

Agile Software Engineering Principles - Assignment 1

by Harley-Earl Burrow

FILE	25365_HARLEY- EARL_BURROW_AGILE_SOFTWARE_ENGINEERING_PRINCIPLES_- _ASSIGNMENT_1_302864_1544509952.PDF (1.05M)		
TIME SUBMITTED	04-DEC-2020 07:10PM (UTC+0000)	WORD COUNT	3927
SUBMISSION ID	138374334	CHARACTER COUNT	21568



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Agile Software Engineering Principles

ASSIGNMENT 1: EVALUATING AGILE DEVELOPMENT PRACTICES AND TOOLS USED AT THE DVLA

Student ID number: 2012822

Harley-Earl Burrow

Word Count: 3299

Table of Contents

Cover Page 1

Contents Page 2

Introduction 3

Main 4

Conclusion 13

References 16

Write a report that evaluates the Agile development practices and tools used at the DVLA. In this report, consider the methods used for recording User Stories, estimation, product backlogs, sprint backlogs and collaboration

Introduction

The Driver and Vehicle Licensing Agency (DVLA) deals with the records of more than 38 million vehicles in the UK as of 2019 [1]. This presents a challenge in the software development within the company, specifically employing robust tools and practices that comply with United Kingdom law and government standards. The chosen development methodology employed by the software development teams within the DVLA is the Agile methodology.

The Agile development methodology is a software development process whereby the methods focus on iterative and incremental development. The process suggests starting with simple and predictable approximations to result with the final product [2]. The Agile software development process promotes the use of continuous improvement [3] throughout the development process to achieve the best product possible.

The incremental focuses throughout the Agile process are called iterations, which are defined as “Short time frames that run from one to four weeks” [4]. These iterations involve the team members working through software development life cycles (SDLC), such as planning, analysing requirements, design, programming, and unit and acceptance testing [4]. Agile methods have become widely adopted and implemented worldwide to deal with issues in more traditional methodologies like DevOps, such as management of improvements and quality control. [5] The Agile method allows for increased productivity and helps to meet customers’ demands, in short chunks called Sprints [4]. Working in this way allows the project to be broken down into smaller, simpler steps and also allows direct participation from the customer during the project development cycle. The customer could send a change request during any time and at any level of the project and using this method it can easily be implemented, allowing higher client satisfaction [5]. Additionally, implementing newer technologies when required becomes more feasible when following the agile methodology [6].

Amongst the various Agile methodologies, such as Dynamic Software Development Method (DSDM), Feature Driven Development, Crystal Methodologies and Scrum, Scrum is among the most widely used for software product development [5]. In the Scrum method, a well-structured approach is taken and is designed to keep the team motivated throughout the project. A “product backlog” provides the direction of the sprint, which is a prioritized list of user stories [7]. User stories are described as a functionality that will be available to either a purchaser or user of a piece of software

or system and are broken into three aspects [8]. The three aspects are namely, a short, written description of the story as a reminder, conversations about the story that help flesh out the details, and tests that convey when a story is complete [9].

The agile methodology employs the use of daily stand-ups, where team members meet every day to go over progress and problems they are facing. The aim of daily stand-ups is to get input from the whole team and update the scrum master, a “round table” is often used, where each team member is asked and answers 3 questions, namely, ‘what did you do yesterday?’, ‘what are your plans for today?’, and ‘Any problems or blockers?’, where blockers are items which may impede the squad members work.

This report will aim to evaluate the Agile practices and tools employed by the DVLA, considering methods for recording user stories, estimation, product backlogs, sprint backlogs and collaboration. Referencing examples of stories, tasks and tests via the method of screenshots. Additionally, reflection upon the importance of continuous integration and testing processes will be included, relating specifically to the technologies and techniques employed by the DVLA.

Main

User stories

User stories are an Agile method of breaking down tasks during projects. User stories have been described as parts of development that have a scope large enough to provide value to a customer, but small enough that the effort required can be estimated with a low risk of being incorrect [10]. User stories are derived Definition of ready (DOR) and definition of done (DOD). The DVLA define DOR items as items that can be added to the backlog and confidently reach DOD items by the end of the sprint, where DOD items are completely functional items. There may be features within the sprint that are not required for a complete DOD product, this helps prioritise importance of user stories.

During the planning phase of the project, known as the sprint plan, tasks are broken up into short sentences or paragraphs to decide how to meet the user demands. A common method used to determine these stories is three simple sentences. For example, ‘As a party planner’, ‘I want cake’, ‘So that people at the party can eat cake’. User stories should be atomic, meaning they should not be relative or reliant on other user stories, made up of indivisible units. Atomic user stories support collaboration and the ability to measure progress during the development cycle. User stories are given estimated sizes, using various methods, such as T-Shirt sizing, or Elephant Sizing (Extra Small, Small, Medium, Large, Extra Large). Planning poker is also adopted to size user stories, whereby

team members are given cards (either physical or virtually), the story is called out, each team member will place a card down with a number on, and all team members flip their cards to reveal their cards. Cards usually follow the Fibonacci sequence ('1,2,3,5,8,13,21') where one is the smallest and 21 is the largest. A good reason for using relative numbers rather than simply counting 1-8, is to avoid tripping up on similar numbers, a user story of 6 may not be much smaller than a user story of 7, however 21 is significantly larger than 13. Additionally, with traditional numbering, the team can fall into the trap of assigning time units to numbers, for example 'a 1 should only take you 2 hours, therefore a 2 should take you 4 hours', this hinders this methodology and is to be avoided where possible.

Jira

Jira is a project management software developed by Atlassian to assist in planning, tracking and managing software development, especially using the Agile methodology.

Within the DVLA, Jira is the software used to display sprint backlog, and Kanban boards. User stories within Jira are assigned codes at the DVLA, to separate them by department, to know who is assigned to what. An additional feature the DVLA make use of is the tags on user stories, during the sprint planning meeting, one of the stories 'Issue Type' tag was changed from a story to a Spike as the general consensus was that the issue was very simple to explore and solve. Additionally, DVLA employ the use of a priority marker within Jira, this allows for more efficient planning of tasks, priority hierarchy ranges from high to low.

Sprint goals are assigned to the sprint backlog when the sprint commences, that attempts to encapsulate the sprint under an umbrella task.

User stories follow the 'As a/I want/So that' structure as seen in figure 1b, story points are discussed and agreed upon on during the sprint planning phase using Fibonacci estimates.

Agile Software Engineering Principles

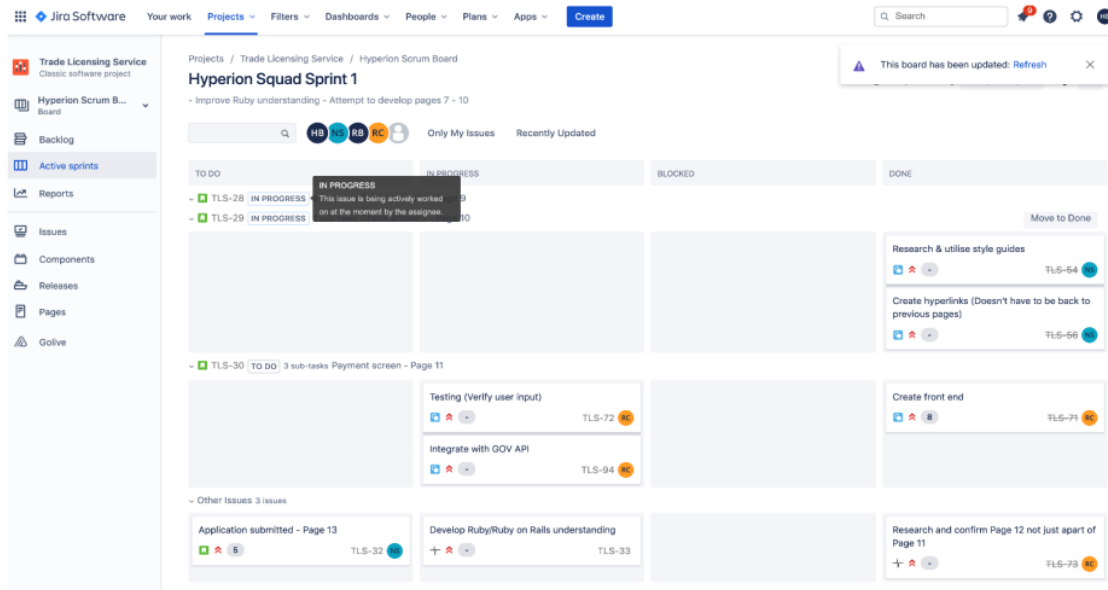


Figure 1a: Sprint backlog on Jira

Figure 1a shows an example of an active sprint backlog, where user stories are assigned to squad members and are moved between sections, To do, in progress, blocked and done. Coloured circles show users assigned to each story.

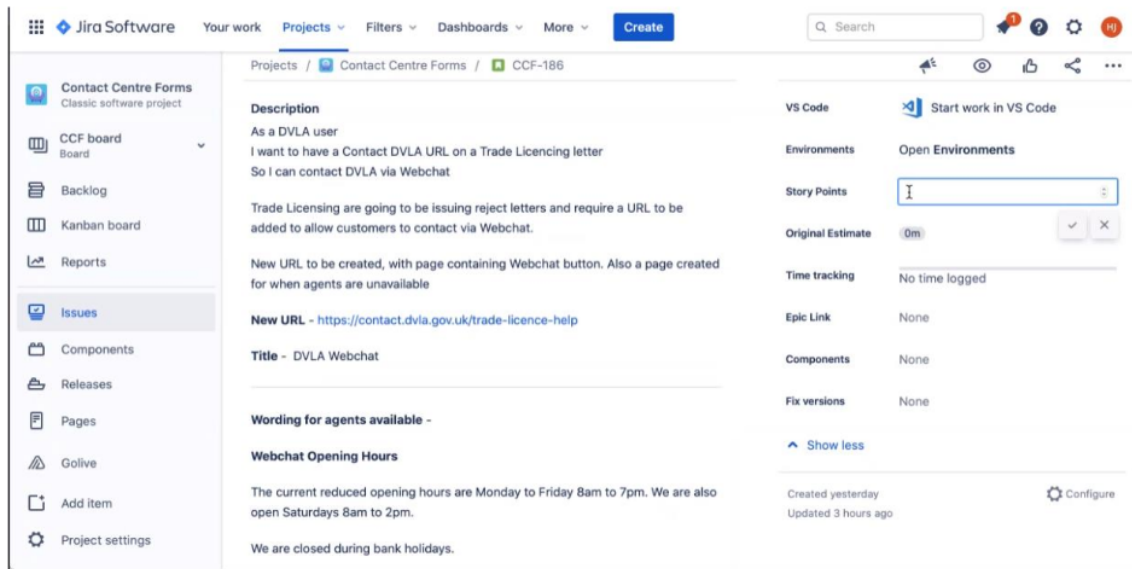


Figure 1b: User story on Jira

Microsoft Teams

Within the DVLA, sprint plans and reviews are held on Microsoft teams, which is an application used for workspace chat, call and video conference capabilities. The structure of the sprint plan meeting within the DVLA is hosted via a voice conference on Microsoft teams. The team is comprised of the squad assigned to that sprint with people from various departments, such as an agile coach, people from quality services and people from testing, to ensure all company standards are captured and met.

During the planning meeting, the sprint backlog, created using Jira, is shared on the screen of the voice conference and various aspects of the backlog and stories it contains are discussed. An example of one story was part of a form for a person to input their age to move onto the next step of the form. The field provided for user input is a date of birth (in DD/MM/YYY format) and the validation selected for user input was that the age result would be between 18 and 99, which was a test requested from the tester.

The story is given a short, punchy title, and a description, in the format 'As a', 'I Want', 'So that', then further details are given for more specifics of that story.

Story sizes are decided amongst the team during the voice call, first the story is discussed and the specifics, who will be responsible for the story. Then, the leader of the call, in this case the Agile coach, will call out that the number should be estimated and counts down, then the team simultaneously enter a number into the chat. The story size is estimated using the Fibonacci sequence method, numbers used are 1,2,3,5,8,13,21, where 1 is the smallest and 21 is the largest. The numbers are then discussed, in one example, most numbers given by the team were 3's, with some 2's and 5's, the call leader will then ask everyone for a happy medium and if they all agree, in this case the medium was 3, in which everyone agreed with.

Docker

The DVLA make use of Docker throughout the development process, which is a containerisation tool, whereby containers bundle their own configuration files and software. This allows branches of code to be committed to and pulled from the repository without the issue of it running fine on one machine but not on another due to missing bundles or packages. This speeds up the development process and follows the founding principles of an Agile software methodology, such as continuous integration, continuous delivery and collaboration.

Bitbucket

For their version control repository, the DVLA utilises Bitbucket, which, similar to Jira, is owned by Atlassian, so they integrate well together. A useful feature within Bitbucket that the DVLA make use of is the approval requirements for pull requests. The software engineering teams at the DVLA have operating procedures, however, most agree on using minimum of 2 approvers for pull requests within Bitbucket. This allows them to have 'two pairs of eyes to check code is correct and conforms to their standards'.

The Agile methodology is applied to this function during the daily stand-up, where team members will request each other to validate and approve their code. Also this is brought up during the round table section of the daily stand-up, where the 3 questions are posed ('what did you work on yesterday, 'what are you working on today' and 'any issues or blockers'), without this methodology it is possible something as minute as asking for an approval of a pull request may go be missed. This could, however, impact the entire sprint and workflow, this is a strong point of the Agile methodology.

Another feature within Bitbucket the DVLA make use of is comments for pull requests (i.e. 'Could you change this code so that...'). However, the DVLA software development squads recognise that it is more effective to have a voice or video call to avoid communication errors.

During the review process of pull requests, it is possible for the reviewer to change the status to 'needs work'. This will stop the pull request from being approved until an issue is resolved or discussed further, however the teams at the DVLA recognise that this could also result in communication errors, and instead recommends a voice or video conference (or face to face chat when working in the office). This potential issue is also resolved using the Agile methodology, whereby any issues or concerns with pull requests can be discussed face to face during the daily stand-up.

Testing - Drone.io

Drone is a tool employed by the DVLA when committing code as part of the continuous integration (CI/CD where CD stands for Continuous delivery) process of the agile methodology. Drone allows for testing during committing to the repository, the pipeline is executed inside an isolated docker container which is downloaded automatically during runtime. This drone file is automatically triggered anytime a push to branch is executed.

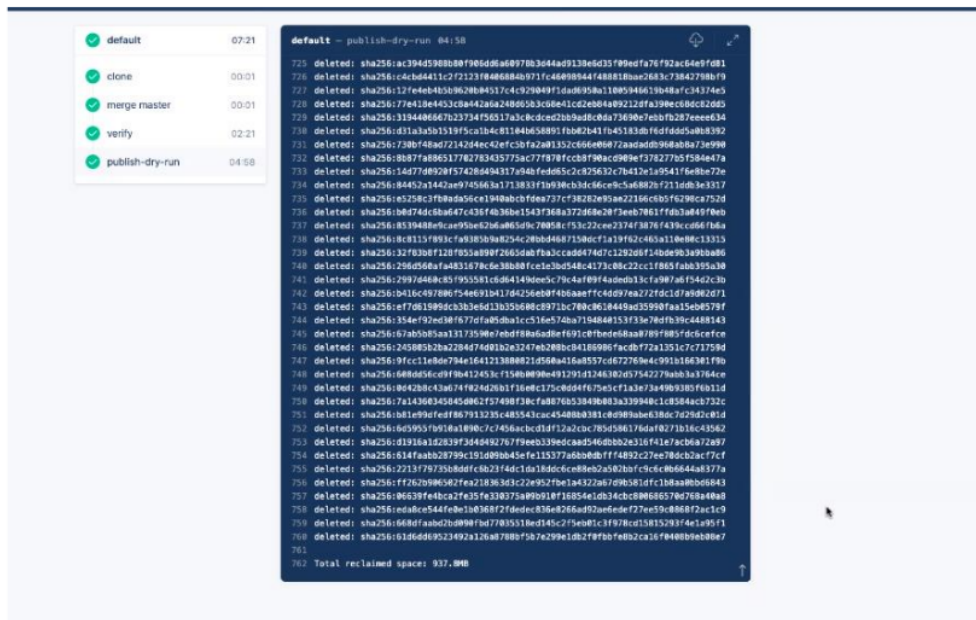


Figure 2a: All drone tests passed

As seen in figure 2a, the merge is cloned, merged to the master, verified then published, all four steps must be passed to allow the code to be merged to the master. Figure 2b shows an active drone test, where a slice of CSS code to restore a comment block to a page needs to be merged to the master branch.

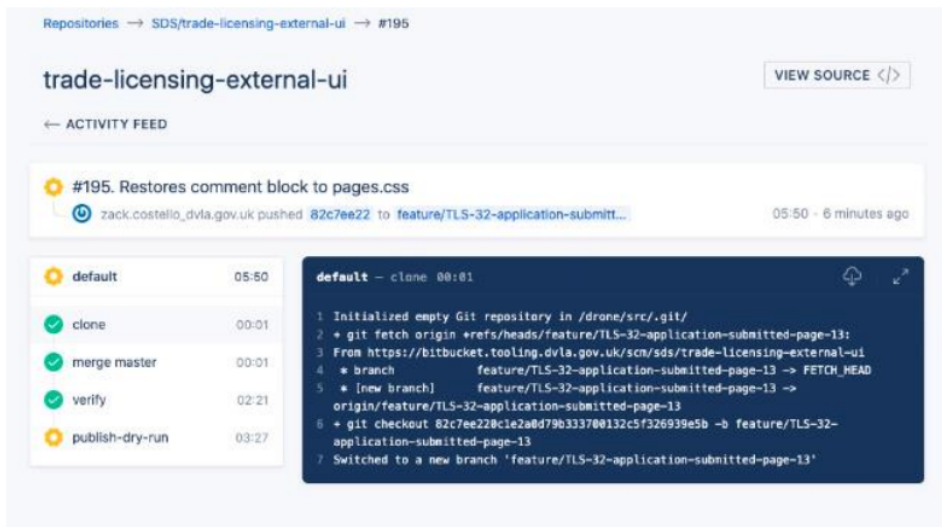


Figure 2b: Active drone testing

Testing - Cucumber

During the Agile process of software development, the project is in a constant state of change and so behaviour driven development is essential, especially during the testing process. The testing tool employed by the DVLA is Cucumber, which runs automated acceptance tests and is carried out using plain-text descriptions of functions in a language known as Gherkin. Cucumber has been identified as a good bridge between technical and non-technical team members, as it provides transparency about the test scenarios and how they are executed [11]. However, a weakness of this approach is that test scenarios have to be generated manually, which can result in a weak test process [11]. Cucumber is an emerging and growing testing tool, which mainly focuses on acceptance testing, it has been identified as a powerful tool that solves a lot of challenges during the testing process of the agile methodology [12].

Pair programming & Collaboration

During a sprint, DVLA employs the use of pair programming, whereby some user stories may require more than one individual to work on simultaneously. Research into pair programming has suggested it holds many benefits for students on computer science courses and can be applied to work environments [13]. The need for pair programming is discussed during the sprint kick off meeting, when the size of the user story is determined.

Also, user stories or sub-tasks may not seem as simple as initially estimated, this is discussed during the daily stand-up, where one member of the scrum team may be struggling to implement a piece of

code, they can then assign this sub-task to another team member, whilst still working on the same bit of code using a branch. This is a useful technique as it plays on the individual strengths of team members. Using this method, both subtasks can be worked on simultaneously, improving efficiency and quality of product.

Another method employed through the DVLA agile process is mob sessions, which function similar to pair programming but with more squad members and has been described as programmers working together in the same time and space, discussing and writing code and solutions together on one screen. [14] Mob programming research has shown it creates a strong bond between team members and feeds from the strength of each members experience, however it has also been shown to irritate some team members [14].

Go Reflect – Sprint Retrospective

An important part of the Agile methodology and continuous integration is the Sprint retrospective. The sprint retrospective is a chance for the squad to look back on the previous sprint and analyse how they worked, and how they could improve in future sprints. The DVLA employed the use of an online team board tool called goReflect.

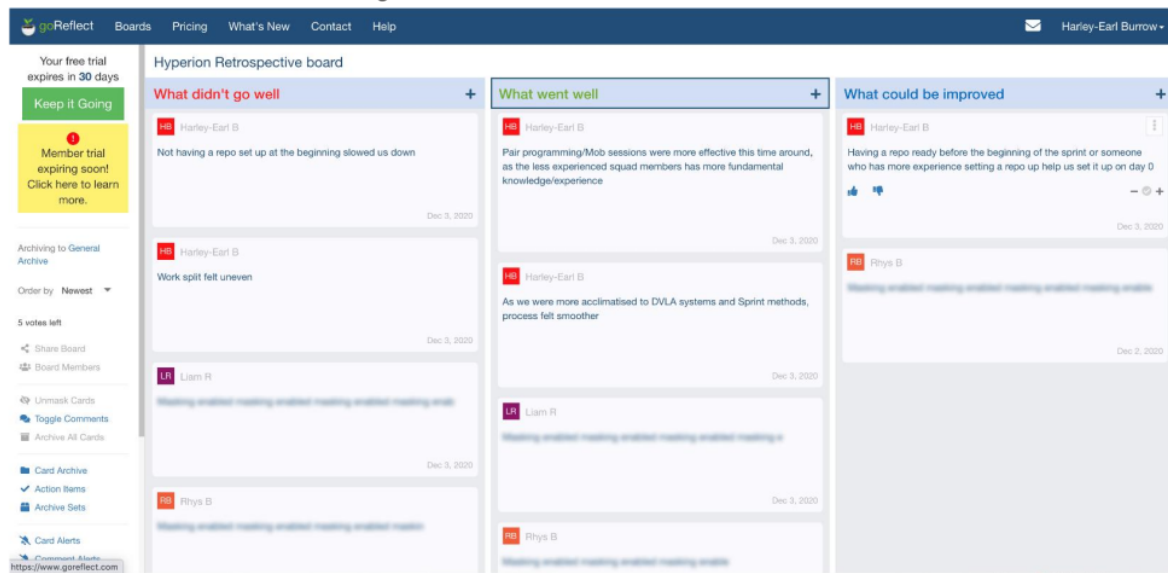


Figure 3a: goReflect board before sprint retrospective with only one users' submissions visible

Prior to the retrospective meeting, team members are asked to add notes to the goReflect board split into sections, namely; 'what went well', 'what didn't go well', and 'what could be improved'. These notes are not revealed until the meeting so each team member can only see what they have

submitted (see figures 3a, 3b & 3c). A sprint retrospective was used at the end of both sprints, and it was evident that the retrospective at the end of sprint 1 increased productivity for sprint 2.

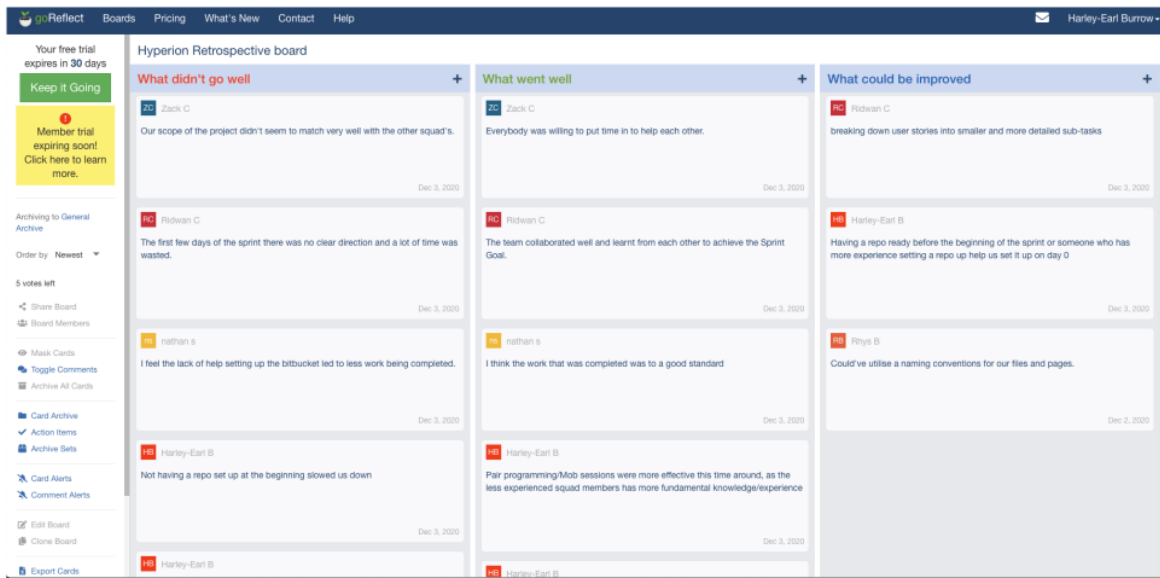


Figure 3b: goReflect board during sprint retrospective with all squads input revealed

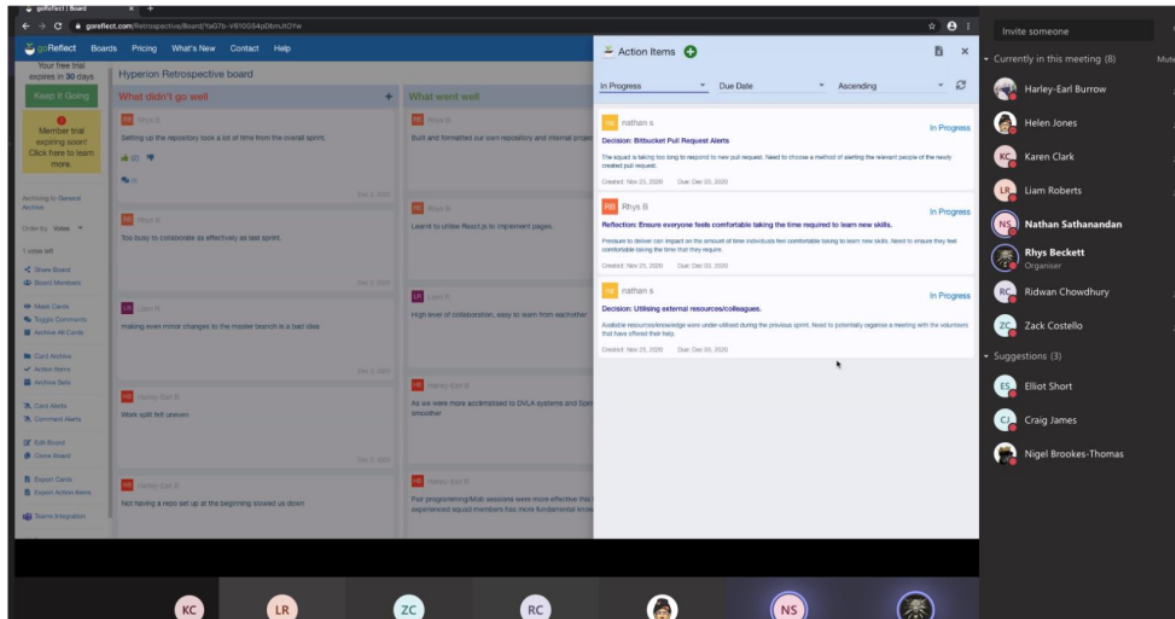


Figure 3c: goReflect board being shared in Microsoft teams during sprint review

Conclusion

On reflection of personal experience during two sprints at the DVLA applying the Agile methodology, it is evident that during the first sprint with a team made up of people on their first sprint, it is difficult to estimate story points accurately, with no frame of reference. A more experienced squad can more accurately estimate how long a User story will take, and how many story points they can complete in one sprint. However, for a team on their first sprint, breaking up the tasks into User stories is more difficult to determine the weighting, an experienced squad may assign a 3 to a user story, whereas the new squad may assign a higher number due to inexperience.

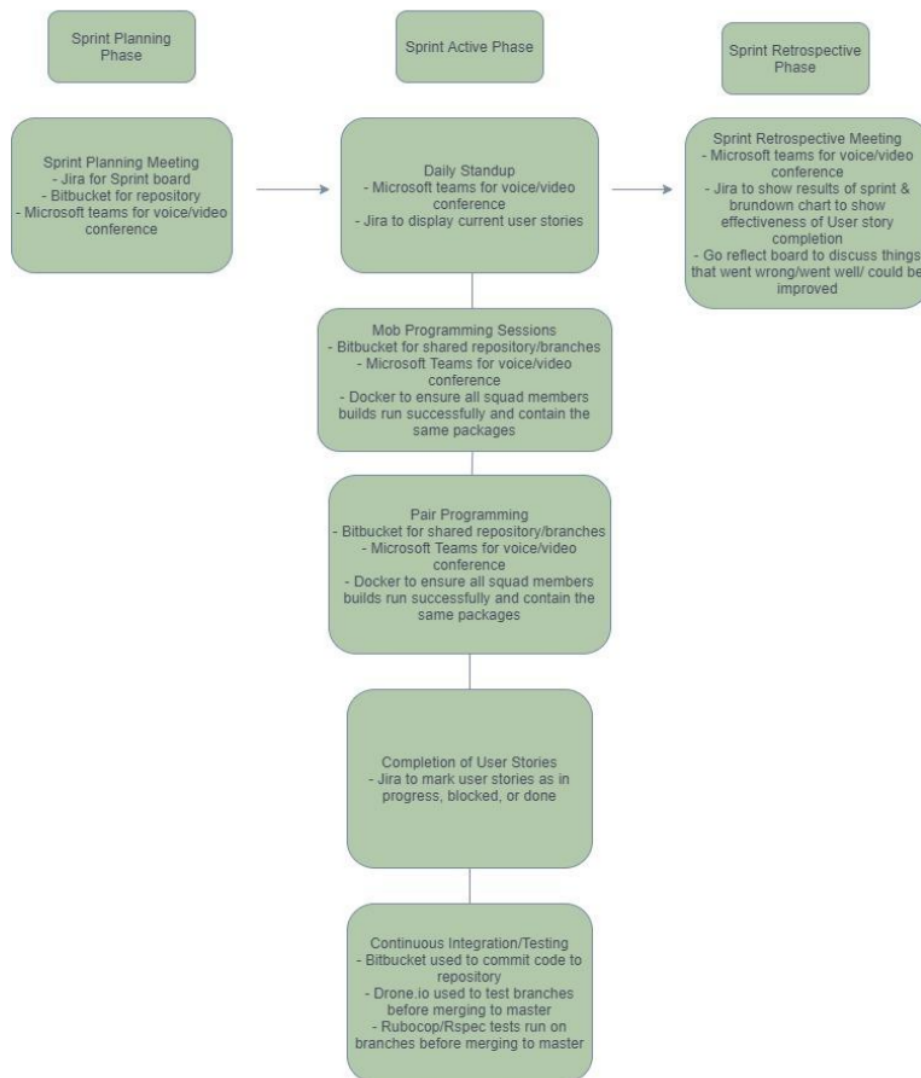


Figure 4a: Agile process and technologies implemented by DVLA

This is likely an anomaly however, as it is rare a development squad will be made up of an entirely new team. Additionally, this is overcome in the DVLA by having an average number of story points per sprint based on previous sprint averages, which is viewed on the burndown chart.

A suggested weakness of the agile methodology is the interaction between the user or customer and the developers not being enough [11], this was apparent in the DVLA where the developers were not sure of what the business analysts needed and so the user story estimation was hindered. Another weakness in the methodology is the difficulty coordinating between teams for large projects [11], this was apparent during the sprints in the DVLA where there were two squads working toward a large project but sometimes struggled to coordinate work allocation.

The testing processes employed by the DVLA with the Agile framework in mind, like any process have their strengths and weaknesses, such as manual generation of tests and mainly focusing on acceptance testing. However, both the Cucumber testing process and the use of Drone appear to have strengths that outweigh their weak points, such as Cucumber being written in an omnipresent language (Gherkin) that can be easily understood by testers, developers, business analysts and customers [12].

A strong point of the Agile methodology employed by the DVLA is the User stories, when compared with other methodologies for estimating work, such as Banana development and requirements decomposition, Scrum resulted in the most effective processes [15]. This was evident in the DVLA workflow, where user stories estimation was mostly accurate and effective.

The technologies employed by the DVLA applying an Agile methodology appear to be streamlined and efficient overall, the core fundamentals of the Agile methodology are applied across various technologies and software. Though they are not without their weaknesses, further evaluation is required as an ever-changing sector grows, and new technologies are adopted and adapted. It is apparent from two project sprints at the DVLA however, that the agile methodology is currently effective in satisfying product owners and business analysts needs in a timely, cost effective manner.

Additionally, research has shown developers face trade-offs between features such as speed and assurance during a continuous integration process [16]. This was apparent during the sprints in the DVLA, where developers had to decide between speed of completing user stories, so the burndown chart was kept in check, and being sure a feature they implemented was completely functional and how the business analysts had requested.

Finally, the importance of continuous integration and testing processes were largely apparent during the agile sprints at the DVLA. Developers were continuously feeding into the repository and continuously running tests on builds using the DVLA technologies. The agile methodologies and use of continuous integration proved effective in delivering the product to the product owner, in a steady, manageable way, with effective testing techniques.

References

1. "Vehicle Licensing Statistics: Annual 2019", *Assets.publishing.service.gov.uk*, 2019. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/882196/vehicle-licensing-statistics-2019.pdf. [Accessed: 13- Nov- 2020].
2. G. Kumar, P.K. Bhatia. Impact of Agile Methodology on Software Development Process. *International Journal of Computer Technology and Electronics Engineering (IJCTEE)*. vol. 2 no.4, p.46-50, 2012
3. M. Ochoodek, S. Kopczynska. Perceived importance of agile requirements engineering practices – A survey. *Journal of Systems and Software*. vol. 143, p.29-43, 2018.
4. S.M. Mohammad. DevOps automation and Agile Methodology. *International Journal of Creative Research Thoughts (IJCRT)*. vol.5, no.3, p.946-949, 2017.
5. Khalid, A., Butt, S. A., Jamal, T., & Gochhait, S. (2020). Agile Scrum Issues at Large-Scale Distributed Projects: Scrum Project Development At Large. *International Journal of Software Innovation (IJSI)*, vol. 8, no.2, p.85-94. doi:10.4018/IJSI.202004010
6. N. Bolloju, S. Alter, A. Gupta, S. Gupta and S. Jain, "Improving Scrum User Stories and Product Backlog Using Work System Snapshots", *SYSTEMS ANALYSIS AND DESIGN (SIGSAND)*, vol. 23, no. 1, pp. 1-8, 2017. [Accessed 10 November 2020].
7. Srivastava, A., Bhardwaj, S. and Saraswat, S., 2017. *SCRUM Model For Agile Methodology - IEEE Conference Publication*. [online] ieeexplore.ieee.org. Available at: <<https://ieeexplore.ieee.org/abstract/document/8229928>> [Accessed 10 November 2020].
8. M. Cohn, *User stories applied*. Addison-Wesley Professional, 2004, pp. 4-8.
9. Ormsby, M., & Busby-Earle, C. (2019, May). Agile Scaled Steps of Doneness: A Standardized Procedure to Conceptualizing and Completing User Stories Across Scrum Teams and Industries. In *International Conference on Evaluation of Novel Approaches to Software Engineering* pp. 364-377
10. Mohammad, A.H., & Alwada'n, T. (2013) Agile Software methodologies: Strength and Weakness. *International Journal of Engineering Science and Technology*, vol. 5, no.3, pp455.
11. Li, N., Escalona, A., & Kamal, T. (2016). Skyfire: Model-based testing with cucumber. In *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)* (pp. 393-400).
12. Rehman, A. U., Nawaz, A., & Abbas, M. (2020). Agile Methods: Testing Challenges, Solutions & Tool Support. *arXiv preprint arXiv:2008.13108*.
13. Rodríguez, F. J., Price, K. M., & Boyer, K. E. (2017). Exploring the pair programming process: Characteristics of effective collaboration. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* pp. 507-512.
14. Kattan, H. M., Oliveira, F., Goldman, A., & Yoder, J. W. (2017). Mob programming: the state of the art and three case studies of open source software. In *Brazilian Workshop on Agile Methods* (pp. 146-160).
15. Anwer, F., Aftab, S., Shah, S. M., & Waheed, U. (2017). Comparative Analysis of Two Popular Agile Process Models: Extreme Programming and Scrum. *International Journal of Computer Science and Telecommunications*, 8(2), 1-7.
16. Hilton, M., Nelson, N., Tunnell, T., Marinov, D., & Dig, D. 2017. Trade-offs in continuous integration: assurance, security, and flexibility. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering* (pp. 197-207).

Agile Software Engineering Principles - Assignment 1

GRADEMARK REPORT

FINAL GRADE

67/100

GENERAL COMMENTS

Instructor

Agile Process Evaluation (40): 26

Examples (30): 22

Testing (30):19

Discussion:

The agile process as applied to the DVLA is discussed in quite good detail.

Examples also look good.

More background research could have been provided on CI and testing.

More references required;

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16
