend, he will eventually have it sent back to him in the zip file.

But with deletes, the order really is important!

I would also bet that this would work the next time a user logs on in 99+% of the cases, if I get the timing issues correct for synchronizing the server and assuming a user doesn't shutdown and then immediately get back in.

Given that that 1% of the time is foreseeable, I'd never put in production a system that doesn't account for that 1% and prevent it from happening, especially when it's relatively trivial for you to

Re: Homegrown synchronization

port code into the user app that you're already using elsewhere.

Now, if WAN User A's update doesn't get applied, then no one gets it including User A. In your scenario, A would still have it. Not sure if this is better. If the update fails and no one has it, except A, then it could be a very long time before anyone notices it. Why, because my WAN users kind of operate as islands. A knows A's data, and C (over at a different WAN site) knows C's data but doesn't know A's too well (and vice-versa) but still wants a look at A's data for comparison purposes. But usually only A will notice if there is a problem (at least in a reasonable amount of time). With the incremental update method, A goes merrily along thinking everything is fine until SOMEONE else points out the problem. But it is A who is in the best position to notice it.

Will A be happy to lose all her most recent data? I just don't see how you could take that risk. Either the data matters or it doesn't. You're depending on timing to go right when you have no control whatsoever over the timing of the events. I think that's completely unwise.

The alternative, which is to apply incremental updates, could mean that updates from other user's (like WAN User B) are not applied to WAN User A's local backend in a timely fashion (which is I think what you are getting at). Why? Because right now MyApp only checks for updates (regardless of whether they are incremental updates or zipped copies of the production backend) at startup. There's no checking going on during a user's session. If necessary, I can maybe add that later. Right now I want to get this working smoothly.

I still don't get it. The question is: how do you get the updates from the server to the user. You're replacing the user's data file, if I'm understanding correctly, even though you can't know for certain that the user's most recent updates have been incorporated into that data file on the server.

I'm saying:

Download the server data file.

Apply the last set of updates to it, and *then* replace the existing data file. That way you know for certain that you've got both the latest server data and all the changes that were last sent up to the server. And it is completely non-dependent on the timing of the updates on the server.

Re: Homegrown synchronization

My comments above are tentative. I don't disagree with you; I'm just relaying my thought process. I may have missed something or be looking at it wrongly. This has been a long and arduous process and I'm starting to go bonkers :)

You've written the code already. Why can't you just incorporate it into the user app?

[]

The sync app loops through files in the INBOX folder on the server to find all of the update files sent by WAN users. Each file does include a date and time, so maybe I could somehow use that.

But that's only the date and time of the *file*, not the data and time of the updates included within it. If somebody doesn't sync until a couple of days after they do updates, the file could be dated several days after the actual data, and that means you can't process them in order of file date — you'd want to process them in the order that the changes were made.

No, actually each update file has the user's logon ID and the date which the update file was created in the file name. So I do know when the update file was created and by whom.

But that doesn't tell you the date of the *data* changes in it. Now, if you can count on the upload completing each time the user exits the app, perhaps that's OK. But I'm not sure that's safe.

After an update file is applied to the production backend, a record is written to a table so that that update is not applied again by the sync app (as part of the Form Timer code). I may want to allow admin people, however, to specify that the sync app should try again on a file that the administrator would select. This functionality has not been added.

Why not just move the file to an archive once the update has been applied to the server?

Re: Homegrown synchronization

Yeah, that's a possibility.

Seems to me there's nothing to lose by doing so.

—

David W. Fenton <http://www.dfenton.com/>
usenet at dfenton dot com <http://www.dfenton.com/DFA/>

.