PROJECT
METRICS COLLECTOR
DESIGN DOCUMENT

Team members

Akshara Boppidi

Pooja Jadhav Eshwarlal

Raja Ravi Teja Ponna

Sponsor

Anthony Giorgio

March 6, 2017

Marist College

Poughkeepsie, NY

Table of Contents

Document Revision History:

Date	Major Changes
02/20/2017	First Draft
02/25/2017	Audience, Objective, External Design
02/28/2017	Internal Design, Software Flow
03/03/2017	Document History, Globalization, Accessibility, Security
03/06/2017	Final Copy

Audience:

This document will address 3 main target audience:

- *User/Customer:* They can use metrics collector for extracting details about their system performance.
- *Developer:* They can implement or enhance metrics collector.
- *Tester:* They can review test specifications, defect reports and test results.

Introduction:

Metrics Collector Design Document provides a description of designing a software that gathers application metrics and statistics on a Linux operating system. In this document we are going to discuss about various software to collect, store and visualize the data generated by the server.

The collected metrics are recorded in a database to maintain a historical record of system activity. We will develop a desktop application that will display memory usage, CPU consumption, process runtime and system load in a neat graphical user interface.

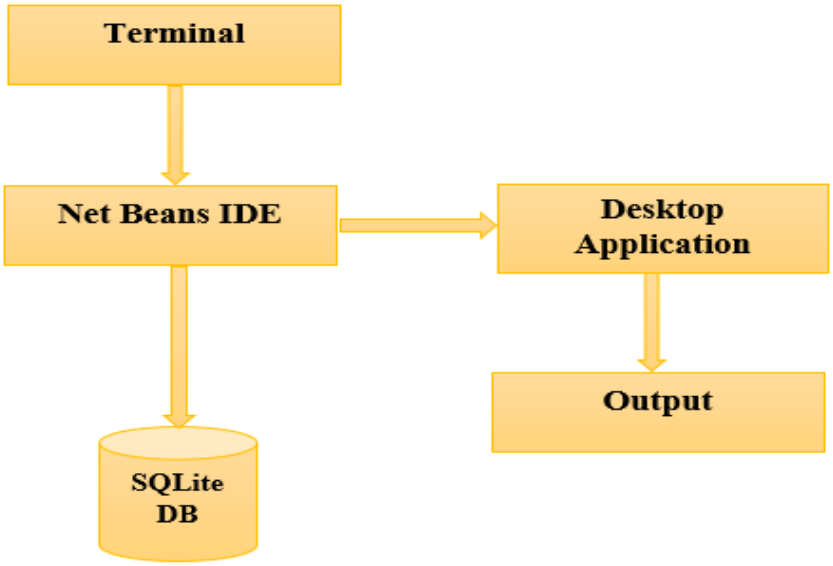
Objective:

Collecting metrics on a machine helps us to track and troubleshoot issues in the system configuration. This requires an understanding of what is to be built and how it is expected to build.

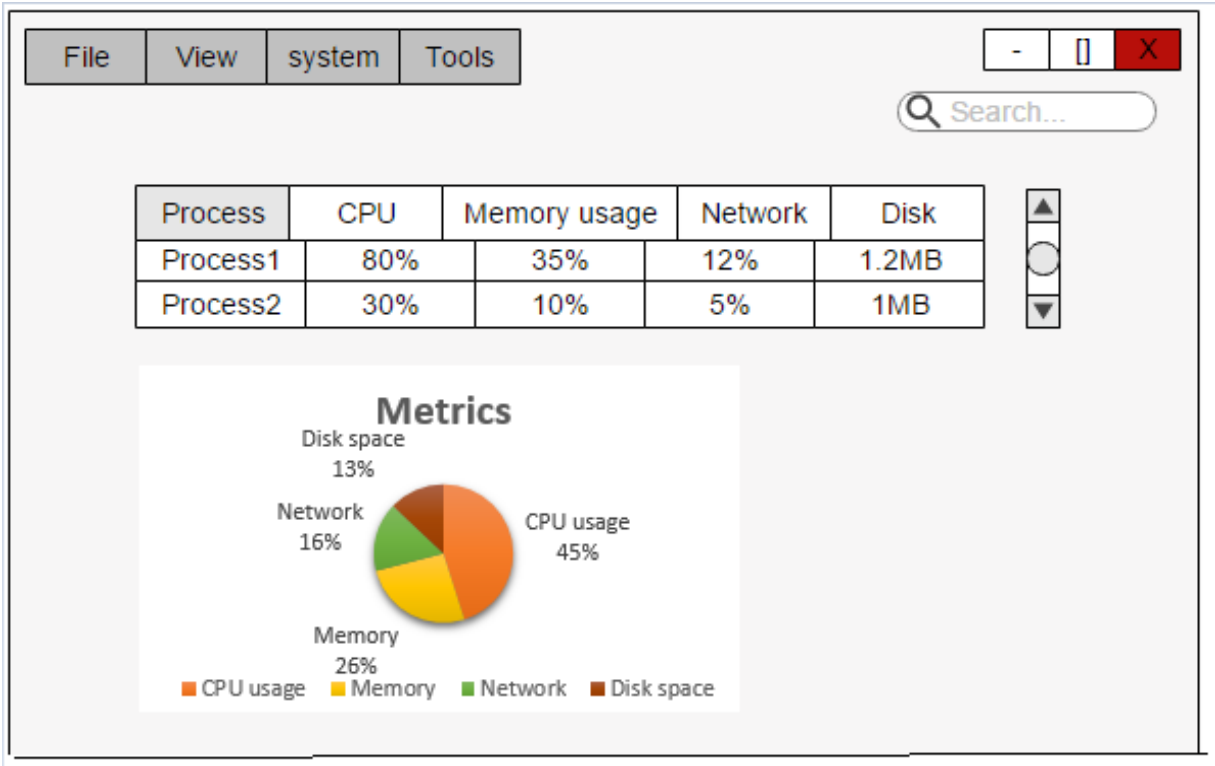
Goals:

- Implementing the database.
- Designing the Desktop application using Net Beans.
- Inserting data using java code.
- Enabling a JDBC driver (API) which makes Java application interact with the Database.
- Testing functionality of the project.

Approach:



External Design:



User can view statistics of the running processes and monitor the system performance.

Internal Design:

Development Standards:

- This project involves agile methodology which uses incremental and iterative development of the software.
- We will be using Java programming language.
- The metrics collector desktop application software is open-source and the code can be reused.

Hardware Resources:

- Operating system : Linux
- Physical machine : Windows
- User Interface : Desktop application

Development Environment:

- Programming language : Java
- Environment : NetBeans IDE
- Database : SQLite
- Build Automation Tool : Maven
- Source Code Repository : GitHub
- Compiler : javac

Software Flow:

- ***Initialization:*** We install all the software mentioned under hardware resources and development environment.
- ***Metrics collection:*** We will be using linux commands such as top, vmstat, lsof, netstat and iostat to collect metrics from the linux operating system.
- ***Data aggregation:*** CPU, memory and network stats will be displayed in the desktop application.
- ***Database storage:*** Collected stats will be stored in the SQLite database.
- ***UI data extraction:*** The data will be extracted from the SQLite database and displayed on the metrics collector which will be viewed by the end-user.
- ***Metrics presentation to user:*** The user can view the metrics in the desktop application.
- ***Termination:*** The application is terminated when the metrics are collected in the database.

Error Handling:

All applications have failures either during runtime or compilation and in order to identify these failures and the reason behind them, we record logs. These logs help the user to rectify what is happening at the any given moment.

Testability:

- Test Environment : Linux operating system.
- Test Scenarios : Testing and comparing between different processes.
- Regression Testing : To check the performance of the application step by step.
- Testing : System test will be performed to record outcomes.

Packaging:

The application can be packaged in NetBeans. It can be installed using NetBeans.

Security:

Any user can install the software and we do not require root permissions to run the metrics application.

Accessibility:

We will use large font and contrast textual items to make reading easier for the vision impaired individuals. Also clear and simple graphics can be beneficial.

Globalization:

The display messages and documentation are in English. There are no translations.

Risks and Dependencies:

- Lack of time.
- If we don't follow design implementation properly, there is a scope for project crash.

Supporting Material:

Glossary:

JDBC: Java Database Connectivity is an application programming interface (API) for java programming language, which defines how a user can access a database.

Maven: It is a build automation tool used for java projects. It is based on Project Object Model which is a XML file (an information file regarding the project and configuration details used).

SQLite: SQLite is a relational database management system which can be used in UNIX and Windows based systems.

NetBeans: A platform which is used to develop Java desktop, mobile, and web applications, as well as HTML5 applications with HTML, JavaScript, and CSS.

References:

[1]IncDigitalOcean™, "An introduction to tracking statistics with graphite, StatsD, and CollectD," DigitalOcean, 2014. [Online].
Available: https://www.digitalocean.com/community/tutorials/an-introduction-to-tracking-statistics-with-graphite-statsd-and-collectd. Accessed: Feb. 15, 2017.

[2]"Metrics," [Online]. Available: http://pcp.io/books/PCP_UAG/html-single/. Accessed: Feb. 17, 2017.

[3]"Software metrics for agile software development - IEEE Xplore document," 2017. [Online]. Available: http://ieeexplore.ieee.org/document/4483261/?section=abstract&part=1. Accessed: Feb. 19, 2017.