Software Requirements Specification

Project Name: Assignment 9 - requirements Analysis Process

Jeffrey Kerley 14310236

Date	**Revision**	**Description**	**Author**
4/6/2020	1.0	Initial Version	Jeffrey Kerley

Contents:

1. Introduction

1.1 Purpose

This Requirement Analysis is designed to show how Chaoss and other similar metric calculating softwares operate. With all of the different pieces of software, groups, projects, and systems, it can be hard to figure out which ones will be viable in 10 or 20 years. This software is designed to track and make predictions about open source software that could fuel the future.

1.2 Scope

Chaoss is meant for open source projects that can range from full teams, one person, and large scale companies. This is a data crunching software so companies that can understand it, and programmers/Computer Scientists, can use this data to enforce which technologies will be long lasting. Startups and people looking to be inspired can use this for investing and work.

The phases of making this software succeed is based around internal api usage and figuring out what data is useful. First setting up a good environment for pulling data is essential. Next, working out how we want to display this data and organize it. Once these are solved the actual data can be found and measured. Finally, working around different bugs and data that is uncommon or hard to find a metric for will be delivered.

1.3 References

Using Chaoss and community posts this software hopes to provide the metrics people are looking for and in an accessible way.

2. Software Product Overview

2.1 System Scope

This system will be available to anyone and you simply view the website to view the metrics. You can see different metrics and input which metrics you want to track. This can be updated automatically or in different ways. The ability to change what metrics you want to track, and how, is why Chaoss can be so strong.If you want to track commits, modification amounts, etc you can enable this.

2.2 System Architecture

The features in this app respond to these various additions:

Feature1: Manage customer metrics they want to track.

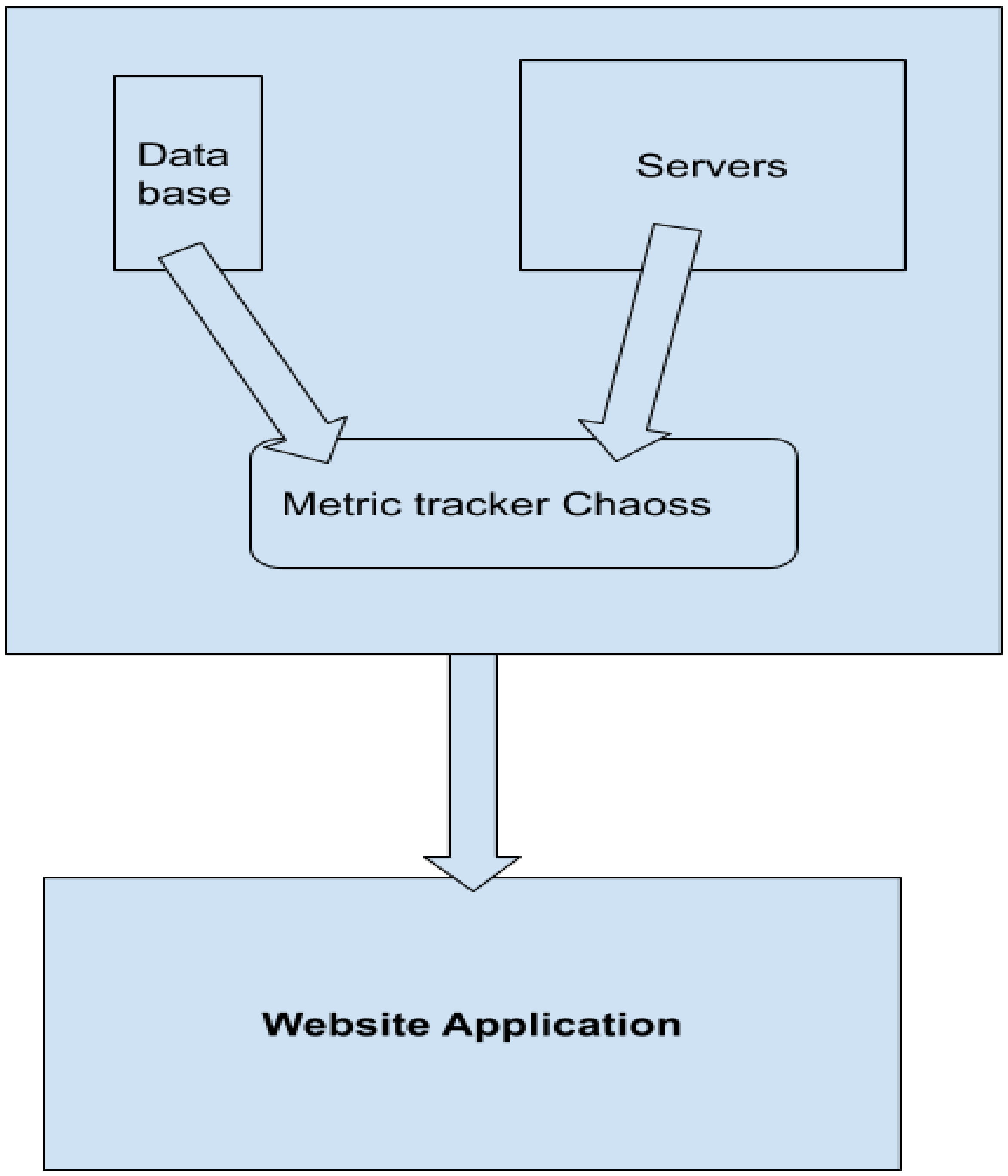
Feature2: Provide and allow additions for metrics to track.

Feature3: Store and save metrics tracked for customers.

Feature4: Display using graphs and text percentages the metrics requested by the user.

Feature5: Allow users a login to easily view metrics each session.

Feature6: Have a clear classification of each metric and job title for viewing and metric changes.


Data base
Servers
Metric tracker Chaoss
Website Application

3. System Use

3.1 Actor Survey

Owner/Proprietor/CEO

The person who will be setting up the original connection with Chaoss. They will not interact with it the majority, unless for major changes that need to be made to the company, or open source project. They have the same role as the Cs/programer who can view the data.

Programmer/CS

The main user of this system, they will be viewing this on the daily for measuring metrics and translating the data to their own resources. This actor is reasonable for upkeeping the metrics, and changes with it. They will be using most of the features presenting earlier. And with each person within this group they will be the workers who will give ideas about what changes need to be made about the software.

Metric Creator

This actor will not use any of the features above, but rather come up with the features and/or metrics to utilize in the software. This is not someone who works on Chaoss, but rather a user who is providing feedback and support for new metrics.

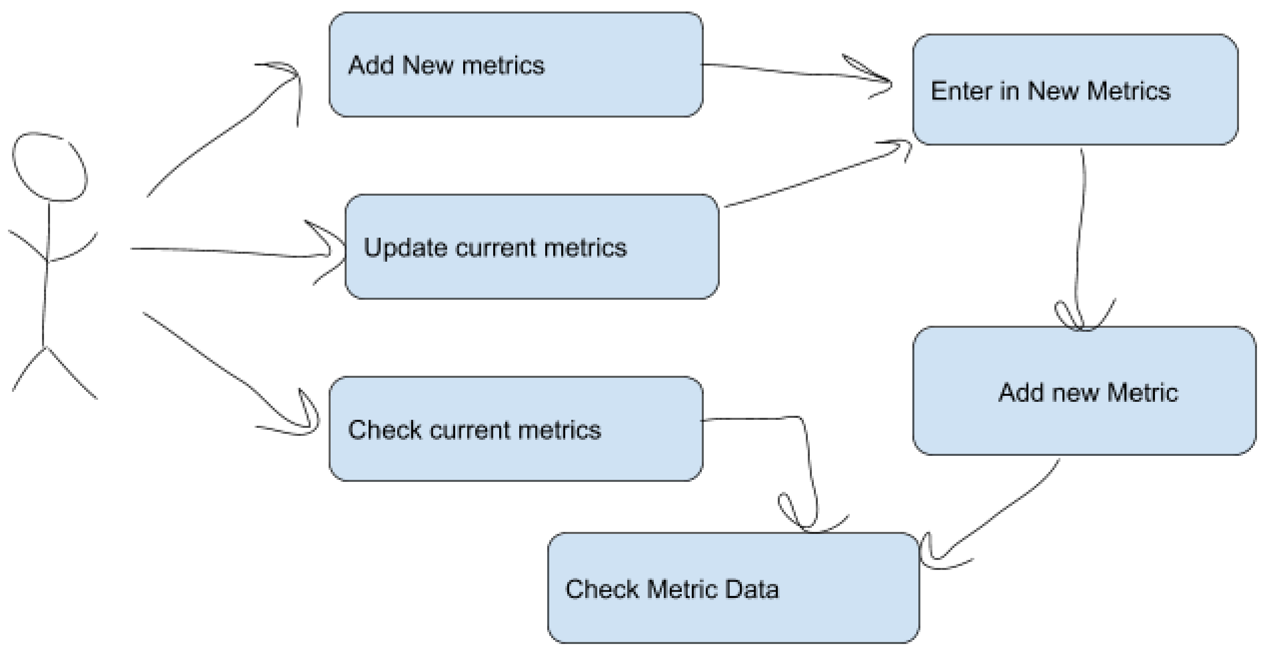
Clients

Anyone who is related to this software will be adding to these metrics, so user interaction on providing feedback and data is vital. These clients of each other software will need the resources provided by clients so the features of data storing are irreplaceable here.

4. System Requirements

4.1 Use Cases

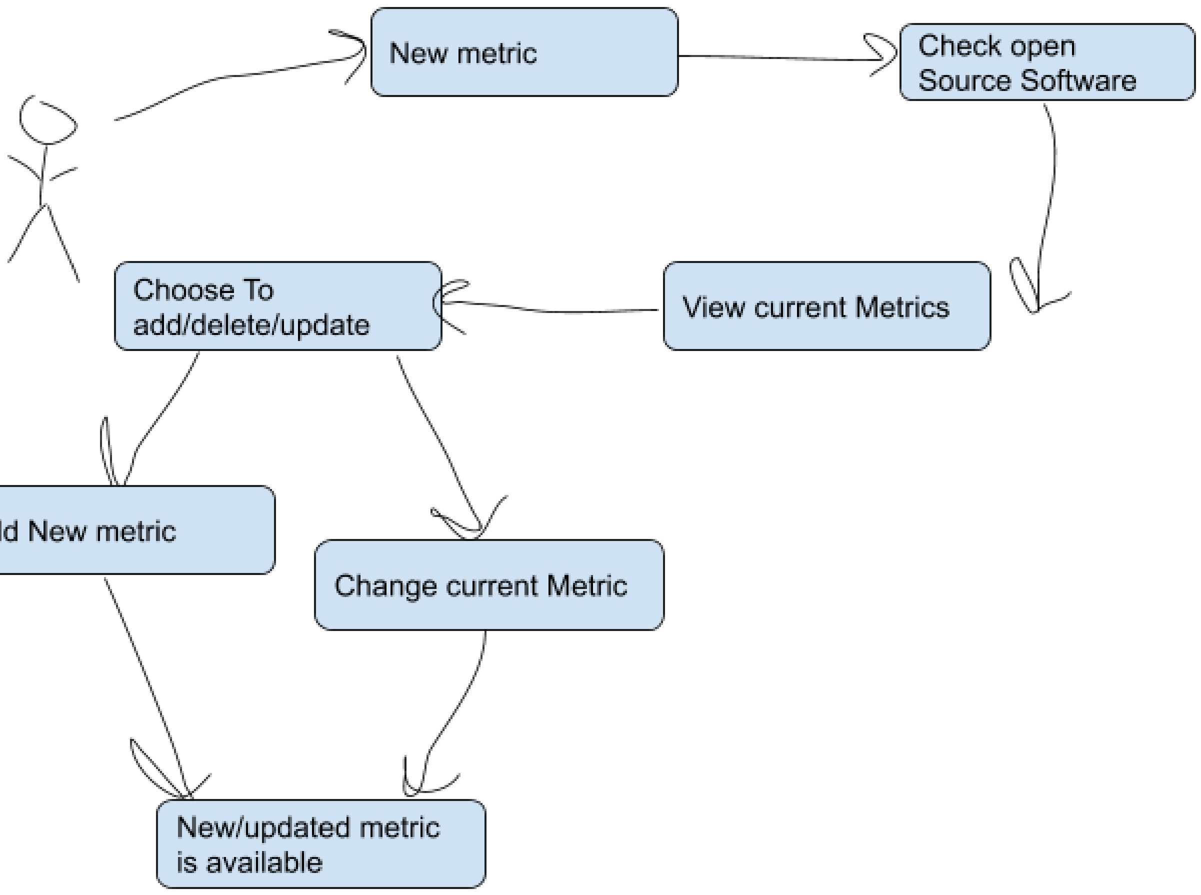
Use Case 1: Collecting Data



Use Case Name	Chaoss Data Collection
System	**Chaoss**
Actors	**Programmer/Cs, Owner, clients**

Flow of Events	**One of the above actors has their own account or company account. They then login to the website. This begins the possibility of checking/ adding/updating metrics.** 1. **Add new metric** 2. **Check new metric**
Alternate Flow of Events	**When the user needs to update current metrics** 1. **Find current metrics** 2. **Update new metrics** 3. **View the new metrics**
Requirements	**People operating this process should have knowledge of how metrics work, and a good understanding of the software metrics they are relating to.**
Conditions	**Employee,owner, client must have account and be logged in or have the access to the metrics.**
Points	**Search metrics, search software**

Use Case 2: Adding a new metric to Software tracking


New metric
Check open Source Software
View current Metrics
Choose To add/delete/update
Add New metric
Change current Metric
New/updated metric is available

Use Case Name	Chaoss Metric Changes
System	**Chaoss**
Actors	**Metric Creator**
Flow of Events	**Metric creator adds their own metric, updates a current metric.** **1. They add a new one.** **2. They then assemble it** **3. It is not viewable**

Alternate Flow of Events	**Metric Creator wants to change a current metric.** **1. Find current metric.** **2. Change current metric.** **3. deletion/addition.** **4. New metric is available.**
Requirements	**People operating this process should have knowledge of how metrics work, and a good understanding of the software metrics they are relating to.**
Conditions	**Must be a Metric Creator who has privileges to apply these new updates.**
Points	**Search metrics, search software, add Metrics**

4.2 System Functional Specification

Operating Side

Login and authentication process for each user.

Menu for each possibility.

View, print, share, and download each metric.

Add new metrics, update, delete.

Metric Information page.

Contact info.

Change user information.

Enter dev Support text box.

System Management

User accounts with lower rights.

Configure mobile website users.

Administration control.

Metric creator control.

Metric Reporting

Allow ability to view public metrics.

Create viewable of graph images of competing metrics.

Pages for metric lists and forums.

Updating has auto complete ideas and formulas.

Storage

Clients, Cs, and owners will have access to a certain amount of storage.

Different paid options available.

Hosted on main server into a database.

Billing

Ability to bill for premium service.

Add account for profits.

Clients will have different levels of security for payment methods.

4.3 Non-Functional Requirements

Use

The software is intuitive and will provide a support page.

Will have to have a basic understanding of how to interact with a website.

Everyday Use

Will have good server use during the day.

Exceptions to each user can be made and changed to fit their time.

Normal working hours.

Performance

The website will run on any modern computer and browser.

Each software metric can support well above the data load of every company/open source.

Support

User will receive good support for any upgrade and metric they want Tracked.

Any updates to the current website will integrate any client over smoothly.

5. Design Constraints

1. **Software Budget**
    a. To achieve the goal of making software metrics understandable and find good software, a good budget to create this will be the biggest limiting factor here.
2. **Metric Importance**
    a. Getting good metric data when so much is prevalent will be hard to achieve and will contribute to the success of this website.
3. **Data Available**
    a. What each software company/ open source using this software has to do is provide good data to track. Without companies providing data this software can only access certain metric to track. With good data the sea of metric will end up being pointless.
4. **Client Openness**
    a. On the end user side, Clients who are operating the open source software, need to disclose their own metrics if they are not being tracked. Although unlawful data tracking happens, users giving access to certain data is required for good metrics.
5. **Understanding Metric Data**

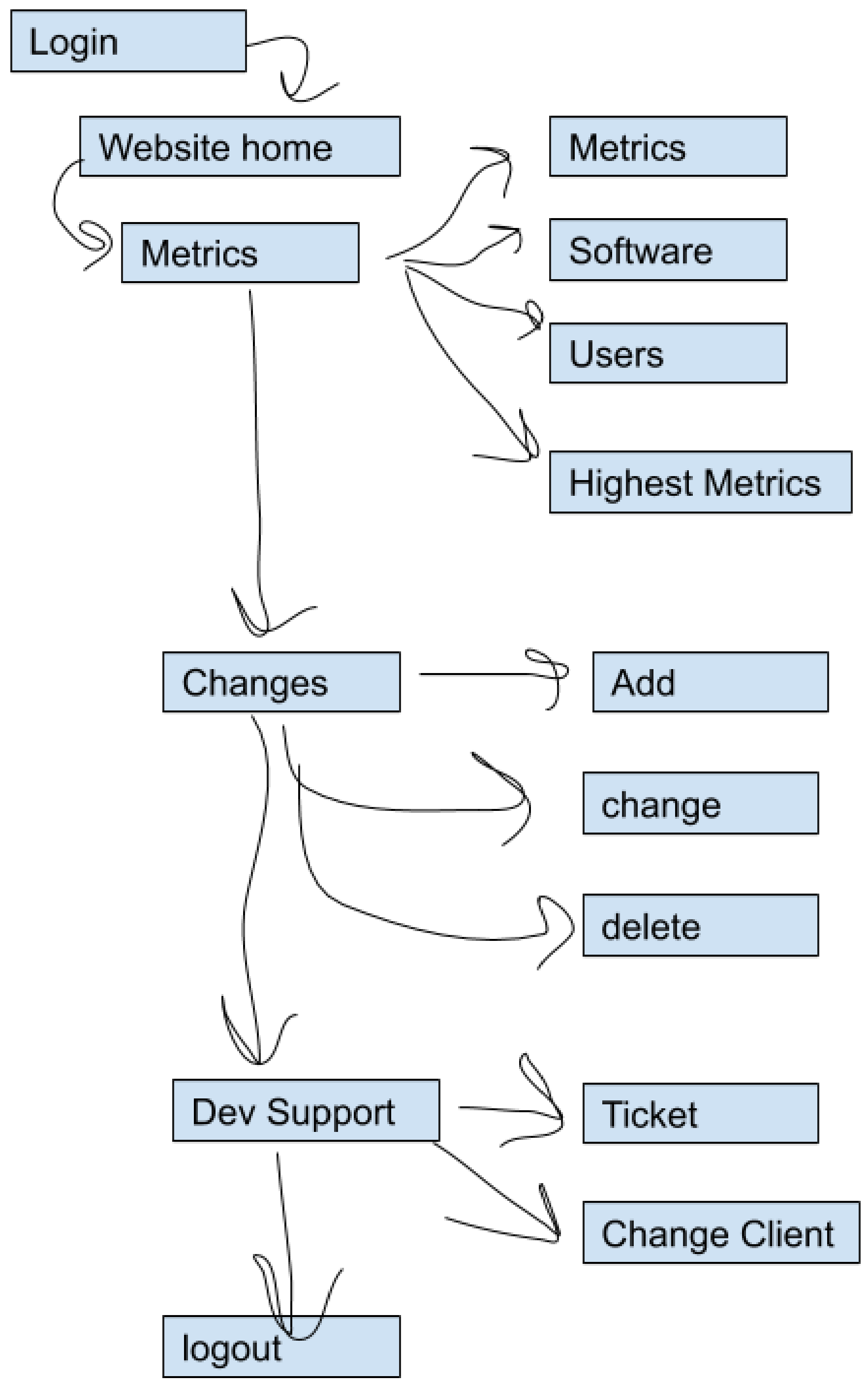
a. Translating the current metrics into viewable data tables is the heart of this software's problem. A good answer to what these metrics need to look like is hard to answer and will require a lot of work to achieve.

6. Purchased Components

All Software will be provided by open source projects or in house development. Any server side products will be licensed through AWS using basic EC2 base: 100$ a year.

7. Interfaces

Metric Adder Interface


Login
Website home
Metrics
Metrics
Software
Users
Highest Metrics
Changes
Add
change
delete
Dev Support
Ticket
Change Client
logout