

Report An Analysis Of Agile Development Practices At The DVLA

by Finchley Alexander Harwood

FILE	23673_FINCHLEY_ALEXANDER_HARWOOD_REPORT_AN_ANALYSIS_OF_AGI LE_DEVELOPMENT_PRACTICES_AT_THE_DVLA_302864_346681333.PDF (1.14M)		
TIME SUBMITTED	03-DEC-2020 05:18PM (UTC+0000)	WORD COUNT	4715
SUBMISSION ID	138219601	CHARACTER COUNT	24597

Report: An Analysis Of Agile Development Practices At The DVLA

Finchley A. Harwood

School of Applied Computing

University of Wales Trinity St. David

November 2020

Abstract

In recent years, the Driver and Vehicle Licensing Agency (DVLA) have moved towards becoming a 'Centre of Digital Excellence', and as part of this they have adopted an Agile approach to software development. Beginning with an introduction of Agile methodologies and why they are effective in modern software development settings, an analysis of the DVLA's implementation of the 'Scrum' framework is given. This is followed by a discussion of collaboration, and a reflection on the importance of continuous integration and testing practices. At the time of writing, the COVID-19 pandemic has meant that DVLA staff must work remotely, this has provided a unique opportunity to analyse the strengths of the DVLA's Agile practices under a testing set of circumstances.

Contents

1	Introduction	3
2	Overview: Agile Methodologies	3
3	Overview: Scrum Framework	4
4	User Stories And Their Estimation	5
4.1	User Stories	5
4.2	User Story Estimation	6
4.3	Analysis Of An Example User Story	7
4.4	Analysis Of A Testing Example	7
5	Product Backlogs	8
5.1	The Product Backlog	8
5.2	Product Backlog Refinement	8
5.3	Analysis Of An Example Definition Of Ready	9
6	Sprint Backlogs	10
6.1	The Sprint Backlog	10
6.2	Analysis Of An Example Sprint Backlog	10
6.3	Analysis Of The Estimation Of A User Story	12
7	Collaboration	13
8	Continuous Integration and Testing Processes	13
9	Discussion	14
10	Summary	15

1 Introduction

The secret to effectively running projects has long been sought by mankind. To be successful, ancient architects and engineers responsible for designing and building structures such as Stonehenge had to think carefully about the underlying project processes from design through to implementation and delivery [1]. In the modern age, although the scope of projects has changed, the goal of effectively running projects remains.

The Waterfall methodology [2] [3] was introduced in the later half of the 20th century as a framework for managing software development projects. Waterfall aimed to apply project management practices that had been used in the manufacturing industry to software development [2]. By the late 1990s, the world was becoming increasingly reliant on the development of complex software to drive progress and it was clear that super-imposing traditional project management ideas onto software development was ineffective.

The following report will analyse and evaluate the implementation of Agile software development at the UK Government's Driver and Vehicle Licensing Agency (DVLA). In recent years, the DVLA have moved towards becoming a 'Centre for Digital Excellence', with the ultimate goal of having an in-house Information Technology department at the forefront of Government [4].

2 Overview: Agile Methodologies

Traditional methodologies, such as Waterfall, rely on writing a complete set of requirements before starting the design, development and implementation of software [5]. This approach works well if the business environment is "stable", but when requirements are constantly evolving, such an approach can become "overwhelmed" [6].

To help to understand the difference between Agile and Waterfall, Highsmith [6] uses the analogy of a commander on a battlefield. No modern general would plan everything before a battle and then expect the battle to play out just as they had planned. Instead, they recognise that the plans they make must adapt to change through *iteration*.

The agile development philosophy is summarised by the "Agile Manifesto" [7]. The methodology values "individuals and interactions over processes and tools", "working software over comprehensive documentation", "customer collaboration over contract negotiation" and "responding to change over following a plan" [7]. This philosophy leads to what Harwood and Kent [8, p. 43] describe as a development process that "[reduces] the risk of delivering a fault by... minimising its ability to remain undetected".

An Agile Development framework is an implementation of the principles outlined in the Agile Manifesto [9]. Some examples of Agile frameworks include ‘eXtreme Programming’, ‘Kanban’ and ‘Scrum’ [10]. The Scrum framework is the most prevalent framework used at the DVLA.

3 Overview: Scrum Framework

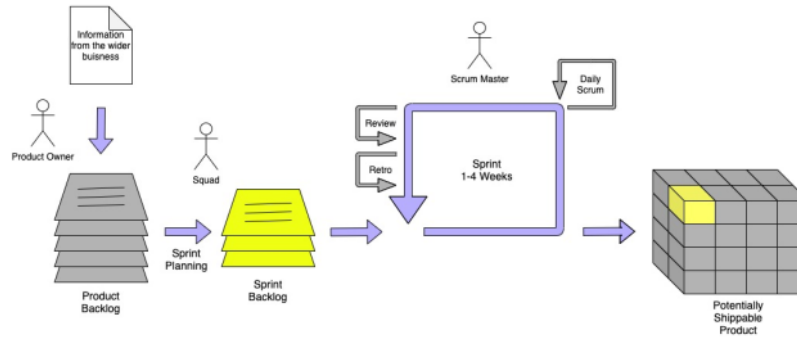


Figure 1: A diagrammatic overview of the Scrum framework, created by the author. This diagram aims to present the ‘pipeline’ through which requirements flow, from discussions between the business and Product Owner through to a ‘potentially shippable product’.

The Scrum Framework, see Figure 1, is focused around time-boxed development iterations called ‘Sprints’, which last between 1 and 4 weeks [11]. The goal of each sprint is to deliver a so-called ‘potentially shippable product increment’ to the customer [11]. A Scrum Team or ‘Squad’¹ is a small group of self-organising developers who work together to satisfy the sprint goal.

The role of the Product Owner (PO) in Scrum is to represent the customer to the Squad; the PO captures requirements in the form of ‘User Stories’ and places them in a prioritised list called the ‘Product Backlog’ [11]. The DVLA refer to their PO’s as Service Designers (SD), which is a reflection of the fact that the IT services created there are in-house. At the beginning of each Sprint, requirements are placed in the ‘Sprint Backlog’ in a meeting called ‘Sprint Planning’ [13].

¹At the DVLA Scrum Teams are known as ‘Squads’, this is perhaps a reflection of the U.S. Army’s naming convention for what are known as ‘Sections’ in the British Army, a small team of roughly 9 soldiers. This link is possibly not as trivial as one may first think, research appears to suggest that this sized grouping of people is ideal for building an effective team based on commonality and a shared identity, see [12]. There is perhaps a tenuous link between the DVLA’s naming convention and the analogy of battlefield discussed in Section 2.

A ‘Daily Scrum’ meeting, held each day, is facilitated by the ‘Scrum Master’, whose role is that of a servant leader to the Squad [11]. At the DVLA, the Scrum Master role is fulfilled by an ‘Agile Development Manager’ (ADM), whose role also involves that of a Line Manager to the Squad members. Two meetings are held at the end of a sprint, the first of these being the ‘Sprint Review’. This meeting allows the team to present the progress made in the sprint to stakeholders. The second meeting is called the ‘Sprint Retrospective’, which is designed to facilitate the Squad’s learning and improvement through evaluation of the previous sprint [11].

4 User Stories And Their Estimation

4.1 User Stories

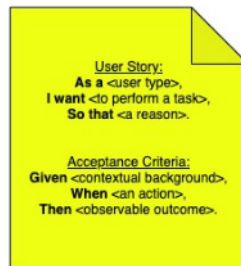


Figure 2: A diagram showing a typical user story, created by the author.

User Stories are a succinct notation for capturing user requirements, based on a user’s non-technical perspective [14]. A User Story consists of three main components: a descriptive piece of text, a record of conversations held between various stakeholders, and a list of acceptance criteria [14]. At the DVLA, the descriptive text follows the format: **As a** [user type], **I want** [to perform a task], **So that** [a reason]. This format is designed to capture only the essential elements of a requirement [14].

A User Story will typically contain multiple ‘Acceptance Criteria’, which are used to test whether the requirements it captures have been met. The DVLA use ‘Gherkin’ notation for writing Acceptance Criteria, this follows the format: **Given** [contextual background (past-tense)], **When** [an action occurs (present-tense)], **Then** [observable outcome (future-tense)]. As with the descriptive text, this format is designed to be understandable to everyone involved in the development process, including the customer [15, Ch 2]. The Gherkin notation can be translated into the Cucumber language, which is then used to develop software

tests [15, Ch 2]. This maintains clarity throughout the development process, by ensuring a direct link between requirements and test cases.

4.2 User Story Estimation

User story *estimation* helps to ascertain how long stories will take to develop. This provides a measurement of how a project is progressing [16]. The DVLA typically use the Fibonacci Sequence in their user story estimation because there is a clear distinction between the numbers of the sequence, avoiding debate over the specific size of a story [16]. The preferred method at the DVLA for user story estimation is that of 'Planning Poker', where Squad members write down their estimate for a story, and then all unveil their estimates at the same time. Thereafter ensues a discussion, and a collective decision on the estimate [16]. This technique helps to avoid members of the Squad being influenced by other member's preconceptions.

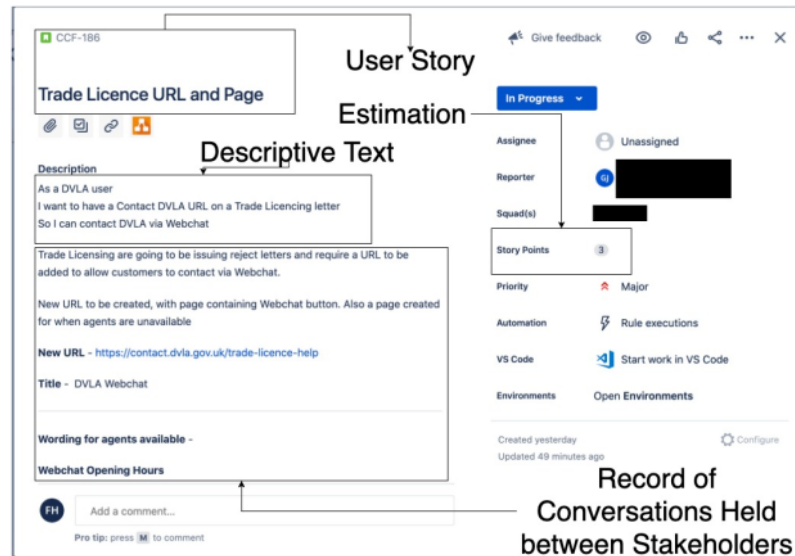


Figure 3: A screenshot taken from Example Squad's JIRA board of a User Story for a web page. The descriptive text, estimation and the record of conversations held between stakeholders are highlighted. This story lacks acceptance criteria, but the text details exactly how the web page should appear.

4.3 Analysis Of An Example User Story

The screenshot of Figure 3 is an example of a DVLA User Story created in JIRA ² (*Jee-ra*) by the PO of 'Example Squad'. The example contains a short piece of descriptive text in the form of **As a - I want - So That** relating to a web page. The story point estimation of '3' suggests that the amount of relative work will be small. Below the descriptive text is a record of conversations held between stakeholders, and other additional information that might be useful to the developer assigned to this story. The example story lacks acceptance criteria written in the form **Given- When - Then**, but in the text there is a detailed explanation of what the web page should look like.

4.4 Analysis Of A Testing Example

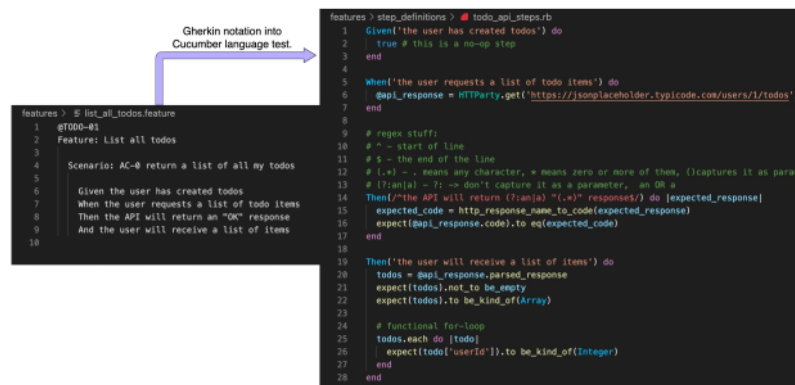


Figure 4: Two screenshots showing the progression of Acceptance Criteria written in Gherkin notation being translated into working Cucumber test code.

The screenshots of Figure 4 give an example of Acceptance Criteria, written in Gherkin notation, and the resulting Cucumber language test. A Gherkin language parser is able to generate the empty Cucumber test code, which is then 'fleshed-out' by the tester. This particular example is to test whether a Json file is returned after a HTTP request.

Cucumber tests coupled with Gherkin notation, facilitate 'Behaviour Driver Development' (BDD) at the DVLA. This is a form of software development that focuses on using Acceptance Criteria to drive development, whilst emphasising

²See Section 7 for more information on collaboration tools used at the DVLA, such as JIRA.

readability [17]. Gherkin and BDD at the DVLA ensures that software better reflects requirements, ultimately leading to a better service for the general public.

5 Product Backlogs

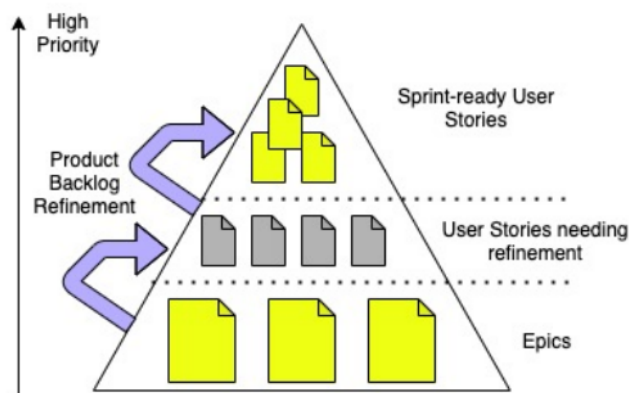


Figure 5: A diagram to illustrate the Product Backlog (created by the Author). The product backlog consists of an ordered-by-priority list of requirements with fully refined user stories at the top.

5.1 The Product Backlog

The Product Backlog consists of complete User Stories, incomplete User Stories waiting for refinement and so-called ‘Epics’, which are larger requirements needing to be broken down into User Stories [18]. The SD has sole responsibility for the Product Backlog, obtaining information from the wider business in order to construct requirements [18]. The Product Backlog takes the form of a *dynamic* list of requirements which evolves over the course of a project. This evolution takes place through the addition, removal, and re-ordering of requirements [18].

5.2 Product Backlog Refinement

The ‘Product Backlog Refinement’ meeting is held between the Squad and the SD [18]. The aim of the meeting is to refine requirements. This is demonstrated in the diagram of Figure 5: Epics are broken down into smaller User Stories, and User Stories are ‘fleshed-out’ by adding details such as Acceptance Criteria. The result of this is the generation of ‘sprint-ready’ user stories. Every Squad at the DVLA have their own Definition of Ready (DoR), which is used to determine

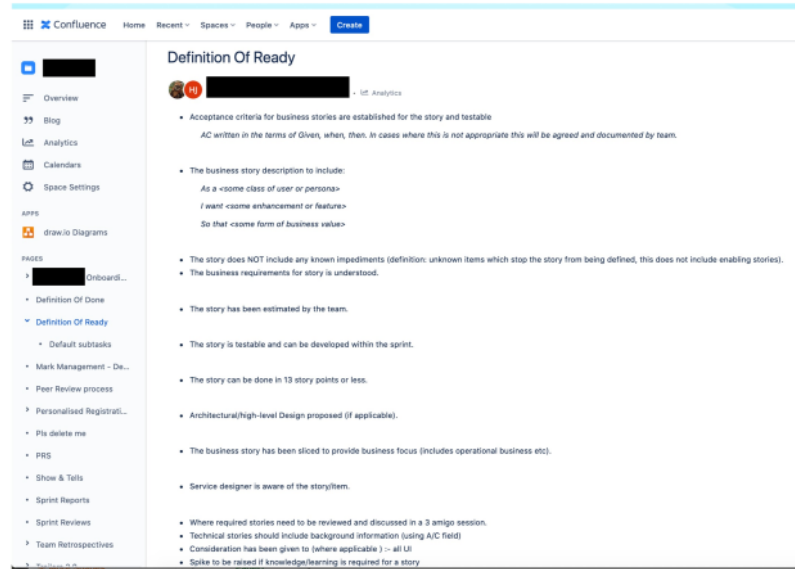


Figure 6: A Screenshot of the DoR for 'Example Squad'.

whether a user story is 'sprint-ready'. The Product Backlog refinement meeting usually occurs once per sprint [19].

5.3 Analysis Of An Example Definition Of Ready

The screenshot of Figure 6 is Example Squad's DoR. There are some interesting points to analyse in this example of a DoR. Firstly, acceptance criteria written in the form of **Given - When - Then** are desired, but when this is not applicable then documentation suffices. This links back to the User Story discussed in Section 4 and why there was no acceptance criteria present. Secondly, the team have decided that a User Story must be 13 points or less, leading to the conclusion that Example Squad consider anything more than this to be an Epic.

Continuing with the analysis of Figure 6, the requirement that a User Story must be discussed in a '3 Amigo's' deserves its own discussion. A '3 Amigo's' Meeting takes place between the SD, a senior Tester and a senior Developer. The main goal of the meeting is to draw upon the combined expertise of the participants in order to refine requirements to a higher level of testability [20].

6 Sprint Backlogs

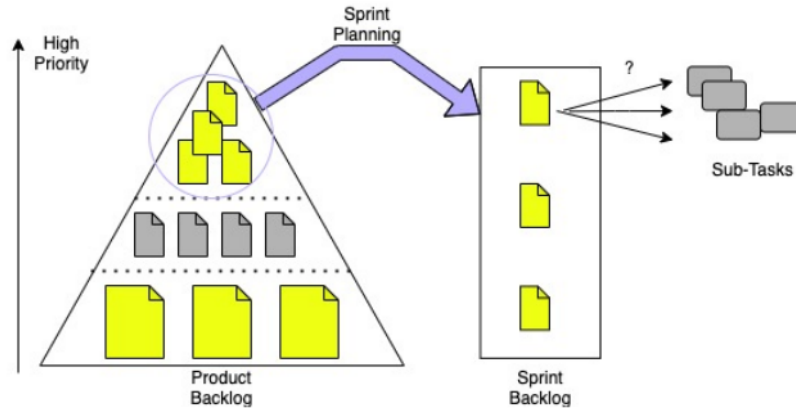


Figure 7: A diagram to illustrate the sprint planning meeting, where high-priority User Stories are estimated and placed in the Sprint Backlog. Sometimes User Stories are broken down into sub-tasks (as demonstrated here with a question mark). Diagram created by Author.

6.1 The Sprint Backlog

The ‘Sprint Backlog’ consists of a plan, a list of items from the Product Backlog, and a goal for the current sprint. The Sprint Backlog is generated in the Sprint Planning meeting, where high priority, sprint-ready User Stories are pulled from the Product Backlog [21]. This meeting occurs at the beginning of each Sprint, and involves User Story estimation, and (often) the breaking down of User Stories into smaller sub-tasks [21].

In its most basic form, the Sprint backlog is displayed as three columns labelled ‘To Do’, ‘Doing’ and ‘Done’, as seen in the diagram of Figure 7. User stories begin in the ‘To Do’ column. The Story (or sub-task) then moved across the board. A User Story can be moved to ‘Done’ when it meets the Squad’s Definition of Done (DoD); the DoD provides a gauge against which User Stories can be tested for completion [22].

6.2 Analysis Of An Example Sprint Backlog

The Screenshot of Figure 9 is an example of a sprint backlog board for the DVLA’s ‘Example Squad’. In this example, the Squad have decided to add an extra ‘Testing’ column. The extra column helps to give a clearer picture of the state of the sprint, and is useful when a different person is carrying out testing.

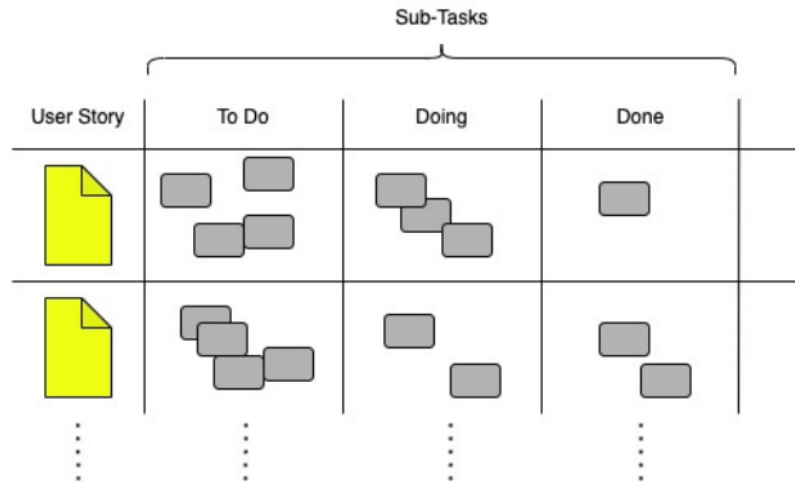


Figure 8: A diagram to illustrate the Sprint Backlog. In this case User Stories have been broken down into sub-tasks, which are then moved across the board. Diagram created by author.

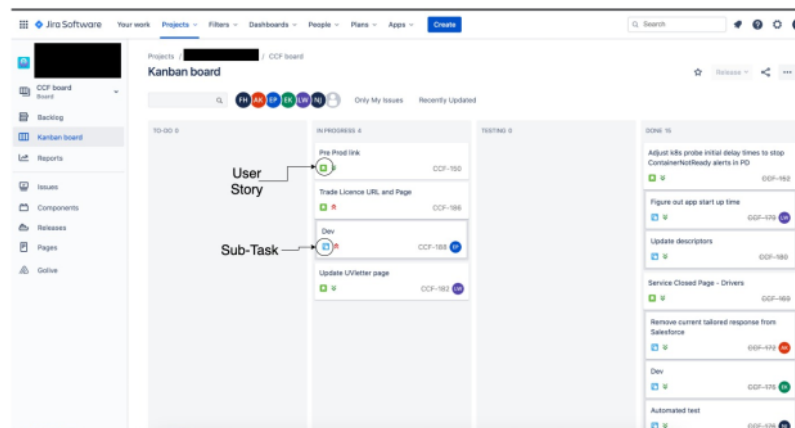


Figure 9: A screenshot of Example Squad's Sprint Backlog. Highlighted are an example of a User Story and a Sub-Task, with the Sub-Task being a 'Dev' task.

As highlighted in the Screenshot of Figure 9, some User Stories have been split up into smaller sub-tasks. The user story is only moved to the 'Done' column when all of its sub-tasks have been completed. Sometimes a Squad may

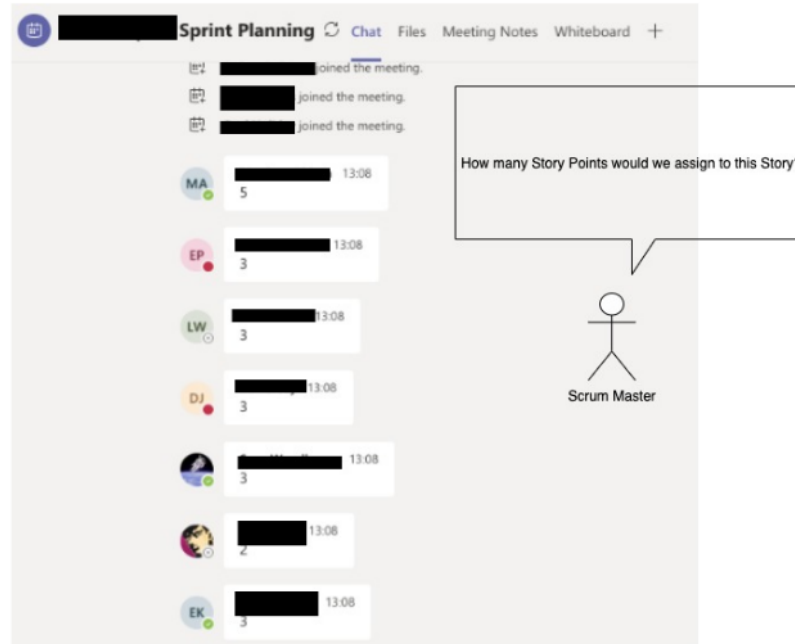


Figure 10: A screenshot taken from Example Squad’s Sprint Planning meeting, facilitated by Microsoft Teams. The Scrum Master asks the members of the Squad to prepare a story point estimation in the chat box. Then, when ready, all members of the Squad send their estimates at the same time.

have default sub-tasks such as ‘Dev’, ‘Test’ or ‘Review’ which are automatically assigned to stories in JIRA. This is especially useful when a Squad always follows a certain work-flow. The Screenshot of Figure 9 contains examples of both automatically and non-automatically assigned sub-tasks.

6.3 Analysis Of The Estimation Of A User Story

Figure 10 is an example of User Story point estimation being carried out in a Sprint Planning meeting. The meeting was hosted in ‘Microsoft Teams’. Firstly, a discussion was held between the SD and the Squad as to which User Stories could be completed in the upcoming sprint. The ADM then went through the User Stories and asked the Squad to prepare their estimation in the chat box. Then, when ready, everyone sent their estimation. In this case the majority of developers picked ‘3’ for their estimation of the User Story (which happened to be the one mentioned in Section 4). The following discussion led to the developers agreeing on an estimation of ‘3’ for the User Story. After every User

Story had been estimated, a total User Story estimation of 68 was arrived at. The ADM asked if the development team was happy with this, or whether they wanted to remove or add stories, since the average Sprint velocity³ of the squad had been 60 in recent weeks.

The example is a nice demonstration of the DVLA implementing the principles of the Agile Manifesto (see Section 2). The SD and the Squad are *responding to change over following a plan* by discussing the highest priority User Stories before commencing the planning session. In addition to this, the fact that it is ultimately the Squad who decide for themselves how much work they will do, and how much work they are capable of, is a demonstration of the DVLA valuing *individuals and interactions over processes and tools*.

7 Collaboration

At the time of writing, the world is in the grips of the COVID-19 pandemic. In order to control the spread of the virus, the majority of employees at the DVLA are working from home. Therefore, the Squads at the DVLA are reliant on online software for collaboration.

Remote collaboration is facilitated through a number of ways at the DVLA. Microsoft Teams replaces face-to-face meetings: it is used for scheduling meetings, sending private messages and conducting video conferences. JIRA replaces the planning whiteboard: this is used for the construction of User Stories, and as a virtual Product Backlog, Sprint Backlog. Confluence contains information about specific Squads, such as DoD's and DoR's and, due to there being no restrictions on access to Confluence boards, it promotes openness within the organisation (the same can be said for JIRA). Bitbucket is used for version control and remote repository hosting. A service called 'Sketch' is used for UI design.

While some of aforementioned tools were used before the pandemic, the pandemic has increased the need for a set of high quality tools. To that effect, the tools are integrated with one-another to varying degrees. For example, one is able to generate a new feature branch in Bitbucket through the JIRA UI.

8 Continuous Integration and Testing Processes

Continuous Integration and Testing (CIT) is the practice of regularly uploading tested code to a central repository [23]. In the case of the DVLA, Bitbucket is the repository hosting tool of choice. A new BitBucket feature branch can be

³The average Sprint velocity of a squad is the average number of User Story points completed per Sprint in the previous few Sprint. The average velocity of the Squad helps to provide a rough guide as to how many User Story points can be completed in the upcoming sprint.

generated from a JIRA User Story. This feature branch is then cloned onto the personal machine of the Squad member working on the User Story. The usual practice is to run the code within 'Docker'; Docker is a containerisation software, which ensures that code, and dependencies remain platform independent.

A tool called 'drone.io' is used to automatically check code when it is pushed to the Bitbucket repository. Drone executes a custom DVLA script that runs functional tests, unit tests and lints the code (ensures it follows naming conventions and the DVLA style). This script is automatically run on Bitbucket through the use of a 'webhook'. If a Squad member wishes to merge their changes into the master branch of the repository, their code must have passed all of the drone checks and then been approved by at least two other members of the Squad.

The Docker image of the master branch is regularly pushed to a so-called 'link environment'. This is working code on a URL accessible from any DVLA registered computer which can be used by departments such as User Experience Design for user testing. This facilitates iterative improvement through user and business feedback. The link environment is hosted using the tool 'Spinnaker'.

The importance of CIT at the DVLA cannot be overstated. That the code is automatically checked every time it is pushed to the central repository and that it must pass these checks and then be approved before it can be merged into the master branch, forms the 'backbone' of high quality, bug-free code at the DVLA. Besides, CIT is key to working in an Agile way. Every time a sprint is finished, one can be assured that the code produced will be of a quality high enough to be considered 'shippable'.

9 Discussion

At the Squad level, the DVLA is completely Agile oriented, with little trace of any other form of methodology. Scrum is the most prevalent framework. Although the DVLA have their own unique 'flavour' of Scrum, with roles such as ADMs and the addition of the not-so-common '3-Amigos', the agency appear to have, on the whole, adopted the framework almost to the letter.

The DVLA is becoming a bastion of knowledge in South Wales when it comes to Agile methodologies. Agile allows the DVLA to remain current and highly adaptive to change. This is hugely important for the functioning of the entire country, especially with changes such as Brexit and COVID-19. For example, the DVLA are rolling out new services such as an online Trade Licensing service to reduce the amount of people required to be in the building to process the current paper service. The use of the Scrum framework has allowed a prototype of this service to be quickly constructed; the user testing of the prototype will help inform the requirements for the first production version iteration.

One could consider the Pandemic to be a good ‘stress test’ of the Agile practices at the DVLA. At the time of writing, due to a good set of tooling and understanding of Agile across the business, the DVLA have managed the transition from in-person to remote working well.

Madadipouy [24], suggests that Scrum should only really be used for short term projects, with an ideal time frame of each new release of software being about 30 days. This is perhaps because Scrum can be susceptible to ‘Feature-Creep’, especially when the requirements are not clearly defined at the outset [25]. It is quite possible that Scrum is not the ideal Agile Framework for the DVLA, however issues such as Feature-Creep are perhaps less of a problem when the customer is the business itself. Presumably the business finds it quite easy to define exactly which features they wish to have in certain services, especially when those services already exist, albeit in a paper format.

At the DVLA, ADMs replace Scrum Masters, serving as Line Managers for the Squad in addition to the servant leader role of Scrum Masters. The practice of using Line Managers for this role would seem to go against the idea of a Servant Leader. It is quite possible that DVLA Squads have an unintended less-open and undemocratic atmosphere as a result.

10 Summary

Agile methodologies are a response to the challenges of modern software development. The Scrum framework is an implementation of the principles laid out in the ‘Agile Manifesto’, and is the framework of choice in most ‘Squads’ at the DVLA. The DVLA have their own unique blend of Scrum with roles such as ADMs and the ‘3-Amigos’ meeting. The former of these perhaps leading to a less ‘democratic’ implementation of Scrum.

Requirements in the form of User Stories are directly connected to software test cases through the use of Gherkin notation and the Cucumber language. Tests and software linting are automatically run on code pushed to ‘Bitbucket’ through the use of ‘drone.io’. This forms the ‘backbone’ of high quality bug-free code at the DVLA. At the DVLA, combining testing with continuous integration is key to working in an Agile way.

The DVLA have been able to adapt to challenges such as Brexit and the COVID-19 pandemic, due to the ability of Agile methodologies to evolve to changing requirements. This ability to adapt will no doubt serve the agency well in their goal to stay relevant and create an in-house IT department at the forefront of Government.

References

- [1] T. Seymour and S. Hussein, "The history of project management," *International Journal of Management & Information Systems (IJMIS)*, vol. 18, no. 4, pp. 233–240, 2014.
- [2] W. Van Casteren, "The waterfall model and the agile methodologies: A comparison by project characteristics," *Research Gate*, February, 2017.
- [3] W. W. Royce, "Managing the development of large software systems: concepts and techniques," in *Proceedings of the 9th international conference on Software Engineering*, 1987, pp. 328–338.
- [4] M. Williams, *DVLA set to become a Centre of Digital Excellence*, 2018 (accessed November 11, 2020), <https://dvladigital.blog.gov.uk/2018/07/02/dvla-set-to-become-a-centre-of-digital-excellence/>.
- [5] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods." *Adv. Comput.*, vol. 62, no. 03, pp. 1–66, 2004.
- [6] J. Highsmith and A. Cockburn, "What is agile software development?" *The Journal of Defense Software Engineering*, vol. 15, no. 10, pp. 4–9, 2002.
- [7] M. Fowler, J. Highsmith *et al.*, "The agile manifesto," *Software Development*, vol. 9, no. 8, pp. 28–35, 2001.
- [8] R. Harwood and G. Kent, *An Agile Development Framework for Business Analysts*. Cutter Consortium, 2019, <https://bookstore.cutter.com/product/an-agile-development-framework-for-business-analysts/>.
- [9] R. Hoda, J. Noble, and S. Marshall, "Agile project management," in *New Zealand computer science research student conference*, vol. 6, 2008, pp. 218–221.
- [10] L. Lindstrom and R. Jeffries, "Extreme programming and agile software development methodologies," *Information systems management*, vol. 21, no. 3, pp. 41–52, 2004.
- [11] M. Hron and N. Obwegeser, "Scrum in practice: an overview of scrum adaptations," in *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.
- [12] W. D. Henderson, *Cohesion, the human element in combat: leadership and social influence in the armies of the Soviet Union, the United States, North Vietnam, and Israel*. DIANE Publishing, 1985.
- [13] R. Popli and N. Chauhan, "Scrum: An agile framework," *International Journal of Information Technology and Knowledge Management*, vol. 4, no. 1, pp. 147–149, 2011.
- [14] G. Lucassen, F. Dalpiaz, J. M. E. van der Werf, and S. Brinkkemper, "Improving agile requirements: the quality user story framework and tool," *Requirements Engineering*, vol. 21, no. 3, pp. 383–403, 2016.
- [15] M. Wynne, A. Hellesoy, and S. Tooke, *The cucumber book: behaviour-driven development for testers and developers*. Pragmatic Bookshelf, 2017.
- [16] H. Zahraoui and M. A. Janati Idrissi, "Adjusting story points calculation in scrum effort time estimation," in *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*, 2015, pp. 1–8.
- [17] C. Solis and X. Wang, "A study of the characteristics of behaviour driven development," in *2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications*. IEEE, 2011, pp. 383–387.
- [18] T. Sedano, P. Ralph, and C. Péraire, "The product backlog," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 200–211.
- [19] M. James and L. Walter, "Scrum reference card," *CollabNet Inc*, 2010.
- [20] Y. Wang and S. Wagner, "Combining stpa and bdd for safety analysis and verification in agile development: A controlled experiment," in *Agile Processes in Software Engineering and Extreme Programming*, J. Garbajosa, X. Wang, and A. Aguiar, Eds. Cham: Springer International Publishing, 2018, pp. 37–53.

- [21] F. M. Fowler, *The Sprint Planning Meeting*. Berkeley, CA: Apress, 2019, pp. 83–88. [Online]. Available: https://doi.org/10.1007/978-1-4842-4164-6_13
- [22] —, *The Sprint*. Berkeley, CA: Apress, 2019, pp. 77–82. [Online]. Available: https://doi.org/10.1007/978-1-4842-4164-6_12
- [23] S. Elbaum, G. Rothermel, and J. Penix, “Techniques for improving regression testing in continuous integration development environments,” in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2014, pp. 235–245.
- [24] K. Madadipouy, “An examination and evaluation of agile methodologies for systems development,” *Australas. J. Comput. Sci.*, vol. 2, no. 1, pp. 1–17, 2015.
- [25] R. T. Rust, D. V. Thompson, and R. W. Hamilton, “Defeating feature fatigue,” *Harvard business review*, vol. 84, no. 2, pp. 37–47, 2006.

Report An Analysis Of Agile Development Practices At The DVLA

GRADEMARK REPORT

FINAL GRADE

82/100

GENERAL COMMENTS

Instructor

Agile Process Evaluation (40): 38

Examples (30): 26

Testing (30): 18

Discussion:

Very good discussion of Agile and the DVLA process with nice examples provided.

Good clear description.

A little more background research could have been included on the testing.

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

PAGE 17
