

ACCA7001 Assignment 1 - Agile report - 2012820

by Liam Samuel Roberts

FILE	25304_LIAM_SAMUEL_ROBERTS_ACCA7001_ASSIGNMENT_1_-_AGILE_REPORT_-_2012820_302864_1135328133.PDF (1.11M)		
TIME SUBMITTED	04-DEC-2020 12:32AM (UTC+0000)	WORD COUNT	3805
SUBMISSION ID	138261390	CHARACTER COUNT	20148

DVLA's Implementation of Agile

ACCA7001 Assignment 1

Liam Samuel Roberts

December 4, 2020

Abstract

This report introduces and evaluates the Drivers Vehicle Licensing Agency's (DVLA) implementation of Agile against the values found in the Agile Manifesto. The DVLA use practices from the Scrum and Kanban frameworks to help integrate the Agile values into their working philosophy. Some of the more important concepts of the implementation are introduced and discussed, like user stories and sprints. Also covered is how the DVLA ensure a working code focus is at the heart of development through different forms of testing and continuous integration, through Cucumber and Drone.io. The report concludes that the DVLA's form of implementing Agile meets or surpasses each of the 4 values observed in the Agile manifesto thus the DVLA's implementation of Agile is proven to be a very strong example of how Agile can govern development work.

1 Introduction

Traditional software development methodologies such as Waterfall focused on a fixed and easy to follow framework as a project progressed through the 6 stages of development, known as the software development life cycle, shown in Fig. 1.[1] Methodologies like Waterfall were discovered to have a number of disadvantages, like testing happening late in the cycle causing increased cost to fix errors and the inability to deal with requirement changes effectively.[2]

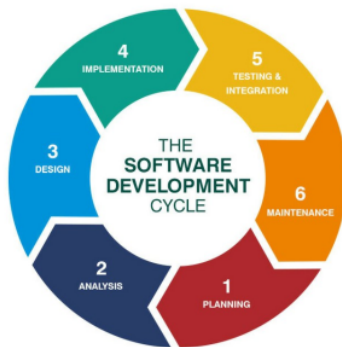


Figure 1: The Software Development Life Cycle.[3]

To deal with the difficulties and disadvantages of older methodologies in the spring of 2000, 17 renowned software engineers held meetings to discuss practices that could increase

development speed and efficiency.[4] The Agile manifesto, as seen in fig. 2, was born out of these meetings, as a set of values which put more emphasis on working code, customer requirements and allowing for changes in requirements to be implemented easily.[4] Also discussed were the 12 principles of Agile, seen in fig. 3. which come from following the Agile Manifesto.[4] In reality the concept of Agile moves away from strict frameworks and linear development paths in favour of installing a belief system that attempts to embrace and mitigate the impact of change and disruptions to the efficiency of the team, putting a focus on collaboration, communication and the working code.

This report aims to introduce the way the Drivers Vehicle Licensing Agency, DVLA implement the Agile values and evaluate the effectiveness of this approach. The main conclusions gathered will be based on how well the DVLA practice fits the Agile Manifesto. Testing practices will also be a feature of this report as the concept of working code is such a significant part of the manifesto.

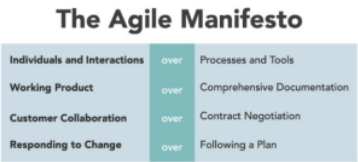


Figure 2: The Agile Manifesto.[4]

12 AGILE PRINCIPLES BEHIND THE AGILE MANIFESTO

1 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	2 Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	3 Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4 Business people and developers must work together daily throughout the project.	5 Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	6 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
7 Working software is the primary measure of progress.	8 The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	9 Continuous attention to technical excellence and good design enhances agility.
10 Simplicity – the art of maximizing the amount of work not done – is essential.	11 The best architectures, requirements, and designs emerge from self-organizing teams.	12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Figure 3: The Agile Manifesto.[5]

2 DVLA Implementation of Agile

As the Agile methodology is not a strict framework, the DVLA choose to adopt working practices from the Scrum and Kanban frameworks to support their Agile approach. The evaluation provided will be based on how the DVLA's practices ensure the Agile manifesto is satisfied.

2.1 User Stories

Requirements in the world of Scrum are written from the perspective of the user and go through several stages of refinement before work can begin. In early stages requirements may take the form of an overarching statement known as a theme.[6] Themes are then refined into initiatives which tend to include the path the user is required to take to satisfy the theme. The next stage of refinement separates an initiative so that each step in the path is its own individual issue known as an epic.[7] Finally epics are refined into user stories. Whereas epics are typically features, user stories are said to be the smallest amount of work required to achieve a goal.[8] This process is illustrated in fig. 4.

Once a user story is refined it is stored in the product backlog. The short nature of user stories enables momentum to build within the Scrum team as completing smaller tasks regularly encourages confidence and provides motivation to the team. User stories typically have an acceptance criteria, also known as a definition of done, which must be met so that the story is marked as complete.[9]

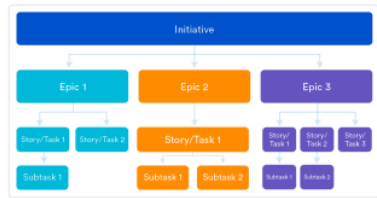


Figure 4: Refinement Road Map[6]

An example of a DVLA user story can be seen in fig. 5. From fig. 5, it is evident that there are three components to a user story. the first component is the type of user, the second is the goal that needs to be completed and the final component is the advantage that completing the story would bring. The DVLA's usage of Jira allows for the Scrum team to assign each other to user stories whilst also displaying the number of story points the story was estimated to be.

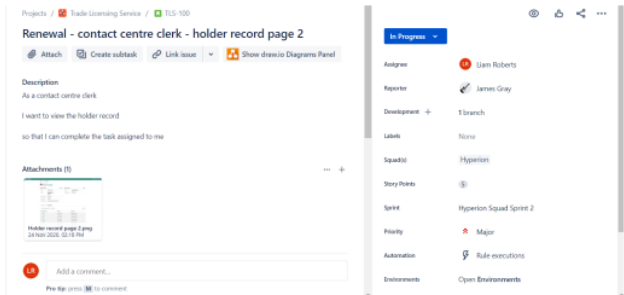


Figure 5: Example of a DVLA user story

2.1.1 User Story Estimation

Following refinement, a user story must be estimated before development work can begin. Estimation is the process of assigning story points to a user story. In the DVLA story, points take the values of Fibonacci series numbers up to a maximum of 13. Story points reflect both the amount of time needed to complete the story and the story's complexity.[10] Prior to assigning story points, the Scrum team will have a discussion about how best to complete the story and

make note of any sub-tasks. After the discussion ends the team will do a round of planning poker.[11] Each team member will assign a card equal to the value of points they believe a story should be and everyone reveals their cards at the same time. If each member agrees on the same value that value is assigned.[11] If not, the highest and lowest estimators explain their decisions and then another round of planning poker begins. This process is repeated until a consensus is reached.[11]

The advantages of user story estimation is that it ensures high level discussion which engages the whole Scrum team, ensuring that everyone has a clear idea of the tasks that need to be carried out. Estimating stories in this way ensures that each team member feels comfortable to express opinions and it is possible that the team will spot similarities between user stories that make collaboration viable.

2.2 Scrum Roles

Scrum ensures solid lines of communication between the development team and the customer by introducing three clearly defined roles of responsibility. These roles are:

1. Product Owner
2. Scrum Master
3. Scrum Team

The product owners job is to take ownership and maintain the product backlog.[12] Additionally they are responsible for liaising with any other stakeholders.[12] In the DVLA the product owner role is assigned to a service designer. The Scrum master acts as the facilitator for the Scrum team. The Scrum masters job is to clear any issues, ensure collaboration between the team and the product owner whilst maintaining the Agile beliefs are followed by the team.[13] In the DVLA the Scrum master is also a line manager however this is not common practice outside of the DVLA. Members of the Scrum team undertake the development work required for a project, organising the work amongst themselves.[14]

2.3 Sprints

The Scrum framework uses an incremental approach to the development process. Each increment is a time limited period known as a sprint. Sprints begin with a meeting known as a sprint planning meeting. Each sprint has a sprint goal and lasts a minimum of two weeks. Each day the Scrum team and Scrum master will attend a daily stand up. Sprints end with a sprint review and a sprint retrospective.

2.3.1 Sprint Planning Meeting

A sprint planning session is a meeting which involves each of the Scrum roles where the main objective is to determine what the teams capacity of story points will be during the next sprint and then promote stories from the product backlog to the sprint backlog until the teams capacity is reached.[15, 16] Story estimation takes place in this meeting. Tasks may be allocated by the Scrum team to individuals during the meeting.[15] This allows members to be aware of what each other are working on meaning that opportunities to collaborate arise naturally. To close the meeting a sprint goal is assigned by the product owner.[17] Typically the sprint planning session is fixed to a maximum 2 hour duration.[15]

2.3.2 Daily Stand Up

The daily stand up is a meeting where each Scrum member answers three simple questions:

- What did you do yesterday?
- what do you plan on doing today?
- Any known impediments?

The daily stand up should last 15 minutes with any issues being dealt with outside of the stand up.[18]

2.3.3 Sprint Review

During the sprint review the Scrum team will demonstrate any working code completed during the sprint to the Scrum master, product owner and any additional stakeholders that have been invited to the meeting.[19] Typically discussions take place between the entire cohort on what should be prioritised during the next sprint as well as any changes that are needed in response to requirement changes.[19] A sprint review is typically fixed to an hour.[19]

2.3.4 Sprint Retrospective

During the sprint retrospective, the Scrum team provide feedback on the sprint in an open and honest reflective format, discussing what went well, what went bad and what could be improved in future. Only the Scrum team and Scrum master attend the retrospective.[20] The Scrum retrospective is fixed to an hour duration.[20]

2.4 Product and Sprint Backlogs

The product backlog is a collection of the user stories that need to be completed during the project. The DVLA keep the higher priority user stories towards the top of the backlog. The product backlog is owned and maintained by the product owner thus stories in the product backlog can not be edited by the Scrum team.[21]

Fig. 6 contains an example of a product backlog for one of the development teams of the DVLA. The DVLA product backlog contains not only user stories but also spikes and major bugs that need to be resolved. Spikes are created to reflect a form of impediment that needs to be addressed before some stories can progress. The Jira software allows the DVLA to link spikes to particular user stories so that when the story being affected by the spike is brought into a sprint the spike is also. Unlike user stories and bugs spikes do not get estimated. Unresolved bugs are added to the product backlog as typically the product owner and users satisfaction with a service would be impaired by them.

The sprint backlog is a collection of user stories that have been refined, estimated and selected to be a part of the active sprint in a sprint planning session. The sprint backlog is owned by the scrum team. User stories in the sprint backlog typically contain a list of sub tasks needed to complete the story. Fig. 7 contains an example of a sprint backlog for a development team in the DVLA. From fig. 7, 'DAGOBAB' is an example of a tag used to note stories that need to be contained within the same release of a project. Each story is also assigned a individual code like 'TLS-100' for quick reference. After a sprint has began, the Jira software populates a Kanban board with the contents of the sprint backlog.

Backlog 262 issues Create sprint ...

Insert Tacho Team Inf TACHO DRIVER - MANAGER RECONCILIATION TALLY UP (PTR SOLUTION) Evolve, TACHO_DAY1, ingest	MSB-4010	✓	-
Ingest: Addressing Dataset Evolve, StateOfRecord, TACHO_DAY1, Volumes, ingest	MSB-3698	⬆	8
⚡ SPIKE: Investigate technical approach to Tacho Org Data Evolve, TACHO_DAY1, ingest	MSB-4593	⬆	0
Ingest: Payment Dataset Evolve, StateOfRecord, TACHO_DAY1, Volumes, ingest	MSB-3672	⬆	8
🔴 Clean up unexpected Notification logs in Production None	MSB-4338	✓	-
Implement Bulk Data generation for Tacho Markers Transaction UNLABELLED-MSB-3980 Evolve, TACHO_DAY1	MSB-4539	✓	8

Figure 6: Product Backlog From a DVLA Development Team

▼ Sprint 48 12 issues 21 29 10 Linked pages 0 ...

Ingest Notification dataset and start presenting Driver data for MIDAS

19/Nov/20 7:21 PM • 03/Dec/20 11:42 AM

...

⚡ Azure function network settings in INT and PROD DevOps, TechDebt	MSB-3572	⬆	0
📄 Ingest: Notification Dataset Evolve, StateOfRecord, TACHO_DAY1, Volumes, ingest	DAGOB AH MSB-3699	⬆	8
📄 Update Tachograph Driver events to include amendReasons field Evolve, TACHO_DAY1, ingest	DAGOB AH MSB-4352	⬆	3
📄 Update Tachograph Driver events to include photo and signature id's to re Evolve, TACHO_DAY1, ingest	DAGOB AH MSB-4353	⬆	3
🔴 CardReplaceApplicationReceivedEvent - error with enumerated types Evolve, TACHO_DAY1, ingest	DAGOB AH MSB-4570	⬆	2

Figure 7: Product Backlog From a DVLA Development Team

2.5 Kanban

The Kanban framework in its simplest form would be a board with three columns.[22] The aims of Kanban are to minimise work in progress, manage work flow and improve collaboration between team members.[22] The DVLA Kanban boards are reflective of that teams active sprints. Jira allows the DVLA to see a Kanban board based on user stories as a whole or focus on the sub tasks of a story.

An example Kanban board used in the DVLA can be observed in fig. 8. The DVLA version of a Kanban board includes a column for blocked sub tasks on top the traditional three columns. Tasks in the blocked column are unable to be completed due to an impediment. The Jira shows which team member is responsible for each sub task.

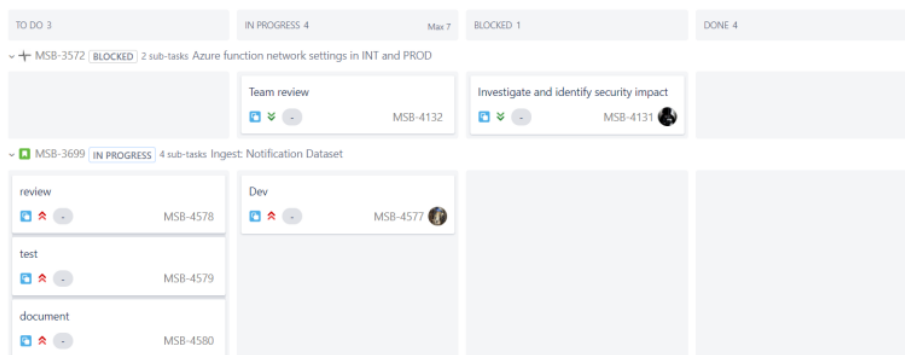


Figure 8: Example of a Kanban Board[22]

2.6 Collaboration

One of the reoccurring themes of this section is how the measures put in place by the DVLA enable strong collaboration. Collaboration in the DVLA is really in the hands of each developer, as effort is made to ensure awareness of each team members goals for the day, organising sessions for mob programming or paired programming becomes very simple. These forms of collaboration ensure discussions take place amongst the team whilst also aiding the professional development and learning of team members. The DVLA harbours a strong community feeling within its software engineering departments where anyone who needs help, can openly ask for it and they will usually find the answers they need quickly, even if they have to go outside of their own team.

3 Testing

Emphasis on testing comes naturally as a part of the Agile process. Agile focuses on the working code and to identify code is working, extensive testing should be carried out. The DVLA use various technologies to fulfil this need. Cucumber is their primary testing framework, whilst using Bitbucket for version control enables the DVLA to integrate continuous integration tools and testing software into their standard workflow.

3.1 Cucumber

Cucumber is used by the DVLA as it's language, Gherkin, is very human-readable meaning a developer could show the tests to a person with no technical knowledge such as product owner and the product owner could understand what is happening in the test.[23] Furthermore a product owner could even help in the creation of tests. Tests written in Cucumber state a scenario and then follow a "given, and, when and then" format to specify what actions need to be taken to satisfy the expected output. This can be seen in fig. 9. Cucumber is often found in behaviour driven development (BDD) environments. BDD is the process of testing code as development is in progress.[24] In general BDD is believed to be the Agile method of testing as stating scenarios allows the tester to test from different user perspectives.[24] Problems are found quickly in BDD environments as the code is tested straight after it is written rather than at the end of the development which leads to savings in the cost of a development.

Scenario Outline: View items by status

Given that Jane has a todo list containing **<tasks>**

And she has completed the task called **<completed task>**

When she filters her list to show only **<filter>** tasks

Then her todo list should contain **<expected>**

Figure 9: Cucumber Example Statement.[25]

3.2 Bitbucket Version Control

Using Bitbucket enables and supports collaboration on projects as each team member is able to create a branch for the user story they are working on. When a user story is completed the branch can simply be merged into the master without disrupting anybody else in the team. There is a specific process for this which can be seen in fig. 10. Before being able to merge back into the master branch the code must pass a few stages of continuous integration tools to try to prevent errors from reaching the master branch.

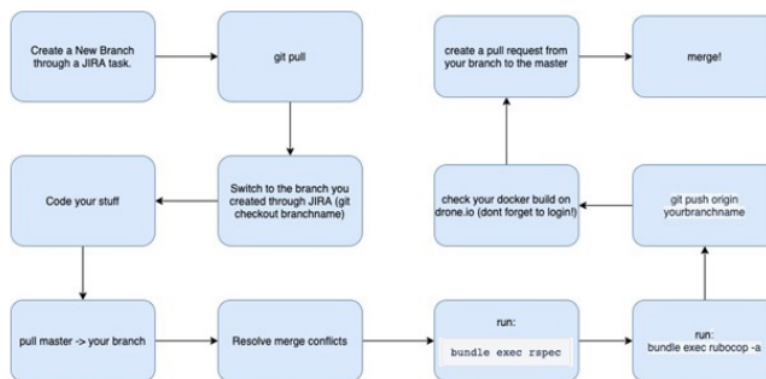


Figure 10: DVLA Workflow Diagram

3.2.1 Continuous Integration Tools

As seen in fig. 10, the DVLA use RSpec, Rubocop and Drone.io as their continuous integration tools. RSpec is a domain specific language used for testing Ruby code. RSpec is seen as the most widely adopted ruby testing framework, especially in the rails community, having been downloaded over 480 millions times.[26] As Ruby on Rails is one of the main languages used in the DVLA the use of RSpec comes naturally. Rubocop is a linting tool and code formatting designed for Ruby which is used by the DVLA use in edition to ESLint to enforce a coding standard across all of it's development teams.[27] Maintaining a coding standard in this way helps to ensure that a developer can read and understand code even if they did not write it themselves opening up collaboration avenues. An example of Rubocop in use can be seen in fig. 11. As seen in fig. 11, after detecting issues with the code, Rubocop reports each issue in the terminal. It is possible for Rubocop to fix any issues detected by it's self. Rubocop also removes any white space from code which is important as white space interferes with Drone.io.

Drone.io is the continuous integration testing tool that the DVLA have built into the Bitbucket version control software. Each time a merge is pushed to the remote repository the code must pass a Drone.io build. Drone.io is a four step process, the first step is to clone the initial code. The second step is to perform a merge with the current version of the master so that step three verifies the newly merged code. In the third step Drone.io automatically runs all of the test written in the project such as unit tests. Drone also runs a variety of other testing and linting tools, including the likes of RSpec and Rubocop but also tools like CSS linting and Gherkin linting. The final step is that drone.io will do a complete docker build of the code to ensure that the code can be published correctly with no issues. Once the drone build passes the branch that was merged with the master gets removed. Another useful feature of Drone.io is that it keeps logs of each build, so that should you need to revert back a commit one can be confident that a build was successful before reverting.

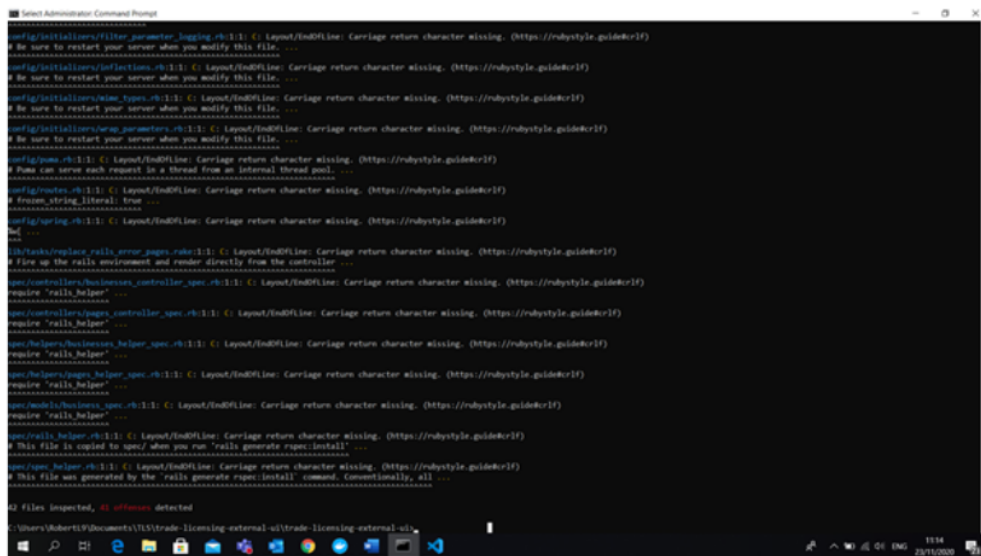


Figure 11: Example of Rubocop Reporting Errors

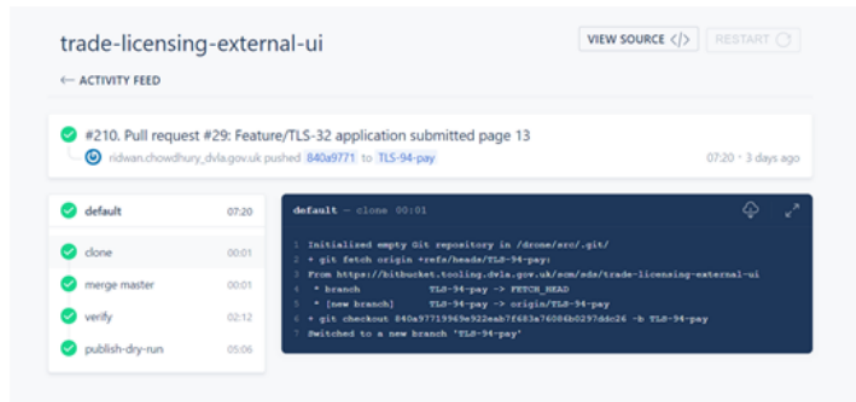


Figure 12: Example of a Successful Drone.io Build

4 Conclusion

The DVLA's focus on the working code is clear to see as the use of Drone.io alone, includes a vast number of testing and linting packages which are more than enough to build confidence in code that successfully passes the build. Using Cucumber as a testing framework also helps foster confidence in code. Cucumber also allows for the customer or product owner to understand a test's scenario, knowing exactly what is being tested and why.

The introduction of the Scrum roles means the customer has an active role in the development process ensuring customer collaboration at all times. The product owner role in addition to working in sprints allows the requirements of the project to change because introducing or modifying a user story that's in the product backlog cause no impact on the development team. The sprint planning session and daily stand up gives freedom to the developer to interact with each other and collaborate in any way they see fit. Furthermore, working using the Scrum framework pushes developers to have more open and honest conversations at both a high level and a low level. The Kanban boards used by the DVLA provide visual representation of each developer's current goals for the sprint meaning developers working on similar tasks will be aware, opening even more collaboration avenues.

Effectively, the DVLA's implementation of Agile either meets or surpasses all four values presented by the Agile manifesto. Therefore the conclusion of this report is that the DVLA's implementation can be considered a strong reflection of the Agile way.

References

- [1] J. Pinherio, "Software Development Life Cycle (SDLC) phases", Medium, 2018. [Online]. Available: <https://medium.com/@jilvanpinheiro/software-development-life-cycle-sdlc-phases-40d46afbe384>. [Accessed Dec. 3, 2020].
- [2] "SDLC - Waterfall Model - Tutorialspoint", Tutorialspoint, [Online]. Available: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm. [Accessed Dec. 3, 2020].
- [3] "Product Lifecycle Management – Milena Software", Milenasoftware. [Online]. Available: https://milenasoftware.com/?page_id=208. [Accessed Dec. 3, 2020].
- [4] "History of Agile", Planview, 2020. [Online]. Available: <https://www.planview.com/resources/guide/Agile-methodologies-a-beginners-guide/history-of-Agile>. [Accessed Dec. 3, 2020].
- [5] A. Mersino, "What is Agile and Why is it Important?", Vitality Chicago Inc., 2019. [Online]. Available: <https://vitalitychicago.com/blog/what-is-Agile-and-why-is-it-important/>. [Accessed: 03- Dec- 2020].
- [6] M. Rehkopf, "Epics, Stories, Themes, and Initiatives — Atlassian", Atlassian. [Online]. Available: <https://www.atlassian.com/agile/project-management/epics-stories-themes>. [Accessed: 03- Dec- 2020].
- [7] M. Rehkopf, "Agile epics: definition, examples, and templates — Atlassian", Atlassian. [Online]. Available: <https://www.atlassian.com/agile/project-management/epics-stories-themes>. [Accessed: 03- Dec- 2020].
- [8] M. Rehkopf, "User Stories with Examples and Template — Atlassian", Atlassian. [Online]. Available: <https://www.atlassian.com/agile/project-management/epics-stories-themes>. [Accessed: 03- Dec- 2020].
- [9] H. McCloskey, "The Agile Definition of Done: What Product Managers Need to Know", Product Plan. [Online]. Available: <https://www.productplan.com/agile-definition-of-done>. [Accessed: 03- Dec- 2020].
- [10] "Estimate Story", Guide.quickscrum.com, 2020. [Online]. Available: <https://guide.quickScrum.com/Scrum-guide/estimate-story/>. [Accessed: 03- Dec- 2020].
- [11] M. Cohn, "Planning Poker: An Agile Estimating and Planning Technique", Mountain Goat Software, 2020. [Online]. Available: <https://www.mountaingoatsoftware.com/Agile/planning-poker>. [Accessed: 03- Dec- 2020].
- [12] "Product Owner Roles and Responsibilities — 280 Group", 280 Group. [Online]. Available: <https://280group.com/what-is-product-management/roles/product-owner/>. [Accessed: 03- Dec- 2020].
- [13] "What is a Scrum Master?", Agile Alliance. [Online]. Available: <https://www.Agilealliance.org/glossary/Scrum-master/>. [Accessed: 03- Dec- 2020].
- [14] "Scrum Team Roles and Responsibilities — Scrum Alliance", Scrumalliance.org. [Online]. Available: <https://www.Scrumalliance.org/about-Scrum/team>. [Accessed: 03- Dec- 2020].

- [15] "What is Sprint Planning?", Scrum.org. [Online]. Available: <https://www.Scrum.org/resources/what-is-sprint-planning>. [Accessed: 03- Dec- 2020].
- [16] "Ideal Modeling Diagramming Tool for Agile Team Collaboration", Visual-paradigm.com. [Online]. Available: <https://www.visual-paradigm.com/Scrum/what-is-story-point-in-Agile/>. [Accessed: 03- Dec- 2020].
- [17] M. Cohn, "Sprint Planning Meeting", Mountain Goat Software, 2020. [Online]. Available: <https://www.mountaingoatsoftware.com/agile/scrum/meetings/sprint-planning-meeting>. [Accessed: 03- Dec- 2020].
- [18] "Daily Standup Meetings: Everything You Need to Know (Standup Agenda, Purpose, Common Pitfalls, and More!)", Geekbot blog. [Online]. Available: <https://geekbot.com/blog/daily-standup-meeting>. [Accessed: 03- Dec- 2020].
- [19] "What is a Sprint Review?", Scrum.org. [Online]. Available: <https://www.Scrum.org/resources/what-is-a-sprint-review>. [Accessed: 03- Dec- 2020].
- [20] "What is a Sprint Retrospective?", Scrum.org. [Online]. Available: <https://www.Scrum.org/resources/what-is-a-sprint-retrospective>. [Accessed: 03- Dec- 2020].
- [21] "What is a Backlog?", Agile Alliance. [Online]. Available: <https://www.Agilealliance.org/glossary/backlog/>. [Accessed: 03- Dec- 2020].
- [22] F. Miller, "What is a Kanban Board? Why and When to Use Kanban?", Productivityland.com, 2019. [Online]. Available: <https://productivityland.com/what-is-kanban-board/>. [Accessed: 03- Dec- 2020].
- [23] "Introduction - Cucumber Documentation", Cucumber.io. [Online]. Available: <https://cucumber.io/docs/guides/overview/>. [Accessed: 03- Dec- 2020].
- [24] "Cucumber - Overview - Tutorialspoint", Tutorialspoint.com. [Online]. Available: https://www.tutorialspoint.com/cucumber/cucumber_overview.htm. [Accessed : 03 – Dec – 2020].
- [25] "Cucumber test results don't display examples name or steps · Issue 1360 · cucumber/cucumber-jvm", GitHub, 2018. [Online]. Available: <https://github.com/cucumber/cucumber-jvm/issues/1360>. [Accessed: 03- Dec- 2020].
- [26] "Introduction to RSpec — The Odin Project", Theodinproject.com. [Online]. Available: <https://www.theodinproject.com/courses/ruby-programming/lessons/introduction-to-rspec>. [Accessed: 03- Dec- 2020].
- [27] "RuboCop :: RuboCop Docs", Docs.rubocop.org. [Online]. Available: <https://docs.rubocop.org/rubocop/index.html>. [Accessed: 03- Dec- 2020].

ACCA7001 Assignment 1 - Agile report - 2012820

GRADEMARK REPORT

FINAL GRADE

75/100

GENERAL COMMENTS

Instructor

Agile Process Evaluation (40): 31

Examples (30): 24

Testing (30): 20

Discussion:

Good discussson of Agile and the DVLA application of Agile

More background research and discussion of CI testing could have been provided

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6



Comment 1

Sprint

PAGE 7

