ACP Issues

I’ve organized the known ACP issues under three headings:

STABILITY

The ACP app is built on top of Excel and the S&P CIQ Excel add-in. Our code is written in C#, and it must communicate with the add-in by writing data into cells of an Excel spreadsheet; there doesn’t seem to be any way around this.

**Problem**: Excel is difficult to manage from another program; it is designed as a single-user desktop application, not as a component in a scalable multi-user solution.

**Solution:** There is none. We can reduce the impact of this problem by reducing our usage of Excel to the minimum possible, but we cannot eliminate it altogether.

**Problem**: The CIQ plug-in is not very reliable. At random intervals, it corrupts its data in such a way that only a full uninstall and re-install of the plug-in will resolve the problem. (This has nothing to do with ACP per se; it happens with any use of the add-in.)

**Solution:** There is no solution, but a workaround is possible. We can automate the process of un- installing and re-installing the add-in. This process could be set up to happen at regular intervals and/or when we detect that the add-in has stopped responding.

**Problem**: The VBA scripts built into the spreadsheets sometimes encounter errors. This causes them to pop up a dialog box on the server and to lock up Excel completely.

**Solution:** We’ve been fixing the VBA errors as we find them, and I believe we’ve found almost all of them, but there’s no way to guarantee we’ve found them all.

**Problem**: The spreadsheet scripts use the Windows clipboard. All Excel spreadsheets – and, indeed, all Windows programs running on the server – share the same clipboard. It is very easy for one user to overwrite the clipboard data that another user needs.

**Solution:** We must remove all usage of the clipboard from the spreadsheets. In most cases, this will be easy. In a few cases – where entire charts are being copied – more work will be needed. Each field on the chart will have to be copied individually.

SPEED

Many of the stability problems also manifest as performance problems; our code retries failed operations several times. Very often, the second or third attempt works, but that means that some operations take two or three times as long as they should.

**Problem**: The server we are running on has low CPU and memory capacity. With just two users, ACP uses 100% of the CPU and 71% of the available memory. Any additional users beyond that will cause the whole system to slow down sharply.

**Solution**: Get a better server.

**Problem**: The communication between our code and Excel is based on Microsoft’s Interop technology which is very slow and is not designed to server as part of a scalable multi-user solution.

**Solution:** We could add support for TCP/IP to our spreadsheets via the WINSOCK control; this would allow us to communicate with the spreadsheets without going through Interop. This seems to be the only practical alternative, but we haven’t made a test to see how well this would work.

**Problem**: Report Generation is very slow.

**Solution:** All of the above issues contribute to this, but there are also factors that are specific to Report Generation. We use VBA scripts built into the spreadsheets to do much of the work; these are not designed for speed. (We’ve converted some parts to C#, but not all.) The logic of the remaining scripts can be moved into the C# code, once we’ve solved the Interop-speed problem (see above).

**Problem**: Report download is very slow.

**Solution:** This happens because the server is already straining to run the application; handling downloads is necessarily slower when all of the system’s resources are already in use. Solving the other speed problems will solve this one.

ORDINARY BUGS

The ACP software – the part of the system we can control – is not yet perfect. There are known bugs.

**Problem**: Once a user is done using ACP, resources are still allocated to him on the server.

**Solution:** Those resources are supposed to be freed within 30 minutes. There’s a complication: as long as the user’s browser is running (even in the background), the resources won’t be freed. On Saturday, March 19, I introduced code to make the ACP server more aggressive about freeing up those resources.

**Problem**: Sometimes, the generated report is missing data or uses the wrong set of peers.

**Solution:** The missing data seems to be due to errors that occur when trying to read data from the Excel spreadsheets during report generation. On Saturday, March 19, I increased the number of times we retry those operations and added code to surrender processor time to the spreadsheet. I’ll be testing those on Sunday March 20.