SEGP2 Plugin Group Milestone 6 Report

Total Hours Spend: approx. ??? hours

Executive Summary

Milestone 6 began by reviewing what were the leftover work and issues needed to be addressed. The following were the tasks that had been set by the group at the beginning of the sprint:

- Finish up the 3rd pillar of plugins: deployment.
- Create plugins and apply what was created and set up plugins development into them.
- Front-end plugins union: Finalize and have the plugins framework set.

The first and second items were taken up by Ida, Mohammad, Fil and Ken, while Xiaodong, Keane and Huang Jian were assigned to the last item.

Ida managed to update the `plugins_manager.rb` to support extension point, introduced by Mohammad, during the installation of a daemon plugin. She also managed to add an uninstall method and accessor script to uninstall the a plugin from Earth. Mohammad provided his expertise to assist Ida in getting the implementation correct. Ida had spend ?? hours on updating the scripts, while Mohammad spend ?? hours to lend his expertise to her, while implementing the file metadata as an extension point.

Ken created the `list_plugins` script to enable the administrator to see what plugins had been installed. Also, he unified all the plugins related scripts (`sign_plugin`, `install_plugin` and `uninstall_plugin`) into `earth_plugins`. This root accessor script will call the appropriate script to perform the required plugin task. Ken had spend 4 hours creating the scripts. He also spend 3 hours to help in preliminary integration tasks.

Fil created a daemon feature that redirects the daemon status to the front-end and he spent 80 hours on it.

Xiaodong, Keane and Huang Jian had conceptualized and implemented their versions of GUI plugins framework into functional prototypes.

Xiaodong and Keane's implementation used the Rails plugin generation mechanism to get the plugins installed. They require the user to add specific codes into the controllers to activate the plugins. On the other hand, Huang Jian's implementation used his own way to get the plugins installed, without any code activation in the controller. Both ways are similar, but the difference is only in the ways the framework is implemented. Thus, they have agreed to work togather further to consolidate the ideas, or finalize one one as the final framework, while the other as a proof-of-concept branch.

All three spend ?? hours in bringing the concepts to functional implementations.

Individual Reports

Attached are the individual reports of each member of the group.

Ken S'ng's Milestone 6 Report

Executive Summary

I suggested to improve the way the daemon plugins are to be managed, as I find it is not very organized to have a several scripts to manage the plugins. I have created a main script that consolidates all the plugin management scripts (installation, uninstallation and signing). Besides that, I also have added a script to list all the currently installed plugins, as there is no way to know whether what plugins has been installed prior to the script was written. Besides that I provided my assistance to another subgroup to investigate how integration can be done.

Tasks and Activities Assigned

- Overseeing the progress of the developments of the plugins mechanism development.
  - Plugins deployment retrofitting
    - Description: This task is geared towards retrofitting plugins install scripts from RSP to support the latest additions to the plugins framework.
    - Affected files: `script/list_plugins`, `script/earth_plugins`, `lib/earth_plugins_interface/plugin_manager.rb`
    - Git commits: `(pamalite/earth) 96828da9db7b7c5f19ac67f0b336727052d18b9e`
    - Estimated time taken (planned): 5 hours
    - Estimated time taken (actual): 4 hours
- Helping out on pre-integration investigation and analysis.
  - Description: This is not outlined in the M6 plan. I was engaged to help out on figuring out how to use `git` to merge all the subgroup repositories as a preliminary step before the actual integration begins.
  - Idea/Solution: N/A
  - Affected files: N/A
  - Git commits: *Not relevant as the commits done were experimental.*
  - Estimated time taken (actual): 3 hours

Resource Contributions

For this milestone, I have estimated that I had spend about 12 hours (including 5 hours meeting time) over the spread of 3 weeks. I have budgeted 45 hours (15 hours per week) for this semester. The other 15 hours were spend on my research project.

Looking at the hours I felt that I have had a productive sprint, though I am only using a fraction of my budgeted hours. The rest of the hours were used for other academics commitments. However, I managed to further integrate the group to get the daemon and GUI plugins retrofitted and implemented, respectively. Besides that, though being the main contact of the group, I still get to do some coding to help the group further, and lend my assistance to the other subgroup on the integration preliminaries.

Rooms for Improvement

- Better time estimation needed for paper work and small coding parts. Previously, the budgeted time was huge as there were a lot of things needed to be done. However, it became a habit to give huge budget, and for this sprint, where the main aim is to prepare for the finishing of the project, only required minimal budget. I should give more time for paper work for this sprint.

Individual Milestone 6 Report

Executive Summary

I had proposed to produce a daemon feature in the form of a plugin to relay the status information from the daemon to the web-based GUI of the Earth application. The motivation for such feature arose from the fact the existing implementation of the GUI provides very little clue to the user about the operation of the daemon. This lack of information become even more crucial if the daemon and the GUI are running on different physical machines, in which case, erroneous assumptions about the daemon operation could effectively limit the reliability of the Earth application. However, further investigation into the actual implementation revealed that such an elaborate plugin approach would introduce unnecessary redesign and complexity to the codebase, all in return for the trivial benefits of providing daemon status information through the web-based GUI. Instead, a more simplistic approach of attaining the same benefit was pursued and implemented in this Milestone sprint. The approach uses the underlying (Linux) file system to relay the relevant daemon status information to the GUI while still maintaining the design separation between the GUI and the daemon.

Tasks and Activities Assigned

- Design and Implement Display of Daemon Status Information on Web-Based GUI.
  - Description: This task involved capturing the daemon status information and displaying it on the web-based GUI in real-time.
  - Idea/Solution: N/A
  - Affected Files:
    - app/controllers/servers_controller.rb
    - app/helpers/servers_helper.rb
    - app/models/earth/server.rb
    - app/views/servers/configure.haml
    - script/earthd
  - Git commits: `(ssurfer/earth.git) 563eb803cfbb8ffb687c86b4c473b718a03dffbd`
  - Estimated time taken (planned): 70 hours
  - Estimated time taken (actual): 80 hours

Resource Contributions

At the beginning of this Milestone, some time was spent identifying which particular feature or aspect of the application to be transformed into a plugin-enabled feature. Aside from deciding on the granularity levels of the plugins, some considerations about the relevancy and appropriateness of the plugins to the primary purpose of the Earth application also had to be made. This part of the task initially appeared to be straight-forward and subsequent efforts were made to quickly transform the implementation of some existing features into the relevant plugin forms. Unfortunately, the retrofitting of the daemon to incorporate plugins using extension points had not been fully completed at this point. As a result, the plugin conversion task had to be put on hold until the relevant plugin framework codes were committed.

In the meantime, the development focus was altered slightly to accommodate these delays and one such change involves the relaying of data between the daemon and the web-based GUI. Presently, the only existing mechanism that is able to facilitate any communication between the daemon and the GUI is the underlying PostgreSQL database. The daemon updates the database based on the changes in the monitored file system and the Rails application retrieves this information from the database for further processing and subsequent display on the browser. This works flawlessly for the file monitoring operations of the daemon, with the GUI passively updating the results when a web request from the application user is received. Unfortunately, this passivity of information relay may not be adequate or suit certain types of operation such as the real-time monitoring of the daemon status. A more direct mean of communication between the daemon and the GUI is more suited for such requirement.

To this end, a more simplistic solution was implemented which involved the periodic updates of a newly created 'daemon status' log file by the daemon. The corresponding GUI front end then simply retrieves the daemon status information from the said log file and displays it on the browser. Alternately, this approach could also be satisfied using the database instead of the log file. However, the approach employing the log file was adopted during this Milestone to quickly complete the proof-of-concept stage in the investigation of this particular design approach and eliminate the inherent complexities that normally arise with the alteration of the database schema. With the successful implementation and demonstration of the proof-of-concept stage results, the feature can then be implemented using the database instead of the log file. This change is necessary to ensure that this particular application feature can operate in a range of situation such as when the daemon and the web-based GUI are running on two different machines. This later expansion to a database-based implementation of the daemon status information update on the GUI is planned for the next development sprint.

Rooms for Improvement

The allocated time of 70 hours which had been initially earmarked for this particular task was in hindsight a bit optimistic. During the planning phase for the current development sprint, it was wrongly assumed that there was some simple way that the daemon could directly relay its status information to the GUI, which would not necessarily involve an intermediate facility such as the log file or the database. The subsequent investigation which revealed that no such simple provision existed also indicated that the incorporation of such feature through direct code manipulation or the creation of an appropriate plugin would take more time than budgeted. This experience clearly highlighted the costly and adverse effect of design decisions based on wrong assumptions on the schedule of software development projects.

– – ––