Assignment 9: Individual Requirements Analysis

Software Engineering

Griffin Schulte

04 April 2020

1. Introduction

1.1 Purpose

The purpose of this document is to specify the system requirements of Augur, a software developed by the CHAOSS community, a Linux Foundation Projects group. CHAOSS is a project whose goal is to create analytics and metrics to help define the health of open source communities. This is accomplished by the development of metrics, methodologies, and software that express various project's health and sustainability. This is important because open source contributors want to know where their work can make an impact and what projects, or areas of a project are worth their effort.

Source: https://chaoss.community/about/

2. Software Product Overview

2.1 Augur

Augur is a software project whose goal is to make sense of data in order to determine the health and sustainability of an open source project. This is accomplished using four human centered data science strategies:

1. Enable Comparisons: Allow people to compare projects with others. This is not a ranking system, but simply a method to help drive analysis.
2. Make time a fundamental dimension in all metrics from the start: "Point in time scores" allow for a comparison of historical health and an estimated trajectory.
3. All data driving visualizations should be downloadable as .csv or some other data exchange format: Transparent data behind a visual is useful and important for users.
4. Make all visualizations downloadable as svg's: This allows for users to input data into reports to gain more resources to continue analysis.

Source: https://oss-augur.readthedocs.io/en/master/getting-started/what-is-augur.html

3. System Use

3.1 Actor Survey

Owner

The owner is responsible for maintaining the administrator(s) and employees of Augur and evaluating the overall health of the software and its workers. This actor will interact with the system by generating reports which will help them identify any resources that may need to be expanded, added, or removed from the system. The owner is also responsible for determining funding for the project and its workers.

System Features:

- Generate / View reports
- Enter / Manage worker information
- Enter funding information

Administrator

The administrator is responsible for reviewing potential changes to the software, pushing changes to the software, assigning employees to specific jobs, and requesting resources from the owner. This actor will maintain more specific variables within the system compared to the owner and work more closely with the employees to maintain the health of the software.

System Features:

- View / Make changes to the software
- Accept / Decline changes to the software
- Assign employees specific jobs
- Request resources from the owner
- Maintain health of the system on a more detailed level

Employee

The employee is responsible for working on and completing their assigned task. This actor will submit work for review by the administrator and fix any errors or missing requirements that may be found within their work.

System Features:

- View / Work on their assigned task
- Submit work for review

User

The user is responsible for entering in information that would help the software identify open source projects and/or communities that may be relevant to them. This actor can then use that information to identify projects to contribute to.

System Features:

- Enter information relevant to their experience
- Enter information relevant to their interests
- View potential projects or communities

4. System Requirements

4.1 Use Cases

Use Case Name	**Review and Accept/Decline Changes**
System	Augur
Actors	Employee, Administrator
Brief Description	This use case explains how an administrator will access, review, and accept or decline changes submitted by an employee.
Basic Flow of Events	Basic flow begins when an employee completes potential changes and submits them to the administrators for review. The administrator views the submitted changes from the employee by viewing new pull requests from within the administrator/employee interface: 1. The system displayed a new pull request to be reviewed. 2. The administrator reviews the pull request. 3. If the changes are complete without error, the administrator can choose to accept the changes and run an update for users or decline the request and send it back for more work.

Use Case Name	**Find Project of Interest**
System	Augur
Actors	User
Brief Description	This use case explains how a user will input their information and search for open source projects or communities that may be of interest to them and impacted by the users work.
Basic Flow of Events	Basic flow begins when a user opens the Augur software and inputs their information relevant to their type of work and interests: 1. The user opens Augur. 2. The opens the search panel and inputs information relevant to their type of work and interests. 3. Augur returns a list of open source projects/communities that align with the user's inputs. 4. The user can go through and view the results to find projects of interest to them.

4.2 System Functional Specifications

Augur User Interface

Search Module

AUI-1: User login

AUI-2: User input information relevant to interests

AUI-3: User search for results based on input

AUI-4: User view results

Settings Module

SM-1: User input information relevant to interests

SM-2: User saves information as preset search option

SM-3: User creates multiple search presets

Help Module

HM-1: User searches for help on specific topic or feature

HM-2: User downloads manual

Augur Administrator/Employee Interface

System Management

SM-1: Administrator configures system parameters

SM-2: Administrators creates and configures employee accounts

SM-3: Administrator configures employee permissions

SM-4: Administrator manages report parameters

Job Scheduling

JS-1: Administrator manages tasks

JS-2: Administrator assigns tasks to employees

Job Review

JR-1: Employee views jobs

JR-2: Employee accepts assigned task

JR-3: Employee submits completion of task for review

4.3 Non-Functional Requirements

4.3.1 Usability

NFR-1: Augur's interface is easy to navigate and easy to find help on if needed from within the help page

NFR-2: All users will be able to become familiar to an advanced degree with the manual provided on the help page

NFR-3: Users will need to be familiar with the MAC OS, Windows, or Linux operating system in order to successfully use Augur

4.3.2 Reliability

NFR-4: Updates – System updates will only be performed during the lowest traffic hours of the day, typically during late night or early morning hours. Updates will only periodically be performed and should not inhibit regular use.

NFR-5: Bugs are estimated to only be present per many thousands of lines of code. Given a bug arises, an easy report feature is available that allows for submission of the problem.

4.3.3 Performance

NFR-6: Augur is built to be fast and responsive for most all machines running Windows, Linux, or MAC OS. Given that this fails, a report feature will allow for easy analysis on potential problems causing the slow responsiveness.

NFR-7: Augur shall support up to 10,000 simultaneous users and will be prepared to expand server space given an influx of traffic.

NFR-8: A typical search should not take more than 5 seconds to return results relevant to the users input data.

5. Design Constraints

The system should be able to run on previous versions and should not be required to be updated in order for functional for a user (1). The system is required to run on a machine which uses Windows, Linux, or MAC OS operating systems (2). Augur is a Flask web application, Python library, and REST server and cannot be ported to be ran in any other configuration (3). The system will not need to be designed to scale (4). The system will run as downloadable executable and cannot be run as a web application via a browser interface (5).

6. Purchased Components

CHAOSS has already purchased licenses for its software including Augur. The production-version of the software will not require new hardware or software for it to function as intended. CHAOSS will purchase new licenses as needed given there are not enough to supply for those working on the software. CHAOSS will also purchase more licenses as needed upon expiration or upon the addition of added software or resources.

7. Interfaces

The primary interface of Augur is an executable that runs on the OS (Windows, Linux, MAC OS) of your system. Augur is not available as a web application. Augur requires internet access to function as intended.

7.1 User Interface

The user interface will become available after successful downloading and installation of the executable. The user interface will only be available after the user has been properly authenticated via a login page upon execution. The user interface will consist of various user inputs that define exactly what the user is looking for within their search. The user interface will also feature a settings page and help page for creating preset searches and helping the user navigate the interface and become familiar with the software.