NPO Accounting action plan (draft)

Christopher Allan Webber

2014-09-29 Mon

1 Introduction and summary

The NPO Accounting project exists to fill a gap where other accounting projects have fallen short. Other accounting systems are either proprietary and un-hackable, or free but not nearly as flexible as needed to meet the needs of an organization such as the Software Freedom Conservancy.

1.1 Current state

The project builds itself off of Ledger, a programmer's accounting tool. With a robust set of built-in reporting and querying tools, an elegant and expressive syntax, Ledger provides the groundwork to build the system you need.

Unfortunately, the downside of building off of Ledger is threefold:

- For a much smaller organization, the technical expertise to customize an accounting workflow based on command line tools may not be there. It would be good to have a system that already does the technical heavy lifting on the backend and provides a simple, clear user interface while still allowing for flexibility.
- For an organization as large as Conservancy, eventually the system grows to a state of technical complexity where in order to do basic bookkeeping, one must be highly technically inclined. Being able to task off basic bookkeeping becomes hard.
- Once the books have grown to the size of Conservancy's, reports can take a very long time to execute if they re-run the command line tool many times. (This is because every time the tool is run, it must re-load the entire accounting history into memory.)

The NPO Accounting project seeks to resolve all of these issues. At the present moment, the last is already addressed, with tooling built so that running multiple commands over extremely large data sets does not bear the weight of having to continuously reload this data. With this foundation in place, we are set to build the rest of the system.

1.2 Future state

So what will the NPO Accounting project look like when the project reaches maturity? The general philosophy of the project is: "batteries included, and the rest is scriptable." By this we mean: you don't *have* to be a technically savvy programmer to use the basics of the system. A web interface will exist which guides you through the most common tasks of entering data and generating reports. But the moment that your organization has special needs, an API exists where you can set a programmer to the task of adding new features. Most features added will be a one-time job: once they are written, non-technical users can continue to take advantage of them through the user interface.

This will require quite a bit of technical work, but the general direction of this work has already been determined, as discussed in the Technical Plan section of this document.

User interface design will be done through a process of spec'ing out the design, sketching quick wireframes, making mockups, then executing.

1.3 A word on branding

Before we get into the technical plan for the project, a word on branding.

"NPO Accounting" is the current term for the project, but probably should not be the eventual name of the project. For one thing, this project should be usable far outside the world of nonprofits. While addressing the needs of nonprofits is still a large goal of the project, we believe that this project will useful for various types of organizations, large and small, as well as a feasible system for personal finance particularly for those who are excited about providing a friendly and attractive user interface for themselves and their families while also being

Aside from this, "NPO Accounting" is simply not a very catchy term. Likely, a new one will be developed soon. One likely possibility is AcornCount. Future versions of this document may be revised to include the new name.

2 Technical plan

2.1 Technologies used

The NPO Accounting project will make use of the following technologies:

- *Ledger:* Flexible command line accounting tool.
- *Python:* Existing bindings for ledger are already written in Python.
- *Asyncio:* A python library for asynchronous execution. Writing the NPO Accounting project in this framework will allow us to build the system in such a way that running long queries or other complicated procedures do not block the user interface, resulting in a better user experience.
- A wide variety of existing nice python web development libraries, such as the *Jinja2* templating language.

2.2 User interface

The user interface for the project will be web-driven.

There will be a "simple queries" page that shows some amount of recent entries with an interface to add new entries as well as edit existing entries. The interface will be simple enough to walk through adding a standard ledger entry, but will also verify the

Reports can be set up to "immediately display" or "be deferred". Deferred reports are ones known to take some time... instead of displaying immediately to the user, they show up in a "pending reports" button. When the report has finished executing, the backend notifies the frontend and a notification becomes available. At this point the user can select the report and view it.

2.3 Scripting API

The backend API for defining new functionality will be written in Python. Plugins will be able to be written to add new queries, new reports, and new

These will consist of:

- Logic to interface with the Ledger API to insert, query, or manipulate data.
- Controller logic for receiving and returning commands to the frontend, as well describing to the frontend how to display and process information.
- Hooks for task processing and asynchronous communication with the frontend.

2.4 Primary/Replicant setup and synchronization

The NPO Accounting project doesn't use a traditional SQL database... instead, it stores transaction history in plaintext files. This actually provides a tremendous advantage... not only are the files easy to manipulate, they can be stored in a modern version control system like git. This means that at any time, one can travel backwards and forwards in history to see what the state of one's accounting files looked like in June 2008. Not just the state of the books from June 2008, but literally the exact files from June 2008! This is a powerful feature for auditing your finances. It also greatly reduces the risk that past work will somehow be accidentally lost. It's always there in the version control history!

However, this does pose a challenge. Assuming that many transactions are scheduled to happen at the same time, some writing (such as adding or editing new transactions) and some reading (generating reports, doing queries, etc)... we don't want users to wait around for reports to finish just to enter their information, especially if those reports are especially complex. Is there any way to prevent this system from blocking?

Borrowing an idea from traditional database design, NPO Accounting will have "primary" and "replicant" checkouts. All of these are checkouts of the repository from git. The "primary" account can do writes to the files and commit them when it is done and has a queue of write submissions. The "replicants" are read-only... they can receive queries and read from files to prepare information for the user, but to avoid collisions, they never write to disk directly. Queries and reports are dispatched to them and returned to be processed for viewing by the user. The "primary" checkout informs the "replicant" accounts when changes have been made to the files, and when the replicants have finished their current tasks, they update their git checkouts to the latest database state.

2.5 Updating the python bindings

In addition, the ledger bindings to python are a bit wanting. Bradley Kuhn has plans to improve them, which will make scripting for this system nicer, but will also benefit all other users of the ledger Python API.

3 Conclusions

The NPO Accounting project has the right combination of technical robustness, flexibility, and focus on user experience to do the job. If we can raise enough funds, we should be able to implement all the above to build a system to help individuals and organizations of all kinds keep themselves financially healthy.