**Project Specification Document**

IBM

Scott Rotvold

Madeline Gordon

Brandon Ebersohl

Adam Murray

Table of Contents

Section Page

[Table of Contents ii](#_Toc411876997)

[1. Project Vision and Objectives 1](#_Toc411876998)

[1.1 Project Scope and Vision 1](#_Toc411876999)

[1.2 Project Goals and Objectives 1](#_Toc411877000)

[2. Project Planning 2](#_Toc411877001)

[2.1 Project Lifecycle 2](#_Toc411877002)

[2.2 Project Setup 2](#_Toc411877003)

[2.3 Stakeholders 2](#_Toc411877004)

[2.4 Project Resources 3](#_Toc411877005)

[2.5 Assumptions 3](#_Toc411877006)

[3. Project Tracking 3](#_Toc411877007)

[3.1 Tracking 3](#_Toc411877008)

[3.2 Communication Plan 4](#_Toc411877009)

[Regularly Scheduled Meetings 4](#_Toc411877010)

[Information To Be Shared Within Our Group 4](#_Toc411877011)

[Information To Be Provided To Other Groups 4](#_Toc411877012)

[Information Needed From Other Groups 4](#_Toc411877013)

[3.3 Deliverables 4](#_Toc411877014)

[3.4 Project Metrics 5](#_Toc411877015)

[4. Requirements (User Stories) 6](#_Toc411877016)

[4.1 Overall Description 6](#_Toc411877017)

[4.2 Users and Roles 7](#_Toc411877018)

[4.3 Use Case Diagrams 7](#_Toc411877019)

[4.4 User Stories (Requirements) 8](#_Toc411877020)

[4.5 User Story Acceptance Criteria 10](#_Toc411877021)

[4.6 Constraints and Limitations 12](#_Toc411877022)

[5. Design 13](#_Toc411877023)

[5.1 Introduction 13](#_Toc411877024)

[5.2 Scope 13](#_Toc411877025)

[5.3 High-Level Component Design 13](#_Toc411877026)

[5.4 Activity Diagrams 13](#_Toc411877027)

[5.6 Data Flow 14](#_Toc411877028)

[5.7 Alternative Designs and Design Rationale 14](#_Toc411877029)

[6. User Interface 15](#_Toc411877030)

[6.1 UI Description 15](#_Toc411877031)

[6.2 UI Mockup 15](#_Toc411877032)

[7. Test Plan 16](#_Toc411877033)

[7.1 Test Plan Description 16](#_Toc411877034)

[7.2 Testing Tools 16](#_Toc411877035)

[7.3 Test Data 16](#_Toc411877036)

[7.4 Test Types and Frequency 16](#_Toc411877037)

[7.5 Test Coverage 16](#_Toc411877038)

[7.6 Test Location 17](#_Toc411877039)

[8. Project Closure 18](#_Toc411877040)

[8.1 Goals / Vision 18](#_Toc411877041)

[8.2 Delivered Solution 18](#_Toc411877042)

[8.3 Remaining Work 18](#_Toc411877043)

[9. Deliverables 19](#_Toc411877044)

[9.1 Study Results 19](#_Toc411877045)

[9.2 Requirements and Design Documents 19](#_Toc411877046)

[9.3 Code 19](#_Toc411877047)

[9.4 Tests and Test Results 19](#_Toc411877048)

[9.5 Build Process Documents 19](#_Toc411877049)

[9.6 Install Process Documents 19](#_Toc411877050)

[9.7 Administrator’s and/or User’s Manual 19](#_Toc411877051)

[9.8 Postmortem Document 19](#_Toc411877052)

[9.9 Final Report 19](#_Toc411877053)

[10. Definitions and Acronyms 20](#_Toc411877054)

1. Project Vision and Objectives

1.1 Project Scope and Vision

The purpose of this project is to create a real-time display of metrics to be viewed in a dashboard via web browser. First, we will research into different technologies needed for the process of collecting and storing metrics. We will store these metrics in a graphing database and gather them using our code written in Python. We hope to create a user-friendly and customizable portal for displaying these metrics that will assist in monitoring and evaluating the health and availability of the system’s infrastructure. The main issue being addressed involves the large size of IBM’s environment and the need for a centralized dashboard to display their metrics. This project will hopefully create for them an organized and structured way to manage their large infrastructure.

1.2 Project Goals and Objectives

|  |  |
| --- | --- |
| **#** | **Goal or Objective** |
| 1 | Create a Jenkins test environment with sample data |
| 2 | Push data from Jenkins servers to graphite-database |
| 3 | Display data with graphite’s default web configuration |
| 4 | Create a user-friendly dashboard that allows users to view the gathered metrics in either graph form or other representations using Grafana |
| 5 | Create a flexible framework that can be deployed within IBM’s Cloud Solution’s development infrastructure using Ansible for automated deployment |

2. Project Planning

2.1 Project Lifecycle

The team will use an agile approach. Our sponsors have outlined our project with three different phases. Our team will research the different technologies that we could use for this project to create a development plan that will be best for our project. We will work in iterations while continuously communicating with each other and with our sponsors.

2.2 Project Setup

|  |  |
| --- | --- |
| **#** | **Decision Description** |
| 1 | GitHub, Windows & Linux, Jenkins, Graphite, Grafana, Python, Ansible |
| 2 | Standards that must be followed: Default capstone coding standards |
| 3 | User Access: Open Source, testing with “dummy data” |
| 4 | Server setup: Virtual Machine running Ubuntu 14.10 |

2.3 Stakeholders

|  |  |
| --- | --- |
| **Stakeholder** | **Role** |
| Jake Morlock | Sponsor, Lead |
| Mathew Odden | Sponsor |
| Adam Reznechek | Sponsor |
| Dean Knudson | Instructor |
| Scott Rotvold | Team member |
| Adam Murray | Team member |
| Madeline Gordon | Team member |
| Brandon Ebersohl | Team member |

2.4 Project Resources

|  |  |  |  |
| --- | --- | --- | --- |
| **Resource** | **Resource Description** | **Quantity** | **Total** |
| Virtual Machine | Virtual machine set up by team member Adam Murray | 1 | $0.00 |
| Capstone Team | Our team of students who will be the primary developers of the project. | 4 | $ 0.00 |
| Jenkins | Open-Source software for continuous integration | 4 | 0.00 |
| Personal Computers | Our team will use our personal computers to access Jenkins and write the code for our project | 4 | $0.00 |

2.5 Assumptions

|  |  |
| --- | --- |
| **#** | **Assumption** |
| A1 | We will have all of the preliminary research and setup completed by February 7th |
| A2 | We will have Jenkins installed on our local machines by February 9th |
| A3 | We will present our work for mid-term presentations on March 10th |
| A4 | We will complete our dashboard model by April 10th and begin testing for bugs |
| A5 | We will set up our project for automated deployment by April 20th |
| A6 | We will have our project ready for a final presentation on May 5th |

3. Project Tracking

3.1 Tracking

|  |  |  |
| --- | --- | --- |
| **Information** | **Description** | **Link** |
| Code Storage | e. g. Project code will be stored in a git-hub account. | https://github.com/the0ldknighte/  IBM\_Capstone\_Project |
| Bug Tracking | e. g. Bug tracking will be done with Trac. | https://csprojects.cs.ndsu.nodak.edu/csci445/  2015/spring/csci445s15ibm/trac |
| Project Schedule | e. g. The project schedule will be stored in the git repository. | https://github.com/the0ldknighte/  IBM\_Capstone\_Project |
| Continuous Integration | e. g. Continuous integration will be done with Jenkins. |  |
| Regression Testing | e. g. Regression testing will use JUnit unit tests and Jenkins. |  |

3.2 Communication Plan

Regularly Scheduled Meetings

|  |  |  |
| --- | --- | --- |
| Meeting Type | Frequency/Schedule | Who Attends |
| Conference Call | Wednesday at 8 a.m. (Weekly) | Project team and mentors |
| Team Meeting | Monday at 8 p.m. (Weekly) | Project team |
| Short Meeting | Weekly in class | Project team |

Information To Be Shared Within Our Group

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Project team | Task assignments | Weekly in class | Team meetings, listing in MS Project file. |
|  |  |  |  |

Information To Be Provided To Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Sponsor and mentor | Final report | At completion of project | Req./Design docs., code, Power Point presentation |
| Sponsor and mentor | Weekly report | Weekly | Email and Git site access |

Information Needed From Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Sponsor and mentor | Requirement changes | Start of each sprint | Conference call or meeting with sponsor and mentor. |

3.3 Deliverables

|  |  |  |  |
| --- | --- | --- | --- |
| # | Deliverable | Delivery Time | |
|  |  | Interim | Final |
| 1 | Prototype Code |  |  |
| 2 | “Dummy Data” |  |  |
| 3 | Deployable product |  |  |
| 4 | Project Documentation |  |  |

3.4 Project Metrics

|  |  |  |
| --- | --- | --- |
| Metric | Frequency | Location |
| Estimated User Story Points | Per User Story at the start of each sprint | MS Project Plan |
| Estimated User Story Points | Per Sprint at the start of each sprint | MS Project Plan |
| Actual User Story Points | Per Sprint at the end of each sprint | MS Project Plan |

4. Requirements (User Stories)

4.1 Overall Description

The primary focus of this project is the client (IBM) wishes to gather important metrics on how their Jenkins setup is working for their Open Stack project. Jenkins is a system primarily used for continuous integration of updates into a large software project. Since Open Stack is a large and open source project, there are large quantities of source updates submitted every day, which have areas that could be tracked using metrics. These metrics may include such traceable aspects as time to commit, time to merge updates, type of function and many others. Jenkins already has ways to find these metrics, however, IBM wants to track them in real time and store data that is displayed in a user friendly way using open source graphing software such as graphite.

The graphite system provides a detailed and easy to understand layout of those metrics so that the users at IBM can pinpoint where they can improve the continuous integration process. The graphite system works using a design pattern as shown below.



**Image taken from the Graphite website:** [**http://graphite.readthedocs.org/en/latest/faq.html**](http://graphite.readthedocs.org/en/latest/faq.html)

This metric tracking and displaying system will then be hosted on an apache web server which will display the required metrics as well as archive old metrics to create more details historical records.

There is a possibility and requested addition to make the project self-deployable and a one file install using a system called Ansible. Ansible will provide the user with an all-in-one setup package to provide an easy setup for the users. This will make the deployment much less of a hassle and will lower the required level of knowledge for the person administering the metric system to their Jenkins database.

4.2 Users and Roles

|  |  |
| --- | --- |
| **User** | **Description** |
| User | A regular user who can view the data and see the graphical displays |
| Admin | A user who also has the ability to edit and update historical data as well as edit the display of data |

4.3 Use Case Diagrams

User and Admin Running System Use Case



Install Project Use Case



4.4 User Stories (Requirements)

**User Story Points Benchmark** – All team members agree that setting up a git repository for use in this system would be worth 5 points. All members have had previous experience with git.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Added** | **Description** | **Status** | **Story Points** |
| 100 | 1/29/15 | As a developer I need to know and understand the technologies I am using | **C** | **45** |
| 200 | 1/29/15 | As a developer I need source control to make the development process go smoothly. | **C** | **5** |
| 300 | 2/5/15 | As a developer I need a system that creates data sets to graph to give visual representation of the data. | **C** | **8** |
| 400 | 2/5/15 | As a developer I need to formalize our web user interface to make the website user friendly. | **C** | **5** |
| 500 | 2/5/15 | As a developer I need a database so I can store the collected metrics to allow for record keeping | **C** | **3** |
| 600 | 2/5/15 | As a developer I want to have a system that can pull metrics and display them in graphs for small scale data to automate the process. | **C** | **21** |
| 700 | 2/10/15 | As a developer I want a system that has automatic deployment (Chef) to make setup easy for users. | **T** | **34** |
| 800 | 2/10/15 | As a developer I want to have a system that can pull metrics and display them in graphs for large scale data for automation purposes | **C** | **34** |
| 900 | 2/10/15 | As a user I want the metric graphs to update in real time to give the most relevant data. | **T** | **45** |
| 1000 | 2/10/15 | As an admin I want access to a webpage that allows me to edit historical data. | **T** | **13** |
| 1100 | 2/10/15 | As a user I want access to a webpage that displays stored metrics and historical data to give a complete story of the data. | **C** | **8** |
| 1200 | 2/10/15 | As a user I want an intuitive UI that allows seamless transition between graphs to give the best possible user experience. | **C** | **13** |
| 1300 | 2/10/15 | As a user I want to be able to edit the display of data to accommodate my needs | **C** | **15** |

4.5 User Story Acceptance Criteria

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Verification** |
| 110 | Research the various technologies involved in continuous integration | Reference mentors to make sure that our understanding is clear. |
| 210 | Setup Github | Make sure that all users can commit changes to Github |
| 310 | System will gather data from server(s) | Acceptance testing to guarantee data is correctly gathered |
| 320 | System will store data in database with timestamp | Acceptance test to ensure proper database design with appropriate time column |
| 410 | User interface will allow user to change graphs | User acceptance test to ensure UI functionality |
| 420 | User interface will allow user to change metrics being displayed | User test to ensure functionality |
| 510 | Ask NDSU to provide a server to host graphite and other required resources | Make sure that the server is accessible and usable. |
| 610 | Small-scale metrics will be pulled from database | Acceptance test to verify correct data is pulled |
| 620 | Metrics will be graphically displayed using Graphite | User test to ensure data is displayed correctly |
| 710 | Program will automatically be installed | User acceptance test to determine if program installs correctly |
| 810 | Implement the correct Jenkins plugins as well as Carbon | Show that data can be passed from a Jenkins server to the graphite server. |
| 910 | Data will automatically and continuously be transferred to database | Acceptance testing to ensure database receives proper data |
| 920 | Graphite will display database data in real time | User test to verify data is displayed properly in real time |
| 1110 | Admins will be designated accounts that have access to the historical editing webpage | Attempt to access historical page using user and admin accounts to ensure proper privileges are designated |
| 1210 | Take the generalized Graphite UI and make it easier to use. | Do functional UI testing to make sure it is user intuitive. |
| 1310 | Make the graphs that are displayed editable using their built in UI settings | Make sure that all users have access to move around what they need. |

4.6 Constraints and Limitations

|  |  |
| --- | --- |
| **Constraint** | **ID** |
| Project will be all open source, no paid services can be used. | 700 |
| Testing will not be done directly upon Jenkins servers as acquired access is difficult externally. | 1000 |
| The group has little python experience so learning will be required. | 300 |

5. Design

5.1 Introduction

The major goal of this project is to take existing graphing, deployment, and metric logging technologies and repurposing those technologies to track the build metrics on a continuous integration system. These metrics provide a useful data set to find what part of the system may be creating problems such as bottlenecks, frequent errors and commonly submitted bug fixes. The technologies chosen to solve this are: Grafana to display the metrics shown, Ansible will be used to make the deployment easier, and finally Jenkins is the current platform used for continuous integration by IBM.

5.2 Scope

The primary scope of this project is to use existing technologies to create a dashboard. In addition to this, IBM has requested that once the primary scope of the project is obtained, a system to automatically deploy the graphing software should be created, which we have chosen Ansible to accomplish.

5.3 High-Level Component Design

The main components of this project are Graphite, Carbon, Whisper, Jenkins, and Grafana. Starting with Graphite it is the main monitoring tool. This will be explained by the individual components which are as follows. The first component which will be explained is Carbon. This is a listening service which passively listens for data. The second component is Whisper which is a database data structure. The purpose of Whisper is to allow many simultaneous reads and writes. This is important for allowing a large amount of data to be handled. The last aspect of our project is Jenkins which is a continuous integration service. This we will not directly be using, setting up automated tests, but instead we will be using metrics from Jenkins servers to feed into Graphite. The last portion of this project is Grafana. Grafana is the graph creation web application which we will be using to display the graphs on a web page.

5.4 Activity Diagrams

Due to the nature of our project we are fundamentally coupled to the user interfaces of the various technologies that we are using. That being said we will not be able to properly give documentation of the steps required to do actions until we have a working user interface.



5.6 Data Flow

The dataflow in the architecture will be based around the Jenkins instances pushing data. This data will then be interpreted by Carbon and written to whisper files. From here Grafana will read the data from the whisper files and display them on the website. This architecture will require us to have knowledge about how long data points should be kept. In addition we will have to interface with Jenkins, probably via plugins, to push this data into Carbon.

In the diagram below, metrics are first pushed into the carbon listener server using an existing graphite plugin in the Jenkins continuous integration server. Next, this data is turned into Whisper objects which are stored in a database on the Grafana server. Finally, those compiled metrics are them pushed to the Grafana web application and then the user may choose which metrics to graph and how to graph them.



5.7 Alternative Designs and Design Rationale

In order to allow automatic deployment of our system our group had two candidate automation technologies, Chef and Ansible. Chef is a configuration management tool that streamlines server configuration based on user written “recipes”. Ansible is a platform for configuring and managing computers by managing nodes over SSH. Although both technologies our viable for our solution we chose Ansible because it proved to have more capabilities for what we need to deploy our project.

8. Project Closure

8.1 Goals / Vision

Overall, our group was able to accomplish the goals set before us by IBM for our capstone project. We delivered a metric tracking system with a real time graphical user interface that is able to communicate with multiple continuous integration servers simultaneously. Through the use of the existing Graphing package Grafana, we were able to configure the baseline codebase to interact with the Jenkins continuous integration metrics in a way that is useful to the IBM team. Grafana is able to display data such as build time, where the build originated from, what was changed etc. This data can then be dynamically displayed in any way the user desires to have them displayed as Grafana allows custom graphing schemas to be saved by each individual user. Finally, an Ansible playbook or automatic deployment schema was created to make the process of installing and configuring the project as easy as possible for the user. Ansible will make the solution as close to a 1 click install solution as humanly possible making the time between install and useful uptime minimal.

8.2 Delivered Solution

Our example Grafana dashboard is located at:

<http://68.69.225.84:5601>

Here we have created sample graphs containing build metrics that we have gathered.

8.2 Remaining Work

The remaining work involves testing the capabilities of Grafana and graphite. This involves sending different types of data from different sources perhaps even with a different back-end database to test capabilities and data handling.

Another area to note, is that the newest version of Grafana has recently been released. There is work to be done testing our project in regards to this new version and its capabilities.

9. Deliverables

9.1 Requirements and Design Documents

https://github.com/the0ldknighte/IBM\_Capstone\_Project/tree/master/Documents

9.2 Code

https://github.com/the0ldknighte/IBM\_Capstone\_Project

9.3 Install/User Guide

https://github.com/the0ldknighte/IBM\_Capstone\_Project/tree/master/Documents

9.4 Postmortem Document

https://github.com/the0ldknighte/IBM\_Capstone\_Project/tree/master/Documents

9.5 Final Report

https://github.com/the0ldknighte/IBM\_Capstone\_Project/tree/master/Documents

10. Definitions and Acronyms

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Ansible | A service used for automated deployment. |
| Grafana | A python based graphing package built upon the Graphite metric graphing base package. |
| Carbon | Graphite’s passive listener service, similar to a socket listener. |
| Whisper | Data structure format used to push metrics to the GUI of Graphite/Grafana |
| Jenkins | The continuous integration service used to automate testing of projects developed by numerous developers at separate locations. |