Team ViceGrips

Requirements Specification:

The VICE Web Application

Volodymyr Saruta

Ian Ambos

Ethan Baranowski

Joshua Melo

Tayler Skirvin

December 3, 2021
Version 1.1

Accepted baseline requirements for the project:

For the Client: ______________________ For the team: ______________________

Introduction

In a technology driven society, people rely heavily on the operation of their devices going smoothly. The public eye does not pay close attention to the specifics of firmware failures, most relying on a successful operation. Just like any other software, firmware will have bugs, and because firmware runs more critical parts of the system, the consequences of failure are larger. This is where validation comes in, a series of tests are performed to ensure that a new patch won't cause serious problems. Currently, Western Digital's validation process encounters issues, further exacerbated by transitioning to a remote workflow. Western Digital's validation process weighs heavily on meetings to coordinate and update each other, sometimes on minor issues, which wastes vital time. Improving this workflow is crucial, and is stressed in our solution. The goal of our project is to fix the above issues with VICE, a web application that will provide a one stop shop for techs and managers to keep track of what they need to do and schedule other issues that need to be fixed.

To make the VICE project possible, intensive planning and work are required. VICE must be able to follow the Validation as a Service (VaaS) model for SSDs onto the cloud. The general expectation of the VICE MVP is the ability to store and schedule workstations that are to be used for testing purposes, and virtualize the task of loading a firmware image onto a Solid State Drive remotely for testing purposes. We additionally need to implement certain analytical capabilities that process and output the results of the different testing suites in a few different ways.

In order to 'store' these workstations, we need to create a database to populate with the information about these machines. This involves having functional capabilities that allow us to read from, delete, and modify observations in the database. The database and capabilities will need to be implemented on the cloud.

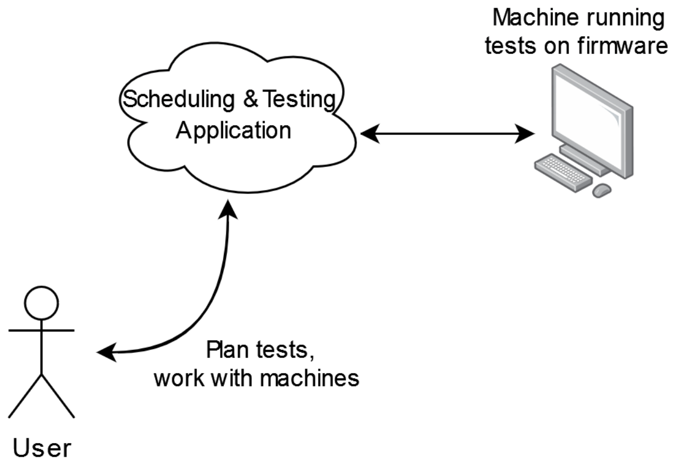
To create a scheduling tool to ensure we don't run into conflicts, we need to make use of this database, and utilize these basic functionalities. The scheduling tool also needs interactive user interface capabilities, and will essentially be the frontend of the project. In order to better understand the basic idea of our task, **Figure 1.0** below shows a simple representation of the problem.

Analytical capabilities will need to be developed once we have access to the testing environment and can better understand what outputs the suite delivers. Some nice-to-haves include using machine learning strategies to interpret what failures happen most often with what test plan, and using the results to move forward with future testing cycles more weighted towards those more probable failures. More realistically, our application will have the ability to export statistics (.csv format) that will allow for further analysis on possible optimizations of testing plans.

The outputs of testing on the virtual machines will need to be transported to multiple different libraries based on Western Digital's needs. Specifically, the issue tracking platform JIRA will be utilized to aid in discovering the most problematic combinations of machine, software, and firmware. Additionally, we will need to be able to actively monitor when issues occur, and notify those responsible for addressing said issues. The higher-level structure of our web application will need to be able to satisfy these requirements:

- Register all existing and new hardware testing platforms on a database (Store database of different machines available)
- Check the status of a platform before or after testing phases
- Allow 'virtualization' of particular platforms, adding the ability to check out a platform, while loading a software operating system image onto the SSD, and running testing suites
- Workflow "cycles" for new firmware - a validation manager tool that allows registration of SSD subsystems and >= 1 test plan (device + OS + test suite) developed for the SSD. We should be able to "launch" cycles for specific test plans, and have them run and produce output.

Figure 1.0

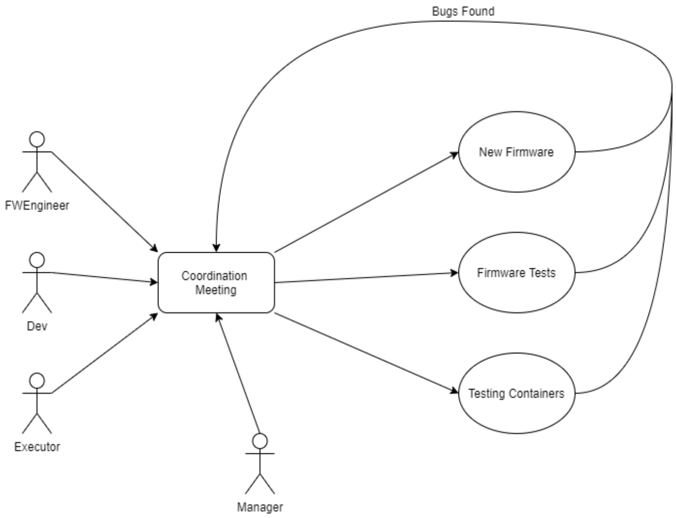

Machine running tests on firmware
Scheduling & Testing Application
Plan tests, work with machines
User

Users scheduling test cycles, planning tests which are input onto firmware machines.

In order to maintain a structure for keeping track of how many machines are in use and for what testing plan, we are going to need to implement a database system.

2: Problem Statement

Western Digital lacks in efficiency for firmware validation workflow, relying heavily on frequent meetings and communication across numerous engineers. Exaggerated by the increase in remote work, minor issues often require discussion, resulting in wasted time for debugging and firmware fixes. Attempting to release, analyze, and test firmware on a large scale comes at a disproportionate cost and disadvantage without a system to accurately view current tasks and updates across a team of validation engineers. Figure 1.1 below, illustrates the straightforward approach of Western Digital's current workflow, demonstrating the importance of creating a visual guide to inform the status of various tasks.



While the current process is lightweight, lack of communication or in fact miscommunication can easily produce additionally unnecessary restraints. Identifying which model is being debugged, performed tests, successful OS and test suites only scratch the surface of how detailed this validation process can be.

Introducing an informative web-application such as VICE removes the majority of these obstacles, eliminating the need for unnecessary meetings. Meaning, the user will be able to identify all up-to-date tasks for particular firmware models. Western Digital currently is missing key elements such as generated reports and descriptions, informative UI elements displaying the characteristics of a particular SSD, and property monitoring for assistance within the workspace. The items listed, encourage accurate communication between engineers, solving their issue of organization, management, and robustness of the firmware validation process.

In summary, Western Digital needs their validation process to be streamlined, creating a well organized and efficient workflow process for the Validation team. Missing capabilities and deficiencies are as follows, which is strove to be implemented through VICE:

- SSD product specification management overview (product details: owner, location, versions, status, with debug virtual machine name and accessible link)
- Debug updates
- Directly accessible test suites
- Firmware status
- Registration for new firmware in line for testing

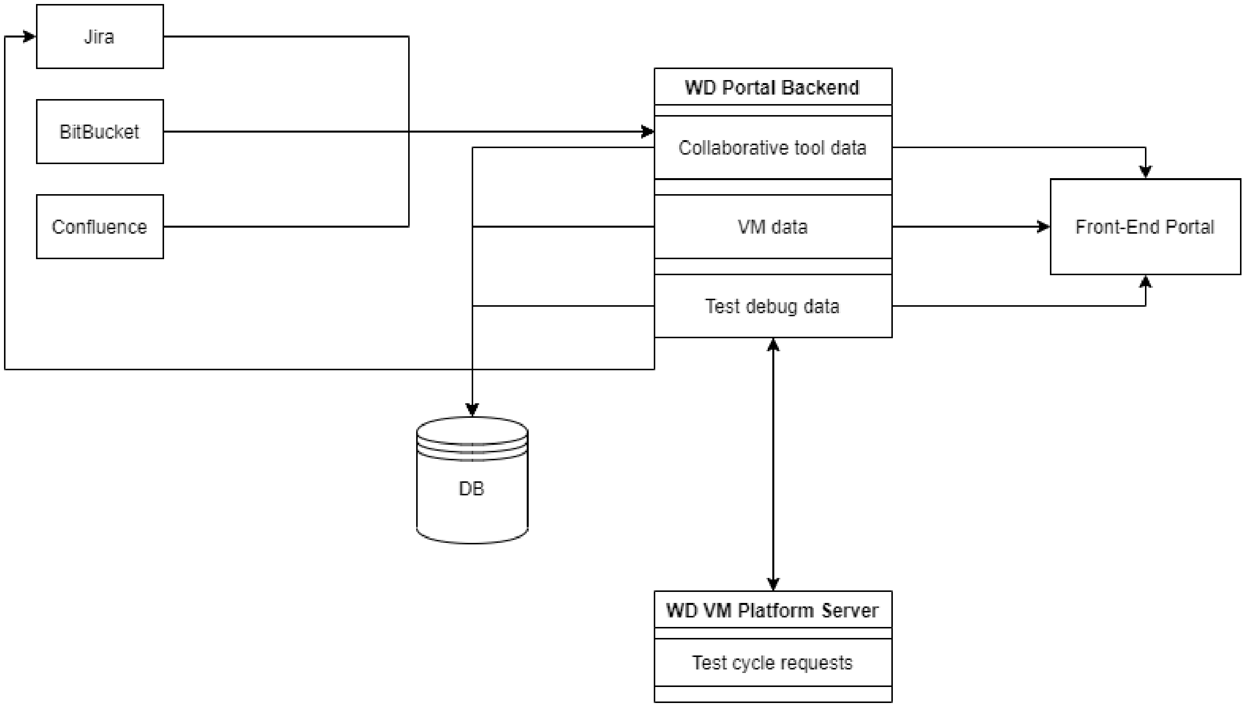
3: Solution Vision

A solution we envision is to have a web application that provides a multitude of functionalities for WD techs to get all the information they need to complete the day's work.

- A firmware engineer will be able to login, see what bugs need to be fixed and be able to remotely log into a virtual machine where they can work
- Techs can go into a testing cycle scheduler where they can create a set of platforms for a firmware patch to be tested on.
- Developers will have a similar view to the firmware engineers, but instead of bugs they will have patches they need to create tests for and an area to upload tests they created onto a testing VM.
- A database filled with firmware, VM and task information will be used to populate space in the web app and the web app will keep data in the database up to date.

Having a portal that is continually updated with data and tasks from a server running tests on firmware will allow employees to stay up to date with all information along the validation process.

Figure 1.2



4: Project Requirements

In this section we will explore each module in more detail, regarding what is needed for them to come to fruition. Each of these requirements are as follows, Functional, Performance and Environmental. Functional requirements will be how our system will work, Performance is how well our system will need to perform and Environmental are parts of the system that are required either through the client or by hardware restrictions. The overall goals for VICE includes:

- VICE will provide system, bug and patch information to firmware engineers
- VICE will provide system, patch and testing information to developers
- VICE will provide system, patch and scheduling information to executors
- VICE will allow users to remotely connect to work VM's and remotely schedule firmware tests

4.1: Functional Requirements:

The functional requirements for VICE are as follows - VICE will:

1. Implement a database to store data about firmware, virtual machines and data needed to interface with external services like Jira.
2. Implement an interface with Jira to pull tasks from and export tasks to.
3. Implement an interface with Confluence to export high level reports to.
4. Allow secure log-in by Western Digital employees and protect said employees information.
5. Provide views for FWEngineers, Developers and Supports to display needed information from the database.
6. Provide utility for the employees to more efficiently perform their job.
7. Provide an admin/manager role that will allow a user to manage other users profiles.
8. Allow the transfer of files to BitBucket for version control purposes.

A database to hold information for every part of VICE

Since our system is going to be a web application that is focused on scheduling firmware validation testing on various virtual machines, we are going to need several functions operating under the hood to reach our goal.

We must have a database capable of storing all information about a machine used for running firmware validation. This database will need to store the following information:

1. The form factor of the SSD on the machine
2. The status of the machine
3. Device location
4. Engineer currently / first working on the machine
5. The version of firmware being tested

Our database needs to be able to update / modify scheduled tests on said machines. We will need to have checks in the database ensuring that scheduling conflicts are not created.

Alongside the database that will be able to create and modify these firmware test schedules, we will need to implement a way to view these schedules in a graphical format. By scheduling a firmware test on a VM, we will have an estimated allotted time in which the test will take place.

We need to be able to view the schedule of the machines based on the database in a graphical format.

Because there are various types of users that will use the web application, it will need to be capable of creating views of the data based on the credentials of the user logged in to the system.

The database must be able to generate reports based on results from running firmware tests and export to various libraries.

Errors resulting from tests will need to be displayed in one view of the web application.

Interface with Jira

The interface with Jira will be able to export debug and testing tasks when firmware fails tests or firmware patches need tests to be made. These tasks will include a unique ID and short description for VICE to use. Each task will be filtered depending on which role needs to see it when taken into VICE. When used in VICE, these tasks will act as identifiers & buttons to access more detailed information. These tasks will be exported and imported from Jira and also will be stored in VICE's database as a backup.

Interface with Confluence

The interface with Confluence will be minimal. While tasks are being created and completed with the previously mentioned Jira interface, this data will be used to create high level reports and graphs through Confluence for managers to easily track the progress and performance of employees. Other data collected from VM's running firmware testing may also be used to create high level reports to aid in balancing workload for each VM.

Secure Log-in and Profiles

Users with profiles will be able to log in securely through the main page of VICE. When users go to the VICE website they will be prompted to enter an email and password. Any page of VICE will need proper authentication via an encrypted cookie that will keep track of a user's session - absence of this cookie will prompt a login. Western Digital will be able to create profiles for each employee that needs one. These profiles will be assigned specific roles upon creation to filter the view that profile will be provided. Profile creation is as follows:

- Admin provides a list of emails to VICE and VICE emails a link for users to setup profiles
- The profile setup page will have a new user create and confirm a password and set their role
- New users will be added to the database and be able to log in using the VICE secure login portal.

Separate Views for Different Roles

Users at Western Digital will be performing different roles and as such need separate views depending on their job. Each view will have some similarities but each will have a unique aspect to assist in the workflow for each task.

A firmware engineer will have a view with features:

- Side bar with a list of bugs to fix
- A changing main window that will provide information depending on which bug is selected
- A window in which to upload new firmware
- Links to remote machines that provide an environment to work in.

The information in the main window will include:

- Product information
    - Name
    - Serial number
- Firmware information
    - Version
    - Form factor
    - Commit ID
- VM information
    - Name
    - Status
    - Owner
    - Location

A Developer will have a view with features:

- Side bar with a list of tests to create
- A changing main window that will provide information depending on the test selected
- A window in which to upload new tests

The main window will have similar information to the firmware engineer but also include a description of the tests that need to be created.

A Support will have a view with features:

- Side bar with list of VM's
- A changing main window that will provide information depending on the VM selected
- A window in which to schedule a testing cycle for a firmware patch

The main window will have similar information to firmware engineers. The window to schedule a testing cycle will have features:

- VM selection dropdown box
- Selection box with list of OS's to test on
- Run time selection dropdown box
- Schedule button
- Cancel button

Utility for Western Digital Employees

The goal in providing a web application portal is to streamline the current process for firmware validation. As such the ability to have different views of the data based on the user's credentials needs to be implemented in such a way that it simplifies testing, development and maintenance.

Admin/Manager Role

An admin/manager role will be required to solve any accidental profile additions and to provide high level information to managers. This role will be able to access all task cards to view more detail while also having additional reports. These reports will be those generated by Confluence. This admin mode will also provide a view to manage profiles. This view will include features:

- List of all profiles
  - User name
  - User email
  - User ID
  - User Role

- Window to modify a profile
  - Textboxes to change name, ID or email
  - Radio checkbox to modify role
  - Delete profile, confirm and cancel changes buttons

Interface to BitBucket

As techs at Western Digital will be working with many different code files, VICE will interface with the version control system they already have in place. VICE will have multiple windows where techs will be able to upload code - from this VICE will take the uploaded file and commit it to the BitBucket repo.

4.2: Performance (non-functional) requirements:

With numerous functional requirements, bottlenecks on performance are crucial to identify. Setting performance measures will help increase the expected usability of the VICE web application, especially in early stages of development and deployment. Implementing performance goals will help increase the overall quality of the product, accurately handling all business-essential tasks in a timely manner. Below we examine performance tests and goals for each functional requirement of VICE.

Accurate Firmware Specifications

Producing proper information on firmware is essential for our web application, as misinformation can produce a snowball effect of bugs and unnecessary troubleshooting. Our goal is to accurately load information from our database within two minutes of execution. At the bare minimum we want to load in the correct information to prevent misleading details from populating our UI, but we strive to accomplish this task as quickly as possible.

Database Population

Additionally, we strive to populate our database with firmware entries within 2-4 minutes. Creating an organized but structured database will aid the efficiency and overall performance of VICE. These individual entries will include data such as: form factor, location, owner, serial number, firmware version, and status of current debugging.

JIRA Reporting

The overarching goal of VICE is to eliminate unnecessary communication between firmware engineers and managers. Therefore proper task delegation and management is crucial within our web application. The goal we set out to achieve is populating the JIRA tasks frequently with updated tasks and accurate details. Meaning, tasks that reside in the UI will not be outdated. We want the tasks to update their status daily, with the addition of illustrating who has assigned the task already.

Confluence Report Exporting

Assigning tasks successfully is just as important as exporting them properly. Therefore our goal is to export the firmware reports with accurate information.

Keeping consistent with the presentation of these reports will also increase the usability and decrease the learning curve for new developers and managers using the VICE web application. Additionally, these reports will be generated within half an hour of exporting a finished task. Once a task is completed, there should be little processing to account for, therefore the generation of these reports should be lightweight for the application to produce. The main time we need to account overhead for is the communication between VM testing infrastructure and our web application.

Secure Login Functionality

VICE is intended to be used by numerous employees with various positions within Western Digital. A login portal is necessary for accounting for these guidelines, as the UI will alter based on the user signed in. Accomplishing these tasks weighs heavily on security and password encryption for profile safety. We intended to load a user profile with the corresponding UI elements and save data within 30 seconds of submission of credentials. This prevents the user from waiting longer than necessary to have an overview of the next day's worth of specifications and tasks. Additionally, it is crucial that a user is logged in correctly. This means the user should be presented with the data pertaining to their job. We also do not want anyone logging into areas without their proper permissions.

4.3: Environmental Requirements:

The environmental requirements required of our team would be the tools our client has requested that we use alongside other technologies. Western Digital has requested to include the following:

- JIRA / Confluence
- Azure / Azure Tables
- Angular / Angular CLI

JIRA / Confluence

With the use of JIRA and Confluence, VICE will be capable of managing tasks and exporting firmware status to managers. Task delegation is essential for the developer to understand where to start when diagnosing SSD's within the web application. Both will act as system utility for developers, managers, and various employees of the validation infrastructure. Issue tracking with JIRA will be implemented within our UI, using a JIRA ID as a unique identifier for the developer. Interacting with the task will change the UI to tailor accordingly.

Azure / Azure Tables

In order to create a cloud infrastructure to house all necessary data, Western Digital has requested the use of Azure for the cloud platform of choice. Azure will be paired with their cloud computing service "Azure Tables" as well, creating a space to store all output data from various virtual machines and firmware tests. It is essential to store this data securely, as it will later be used for generating reports via Jira / Confluence.

Angular / Angular CLI

Creating large scale applications requires a great amount of delegation of responsibility, therefore each framework / library accomplishes different and unique tasks. Front end structure is just as important as a functional database, therefore VICE will utilize Angular and Angular CLI. With the use of this framework, we will be capable of making routes that communicate to JIRA and Azure to accurately assess the request of the user. Using their command line interface also increases modularity within our program.

5: Potential Risks

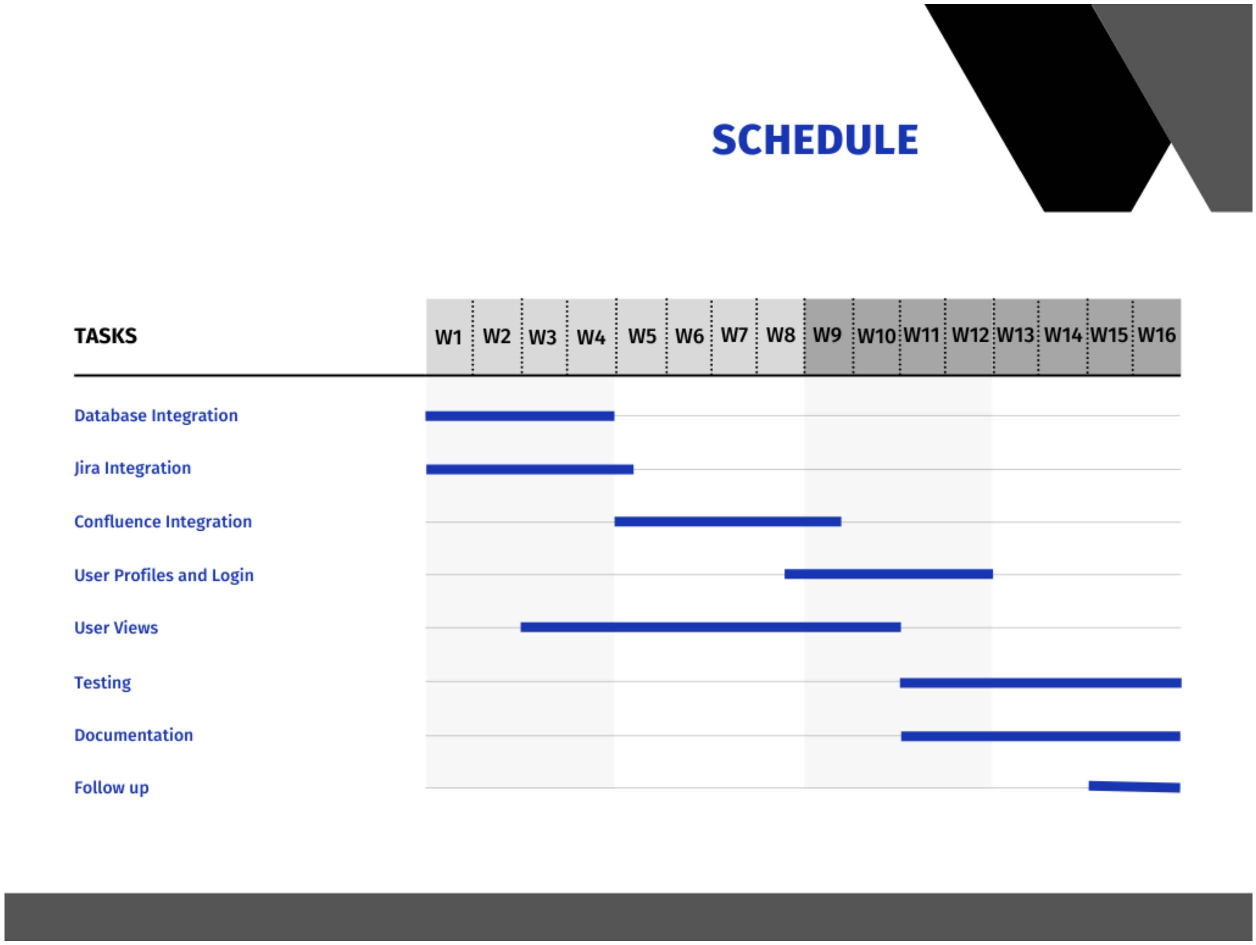
With large scale projects comes many avenues prone to potential risks, often varying in scale. Analyzing what is classified as a potential risk (aka likelihood of occurring or substantial long term effects), helps pinpoint what could possibly go wrong after deployment. Development success can be altered if there is a failure to decipher roadblocks early on. For an overview, potential risks for VICE rely heavily on runtime errors, ability to accurately relay information on firmware, thus preventing improper testing and allocation of firmware.

Risks associated with firmware validation processes can have extensive impact if not dealt with carefully. Failure to store user data correctly due to a firmware error can snowball into a plethora of security concerns due to improper identification of potential obstacles. Consequently, this increases security risks to businesses and consumers. More specifically, VICE could have a potential risk of inaccurately loading firmware information into the UI. This in turn can lead to performing unnecessary bug fixes, or performance degradation. Actively pursuing testing measures on inaccurate information, leaves room for numerous errors and miscommunication amongst validation engineers, managers, and the many individuals VICE plans to accommodate. Additionally, VICE faces a potential risk of encountering runtime errors, once an object is loaded into the dashboard. This could also be a result of a task assignment with Jira as well. The combination of many technologies increases dependencies, creating a potential for decrease in application performance. Each entry within our database easily can grow exponentially, therefore a runtime error does not seem out of the ordinary if constructed incorrectly. Using an object orientated approach allows for adaptability as newer technologies are released as well, preventing the application from easily becoming outdated.

Western Digital is looking to prevent miscommunication and increase the flow of the validation process. Failure to produce a system that strives to maintain consistency across the validation department would in fact produce the same outcome as the current workflow practices. In effort to prevent data loss, runtime errors, alongside other possibilities, maintaining a structured development approach is necessary. As data loss is crucial to avoid, it is important to relay the proper information to firmware engineers using VICE for debugging and troubleshooting.

6: Project Plan

For our plan going forward we will create the most essential, concrete modules first and build towards more dynamic modules that will need client input to get to a satisfactory product.



As seen here, we will integrate many extensible backend services that will have less chance for later changes. Jira and Confluence are both external services that allow for much flexibility on the services side; we will provide an interface for the flexibility so changes can be made easily on their side if the need arises. During the creation of many backend modules, part of the team will focus on the more dynamic user views. The process of creating the user views and interface will be iterative where mock-ups and prototypes will be created and shown to the client. End-users like firmware engineers will also be involved to create a productive environment.

7: Conclusion

VICE is a web application tailored for streamlining firmware validation processes, in our case for Western Digital. The construction of an application capable of relaying important testing platforms alongside product specifications outweighs completing this task without a centralized system. Communicating via meetings and hearsay creates too many opportunities for miscommunications and errors. Operating and validating hardware on a larger scale illustrates the importance of consistency across a team of engineers, therefore adapting to that need is crucial for an increase in productivity, resource allocation, with the added benefit of reduced costs.

Catering to the needs of Western Digital and their validation team, we plan to implement numerous features creating a flexible but reliable workflow for firmware engineers. In order to successfully relay this information, displaying patch data and virtual machine (VM) specifications is necessary. Additionally, embedding a testing cycle scheduling system with Jira will effectively organize the tasks needing attention versus those already examined. Allowing firmware uploading authorizes new hardware to be debugged and troubleshooted. Storing raw data on each hardware specification allows the client to quickly view any misleading information that could potentially arise with the addition of numerous hardware units. The validation process used by Western Digital currently, relies heavily on peer to peer communication, but introducing a lightweight UI accounts for proper SSD form factors and testing status eliminates the need for recurring meetings and emails.

In summary, VICE is intended to modernize current business techniques within the firmware validation realm. While being mindful of potential security risks and logical errors, we intend to provide responsiveness and accuracy for Western Digital's validation team.