eosDAC Development team report

Period 2020/07

The intent of the development team reports has been to help translate technical language into a descriptive narrative to help investors, DAC members, and the general public, to understand where we are, progress made, and pending issues.

The main advance of previous month was the creation of the test environment. Thanks to this, it was possible to invite different stakeholders to create DACs and be able to report issues and get their feedback. In several cases there were problems, and manual assistance was needed to “unlock” the DAC creation process. The priority for this month was to help these users to create test DACs, and to identify which cases actually stopped the release of the DAC Factory, and which cases could be dealt with in a next stage. This greatly reduced the amount of to-dos.

What was done in this month?

During the month, the following cases were closed:

- **Case # 42 charging CPU to the DAC, not to the user**
  It was concluded that it this not necessary anymore thanks to the integration with Greymass / Anchor. Nothing was developed.

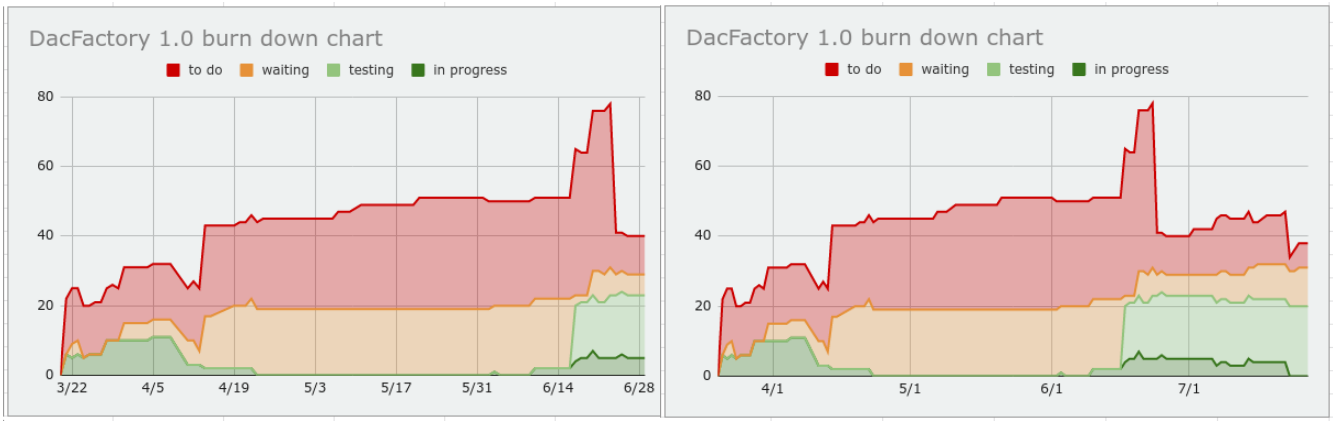
- **Case # 24 HTTPS certificates for dacfactory subdomains.**
  A multi-domain certificate has been configured for the created DACs. Nothing was developed.

- **Case # 92 redirect HTTP pages to HTTPS.**
  Now that we have HTTPS, the URLs are redirected to the secure address. Nothing was developed.

- **Case # 101 error with 2 decimal DACs.**
  Thanks to the tests carried out, multiple errors have been detected, the most critical one was that the system did not allow the creation of DACs with less than 4 decimal places. Dallas developed the solution along with automated tests to test extreme cases (no decimal places, or up to 8 decimal places). The solution is already applied in Jungle 3.

What is pending?

Thanks to the reduction of pending tasks, we currently have only 6 tasks to complete. Of these, 3 are visual problems, 2 are error handling, and only one of them is a functional problem. In the graph we see the red area representing the to-dos.



Comparison of June vs. July charts.

In this graph we can see that the development team has no ongoing tasks (dark green area), and has few tasks to do (red area). The graph shows that the largest area is in the validation and approval of tasks, or quality control (light green and orange area). The cases to be tested and approved total 31 tasks against only 6 to start. The main focus during this month has been to test, creating test DACs, but there has been no documentation of the process.

To formalize the tests that serve as supporting documentation of approval for the project release, case # 105 “Create flowchart for testing worker proposals” was created. Dallas is working on documenting the worker proposal module to be the basis for testing.

Testing is # 1 priority for the first week of August.

Non-technical advances

Added more information to the documentation wiki at https://github.com/eosdac/documentation:

- A specific page centralizing the documentation on custodians was created. A table of the custodians' call recordings for easy access was added to that page.

- Started creating documentation per module of the eosDAC, with the first module about the new worker proposal system.

Conclusions

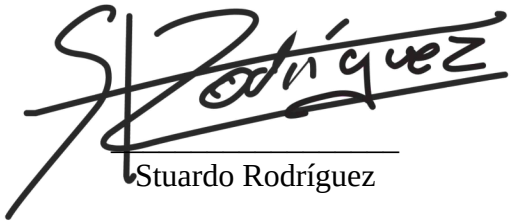
In June’s report, the main weakness identified was the lack of definitions of specific responsibilities and the amount of hours that development team would invest in the development of the project. The whole weight continues to fall on Michael Yeates, who is responsible for servers, block production, website updates, content creation, AlienWorlds project creation, maintenance of authentication libraries for EOS and WAX, development, testing, and authorization of code changes of the DAC client and DAC factory. To free this bottleneck, it takes one or two people to make progress in parallel.

We see in the graph that the greatest amount of work is not development, but quality control, and the majority of pending development can be done by a front end programmer, without experience in blockchain. We still do not know the financial statements to to know how much we can pay to fill those positions.

Could the same programmer who developed the system perform quality control? This is a typical question for empirical software development companies or startup companies with no experience in the software development area. A company does not want to spend more resources hiring someone new to validate work already paid. They just expects it to work.

The truth is that when a programmer self-validates his own code, he unconsciously steps back on his own footsteps in the sand. We realize this with the example of the first DAC created by someone who didn't code the system. The DAC could not be created by using a different configuration than what has always been used.

The cost of not validating the work by someone else, in financial systems like blockchain is very high. Today we cannot know if we can hire someone for that task because we do not have formal financial statements. It is my personal opinion that in order to move forward, we need financial statements to know if we can hire someone from QA.



Stuardo Rodríguez