The Role of Product Owner: Duties, Effects and Qualities in Software Development in Iceland

Pétur Bjarni Pétursson

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
University of Iceland
2020

THE ROLE OF PRODUCT OWNER: DUTIES, EFFECTS AND QUALITIES IN SOFTWARE DEVELOPMENT IN ICELAND

Pétur Bjarni Pétursson

60 ECTS thesis submitted in partial fulfillment of a
Magister Scientiarum degree in Computer Science

Advisor
Ebba Þóra Hvannberg

External Examiner
Marta Kristín Lárusdóttir

Committee
Matthias Book

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, May 2020

The Role of Product Owner: Duties, Effects and Qualities in Software Development in Iceland
60 ECTS thesis submitted in partial fulfillment of a M.Sc. degree in Computer Science

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Dunhagi 5
107, Reykjavik, Reykjavik
Iceland

Telephone: 525 4000

Bibliographic information:
Pétur Bjarni Pétursson, 2020, The Role of Product Owner: Duties, Effects and Qualities in Software Development in Iceland, M.Sc. thesis, Faculty of Industrial Engineering, Mechanical Engineering and Computer Science, University of Iceland.

Keywords:
Product Owner, Scrum, Role, Agile, Software Development

Reykjavik, Iceland, May 2020

Abstract

Agile software development methods such as Scrum have been widely used since 2001(Beck et al., 2001). Scrum is a framework that defines various roles and methods to successfully develop a software product. One of those roles is the Product Owner, whose responsibilities include maximizing the product value. This study researched the role of Product Owner in the context of software development in Iceland. The research questions sought to answer what activities Product Owners performed, how they affected development cost, time and scope, what methods they used for affecting product value, similarities with other roles and what skills or qualities were expected of a Product Owner.

The study was carried out using qualitative research methods, data gathered by interviews and observations and grounded theory used in data analysis. The results found that in addition to activities established in theory the Product Owners were involved in supervising the progress of product development, a duty traditionally done by a Scrum Master. The Product Owners affected time and scope more than cost, and the study identified novel skills which were associated with the Product Owner such as technical skills, domain knowledge and creativity.

Útdráttur

Agile aðferðir við hugbúnaðarþróun eins og Scrum hafa verið mikið notaðar síðan árið 2001. Scrum er aðferðafræði sem skilgreinir ýmis hlutverk og aðferðir til að þróa hugbúnaðarvöru á sem skilvirkastan hátt. Eitt þeirra hlutverka er vörustjóri (enska Product Owner), sem ber ábyrgð á því að hámarka virði vörunnar. Þessi rannsókn skoðaði hlutverk vörustjóra í samhengi hugbúnaðarþróunar á Íslandi. Rannsóknarspurningarnar sem leitað var svara við voru hvaða verkefnum vörustjóri sinnir, hvernig vörustjórar hafa áhrif á þróunarkostnað, -tíma og -umfang, hvaða aðferðum þeir beita til að hafa áhrif á virði vöru, líkindi með öðrum störfum og hvaða hæfni eða hæfileika er búist við að vörustjóri hafi yfir að ráða.

Rannsóknin var framkvæmd með eigindlegum rannsóknaraðferðum, gögnum safnað með viðtölum og athugunum, og Grounded Theory aðferðin notuð við greiningu gagna. Niðurstöðurnar sýndu að auk þekktra verkefna úr fræðum sinntu vörustjórar yfirliti með framvindu hugbúnaðarþróunarinnar, ein af hefðbundnum skyldum Scrum Master. Vörustjórarnir höfðu meiri áhrif á tíma og umfang en kostnað, og rannsóknin leiddi í ljós nýstárlega færni sem vænst er af vörustjóranum, það er sköpunargáfu, tæknikunnáttu og þekkingu á sviði vörunnar.

Preface

The motivations for researching the role of Product Owner as defined in the Scrum methodology arose from the work experience of the researcher. He noticed that companies that considered themselves working with Agile methods such as Scrum or Extreme Programming often had differing implementations of these approaches. They also differed in what Agile values were top priority, which principles were adhered to and what roles of any given methodology were adopted.

For example, sometimes the customer could not dedicate enough time to be as involved as needed, or the budget did not allow much flexibility for changes once development had begun. In those cases more time was spent on cost estimates than cooperation, whereas the Agile Manifesto (Beck et al., 2001) values customer collaboration over contract negotiations.

Another observation was that there sometimes was no explicitly defined Product Owner in a company. In those instances, some of the responsibilities of the role were still performed but by the Project Manager, a term sometimes used as a broad, catch-all title.

Contents

List of Tables

Acknowledgments

I would like to acknowledge my gratitude to the individuals who helped me in completing this thesis. My instructor Ebba Þóra Hvannberg for her continuous guidance and patience over the course of my studies, my parents for their support and encouragement and my sister for her constant support, advice giving and inspiration. Thank you.

1. Introduction

Software development methodologies have long been a topic of research. One of the first software engineering conferences was held in 1968 to discuss problems of software development along with possible methods and techniques to solve them (Naur and Randell, 1968). Agile is a category of software development methodologies, formally defined and popularized by the publication of the Agile Manifesto (Beck et al., 2001). One such Agile method is Scrum. Scrum is one of the most widely used Agile methods in software development worldwide, with research indicating Scrum being the most popular Agile method in Iceland (Bjarnason, 2018).

A particular role defined in the Scrum method is the role of Product Owner (PO). Among the responsibilities of the PO is maximizing the product value. Additional functions of the role have been identified, for example by Bass and Haxby (2019) who discuss the PO functions of being a technical architect or communicator. Expectations of the team towards the PO have been surveyed by Unger-Windeler and Schneider (2019), noting that being trustworthy was an attribute of the perfect PO.

Within the context of software development in Iceland, the role of PO has been less studied. A case study by Sverrisdottir et al. (2014) found the PO role varied between Icelandic companies. Little research was found on how a PO affected product value, overlap in PO responsibilities with other positions such as Project Manager (PM), or how a PO affected product development cost, time and scope, the three common criteria in software projects known as the Iron Triangle (Agarwal and Rathod, 2006).

An interview study was carried out to supplement existing research on the PO role in software development. The goal of the study was to gain a better understanding of the activities of a PO, what was expected of a PO and how POs affected product development. The following are the research questions the study intended to answer:

1. What is the role of Product Owners in software development?

2. How does the role of Product Owner affect the Iron Triangle of cost, time and scope of the product development?

3. What methods or tools do Product Owners use to affect the value of the product?

4. How does the role of Product Owner in an industry setting compare to theory?

5. In what ways are the roles of Product Owner and Project Manager alike, and in what ways do they differ?

6. In an ideal setting what are the skills or qualities expected of a Product Owner?

Ten participants from eight different companies were interviewed along with observations performed of them at work. The data was analysed with qualitative methods, using grounded theory and open coding (Runeson and Höst, 2008) to formulate answers to the research questions.

The study results indicated that in addition to activities observed in prior research, the PO performed certain activities attributed in literature to the Scrum Master. The POs were more reactive than proactive with regards to affecting value, and mostly affected the time and scope of the product development. With regards to expected skills and qualities, the results indicated that domain knowledge, a technical background and a creative mindset are of more benefit to a PO than has been suggested so far.

The following sections will discuss and introduce the theoretical background of the topic, then describe the applied methodology, followed by the results, a discussion of the results and the study conclusions.

2. Theoretical Background

The research was influenced and guided by existing research and literature in software engineering and software development. The following sections introduce different software development methodologies, Scrum and its concepts, and outcome evaluation in software development.

2.1. Software Development Methodologies

Methods for software development have been progressing and evolving ever since 1968, when a conference was held discussing software engineering and software development (Naur and Randell, 1968). Since then, various methodologies have emerged in an attempt to formalize the process of developing software. Randell (1968) compared projects concerned with the methodology of designing computing systems as well as discussing some of the problems still faced in systems design at the time. The next section introduces one of the first software development methodologies.

2.1.1. Waterfall Model

One of the oldest formalized methodology is known as the Waterfall Model defined by Royce (1987). The development can be divided into four stages, namely requirements analysis, design, coding and testing. Each stage of development is finished in full before proceeding to the next. The model derived its Waterfall name from the cascading manner in which the development process flows from one step to the next.

2.1.2. Iterative Software Development

Conversely, Iterative and Incremental Development (IID) practices focus on iterative, evolutionary and incremental steps during the development process. Larman

and Basili (2003) reviewed the history of various such practices and noted that the earliest specific reference to iterative development was made in 1968, with some examples of iterative product development even earlier than that. IIDs all have common characteristics, namely developing in multiple iterations, working within some specific time frame and avoiding sequential, single-pass approaches such as in the Waterfall model. By the end of the 20th century, IID practices were becoming widespread and many variants had entered the scene.

2.1.3. Agile Manifesto

In February 2001, a group of 17 individuals representing various IID software practices met in Utah to discuss common ground. Together they authored the Agile 'Software Development' Manifesto (Highsmith, 2001).

The manifesto lays out the core values and principles behind Agile software development or Agile methods. These values and principles are shared by the various methodologies whose representatives took part in authoring it. Dybå and Dingsøyr (2008) list the main agile development methods as being Scrum, Dynamic Software Development Method (DSDM), Feature-driven development, Lean software development, Extreme Programming and Crystal methodologies.

2.2. Scrum

The agile development method Scrum was developed by Schwaber and Sutherland, who wrote the official Scrum Guide and were among the authors of the Agile Manifesto. In the Scrum Guide, Schwaber and Sutherland (2017, p. 3) define Scrum as a "framework within which people can use to address complex adaptive problems, while productively and creatively delivering products of the highest possible value."

The earliest mentions of scrum came in 1986. The name is a reference to the rugby term scrum and was first used in the context of a product development process in Takeuchi and Nonaka (1986). The scrum approach was compared with the old approach, metaphorically described as a relay race where a group of specialists passed the baton to another in a sequential way from the beginning to end of a project. In contrast, the rugby approach instead relied on the constant cooperation of a hand-picked multidisciplinary team working together from start to finish.

Scrum is one of the popular Agile software development frameworks, with 75% of Agile software development using Scrum (digital.ai, 2020). Some have even sug-

gested that Scrum is becoming the de-facto standard in the software development industry (Marchenko and Abrahamsson, 2008). Research by Shastri et al. (2016) found that Scrum was the most popular Agile method, with 91% of respondents using either standalone Scrum or a hybrid of Scrum and one or more other Agile methods.

Within the Scrum framework, a number of artifacts, roles and events have been defined. The following sections explore them in further detail, based on their official definitions in Schwaber and Sutherland (2017) and earlier literature where noted.

2.2.1. Artifacts within Scrum

Product Backlog

The Product Backlog, often referred to simply as the backlog, is a prioritized list of all business and technical functionality that needs to be delivered in the product. It is the sole source of any required changes to a product and contains all features, functions, fixes and requirements that are to be made to the product in future releases.

These changes are referred to as backlog items or tasks. Each item includes a description, priority, estimated effort of development and value. They also include a test description or acceptance criteria which demonstrates the items completeness when work is finished. The backlog is dynamic and evolves as development of the product progresses and as the environment in which the product will be used changes (Schwaber and Beedle, 2001, p. 32-34; Schwaber and Sutherland, 2017).

Sprint Backlog

The Sprint Backlog consists of a list of items the team commits to finishing within a single Sprint (a fixed time period of development described further below). These tasks are items selected from the Product Backlog and contain all the details necessary to convert them into working software. Throughout the Sprint the team modifies and updates the Sprint Backlog if needed. The Sprint Backlog serves as a visible, real time image of the work the team plans to accomplish within a specific time period. Only the team can modify the Sprint Backlog during a Sprint, and as the team becomes more familiar with Scrum processes it becomes better at planning how much they can accomplish in a single Sprint (Schwaber and Beedle, 2001, p. 49, 71).

Potentially Shippable Product

Over the course of each Sprint the team is required to build some increment of the products functionality. As the PO might choose to implement the added functionality immediately, each increment must be properly tested, well written and documented. By the end of each Sprint a Potentially Shippable Product should have been built, containing all of the parts of a completed product with the addition of the finished backlog items selected for that particular Sprint (Schwaber, 2004, p. 12-13, 141).

2.2.2. Roles within Scrum

The Scrum framework defines a Scrum Team as consisting of three roles, namely Product Owner, Scrum Master and Development Team. These Scrum teams should be cross-functional and self-organizing and possess all the necessary skills and competencies to deliver incremental products via Sprints.

Product Owner

The PO is instrumental in securing funding for the product by creating the initial requirements along with its release plans, and by defining Return on Investment (ROI) objectives. The PO has the responsibility of ensuring the maximum value of the product that emerges from the work of the developers. This means that the PO usually prioritizes development of functionality which would solve vital business problems, and that their focus is on the products ROI (Schwaber, 2004; Schwaber and Sutherland, 2017).

The PO coordinates the development efforts between the developers and other stakeholders of the product, and takes decisions within the development periods and between them to ensure that the Development Team is making something of value (Pichler, 2010). Additional activities and functions of the role have been observed in research. Bass et al. (2016) identified various activities performed by POs in small companies, and noted some of the performed activities such as testing were outside of the conventional PO definition. Bass (2015) also describes multiple product ownership functions based on practitioner descriptions from companies utilizing Scrum and Agile methods.

Schwaber and Beedle (2001, p. 34-35) state that in Scrum the PO is the sole individual in control of and responsible for the Product Backlog. This includes the backlogs

content, ordering of the backlog items and making sure the backlog is visible and available to the team.

Although the founders of Scrum explicitly mention the responsibility of PO as belonging to a single individual, there are examples of Scrum implementations where this responsibility is shared. Judy and Krumins-Beens (2008) explored a case in which collective product ownership led to the PO having retained the product accountability while authority over the product vision, prioritisation and execution was shared with the team. Kristinsdottir et al. (2016) reported a case where there were two POs for one product.

Scrum Master

Scrum Master is a management role introduced by Scrum. It is the Scrum Master's responsibility to make sure that Scrum practices are followed within the organization. They help everyone understand the theory, practices, rules and values behind Scrum. The Scrum Master is a servant-leader for the Scrum Team and helps them in multiple ways. For example by making sure all Scrum events and artifacts are properly carried out, conducting Daily Scrums and monitoring the development progress to see if there are impediments hindering developers which the Scrum Master could solve along with representing management to the team and vice versa (Schwaber and Beedle, 2001, p. 31-33).

Development Team

The Development Team, or developers, consists of self-organizing professionals whose job it is to take items from the Sprint Backlog and turn them into a potentially releasable product. The PO and Scrum Master are not considered as part of the development team, unless they take part in working on the backlog items.

2.2.3. Events of Scrum

Sprint

A Sprint is one of the core concepts of Scrum. It is a fixed period of time, usually one month or less, in which the Scrum team works to deliver an increment of product functionality. After a Sprint is finished another starts. This cycle is repeated until the product is deemed ready for release. Whether a product is potentially

releasable is assessed based on empirically managing cost, time, functionality and quality (Schwaber and Beedle, 2001, p. 7-10).

As defined in the Scrum Guide by Schwaber and Sutherland (2017), a Sprint encompasses Sprint Planning, Daily Scrums, the development work, Sprint Review, and Sprint Retrospective.

Sprint Planning: For each Sprint, items are selected from the Product Backlog which the Development Team commits to finishing within the Sprints time frame. It is solely up to the Development Team to assess what it can accomplish over the upcoming Sprint. The PO can help clarify backlog items. For example if an items description is not detailed enough for the developers to assess its complexity, the PO can add details or split the item into smaller items (Schwaber and Beedle, 2001, p. 47-49).

Daily Scrum: Every day of the Sprint, a short status meeting is held. During these daily meetings the Development Team goes over the work done since the last Daily Scrum, what work will be done until the next one, and whether there are any obstacles that might slow down their work on backlog items (Schwaber and Beedle, 2001, p. 40-45).

Sprint Review: After each Sprint a Sprint Review is held. Here, the Scrum Team along with stakeholders of the product meets and reviews what was done in the Sprint. This is an informal meeting as opposed to a status meeting. The latest increment of the product is presented, with the intent to elicit feedback and foster collaboration. The Development Team discusses what was completed, what went according to plan and what went wrong. The PO explains the Product Backlog and what items have been finished and what remains. The Sprint Review also serves as an opportunity for the group to evaluate how the use of the product may have changed, whether the market expectations or requirements have evolved and how such changes affect what should be done next. In this way, the Sprint Review can provide information for the next Sprint Planning event (Schwaber and Sutherland, 2017).

Sprint Retrospective: The Sprint Retrospective usually occurs after the Sprint Review and before the next Sprint Planning meeting. It is attended by the Scrum Team and is intended to be positive and productive. Whereas the Sprint Review evaluates the outcome and finished work of a Sprint, the Retrospective considers how the last Sprint went with regards to people, processes and tools. The outcome of a Sprint Retrospective is the identification and planning of possible improvements that might increase product quality e.g. by improving work processes (Schwaber and Sutherland, 2017)

2.3. Outcome Evaluation

2.3.1. Scrum

Scrum aims to deliver products of the highest possible value. A product is continuously and incrementally improved, with a potentially releasable product available at the end of each Sprint (Schwaber and Sutherland, 2017). This means that along with a releasable product, the product value is a factor in considering the outcome success.

Product Value

The PO is responsible for maximizing the value of the product. How this is done may differ between organizations, teams and individuals (Schwaber and Sutherland, 2017). Schwaber (2004, p. 20-21) goes into more detail and refers to value as business value. He writes that the PO is responsible for the ROI of the project, using the Product Backlog to prioritize requirements that provide the greatest business value and ROI for their organization.

Rico (2008) studied the ROI of Agile methods. He differentiated between soft-side ROI and hard-side ROI. Soft-side ROI referred to qualitative benefits like improved morale, and hard-side ROI referred to quantitative benefits such as cost or time savings. The conclusion Rico drew was that using Agile methods resulted in increased hard-side ROI when compared to traditional methods.

Deemer et al. (2012) note that the PO is responsible for maximizing ROI by choosing the highest-value tasks to work on, but value can be an unclear term in practice. For commercial software profitable tasks could be considered as high value, while for internal applications task prioritization might be influenced by other factors such as a desire to satisfy specific users, aligning the product with the company's strategic objectives or mitigating risks.

2.3.2. Project Management

In traditional project management, the success of a project is sometimes deduced from a combination of three factors, cost, time and scope. Success can then be measured by whether the project finished within the target time, if it was completed for less than the target cost and whether the intended scope or functionality of the

project was attained. These factors are often referred to as the Iron Triangle of cost, time and scope and are often used in software projects (Agarwal and Rathod, 2006).

The Iron Triangle as a success criteria has been linked to project management since the 1950s, but is not the only criteria within project management. Atkinson (1999) proposed a new framework for information systems, the Square Route, which he defines as the Iron Triangle with the addition of three criteria, namely benefits to the organization, benefits to the stakeholders, and the quality of the information system. Examples of the additional criteria include profits and efficiency for the organization, stakeholder satisfaction, and system quality measured by factors such as reliability and maintainability.

3. Methodology

After deciding on the role of PO as a topic and having outlined the research questions, a suitable research methodology was chosen. Whereas quantitative methods are useful for testing a specific hypothesis, Creswell (2008, p. 181-184) lists various characteristics of qualitative research that made it preferable for addressing the research questions. Qualitative research takes place in a natural setting and is more emergent than other methodologies. Qualitative research utilizes multiple methods that are interactive and humanistic and are fundamentally interpretive. The researcher develops a description of the phenomenon being studied, analyzes gathered data for themes or categories and finally making an interpretation or drawing conclusions about its meaning. As the researcher learns what to ask and whom to ask, the research questions may change and be refined.

3.1. Inquiry Strategies

Creswell (2008) notes that the qualitative researcher should use one or more strategies of inquiry. An interview study utilizing certain characteristics of Grounded Theory was chosen, based on his descriptions and suggestions with regards to different qualitative strategies.

3.1.1. Interview Study

An interview study design was outlined which followed similar guidelines as when conducting a case study. Runeson and Höst (2008) define a case study as an empirical method for investigating contemporary phenomena in their context. A case study will never provide statistically significant conclusions. It does however use different kinds of evidence linked together to support a strong and relevant conclusion. The key characteristics of a case study are:

- It is flexible and able to keep up with complex and dynamic real world settings such as software development.

- Its conclusions are based on a clear chain of evidence gathered from multiple sources in a planned and consistent manner.

- It adds to existing knowledge either by being based on established theory or building a new one.

Runeson and Höst further outline the five major process steps involved in a case study:

1. Case study design: The objectives and a case study plan is defined. Such a plan should describe the objectives, what will be studied, the theoretical frame of reference, the research questions, how to collect data and where to find the data.

2. Preparation for data collection: The procedures and protocols used during data collection are defined.

3. Collecting evidence: Execution of the data collection procedures for the studied case.

4. Analysis of gathered data

5. Reporting

Following these steps, an interview study design was outlined. The study objectives and the research questions have been noted in chapter 1 Introduction, while the frame of reference was outlined in chapter 2 Theoretical Background.

The following sections explain what was studied, or the unit of analysis, how and from what participants data was collected, and the methods used for data analysis. The final step of the interview study process as outlined above, reporting the results, follows in the next chapter.

3.2. Unit of Analysis

The subject of this study was the role of PO. In order to increase the precision of the research, data was also gathered from PMs or equivalent manager roles that worked with POs.

3.3. Data Collection

Data was collected with two methods, semi-structured interviews and observations. Semi-structured interviews were performed with people working in the software development field, including POs, PMs and individuals working in equivalent roles. Observations were performed on POs and PMs at their place of work.

These two empirical research methods, interviews and direct observation, were chosen for their complimentary nature. Interviews elicit the spoken thoughts and views of the participant, whereas observations may provide a deeper understanding of the topic and better triangulation of data (Runeson and Höst, 2008). For example, if a participant stated in an interview that they performed certain tasks multiple times per day, an observation could reveal whether there was some discrepancy between the stated frequency and reality.

3.3.1. Interviews

The Interview process was based on guidelines for semi-structured interviews by Runeson and Höst (2008). They note that semi-structured interviews are common, with questions planned beforehand but not necessarily asked in the same order as listed. Depending on the flow of the interview, the researcher might ask questions in a different order. In such cases the question list serves the purpose of making sure that all questions have been answered, regardless of order. Semi-structured interviews also allow for certain improvisation and exploration of the studied topic.

An hourglass model was chosen for the interview and the interview questions, beginning with open questions, moving into more specific ones and ending with more open questions again towards the end. In order not to influence the participants responses the exact nature of the research questions was only described in broad terms prior to the interview.

A digital recorder was used to record the interview in full. Each interview started by informing the participant of the goal of the interview, gathering their consent, discussing confidentiality and explaining the nature of the study. Next, the participants were asked some introductory questions about themselves and their background. From there the interview continued, using the list of questions to guide the interview along. When answering each question, participants were allowed to discuss their answer at length. The researcher noted when they answered multiple questions in a single response and selected the next question from the list based on what he considered contextually fitting and would keep the conversation flowing. At the end of the interview the subjects were asked if they had any thoughts or insights

that came to mind during the interview that they wanted to add.

The interview questions were written with specific research questions in mind. Some research questions were answered by the combined responses to multiple interview questions. Some questions were included in order to gather additional data and allow the participants to share insights which might not have been considered by the researcher. These included questions attempting to elicit war stories from the participants regarding any exceptional experiences working with or without a PO. Introduced by Lutters and Seaman (2007), the War Story Procedure asks participants to narrate a highly memorable instance in which adversity was overcome with exceptional effort. The responses to these questions did not include any such war stories, but they did contain mentions of attributes considered as beneficial in a PO.

See Appendix B for the interview questions as they were worded at the end of the interview phase.

3.3.2. Observations

Observations were performed by shadowing the POs over the course of a day at their work. The researcher met with the team, sat in on any meetings the POs had during that day, and wrote down notes on tasks, duties and other points of interest as they occurred. See Appendix C for an overview of the observation protocol.

The participants were encouraged to verbalize and describe their thoughts and actions while solving tasks in order to capture the cognitive processes they applied to their actions. This is similar to a methodology known as contextual inquiry. Raven and Flanders (1996) discuss and describe some examples of how technical communicators use the methodology to gather useful data from users and note that there are different implementations of the contextual inquiry process. The implementation used in this research was a work based interview, described by Raven and Flanders as a method where participants are interviewed and observed while they are engaged in actual work activities. Observation notes were written in generic terms with no sensitive data from the companies or any identifying information gathered.

Two observations of work were carried out. A few of the interview participants were reluctant to take part in an observation but of those willing only two were able to gain permission from the company. The performed observations lasted a whole workday, with the researcher shadowing the participants at their place of work.

3.4. Participants

To recruit candidates, a link to a short Google Forms questionnaire was posted to a group on social media dedicated to the topic of product and project management in Iceland. In order to increase the chances of a reader responding, the questions were kept few and simple, and the post introducing the research kept short and to the point. The questionnaire asked about the name, job title and current company of respondents, along with company-specific questions such as whether they considered their company to use Agile methods and what tools and metrics they used in their work. See Appendix A for the questionnaire in full.

Out of those who responded, any who did not meet all of the following criteria were filtered out:

- Currently working in software development or directly with developers
- Job title either related to product ownership or project management
- Considered their current place of work to be following Agile methods

Those who fit the criteria were sent a follow-up e-mail, asking them to participate in an interview. They were also asked if it were possible to allow the researcher to observe them in their day to day routine at their place of work.

To get a descriptive view of the industry setting, there were a few goals that the researcher set early on, which influenced participant selection and solicitation. Firstly, to gather in total at least 10 participants. Secondly, that the participants were a mix of either POs (or an equivalent product ownership position) or in a management position working directly with developers or POs.

Out of those six who answered the short questionnaire, two were excluded from further consideration due to not meeting the aforementioned criteria, and one did not answer the follow-up email. Additional participants were located by contacting acquaintances of the researcher within the industry by e-mail, social media and telephone. All those contacted were willing to help by doing interviews or introducing the researcher to additional participants. These introductions caused a snowballing effect in finding additional interview subjects. When contacting additional participants, care was taken to maintain a balanced mixture of job titles among chosen participants.

In the end, 10 participants from eight different companies were interviewed. The companies ranged in size from 15 to 300 employees. The smaller companies were solely focused on software development both for internal use or external clients. The

larger companies were in different business sectors and had software departments which were developing products for internal use. The participants' prior work experience in the software industry ranged from 2 to 25 years with some that had used Agile methods for years or even decades. The job titles varied between participants, but all of them were in the area of software development. Five of the participants had the title PO or an equivalent title. For clarity all of them were referred to as POs in the following chapters and sections. The other five participants were PMs, Program Managers or Development Managers. Collectively they are referred to as managers in the following sections. The gender split was even, and the age of participants ranged from 25 to 50.

3.5. Data Analysis

Grounded Theory as described by Creswell (2008, p. 15, 191) was used for data analysis. Grounded Theory involves multiple stages of data collection, along with the refinement of information categories and their relations. These stages include creating categories of information via open coding, selecting one of those categories and positioning it within some theoretical model, and then interpreting a theory based on the relationship between the categories.

Following interviews and observations, transcription of audio files for interviews was done and notes added. Each recorded interview was played back at half speed and typed into a text editor. Generic transcription guidelines were followed and filler words, false starts (the subject starts a sentence and stops) or instances where the subject made sounds or was silent while thinking of an answer were for the most part omitted. Only in cases where such words, sounds or lack thereof added information or gave possible insight into the thought process of the subject were they included in the transcript. After the initial write-up, the interview was replayed and the transcript reviewed for typos or input errors.

Interview transcripts and other observation data such as notes and images were then related to the interview ID of the respective participant and uploaded to Dedoose (Version 8.3.2; Dedoose, 2020), a Qualitative Analysis application used for transcript analysis and coding . For example, notes and artifacts relating to interview 001 were stored as *"001 Artifact 1.jpg"* and *"001 - Interview.docx"*.

3.5.1. Data Coding

In studies where interviews are the main data source, the first step of analysis is coding of the transcribed data for a better understanding of common elements. Runeson and Höst (2008, p. 151) describe data coding as follows: "First the data is coded, which means that parts of the text can be given a code representing a certain theme, area, construct, etc. One code is usually assigned to many pieces of text, and one piece of text can be assigned more than one code".

The application Dedoose (Version 8.3.2; Dedoose, 2020) was used for coding, analysis and organization of the research data. Dedoose is a cloud-based tool offering functionality for managing qualitative data, such as excerpting, coding and visualization of data. The text files containing the transcribed interviews were uploaded and labelled with participant ID. Coding began by roughly grouping chunks of texts and labelling them according to what interview question they were answering.

During preliminary coding, the participant responses were first read through and paragraphs marked which the researcher interpreted as answering a specific interview question from the scripted questions. For example, if the participant was asked question A.3.1 about cost estimations of development, and in the response they discussed time estimates as well, the relevant paragraph would be marked [A.3.1, A.3.3], as both questions were answered at the same time.

Next the interviews were reread, and paragraphs and sections coded in more detail. The codes were short sentences which summarized the sentiment, statement or fact being stated by each participant in that text. The codes were written in such a way that each code contained only a single sentiment, statement or fact. The codes were phrased in such a way that they could be read as answering the interview question. In cases where similar codes came up in relation to different interview questions, the relevant texts were reviewed and the codes reworded to avoid any misunderstanding during analysis. Each code was in that manner connected to a specific interview question, which in turn was intended to inform the answer to a specific research question.

3.5.2. Content Analysis

To understand the main categories and concepts in participants answers and in line with the exploratory nature of the study, the data was then analyzed. Using the coded data, categories and patterns were generated which would then be used to address the research questions.

When coding of the interview data was complete, the codes were exported to a spreadsheet program. There they were sorted by the interview question they responded to and what research question the interview question addressed. Preliminary grouping was done in the spreadsheet program with similarly worded codes being put in adjoining lines in order to simplify later processing.

During the same preliminary grouping, codes were set aside that did not address any of the research questions. Other codes that were removed were duplicates of existing codes, codes containing identifiable information or codes describing second hand knowledge. For example, if a participant discussed activities they heard a friend in another company describe, the code was removed.

Next, all the codes for each research question were manually grouped together according to their conceptual similarities with regards to the interview question and the research question behind them. The categories that emerged were then reviewed for errors and some categories merged or divided. Lastly, those categories were assembled and the results extrapolated from them to answer the research questions.

4. Results

The following sections address the answers to the research questions based on the gathered and analysed data.

4.1. The role of PO in software development

Research Question: What is the role of Product Owners in software development?

The participants were asked to describe their job and what activities or tasks they performed on a regular basis (daily, weekly or monthly). Only responses from POs were considered when answering this research question, unless the participant was describing a task or activity they performed with a PO. The categories which arose from the responses were the following:

Prioritizing

Activities relating to the backlog or prioritizing in a broader sense came up in almost all of the interviews. A majority of POs mentioned keeping the backlog prioritized, making sure it was up to date and regularly reviewed, as some of their main duties.

Two respondents added that they prioritized tasks within projects for the development team. One of them considered the PO role as including the organization of projects as a whole for the developer team, comparing the PO to being project manager over a specific product.

Progress Supervision

A portion of the responses were related to monitoring the status of development and relaying the status to superiors. These duties were grouped together as progress supervision.

All of the product owners mentioned attending daily meetings with the team, but their participation in them differed. One said that while attendance of the PO to the daily stand-up meetings was not mandatory, they still found it beneficial to join and listen in and see how the tasks were progressing. Another PO felt the benefit in attending daily meetings was twofold. Being able to answer questions on tasks in progress for that day and to clear up any ambiguities in related user stories, and being able to meet the whole team every day, since some were located abroad.

One PM noted that they had reevaluated how often they held meetings involving the PO. After discussing with the PO and the development team separately, they found out that too frequent meetings did not add any value or efficiency for either of them.

All of the participants attended meetings with superiors or executives. These varied in frequency and form depending on the company infrastructure. While some reported to and attended meetings with a larger product market group, another was on the executive governance board which met regularly and went over products and projects in progress. These meetings were all considered to fit into the same duty of having an overview of the progress and relaying it forward.

Defining Tasks

Several activities were mentioned which had a direct effect on how tasks were defined and how they were understood by the developers. Three of the participants noted writing and creating user stories as part of their role as PO. User stories are a method used for writing software feature descriptions from the users point of view. A user story most often describes the users role, the function they want performed and the business value the function provides them (Gannon, 2013).

In addition to writing user stories, several participants mentioned leading refinement or grooming meetings where tasks were reviewed and refined. The purpose of those meetings was e.g. to introduce the tasks to the developers, refine user stories or review tasks for missing information that might delay development. During this refinement a task could be rewritten, broken down into smaller tasks or put on hold if parts of if were unclear and some aspects of it needed clarification.

Communicator

A recurring theme in all of the interviews was being in contact with a wide variety of individuals, soliciting and relaying information as necessary. These duties were

divided into two categories, mediating and user relations. The former category contained responses regarding communications with internal stakeholders and teams, the latter communications with end users.

Mediating: A number of participants were active in stakeholder management and making sure those were satisfied with the development efforts. A large part of the PO role involved compiling information and relaying between the business, developers and the end users. One participant underscored this aspect, and explained their job to those not working in tech as such:

> I'm an interpreter between the business side and the development team. How I usually describe it is that the computer scientists speak their own special language, the business another language and it's the same with the end users. So I'm making all these worlds understand each other. (Interviewed PO)

The same participant also noted that a lot of time was spent on keeping team members informed:

> Because we're outsourcing, a lot of my time goes into keeping them informed. Because they're not on the floor with us, and able to hear what's going on, then I step in and do a lot of keeping them informed, bringing information to them, relaying information from them to others and such. I'm a bit of a mediator in that way, I think. (Interviewed PO)

User Relations: One PO regularly gathered functionality reports from users, answered questions they had and responded to feature requests. Three participants assisted in technical and functional support for end users and customers, by answering bug reports or functionality inquiries that came via the customer support. For example, one PO was considered the main contact for customer support for their product, and helped resolve issues which required more in-depth knowledge of its inner workings than customer support had on hand. They then also decided whether the issue needed to be escalated further, and if so brought the issue to the developers.

Feature Ownership

Duties directly involving the products features were grouped together, and then divided into two categories, feature management and quality assurance. These differ from the categories of prioritizing and defining tasks mentioned above, which involved ordering tasks by priority and adding information to a task. Feature management referred to the responsibility of deciding if features were to be divided into

tasks and put on the backlog. Quality assurance referred to tasks such as one described by a participant as "making sure that what comes out of development is the right thing".

Feature Management The PO decides what features go into development and when they are ready for release. The PO receives requests from various stakeholders, and decides whether or not those features should go into the product. One participant felt that a difficult challenge of product management, was having to say no a lot to feature requests.

Quality Assurance The feature ownership responsibility included making sure that the developed product had the correct features and that those worked as expected. As part of her role, one PO was actively working with the product testers and doing pre-release tests of the product. Another participant considered it part of her role to make sure the product in development was the right product and to ensure all development was of high quality.

Approach and Mindset of a PO

In addition to the above categories there were two that described the approach or mindset with which the participants fulfilled their role as PO, as opposed to specific activities or duties.

Team Support Support of the development team and their efforts was mentioned in several responses as being part of the PO role. One participant felt that trusting the team to do their job minimized overhead, and the PO did their best to be always available to the team if needed. She noted that there was a difference between development teams in how they work. It was the POs job to figure out what arrangement works best for each team, thereby helping and supporting them. Another stressed that the PO is not the teams superior and the PO is not responsible for team morale. But despite it not being in their job description, she still did a lot to maintain good morale and spirits within the team. Some examples included connecting with team members on a personal level, making sure remote members were in the loop on inside jokes, and reading up on various team building exercises.

Adaptiveness Another category of responses describing an approach to the PO role was named adaptiveness. Three of the POs mentioned having to be flexible and willing to try different things, to see what works and what does not. A PO should not be afraid to change or switch methods that are not working, either for the development team or for the PO themselves, and should be open minded and willing to review prior decisions they have made as development progresses.

Takeaway

The PO role consists of prioritizing tasks, both in the backlog and within projects, defining tasks by writing user stories and reviewing tasks, supervising the development progress, facilitating the flow of information between stakeholders, communicating with end users, and making functionality decisions regarding the product along with ensuring the products quality.

Furthermore, participants mentioned the role requiring an approach that is both adaptive and supportive, modifying and changing work processes if needed and being supportive of the development team and their efforts.

4.2. The role of PO in affecting the cost, time and scope of the product

Research Question: How does the role of Product Owner affect the Iron Triangle of cost, time and scope of the product development?

POs were more focused on scope and time estimates and less involved in cost estimates. Conversely, the managers were more involved in time and cost estimates and less in scope estimates.

Cost

Out of the five POs interviewed, none were estimating cost of product development before work began. Two did note that they were indirectly estimating cost, by multiplying the time estimations for the product development with some cost factor. The managers almost all considered themselves estimating the development cost, but these were based on the time estimations as well.

One PO stated that the reason development cost was not actively tracked was because in their company, the development cost of a product was not directly billed to anyone since the product was intended for internal use.

Time

Most POs were estimating development time, and then reevaluating those estimates as development progressed. Two POs mentioned that time estimations were not a top priority for them. One worked on a product where the company was not billing external customers so time estimates were not a priority. The other said that she was doing minimal time estimations with the development team, but that she still learned a lot about the teams skills and efforts through such estimates. One participant felt their time was better spent focusing on the product itself rather than time estimates.

All of the managers were doing rough time estimates and keeping an eye on time spent on projects. Monitoring of the time spent on a project was done both formally via work logs and indirectly with one describing their time monitoring as more reactive than proactive. These rough estimates were usually measured in weeks or months.

Scope

All of the POs defined and estimated the product scope before development. The extent of those estimates and methods to reach them differed. Most did those estimates in cooperation with the developers or working with the client. One mentioned using the no estimates method, which involved estimating whether a task as it was written would take more or less time than a week. If it took more than a week the task was broken down into more tasks. The team used this method because they felt point based estimations took too much time with too little benefit. An example of a point based estimation method is the Story points method, frequently used to estimate size in Agile Software Development (Usman et al., 2014). The same PO also mentioned that they felt their time was better utilised writing the acceptance criteria and user story definitions than defining the scope.

One PO noted that when the scope is properly defined beforehand, things run a lot smoother. She stressed the need to divide complicated tasks down into smaller pieces and added "You don't eat an elephant in one bite, you need to do it in small steps."

A sentiment shared by both POs and managers was that during development, scope will almost always change. The reasons given for why it changes included insufficient data from a client, lack of testing or dependencies on other teams.

Two participants had methods they used to minimize changes to scope when devel-

opment had started. One kept continuous communication between all the product stakeholders on current priorities and the developmental time frame, allowing everyone to spot deviations or a mismatch in expectations earlier in the process. The other method involved coaching customers in the mindset of having a strategy and making choices early in development, which makes it less likely the customer would make drastic changes later. As they described it, when the customer has chosen one implementation path, the others are eliminated. You can then point to that decision later, when discussing whether to add new features into product scope, and see if your product strategy has changed since then or if the new features fits the chosen path.

Takeaway

POs define and estimate the product scope before development, as well as keeping an eye on whether the scope changes during development. POs estimate the development time before and during development. POs do not affect the product development cost, only indirectly through the defining of scope and estimates of the development time.

4.3. Methods and tools used by POs to affect value

Research Question: What methods or tools do Product Owners use to affect the value of the product?

Many of the POs were not actively measuring value of the product during development. Some assessed its value after release by comparing product usage and the effort put into the product. Others measured value based on what the customer perceived as value. For example, one PO estimated value based on the time saved for customers by automating certain tasks. The same PO stated that they were more reactive rather than long-term planning when it came to value estimates, a description which was fitting for many of the participants.

The participants named various methods and tools they used and considered as affecting the value of a product or project. The following sections discuss them.

4.3.1. Methods

Some participants did not mention any particular method by name, but did mention various Agile methods and practices. Building up strong teams, having daily meetings to keep track of progress, having constant communication and everyone voicing their opinion in development. One PO noted that it is important not to be too literal in adhering to Agile guidelines. They used Scrum more as a guideline than a set of rules. For example, they stated using a hybrid of Scrum and Kanban, another Agile method, as it suited them better.

Methods focusing on the user and customer were regularly mentioned. Customer journey mapping allows the PO to focus on the imagined experience of a customer interacting with a certain product or accomplishing some certain task. The PO can then create specific projects from those findings and deliver increased value to the customer. User-centric methods such as product discovery, user testing, user research and user acceptance testing were also found to be helpful in increasing value and getting feedback to make sure that the right things were being developed.

4.3.2. Tools

Most participants were using either Jira or Asana, that are online project management tools. Jira was used to keep track of projects, stories and time spent on projects. Asana was used for similar purposes in addition to often being used for working with third parties. Figma, a collaborative interface design tool, was used by some POs for design work when working with designers or third parties.

Physical tools and ways to make things visual were also mentioned by two participants. One company had a special glass-walled meeting room with questions on the walls which facilitated brainstorming about some workflow being analysed, and whiteboard pens for writing on the walls. Using a pen and whiteboard was the preferred method in a meeting for another participant, who noted that making things visual was a good way to ensure everyone had a mutual understanding. Sketching quick points down as a meeting progressed and being able to point to it and ask "Is this what we were talking about, or is this not what you envisioned?" saved a lot of time and effort for that participant. Other commonly mentioned tools were Slack, Microsoft Teams and Workplace, which are designed around facilitating communication.

Takeaway

The methods and practices that POs use are daily meetings, building strong teams and increasing communication between stakeholders. Various user-centric methods are less used, such as user testing, user research and customer journey mapping.

The tools most commonly used by POs are Jira and Asana, but others less frequently used are Figma, Slack and Microsoft Teams.

4.4. A comparison of the role of PO in an industry setting to theory

Research Question: How does the role of Product Owner in an industry setting compare to theory?

To answer this question, the researcher compared the activities of section 4.1 to the definition of PO according to the Scrum Guide by Schwaber and Sutherland (2017). The activities were also compared with research done by Bass (2015) and Bass et al. (2016), where various functions and activities that fall within the PO role have been identified. The resulting comparison can be seen in Table 4.1, where ✗ denotes a lack of an equivalent function being defined as part of the PO role.

Table 4.1: PO activities

This Study	Scrum Guide	Bass et al.
Prioritizing	Backlog Prioritisation	Prioritiser
Defining Tasks	Backlog Refinement	Groom
Feature Management	Release Planning	Release Master
User Relations	✗	Customer Relationship Manager
Mediating	✗	Intermediary, Communicator
Quality Assurance	✗	Tester, Gatekeeper
Progress Supervision	✗	✗

The Scrum Guide (Schwaber and Sutherland, 2017) outlines a number of activities which the PO is responsible for or takes part in. The PO is the person responsible for a backlog items priority. They take part in refining the backlog items, adding details, estimates and ordering them. Finally, it is at the POs discretion whether or not to release the outcome of a Sprint. These activities are roughly equivalent to

the activities observed in this study, namely prioritizing, defining tasks and feature management. Bass (2015) identified similar functions, namely Prioritiser, Groom and Release Master.

User relations was found to be equivalent to the function of Customer Relationship Manager, observed by Bass et al. (2016) as an additional responsibility of the PO in small companies. Bass et al. describe the function as performing second level support with regards to the user or customer as well as various other tasks such as answering customer inquiries or assisting with installations.

Mediating as defined in this study encompassed two functions in Bass (2015), Intermediary and Communicator. The former denotes the role a PO plays in interfacing with upper management and disseminating domain knowledge to teams. The latter describes a PO connecting teams in different locations and ensuring information flow between teams in formal or informal communication channels.

The activity of quality assurance had counterparts in the functions of Tester and Gatekeeper, which were observed as roles of the PO in small companies (Bass et al., 2016). Tester referred to the observation that POs sometimes performed testing activities and quality assurance tasks, similar to the activities noted in this study. Gatekeeper encompassed the notion that the PO determined whether a feature is complete.

An activity which was not explicitly mentioned as being part of the PO responsibilities was the category progress supervision. The Intermediary function observed in Bass (2015) describes the PO as an interface for senior executives, enabling information flow of their vision and business strategy down to the development teams. Conversely, the activity of progress supervision describes the general monitoring of the development and reporting on it to their superiors. Where the Intermediary is more of a top-down flow of information, progress supervision is a bottom-up flow of information.

This function of supervising progress and working with management is described in Schwaber and Beedle (2001) but as part of the duties of the Scrum Master and not the PO. Schwaber (2004, p. 99-100) discusses similar duties, noting that the Scrum Master needs to ensure progress visibility towards the organization as needed.

Takeaway

The role of POs in an industry setting was for the most part reflected in other research and literature. The PO activity of progress supervision did not have an equivalent function in the reviewed research, but Scrum literature attributes similar

duties to the Scrum Master.

4.5. Comparisons of the roles of PO and PM

Research Question: In what ways are the roles of Product Owner and Project Manager alike, and in what ways do they differ?

The participants noted some conceptual similarities with the roles of PO and PM. Both have the obligation to set goals, follow through and report on those goals. They also both make sure that those involved in the project or product they are overseeing are on top of things. They also care for the product and want to see it succeed, albeit from different perspectives.

With regards to tasks and duties, a common sentiment was that the PM is more focused on the project as a whole, independent from individual products that comprise the project. Projects with multiple products, multiple teams involved or spanning multiple countries were considered something a PM would supervise. In projects with multiple teams, the PM was responsible for coordinating product owner efforts and synchronizing their efforts with the overarching project goals. The PM is also in charge of the implementation and progress of the project as a whole.

POs focus on a more defined problem space, namely a single product and the value delivered by it. They need to know their product inside out, and have to make sure it fits the requirements. Where the PM prioritizes the timeline and execution of the project in a broader sense, the PO prioritizes within the project and for their own team only.

POs look at the wants and needs of both end users and the business in regards to the product. They are more in contact with these individuals than the PM. They instruct and delegate necessary tasks to the developers.

POs are sometimes limited to their own department and lack the authority to persuade other teams to finish their part of a solution, or compel departments to finish work that the team is waiting on. Another manager shared this sentiment, noting that a PO for one of their clients often ran into roadblocks internally, having to wait for other departments to make decisions or collect data, and the development suffered for it. This was an advantage that the role of PM had. Having the authority to order teams or departments into action and with access to other departments, the PM could clear internal hurdles and bring everyone whose input is needed to the table.

Having a PO and PM in the same project was noted by some as being detrimental, adding unnecessary overhead as projects wound down. Another participant voiced a different view. He said that the difference between PO and PM was in the mindset with which a project was approached. If it was ongoing development, with constant funding for maintenance and progress, you needed a PO. If the task was an one-off investment with a well defined start and end point, it was a project which required a PM. Software product development is in its nature unpredictable, so you need product management in the form of a PO. Project management, on the other hand, is a good tool for something that is more predictable.

According to one participant, having both roles makes sense in large projects, where the PM would have oversight over the various production threads. Those threads would then be under the purview of a PO, who would act as the interface for the PM into the production.

Takeaway

The roles of PO and PM are conceptually similar, both setting goals, following and reporting on the progress. However, the PO is more focused on the singular product, what value it delivers, how it works and what the users and the business want from it. In contrast, the PM is focused of the project as a whole and less involved in the specifics of its components.

4.6. The skills or qualities expected of a PO in an ideal setting

Research Question: In an ideal setting what are the skills or qualities expected of a Product Owner?

The participants were asked to describe what qualities and skills a PO should have, and what activities a PO would perform in their company under ideal conditions. Their responses were then grouped based on conceptual similarity into categories that answered the research question. All the categories were either task-related, competence-related or a mixture of both. The following sections detail each category.

Communication Skills

Nearly all participants mentioned the need for communications skills in their expectations of a PO. These were both explicit and implied mentions, with some participants outright stating the PO needs communication skills, while others named activities which involve communication. Examples included being able to successfully elicit information from end users, communicating information smoothly between stakeholders and developers, and having the ability to disseminate and share information in a manner the listener understands.

Prioritizing

Most participants named the backlog, and keeping it clear, prioritized and clutter-free, as one of the activities expected from a PO. In general the ability to be organized and able to prioritize things was considered a beneficial skill for a PO.

Domain Knowledge

According to the participants a vital ability for POs to have was an understanding and knowledge of the domain or problem space in which a software will solve a problem or need. As one participant put it when referring to domain knowledge, "In order to be truly effective at your job, you really need it."

Technical skills

Having technical skills and a technical understanding was considered beneficial by many of the participants. Among reasons given was being able to communicate technical details with more ease, understanding the programming behind the product over to being able to connect and bond with the developers on a more personal level.

One participant felt there often was an aversion by programmers to work with or answer to typical management types who had less understanding of technology. He believed that having taken a programming course or having a background in science or engineering made it easier for a PO to be accepted by the developers as more of an equal rather than just another management type.

Another participant admitted to often thinking that it would be nice to be a pro-

grammer as well, and being able to dive into the back end code and see how things worked. The participant felt she needed to rely on thoroughly explaining issues and requirements in great detail, in order for the developers to understand well enough what was being asked for. If she had been a programmer she would be able to look at how the code works as it is currently implemented, but also explain to the team the requirements more efficiently since they had some shared technical understanding.

Yet another participant felt that having neither domain nor technological knowledge would make fulfilling the duties of PO very difficult. Having at least one of the two, a PO would be able to perform adequately and over time could fill in the gaps and gain insight into the technical stack or the business domain.

Leadership

The need for leadership traits and management skills was a recurring element in participant responses. As one participant put it:

> You need to be able to work with all sorts of people, and have an easy time working with people, those sort of leadership traits.. that is to say, be able to motivate others, motivate the group forward. (Interviewed PO)

The view of PO as a leader of the product development was further supported by responses mentioning that the PO needs to ensure everyone is in alignment with where a product is headed, and has to unite all the different stakeholders of a product behind a common goal.

Creativity

Creativity and related concepts were mentioned by the participants in a wide range of contexts. The need for an open mind to different solutions to problems, having an innovative mindset towards meeting requirements, and being solution oriented all came up with regards to the product development in general. There is also a certain creative mindset necessary in writing and refining user stories, an activity mentioned by participants as well.

Support

Some of the participants mentioned various tasks and qualities which were summarized as support of the developers and the development in general. This included being a point of contact for product questions and issues the end users reported, filtering requests from other departments, insulating the team from distractions and making sure they had everything they needed to be able to start working on the next task on the backlog.

Takeaway

The study concluded that in an ideal setting, a PO should possess communication skills and is required to be in contact with various groups and individuals, needs a good grasp of the business or domain a product is intended for, and needs to be organized and able to prioritize things. Additionally, the role of PO is a leadership role, requires a creative and innovative mindset, having a technical understanding is beneficial, and requires the PO to support the team.

5. Discussion

In this chapter the main findings of the study will be discussed along with addressing limitations and threats to validity of the study.

5.1. PO activities

All but one of the PO categories of activities in this study had similar PO functions described by prior research or literature. The remaining activity, progress supervision, had similarities with Scrum Master duties (Schwaber and Beedle, 2001, Schwaber, 2004, p. 99-100). This suggests the PO is performing some Scrum Master duties, rather than being a novel activity of the PO role.

Two categories observed in this study, namely adaptiveness and team support, represented approaches to the job itself rather than particular activities or functions. These approaches are characteristic of Agile software development, and similar sentiments were found in the fifth and twelfth principles of the Agile Manifesto (Beck et al., 2001). The fifth principle underscores the value of supporting the team and trusting them, while the twelfth suggests the team regularly reflect on its effectiveness and adopt changes to improve it. The researcher suggests that these two categories observed in the work of PO in the Scrum method adhere well to the principles of Agile methods. It is nevertheless interesting that the POs stressed these approaches in particular when describing their job and duties.

5.2. PO effects on development and value

In evaluating how the PO impacts cost, time and scope of the product, one of the findings was that only a few participants were directly considering the cost of the product development. This was surprising, as most PO definitions highlight the POs responsibility of maximizing the product value (Schwaber, 2004; Schwaber and Beedle, 2001). With that responsibility in mind, one might expect to see more

cost considerations, for example in order to calculate the ROI of individual backlog items that are being prioritized. I suspect the reason for this is that for many of the participants, the product development cost was the development time multiplied by some factor. The implication being that when the POs are considering the ROI of a task or feature, estimating both cost and time would be redundant. Or as the the proverb goes: Time is money.

An unexpected finding with regards to product value was that some of the POs did not think of any specific methods or approaches when asked how they affected value as PO. Taken at face value, this would indicate that some of the POs were not using any particular methodologies or processes to affect product value. However, the results may partially be attributed to the interview questions. There are indications that the POs understood them to be asking about complex processes. Some of the POs that did not name any specific methods later discussed prioritizing tasks by time saved for the end user, suggesting that they were indeed making a rough mental estimate of each task's value. These findings indicate that POs utilize a variety of methods and processes in order to affect the product value, even if they do not consciously consider some of them methods. Future research might consider framing similar questions in a more explicit and direct manner.

5.3. Comparing the roles of PO and PM

The results on similarities between the two roles of PO and PM did not reveal any overlap in responsibilities. Most of the participants felt there were conceptual similarities between the roles, but that the roles differed in their respective focus and authority.

When comparing these roles some of the participants felt a PM had more authority than a PO and could more easily push or persuade other teams into action. Future research might look at how the PO interacts with other project management roles or whether the PO role requires greater authority outside their team.

5.4. PO skills and qualities in an ideal setting

A number of intriguing results emerged when looking at PO skills and qualities in an ideal setting. Seven categories were identified, of which three were of special interest. Four of the categories, namely communication skills, prioritization, leadership and support, were already established in prior research or literature. Communication

skills and prioritizing are parallels to the activities of mediating and prioritizing, discussed in section 4.4. Support is mentioned in the fifth principle of the Agile Manifesto (Beck et al., 2001). The leadership aspect of the PO role has been established in prior research and Kristinsdottir et al. (2016) suggested that the PO was even more of a leadership role than previously found.

The remaining three categories, technical skills, domain knowledge and creativity, seem less established as necessary skills or qualities of a PO, only being implied or suggested in the reviewed research. Technical Skills were suggested as potentially beneficial by research such as Sverrisdottir et al. (2014). Domain Knowledge was considered to be of benefit in some literature, as Rubin (2012) asks:

> It's difficult to be an effective product owner if you're new to the product domain. How can you set priorities among competing features if you don't know the subject matter? (p. 172)

Creativity has been hinted at as well, with research such as Unger-Windeler et al. (2019) describing a PO function of being the product visionary, which could be considered creative.

The findings of this study are interesting in that they suggest these three skills and qualities are even more beneficial to a PO than has been explicitly stated in research or literature so far. Further research is needed to learn more about evidence of applying these skills during POs' work and the importance of these skills compared to other skills.

5.5. Limitations of the Study

Some limitations of the study and threats to validity of the results have been considered. One possible threat to validity was having a single researcher both coding and transcribing the interviews. This was mitigated by having the main advisor do sample reviews of the transcribed interviews and coded excerpts.

Another limitation of the study was a lack of access to the participants in order to perform observations. Permission was in many cases denied by the company or organization, increasing the reliance on interview data. As the interviews were thorough and extensive, this did not substantially impact the results, but future research could include additional observations on specific PO activities.

6. Conclusions

In this thesis, an interview study was conducted to research the role of PO in the context of software development in Iceland. By using grounded theory and open coding to analyze interview data, the results show that in addition to activities such as prioritizing and defining tasks, POs supervise the development progress, a duty usually done by a Scrum Master. The PO affects time and scope of the product development, while only indirectly affecting cost. Methods and practices such as daily meetings, facilitating stakeholder communications and user testing are used by the PO to affect product value along with tools such as Jira, Asana and various communication software. The skills and qualities expected of a PO include communication skills, prioritizing skills and leadership traits, as well as creativity, a technical understanding and domain knowledge.

These findings reveal traits of the PO role which have been less regarded so far, as well as activities undertaken which are beyond the roles' original scope. Awareness of how the role is evolving and what competencies are considered valuable in a PO should be a priority for Agile software companies going forward, as these are likely to contribute to a more successful software development.

Further research is suggested to study the importance of these skills compared to other skills, how these traits are applied during POs' work and whether the duties of Scrum Master are often being performed by PO.

References

Agarwal, N., & Rathod, U. (2006). Defining 'success' for software projects: An exploratory revelation. *International Journal of Project Management*, *24*(4), 358–370. https://doi.org/10.1016/j.ijproman.2005.11.009

Atkinson, R. (1999). Project management: Cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International Journal of Project Management*, *17*(6), 337–342. https://doi.org/10.1016/S0263-7863(98)00069-6

Bass, J. (2015). How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, *20*, 1525–1557. https://doi.org/10.1007/s10664-014-9322-z

Bass, J. M., Beecham, S., Noll, J., & Razzak, M. A. (2016). All Hands to the Pumps: The Product Owner Role in Small Companies, 16.

Bass, J. M., & Haxby, A. (2019). Tailoring Product Ownership in Large-Scale Agile Projects: Managing Scale, Distance, and Governance. *IEEE Software*, *36*(2), 58–63. https://doi.org/10.1109/MS.2018.2885524

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., & Jeffries, R. (2001). Manifesto for agile software development. Retrieved April 23, 2020, from https://agilemanifesto.org/

Bjarnason, E. Ö. (2018). *Hvernig er Agile að reynast íslenskum hugbúnaðarhúsum?* (Thesis). Retrieved May 14, 2020, from https://skemman.is/handle/1946/31304

Creswell, J. W. (2008). *Research design: Qualitative, quantitative, and mixed methods approaches* [OCLC: 254461637]. Sage Publ.

Dedoose. (2020). Version 8.3.20, web application for managing, analyzing, and presenting qualitative and mixed method research data (2020). www.dedoose.com

Deemer, P., Benefield, G., Larman, C., & Vodde, B. (2012). A lightweight guide to the theory and practice of scrum. *Ver*, *2*, 2012.

digital.ai. (2020). 14th Annual State of Agile Report [Library Catalog: explore.digital.ai]. Retrieved June 3, 2020, from https://explore.digital.ai/state-of-agile/14th-annual-state-of-agile-report

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, *50*(9), 833–859. https://doi.org/10.1016/j.infsof.2008.01.006

Gannon, M. (2013). An agile implementation of SCRUM [ISSN: 1095-323X]. *2013 IEEE Aerospace Conference*, 1–7. https://doi.org/10.1109/AERO.2013.6497388

Highsmith, J. (2001). History: The Agile Manifesto. Retrieved May 13, 2020, from https://agilemanifesto.org/history.html

Judy, K. H., & Krumins-Beens, I. (2008). Great Scrums Need Great Product Owners: Unbounded Collaboration and Collective Product Ownership [ISSN: 1530-1605]. *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, 462–462. https://doi.org/10.1109/HICSS.2008.186

Kristinsdottir, S., Larusdottir, M., & Cajander, Å. (2016). Responsibilities and Challenges of Product Owners at Spotify - An Exploratory Case Study. In C. Bogdan, J. Gulliksen, S. Sauer, P. Forbrig, M. Winckler, C. Johnson, P. Palanque, R. Bernhaupt, & F. Kis (Eds.), *Human-Centered and Error-Resilient Systems Development* (pp. 3–16). Springer International Publishing.

Larman, C., & Basili, V. (2003). Iterative and incremental developments. a brief history [Conference Name: Computer]. *Computer*, *36*(6), 47–56. https://doi.org/10.1109/MC.2003.1204375

Lutters, W., & Seaman, C. (2007). Revealing actual documentation usage in software maintenance through war stories. *Information and Software Technology*, *49*, 576–587. https://doi.org/10.1016/j.infsof.2007.02.013

Marchenko, A., & Abrahamsson, P. (2008). Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges. *Agile 2008 Conference*, 15–26. https://doi.org/10.1109/Agile.2008.77

Naur, P., & Randell, B. (Eds.). (1968). Software engineering-report on a conference sponsored by the NATO Science Committee Garmisch, Germany, 226. Retrieved May 20, 2020, from http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF

Pichler, R. (2010). *Agile product management with Scrum: Creating products that customers love* [OCLC: ocn457156599]. Addison-Wesley.

Randell, B. (1968). Towards a methodology of computing system design., 204–208. Retrieved May 20, 2020, from http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF

Raven, M. E., & Flanders, A. (1996). Using contextual inquiry to learn about your audiences. *ACM SIGDOC Asterisk Journal of Computer Documentation*, *20*(1), 1–13. https://doi.org/10.1145/227614.227615

Rico, D. F. (2008). What is the Return on Investment (ROI) of agile methods. *Methods*, 1–7.

Royce, W. W. (1987). Managing the development of large software systems: Concepts and techniques [Original Date: 1970-08], 328–338. Retrieved May 20, 2020, from https://leadinganswers.typepad.com/leading_answers/files/original_waterfall_paper_winston_royce.pdf

Rubin, K. S. (2012). *Essential Scrum: A practical guide to the most popular Agile process*. Addison-Wesley.

Runeson, P., & Höst, M. (2008). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131. https://doi.org/10.1007/s10664-008-9102-8

Schwaber, K. (2004). *Agile project management with Scrum*. Microsoft press.

Schwaber, K., & Beedle, M. (2001). *Agile Software Development with Scrum* (1st). Prentice Hall PTR.

Schwaber, K., & Sutherland, J. (2017). The scrum guide-the definitive guide to scrum: The rules of the game. *SCRUM. org, Nov-2017*. Retrieved April 24, 2020, from https://www.scrumguides.org/scrum-guide.html

Shastri, Y., Hoda, R., & Amor, R. (2016). Does the "Project Manager" Still Exist in Agile Software Development Projects? [ISSN: 1530-1362]. *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*, 57–64. https://doi.org/10.1109/APSEC.2016.019

Sverrisdottir, H. S., Ingason, H. T., & Jonasson, H. I. (2014). The Role of the Product Owner in Scrum-comparison between Theory and Practices. *Procedia - Social and Behavioral Sciences*, *119*, 257–267. https://doi.org/10.1016/j.sbspro.2014.03.030

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard business review*, *64*(1), 137–146.

Unger-Windeler, C., Klunder, J., & Schneider, K. (2019). A Mapping Study on Product Owners in Industry: Identifying Future Research Directions. *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, 135–144. https://doi.org/10.1109/ICSSP.2019.00026

Unger-Windeler, C., & Schneider, K. (2019). Expectations on the Product Owner Role in Systems Engineering - A Scrum Team's Point of View. https://doi.org/10.1109/SEAA.2019.00050

Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Effort estimation in agile software development: A systematic literature review. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, 82–91. https://doi.org/10.1145/2639490.2639503

Appendices

A. Participant Questionnaire

Agile á Íslandi

Pétur Bjarni heiti ég og er að vinna að meistaraverkefni í tölvunarfræði. Viðfangsefnið er "Agile" í hugbúnaðargerð, og hver staða þess og notkun er á Íslandi (formlega eða óformlega).
Ég er s.s. að leita að aðilum í hugbúnaðargeiranum sem eru í verkefna- og vörustjórn (eða svipuðum störfum), sem væru til í að svara nokkrum spurningum og væru opin fyrir því að haft yrði samband síðar fyrir ítarlegra follow-up.
Öll svör eru meðhöndluð af fyllsta trúnaði, og allar niðurstöður verða gerðar nafnlausar m.t.t. fyrirtækja og þátttakenda.
Allir svarendur hafa kost á að fá sendar niðurstöðurnar þegar verkefninu lýkur.

*Required

- * Email address:
- * Nafn:
- * Starfstitill:
- * Fyrirtæki:
- Hver er sirka fjöldi starfsmanna í hugbúnaðar/vefþróun hjá fyrirtæki?
- Er unnið eftir "Agile" á þínum vinnustað?
  Þá er átt við hvort fyrirtækið skilgreini sig sem "Agile", þú sjáir einhver einkenni "Agile" í verkferlunum (eftir þinni eigin skilgreiningu á "Agile") eða slíkt.
- Hvaða tól eru notuð í þróun og utanumhaldi verkefna á vinnustaðnum?
  Dæmi: Git, Gitlab, Asana, Trello, Jira o.s.frv.
- Hvaða metla (metrics) er helst stuðst við til að meta framvindu og gæði verkefna?

Dæmi: Kostnaður verkefnis, tímafjöldi verkefnis, LOC (lines of code), o.s.frv.

- * Má hafa samband við þig síðar í verkefninu með ítarlegri spurningar?
  [Já / Nei]

- * Má bjóða þér að fá uppfærslu í pósti þegar niðurstöður liggja fyrir?
  [Já / Nei]

B. Interview Questions

These interview questions will serve as conversational jumping-off points, and are grouped by which research question they are intended to address. They may be asked in a different order, be differently worded or answered in such a way that the responses cover multiple questions, due to the unstructured nature of interviews.

B.1. Introduction Information and Opening Questions

1. Aðeins að segja þér um viðtalið , en eins og þú veist kannski þá er ég að vinna að meistaraverkefni í tölvunarfræði við Háskóla Íslands. Verkefnið snýr í stuttu máli að Agile hugmyndafræðinni, og ég er að kynna mér hlutverk vöru- og verkefnastjóra (á ensku "Product Owner" og "Project Manager") í hugbúnaðargerð á Íslandi og sjá hvernig hlutverkin eru að þróast.
   Sem stendur er ég að safna viðtalsgögnum og að fylgjast með í eigin persónu, en öll gögn verða gerð nafnlaus og órekjanleg til stakra svarenda.
   Það er ekkert rétt eða rangt í þessu, ég er bara að kynnast reynsuheim svarenda.

2. Hvað heitir þú, og hver er starfstitill þinn innan fyrirtækisins?

3. Hefurðu unnið lengi í þessum geira?

B.2. What is the role of Product Owners in software development?

1. Hvernig myndir þú lýsa starfi þínu (vörustjóri/verkefnastjóri)?

2. Hvernig myndirðu lýsa fyrirtækinu, m.t.t. stærðar og skipurits / valdastrúktúr. Hver er t.d. næsti yfirmaður þinn?

3. Sem hluti af því starfi, hvaða tilteknu verkefnum og skyldum sinnirðu daglega/vikulega/mánaðarlega?

4. Hvaða önnur viðfangsefni tekstu á við í vinnunni, sem þú myndir ekki telja hluta af þeim starfstitli? Ertu með fleiri en einn "hatt" eða titil í vinnunni?

B.3. How does the role of Product Owner affect the Iron Triangle of cost, time and scope of the product development?

1. Ef þú hugsar um verkefnið sem þú stýrir núna, áætlaðir þú kostnað við vöruþróunina áður en þið hófust handa?

2. Fylgist þið reglulega með kostnaðinum eða endurmetið?

3. Ef þú hugsar um verkefnið sem þú stýrir núna, áætlaðir þú tímann sem færi í vöruþróunina áður en þið hófust handa?

4. Fylgist þið reglulega með tímanum sem hefur farið í þróunina?

5. Ef þú hugsar um verkefnið sem þú stýrir núna, áætlaðir þú umfang (e. scope) vörunnar sem á að þróa áður en hafist er handa? Með umfangi (scope) er átt við hve víðtæka virkni (e. functionality) varan mun hafa .

6. Fylgist þú reglulega með því hvort umfangið sé að breytast?

7. Metið þið eða mælið gæði vörunnar almennt? T.d. með því að skoða "usability", "reliability" (áreiðanleika), performance og/eða supportability hennar?

8. Fylgist þú reglulega með því hvort gæðin séu þau sömu eða að taka breytingum?

B.4. What methods or tools do Product Owners use to affect the value of the product?

1. Í þeim verkefnum sem þú ert þátttakandi í, hvaða aðferðafræði, ferlum eða aðferðum, í breiðum skilningi, beitir þú sem (vörustjóri/verkefnastjóri) til að hafa áhrif á virði (e. value) vörunnar/verkefnisins (fyrir endanotendur eða viðskiptavin)?
   Dæmi (ef ekkert kemur til hugar hjá viðmælanda): "Gerið þið cost/benefit analysis, metið með viðskiptavini virði hverrar virkni eða hvað?"

2. Hversu gagnleg finnst þér sú aðferð?

3. Hvaða tólum eða verkfærum beitir þú til að hafa áhrif á virði (e. value) vörunnar/verkefnisins, hvort sem er í tengslum við fyrrnefndar aðferðir eða almennt?
   Dæmi (ef ekkert kemur til hugar hjá viðmælanda): "Jira, Excel, Wordskjal? Eitthvað sérsmíðað?"

4. Hversu gagnleg finnst þér það/þau tól?

5. Mælið þið virði vörunnar á einhvern hátt? Þá er átt við virði eins og tímasparnað, tekjur eða einhverja bót/hag fyrir endanotanda eða viðskiptavin), gætirðu nefnt dæmi um slíkt?

6. Manstu eftir atviki þar sem það að hafa góðan PO hafði góð áhrif (þú sjálf/ur eða reynsla frá fyrra teymi eða störfum)? Hvað var það helst sem þú telur að hafi verið lykilatriði að svo fór?

7. Manstu eftir einhverju atviki þar sem það að hafa engan eða óreyndan PO hafði slæm áhrif (þú sjálf/ur eða reynsla frá fyrra teymi eða störfum)? Hvað var það helst sem þú telur að það hefði breytt?

B.5. In an ideal setting what are the skills or qualities expected of a Product Owner?

1. Hvaða hæfni eða hæfileika telur þú að PO þurfi að ráða yfir, og af hverju?

2. Við fullkomnar aðstæður, hvaða skyldur og verk þættu þér falla undir verksvið PO í þínu fyrirtæki?

B.6. In what ways are the roles of Product Owner and Project Manager alike, and in what ways do they differ?

1. Er fyrirtækið þitt með bæði vörustjóra og verkefnastjóra?

2. Á þínum vinnustað, hver myndirðu segja að væri munurinn á vörustjóra og verkefnastjóra?

3. Hvað myndirðu segja að væri líkt með þessum hlutverkum, ef eitthvað?

B.7. Additional questions

1. Eru einhver atriði eða viðfangsefni sem þú vilt bæta við eða útskýra nánar?

C. Observation Protocol

Observations will be performed at the workplace of Product Owners / Project Managers, using contextual inquiry and having participants verbalize and describe their actions when they work.