

# **Supporting the Tailoring of the Product Owner Role to Hybrid Development Environments**

Der Fakultät für Elektrotechnik und Informatik  
der Gottfried Wilhelm Leibniz Universität Hannover  
zur Erlangung des akademischen Grades

**Doktor-Ingenieurin**

(abgekürzt: Dr.-Ing.)

genehmigte Dissertation von Frau

**M. Arts Carolin Unger-Windeler**

geboren am 04.02.1986 in Hannover, Deutschland

**2020**

1. Referent: Prof. Dr. Kurt Schneider
  2. Referent: Prof. Dr. Ralph Ewerth
- Vorsitzender der Prüfungskommission: Prof. Dr. Holger Blume

Tag der Promotion: 28.09.2020

# Abstract

Product Owners have an important role in the agile software development process. While the description of the Product Owner role heavily depends on its particular agile framework, the application of a single development framework is seldom in practice. In fact, customized hybrid development approaches, where frameworks/methods are tailored or combined with others, are state of the art. Although it is common knowledge that processes need to be tailored to project needs, as they become, otherwise, a project risk – the tailoring of the Product Owner role has been neglected in research so far. Consequently, there is a lack of knowledge about how to tailor the Product Owner role to hybrid development environments. As this knowledge gap can put projects in a hybrid environment to a risk, the goal of this thesis is to close this gap.

To achieve this, a knowledge-base of Product Owner peculiarities needs to be established and consolidated with knowledge from the area of Software Process Tailoring. To generate the knowledge-base of the Product Owner peculiarities, a number of case studies as well as a systematic mapping study was conducted to identify Product Owner tasks, characteristics and structures in hybrid development environments. This resulted in the identification of 13 frequently conducted tasks, 6 favorable characteristics and 12 different structures of Product Owners that apply in hybrid development environments. From the area of Software Process Tailoring, 14 influencing factors on the Product Owner role were extracted along with its respective action items. The consolidation of this knowledge results in a catalog which combines the influencing factors, the Product Owner tasks, characteristics & structures as well as the respective implications on the Product Owner role according to the research results of this thesis. Based on this catalog, any project environment can be assessed and distinct recommendations for a tailored Product Owner role can be deduced. Overall, this thesis generated 84 different recommendations on how to tailor the Product Owner role to a particular hybrid development.

With this, so far missing knowledge was gained to systematically support the tailoring of the Product Owner role to hybrid development environments and thus, to support projects to complete successfully. To share the gained knowledge to other researchers as well as practitioners, this thesis also provides an expert system in the form of a proof of concept web-application. The so-called *Hybrid Product Owner (short: HyPrO) Expert System* represents the research results of this thesis. Its user-friendly interface enables the user to assess the project environment and displays the respective recommendations. The *HyPrO Expert System* validated the results of this thesis, as it surpassed human experts by providing more comprehensive recommendations in a comparative case study.

**Keywords:** product owner, agile software development, hybrid development environment, expert system



# Zusammenfassung

Product Owner spielen eine wichtige Rolle im agilen Softwareentwicklungsprozess. Während die Beschreibung der Product Owner-Rolle stark von ihrem jeweiligen agilen Framework abhängt, ist die Anwendung eines einzelnen Entwicklungsframeworks in der Praxis selten. Tatsächlich entsprechen maßgeschneiderte hybride Entwicklungsansätze, bei denen Frameworks / Methoden verändert oder mit anderen kombiniert werden, dem Stand der Technik. Obwohl allgemein bekannt ist, dass Prozesse auf die Projektanforderungen zugeschnitten werden müssen, da sie sonst zu einem Projektrisiko werden, wurde die Anpassung der Product Owner-Rolle in der Forschung bisher vernachlässigt. Infolgedessen fehlt es an Wissen darüber, wie die Rolle des Product Owner an hybride Entwicklungsumgebungen angepasst werden kann. Da diese Wissenslücke für Projekte in einer hybriden Umgebung zu einem Risiko werden kann, ist das Ziel dieser Arbeit, diese Lücke zu schließen.

Um dies zu erreichen, wurde eine Wissensbasis über die Besonderheiten der Product Owner-Rolle aufgebaut und mit Kenntnissen aus dem Bereich des Software Process Tailoring konsolidiert. Um die Wissensbasis über die Product Owner-Besonderheiten zu generieren, wurden eine Reihe von Fallstudien sowie eine systematische Mapping-Studie durchgeführt. Dies führte zur Identifizierung von 13 häufig durchgeführten Aufgaben, 6 favorisierten Eigenschaften und 12 verschiedenen Strukturen von Product Ownern in hybriden Entwicklungsumgebungen. Aus dem Bereich des Software Process Tailoring wurden 14 Einflussfaktoren auf die Product Owner-Rolle zusammen mit ihren jeweiligen Aktionselementen extrahiert. Die Konsolidierung dieses Wissens führt zu einem Katalog, der die Einflussfaktoren, die Product Owner Aufgaben, Merkmale und Strukturen sowie die jeweiligen Auswirkungen auf die Product Owner-Rolle gemäß den Forschungsergebnissen dieser Arbeit kombiniert. Basierend auf diesem Katalog kann jede Projektumgebung bewertet und eindeutige Empfehlungen für eine maßgeschneiderte Product Owner-Rolle abgeleitet werden. Insgesamt ergab diese Arbeit 84 verschiedene Empfehlungen, wie die Rolle des Product Owners auf eine bestimmte hybride Entwicklungsumgebung zugeschnitten werden kann.

Damit wurde bisher fehlendes Wissen gewonnen, welches die Anpassung der Product Owner-Rolle an hybride Entwicklungsumgebungen sowie einen erfolgreichen Projektabschluss systematisch unterstützt. Um das gewonnene Wissen an andere Forscher und Praktiker weiterzugeben, bietet diese Arbeit auch ein Expertensystem in Form einer Proof-of-Concept-Webanwendung. Das sogenannte *Hybride Product Owner (kurz: HyPrO) Expertensystem* repräsentiert die Forschungsergebnisse dieser Arbeit. Über die nutzerfreundliche Oberfläche kann der Benutzer die Projektumgebung bewerten und die entsprechenden Empfehlungen anzeigen lassen. Das *HyPrO Expert System* validiert die Ergebnisse dieser Arbeit, indem es in einer vergleichenden Fallstudie umfassendere Empfehlungen als menschliche Experten lieferte.

**Schlagerwörter:** Product Owner, Agile Softwareentwicklung, Hybride Entwicklungsumgebung, Expertensystem



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation and Background . . . . .	1
1.2	Hypothesis . . . . .	2
1.3	Research Approach . . . . .	2
1.4	Structure . . . . .	3
<b>Part I</b>	<b>Basics: The Product Owner and Hybrid Software Development</b>	<b>5</b>
<b>2</b>	<b>The Product Owner</b>	<b>9</b>
2.1	The Product Owner in Agile Software Development . . . . .	9
2.2	The Product Owner in Scaled Agile Software Development . . . . .	11
<b>3</b>	<b>Hybrid Development</b>	<b>15</b>
3.1	Hybrid Development as State of the Art . . . . .	15
3.2	Software Process Tailoring for Hybrid Software and System Development in Industry	16
3.3	Definition of Hybrid Development Environment . . . . .	17
<b>4</b>	<b>Part I Summary</b>	<b>19</b>
<b>Part II</b>	<b>Structuring the Area of Research</b>	<b>21</b>
<b>5</b>	<b>Related Work on the Product Owner Role in Industry</b>	<b>25</b>
5.1	Tasks and Characteristics . . . . .	25
5.2	Product Owner Teams – Terminology and Structures . . . . .	28
<b>6</b>	<b>Identifying Research Directions – A Mapping Study on Product Owners in Industry</b>	<b>31</b>
6.1	Research Objective . . . . .	31
6.2	Research Design . . . . .	31
6.3	Results . . . . .	36
6.4	Identified Research Directions for this Thesis . . . . .	43
<b>7</b>	<b>Part II Summary</b>	<b>45</b>

<b>Part III</b>	<b>State of the Practice: Product Owners in Hybrid Environments</b>	<b>47</b>
<b>8</b>	<b>Product Owners Communication Activities in a Hybrid Environment</b>	<b>51</b>
8.1	Methodology . . . . .	51
8.2	Product Owners Communication Activities . . . . .	55
8.3	Quantification of Communication Activities . . . . .	56
8.4	Product Owners Communication Partners . . . . .	59
<b>9</b>	<b>Requirements Flow and Collaboration in a Hybrid Environment</b>	<b>61</b>
9.1	Related Work . . . . .	61
9.2	Methodology . . . . .	62
9.3	Requirements Flow in a Hybrid Environment . . . . .	66
9.4	Involved Roles in a Hybrid Requirements Engineering Process . . . . .	74
9.5	Product Owner Tasks within a Hybrid Requirements Engineering Process . . . . .	75
9.6	Collaboration between Product Owners and Traditional Management Roles . . . . .	76
<b>10</b>	<b>Expectations on the Product Owner Role in a Hybrid Environment</b>	<b>79</b>
10.1	Methodology . . . . .	79
10.2	Expectations on Tasks . . . . .	83
10.3	Expectations on Characteristics . . . . .	86
<b>11</b>	<b>Part III Summary</b>	<b>87</b>
<b>Part IV</b>	<b>The HyPrO Expert System</b>	<b>89</b>
<b>12</b>	<b>Type of Research Artifact</b>	<b>93</b>
<b>13</b>	<b>Model</b>	<b>95</b>
13.1	Knowledge-base: Product Owner . . . . .	95
13.2	Knowledge-base: Software Process Tailoring Criteria . . . . .	102
13.3	Resulting Catalog . . . . .	105
13.4	Inference Engine . . . . .	106
<b>14</b>	<b>Instance</b>	<b>113</b>
14.1	User Interface . . . . .	113
14.2	Calculation of Results . . . . .	119
14.3	Key Design Decisions . . . . .	121
<b>15</b>	<b>Validation</b>	<b>123</b>
15.1	Comparative Case Study . . . . .	123
15.2	Results . . . . .	129
<b>Part V</b>	<b>Conclusion</b>	<b>139</b>
<b>16</b>	<b>Contributions of Thesis</b>	<b>143</b>
16.1	Theoretical Work . . . . .	143
16.2	Empirical Work . . . . .	144



16.3 Constructive Work . . . . .	144
<b>17 Conclusion</b>	<b>147</b>
<b>18 Limitations and Future Work</b>	<b>149</b>
<b>Part VI Appendix</b>	<b>151</b>
<b>A Systematic Mapping Study Reference List</b>	<b>155</b>
A.1 Startset Product Owner . . . . .	155
A.2 Snowball Backward Product Owner . . . . .	156
A.3 Snowball Forward Product Owner . . . . .	156
A.4 Startset On-site Customer . . . . .	156
A.5 Snowball Backward On-site Customer . . . . .	157
<b>B Requirements Engineering Process - Interview Questions</b>	<b>159</b>
<b>C Catalog</b>	<b>163</b>
C.1 Category: Team . . . . .	163
C.2 Category: Internal Environment . . . . .	165
C.3 Category: External Environment . . . . .	167
C.4 Category: Objective . . . . .	169
<b>D Truth Tables</b>	<b>175</b>
D.1 Truth Table for Category: Team . . . . .	176
D.2 Truth Table for Category: Internal Environment . . . . .	177
D.3 Truth Table for Category: External Environment . . . . .	178
D.4 Truth Table for Category: Objective . . . . .	179
D.5 Combined Categories Truth Table . . . . .	184
<b>E Values &amp; Scales</b>	<b>187</b>
<b>F Test Case Scenarios</b>	<b>189</b>
<b>G Publications</b>	<b>193</b>
<b>List of Figures</b>	<b>195</b>
<b>List of Tables</b>	<b>197</b>
<b>Glossary</b>	<b>199</b>
<b>Bibliography</b>	<b>201</b>
<b>Curriculum Vitae</b>	<b>211</b>



# 1

## Introduction

### 1.1 Motivation and Background

Since several years, software producing companies face the necessity to develop and distribute high-quality software at a high pace [38]. Especially companies in systems development struggle with delivering working software at an early stage while remaining flexible and adaptable to changes [38]. Agile software development promises to satisfy these needs [12, 26, 59]. Consequently, a lot of companies strive towards a more agile development approach [51, 52] and the methods have been integrated in various projects, industries and wide-ranging application domains [30, 32, 33, 55, 65, 75, 107]. However, particularly in large organizations that combine software and hardware production, implementing agile methods is often reported as difficult [14].

This difficulty may be one possible reason why hybrid development methods are state of the art [50]. Based on the results of the large-scale HELENA<sup>1</sup> study [58], nearly three of four companies follow a hybrid development approach combining different development methods and practices [50]. These customized hybrid methods often include traditional and agile development methods and practices [101], whereas Scrum and the Waterfall model are among the most applied development methods [50, 101]. Consequently, many aspects within the development process need to be tailored to fit to this particular hybrid model.

Whenever the Scrum framework is applied, the role of the Product Owner comes into play [90]. This role is an important aspect within the development process. Although it is commonly accepted that any software process needs to be tailored to the particular project's requirements, as it becomes, otherwise, a project risk [4, 5, 43, 76], the tailoring of the Product Owner role has been almost neglected in research until now.

To fill this gap, this thesis addresses this topic with the overall goal to support the tailoring of the Product Owner role to hybrid development environments.

The identification of this gap and the motivation to fill it derived from the author's personal experience of working in a hybrid environment, where the Scrum framework and its different roles were integrated into a traditional product development environment. While most of the

---

<sup>1</sup>HELENA: Hybrid dEveLopmENt Approaches in software systems development, online: <https://helenastudy.wordpress.com>

integration efforts were considerably successful, the integration of the new and unknown role of the Product Owner was perceived as not satisfactory. What was missing, was any kind of support to apply this role differently than defined in a certain framework such as Scrum, Nexus or SAFe. The issue with the definitions of the Product Owner role in these frameworks is that they are too abstract to deduce a tailored version of it for even just a slightly different environment. Hence, to be able to tailor the Product Owner role, the initial attributes that constitute this role needs to be identified.

Thus, the goal of this thesis is defined as follows:

The goal of this thesis is to analyze Product Owner tasks, characteristics and structures, to deduce recommendations to support the tailoring of the Product Owner role to hybrid development environments.

## 1.2 Hypothesis

While the consideration of all relevant factors to tailor the development process is quite complex already, it gets even more complex when the Product Owner needs to be tailored as well. This is because the more factors (which may depend on each other [56]) need to be considered, the more complex it gets for a human to ponder all options and make the right decision. Also, due to the fact that the tailoring of the Product Owner to hybrid development environments is considerably new as well as perceived as difficult, it stands to reason that there are not many experts with sufficient knowledge to tailor this role accordingly. However, supposed experts in this field could be consultants who have specialized on the introduction of Product Owners within certain frameworks, experienced Product Owners or general project managers who have several years of experience when it comes to manage projects and the various roles within a software development team. However, as the level of experience depends on the very individual, the recommendations on how to tailor the Product Owner role to particular hybrid development environments might vary and hence, are not very comprehensive. Therefore, it stands to reason that a knowledge-based expert system could support the tailoring of Product Owners to hybrid development environments.

Therefore, the author of this thesis hypothesize the following:

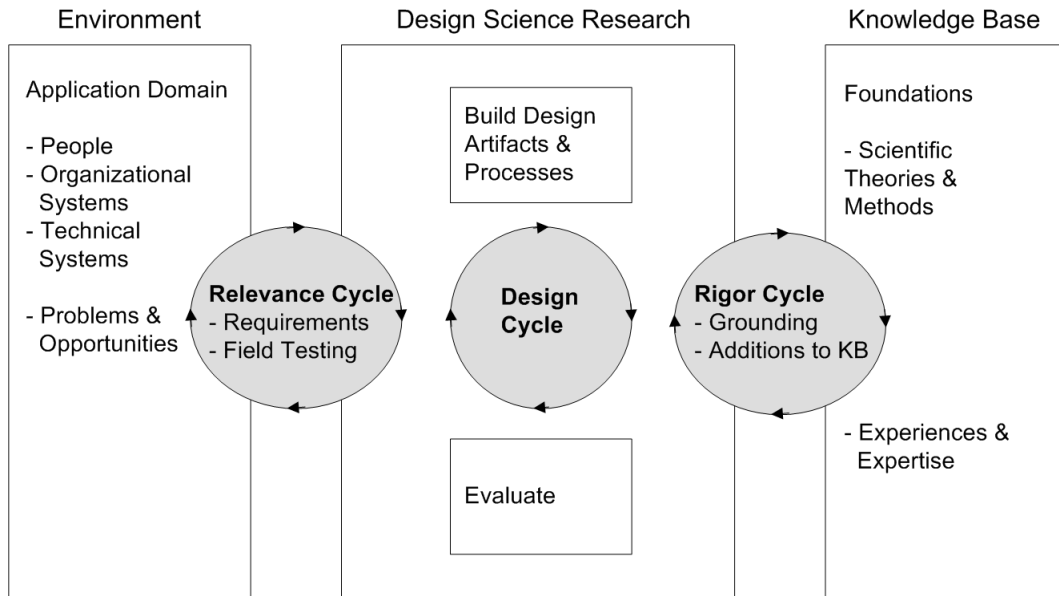
**Hypothesis: The tailoring of the Product Owner role to hybrid development environments can be supported by a knowledge-base expert system which provides equal or more comprehensive recommendations than human experts.**

## 1.3 Research Approach

The above described motivation and hypothesis for this thesis goes along with the motivation of design science research:

”(...) the desire to improve the environment by the introduction of new and innovative artifacts and the processes for building these artifacts.” (Simon 1996, as cited in [35, p.2])

Consequently, the design science research cycles (by Hevner [35]) are followed for this thesis, as depicted in Figure 1.1 and as described as follows:



**Figure 1.1:** Three-cycle view of Design Science Research (based on [35])

**The Relevance Cycle** identifies opportunities and problems in an actual application environment, whereas this environment consists of its people, organizational and technical systems that all interact and work towards a goal [35]. In this thesis, the Relevance Cycle is applied as described in the author’s motivation and background (Section 1.1). Although the author identified the problem in the actual application environment of a certain company and hence a certain domain, the identified problem is applicable to all hybrid environments regardless of the domain or company. Also, within that cycle, the resulting artifact should be validated.

**The Design Cycle** is the heart of each design science project and iterates between its design and its evaluation to constantly improve the design [35]. In this thesis, the Design Cycle is applied in the instantiation of the *HyPrO Expert System* described in Part IV.

**The Rigor Cycle** provides existing knowledge to the research project to ensure the innovation of the newly designed artifact [35]. In this thesis, the Rigor Cycle builds the scientific foundation and is applied in various sections. The existing knowledge is discussed in Part I and II. New knowledge has been added to the knowledge-base in Part III and Part IV combines all relevant information.

## 1.4 Structure

The structure of this thesis follows the iterative approach according to the design science research approach described in Section 1.3. However, iterative approaches are hard to describe in a thesis. Therefore, following structure is applied:

Part I provides the basics of the two most important Software Engineering topics addressed in this thesis, the different descriptions of the Product Owner role in the most common (scaled) agile frameworks and the importance of hybrid software development. Also, a definition of hybrid development environments is provided for this thesis.

Part II structures the area of research for this thesis based on related work and a systematic mapping study. Furthermore, the results of the systematic mapping study identified some gaps in current knowledge on Product Owners in hybrid environments.

Part III addresses the identified gaps and adds newly gained knowledge to the existing knowledge of this field.

Part IV represents the design cycle of the design science research approach and combines the knowledge from Part I, II and III. It describes the generation of the model, the instantiation of the *HyPrO Expert System* as well as its validation.

Part V concludes the thesis by listing the contributions, limitations and presents potential fields for future work.

## Part I

# Basics: The Product Owner and Hybrid Software Development





---

<b>2</b>	<b>The Product Owner</b>	<b>9</b>
2.1	The Product Owner in Agile Software Development . . . . .	9
2.1.1	Scrum . . . . .	10
2.1.2	eXtreme Programming (XP) . . . . .	11
2.2	The Product Owner in Scaled Agile Software Development . . . . .	11
2.2.1	Nexus . . . . .	11
2.2.2	Large-Scale Scrum (LeSS) . . . . .	12
2.2.3	Scaled Agile Framework (SAFe) . . . . .	12
<b>3</b>	<b>Hybrid Development</b>	<b>15</b>
3.1	Hybrid Development as State of the Art . . . . .	15
3.2	Software Process Tailoring for Hybrid Software and System Development in Industry	16
3.3	Definition of Hybrid Development Environment . . . . .	17
<b>4</b>	<b>Part I Summary</b>	<b>19</b>

---



# 2

## The Product Owner

This chapter provides the theoretical foundations of (scaled) agile software development and provides a short description of the most common development approaches. As the main topic of this thesis is the tailoring of the Product Owner role in a hybrid development approach, it emphasizes on the definitions of this role for each framework and that customized hybrid development approaches are state of the art.

### 2.1 The Product Owner in Agile Software Development

Since 2001, agile software development is a term that stands for a set of frameworks, methods and practices that are based on values and principles expressed in the *Manifesto for Agile Software Development* [9].

While the 4 values are defined as follows [9]:

**Individuals and Interactions** over processes and tools.

**Working Software** over comprehensive documentation.

**Customer Collaboration** over contract negotiation.

**Responding to Change** over following a plan.

While it is clearly stated that there is value in the items on the right, the values on the left are seen as more important [9]. Hence, they basically can be read as follows:

- It is still important to have processes and tools, although the effective collaboration of individual persons is even more important.
- A comprehensive set of documentation is necessary, however, the main focus should be on developing working software.
- A contract is important, but should not be holding back from working with the customer closely.
- Having a project plan is important, but it should have some space to accommodate changes.

With this, all of these frameworks promise increased customer satisfaction with lower defect rates, faster development times, and a solution to rapidly changing requirements [15].

While there are many different methods and practices available to support the software development life cycle, e.g. eXtreme Programming (XP), Kanban, Feature-Driven Development

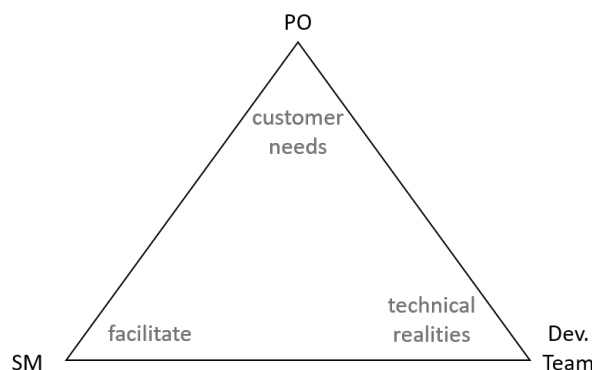
(FDD) or Lean software development – Scrum and Scrum/XP Hybrid happen to be the most commonly used agile methodologies [108].

### 2.1.1 Scrum

Scrum is not a process to be followed, but rather a framework to manage and develop complex products [92].

It is a lightweight and simple to understand framework that consists of Scrum Teams and their associated roles, events, artifacts and rules. The heart of Scrum is a Sprint, a time-boxed event where a “Done”, useable, and potentially releasable product increment is developed. The increment to be developed during a sprint is defined through a set of requirements that is listed in the product backlog (artifact)[92].

Scrum Teams and their associated roles are: the Product Owner, the Scrum Master and the Development Team. These three roles can be represented in a triangle as in Figure 2.1. The Product Owner represents the customer needs as available and ordered content in the product backlog. The Scrum Team members are representing the technical realities of software development while developing the increment during a sprint against the requirements in the product backlog. The Scrum Master has to facilitate the collaboration between the Product Owner and the Development Team [92].



**Figure 2.1:** Basic Representation of a Scrum Team[106]

**Product Owner in Scrum** In the Scrum framework, the role of a Product Owner (PO) is represented by a single person who is responsible for requirements elicitation and for requirements prioritization [91, 92]. Furthermore, he is responsible for managing the Product Backlog by ensuring that the Product Backlog is visible, transparent and understandable to everyone involved in the development process. To do so, he has to express the Backlog items clearly and prioritize them to best achieve project goals and missions [92].

Overall, the Product Owner is responsible for maximizing the value of the product. However, ”how this is done may vary widely across organizations, Scrum Teams, and individuals” [92, p.6].

### 2.1.2 eXtreme Programming (XP)

In 2000, one year prior to the *Manifesto for Agile Software Development*, Kent Beck published the book "Extreme Programming explained" [10]. Extreme Programming (XP) already emphasized the development of small Increments within short development phases and describes some practices, such as "pair programming" and "continuous integration"[10, 11]. To apply these practices, the team needs to be extremely disciplined – hence, the name.

**On-site Customer in XP** In XP, the on-site customer is a person who knows the domain well, is able and responsible for making business decisions and to be on-site with the rest of the XP team [69]. Beck [10] defines an on-site customer to be "a role on the team for choosing what stories the system has to satisfy, what stories are needed first and what can be deferred" [10, p. 177].

## 2.2 The Product Owner in Scaled Agile Software Development

While the described methodologies in 2.1 are perceived as best suitable for small teams working on greenfield projects[10, 16] there have evolved several frameworks that support challenging environments with large-scale development efforts and / or distributed development teams.

### 2.2.1 Nexus

Nexus is a framework that uses the Scrum framework (see Section 2.1.1) as its foundation but scales it up to weave together the work of three to nine Scrum Teams to work on a single Product Backlog and create an Integrated Increment. The main difference to Scrum is that more attention is paid to dependencies and collaboration between multiple Scrum Teams rather than just one [89].

To coordinate, coach and supervise the effort of multiple Scrum Teams, a new role is introduced: the Nexus Integration Team. It consists of the Product Owner, a Scrum Master and Nexus Integration Team Members [89]. All members of this team are skilled in the use of tools, various practices, and the general field of systems engineering. They have to detect dependencies within the Nexus and frequently integrate all artifacts to the definition of "Done" [89].

**Product Owner in Nexus Integration Team** The Product Owner role in the Nexus framework is defined as in the Scrum framework (see Section 2.1.1). Hence, the Product Owner is a single person that manages a single Product Backlog. The Product Owner remains accountable for managing the Product Backlog so that maximum value can be derived from multiple Scrum Teams at the same time. Other than in Scrum, the Product Owner is a member of the Nexus Integration Team and hence, is also responsible to coordinate and steer the process accordingly. However, "how this is done may vary widely across organizations, Nexuses, Scrum Teams, and individuals" [89, p. 6].

## 2.2.2 Large-Scale Scrum (LeSS)

Large-Scale Scrum (LeSS) is a set of rules to support the collaboration of multiple teams working together on one product. [60]. Its concept is the right combination of principles and elements of Scrum in a large-scale context.

**Product Owner in LeSS** Larman and Vodde [60, 61, 62] suggest to define the Product Owner as a group of people rather than a single person to cope with the increased work load in a large scale environment. The so called *Product Owner Team* consists of the Product Owner, at least one other Area Product Owner (APO) and perhaps others to support the requirements clarification for the individual teams [62]. Although all of the Product Owner Team members should make cost, schedule and content decisions together for the whole product, the Product Owner has the final saying. However, to better support many teams, the Backlog is split into different Area Backlogs. Each Area Backlog is owned by an individual Area Product Owner solely. Each Area Product Owner clarifies and prioritizes his Area Backlog Items independently. The Area Product Owners are specialized in this customer-centric area and acts as the main point of contact for the assigned teams in that area [62] (see Figure 2.2).

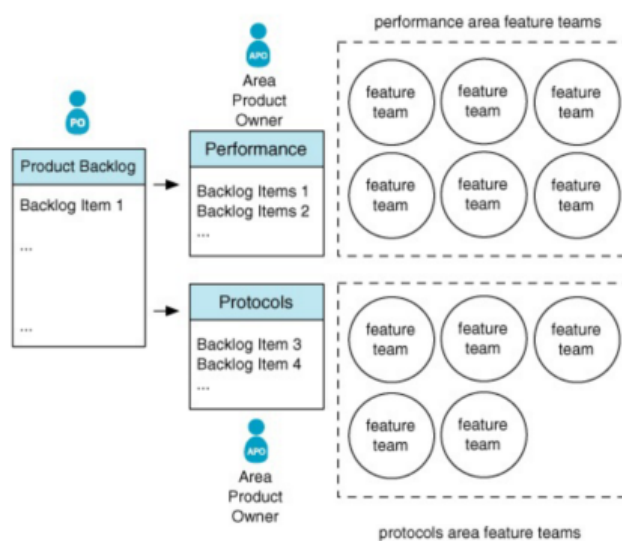


Figure 2.2: LeSS Product Owner Team and Feature Teams[62, p.265]

## 2.2.3 Scaled Agile Framework (SAFe)

The Scaled Agile Framework (SAFe) is a knowledge-base of principles, practices and competencies to guide enterprises scaling agile and lean practices to their needs [23].

According to [108] beside Scrum (or a hybrid that contains Scrum) SAFe is solid in the lead of the most commonly used frameworks (30% report that SAFe is the approach their organization follows).

**Product Owner in SAFe** According to SAFe, the Product Owner should collaborate closely with Product Management as well as the development team. Safe recommends to have a so called *fan-out model* including one Product Manager who works with up to four Product Owners, while each Product Owner is responsible for the Product Backlog for a maximum of two teams [23]. While the Product Manager is responsible for providing the vision, roadmap and the program backlog, the Product Owner acts more on a team level as he owns the team backlog, prioritizes it and accepts/reject the deliverables from his team(s).

The structure of the Product Owner, Product Manager and Development Team is represented in Figure 2.3:

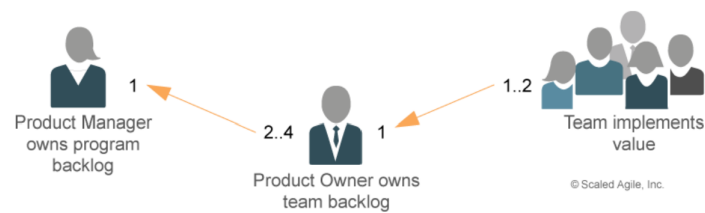


Figure 2. Fan-out model for Product Manager, PO, and Agile teams

**Figure 2.3:** Fan-out model for Product Manager, PO, and Agile teams [23]





# 3

## Hybrid Development

The overall goal of this thesis is to support the tailoring of the Product Owner role to hybrid development environments. To understand the importance as well as difficulty to achieve this, this chapter addresses that customized hybrid development approaches are state of the art, how related process tailoring is addressed in industry as of now and provides a definition for hybrid development environments for this thesis.

### 3.1 Hybrid Development as State of the Art

It is common knowledge that software processes are tailored to the particular environment of a project [4, 5, 43, 56, 76, 118, 119]. While directly applying an existing process without any tailoring is perceived as a project risk [4, 5, 43, 76], tailoring existing processes – rather than creating them from scratch – provides the benefits of already defined roles and activities [119]. This also applies to the agile frameworks and methods. Although they are promising methods to deliver high quality software at a high pace [12, 26, 59], they can have situation-dependent shortcomings that can lead to a project failure if they are not considered [15]. Furthermore, agile practices are particularly difficult to integrate in large scale organizations that combine software and hardware production [14].

Consequently, it is not surprising that nowadays hybrid development methods are state of the art [50]. Here software development projects are not isolated activities [44, 45]. They usually exist as sub-projects in an environment composed of hardware development, marketing and production planning, which all must be managed and coordinated concurrently [22]. According to the results of the large scale HELENA<sup>1</sup> study [58], nearly three of four companies follow a hybrid development approach combining different development methods and practices [50]. These customized hybrid methods often include traditional and agile development methods and practices [101], whereby Scrum, Iterative Development, Kanban, Waterfall and DevOps are the most frequently used frameworks and methods [50].

The adaption and tailoring of a process is commonly motivated by explicit goals, such as

---

<sup>1</sup>HELENA: Hybrid dEveLopmENt Approaches in software systems development, online: <https://helenastudy.wordpress.com>

improving productivity, quality as well as the planning and estimation [50], or is driven by the needs of the client, market, business or issues with the standard process of the company [50].

### 3.2 Software Process Tailoring for Hybrid Software and System Development in Industry

What factors influence the tailoring process, and hence the development of a customized hybrid development approach, has been subject to research for quite some time already. As a result multiple approaches to support this process have evolved.

To overcome shortcomings of both, traditional and agile methods, Boehm and Turner [16] proposed to consider balancing agile and plan-driven methods. To do so, they provided a method to assess the profile of a project or organization in order to apply an appropriate balance between agile and plan-driven methods [16]. The assessment of five critical factors determines whether a plan-driven or agile approach would fit best to the particular project. The determination is based on a set of conditions under which each development approach is most likely to succeed – so called "home grounds". Figure 3.1 represents the five factors in the *Home Ground Polar Chart*.

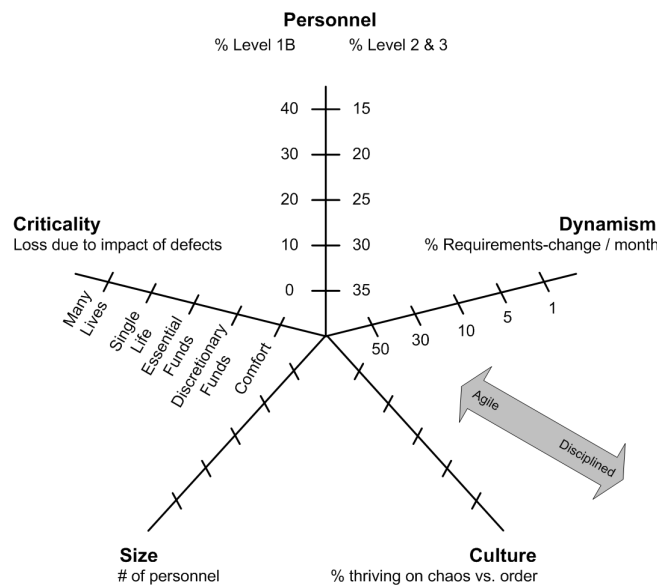


Figure 3.1: Home Ground Polar Chart (based on [16])

Tiwana and Keil developed a one-minute risk assessment tool to identify the most critical area of a project so that certain actions can be taken to reduce the risk of failure in this area [102].

Based on this, Xu and Ramesh [119] developed a challenge assessment questionnaire for practitioners to assess the relative importance of challenges that need to be tailored. Additionally, they provide strategies on how the identified challenges can be addressed [119].

Kalus and Kuhrmann conducted a systematic literature review on this topic and investigated on concrete tailoring criteria for the tailoring of the software process [43]. They present an overall collection of 49 tailoring criteria which are clustered into four categories: Team, Internal Environment, External Environment, Objective [43]. Furthermore, they provided 20 actions

that can be used to tailor the process and hence, to overcome the risk of the failure of the project [43]. However, the results of their study state that, although the influencing factors on the tailoring process are well understood, the initial criteria to decide for the right action items are often interpreted on a project-to-project basis [43].

This coincides with the findings from Kuhrmann et al. [57]. They conducted a survey on hybrid software development approaches and report that 83.9% of them emerge from experience and learning from past projects, whilst only 33.9% of the project-specific tailoring follows defined rules and that a project manager carries out the tailoring in the beginning of a project.

According to the above, the topic of software process tailoring is considered as an important aspect for this thesis and its knowledge is reused to build the research artifact resulting from the design science research approach: the *HyPro Expert System*. The application of this knowledge is further discussed in Chapter 13.

### 3.3 Definition of Hybrid Development Environment

Although hybrid development approaches seem to be state of the art and how software processes can be tailored have been subject to research, there is a lack of a clear definition of 'hybrid development'. One definition for hybrid software development approaches has been provided by Kuhrmann et al. [57, p.2]:

”A hybrid software development approach is any combination of agile and traditional (plan-driven or rich) approaches that an organizational unit adopts and customizes to its own context needs (e.g., application domain, culture, processes, project, organizational structure, techniques, technologies, etc.).”

However, this definition does not quite fit to this thesis. This is because the goal of this thesis is to support the tailoring of the Product Owner role to hybrid development **environments**, rather than plain software development. Furthermore, the above definition seem to require to combine agile and traditional software development methods – which is not necessarily the case in hybrid development environments. Here, according to the understanding of the author of this thesis, multiple development approaches can coexist and may (but does not necessarily) require certain aspects to be altered. However, the altering of a process does not necessarily require to combine agile and traditional methods but can be achieved through scaling or tailoring of certain aspects.

Therefore, the author of this thesis defines hybrid development environments as follows:

**A hybrid development environment**  
describes a product development process, where multiple development approaches coexist.  
This can require agile software development methods / frameworks to be  
scaled, and / or tailored, and / or combined  
with traditional development approaches to its own context needs.



# 4

## Part I Summary

This part provides the basic descriptions of the Product Owner role in the most important methods and frameworks. Table 4.1 summarizes the main tasks, characteristics and structures of Product Owners described in this chapter. The overview provided by this table makes clear that the implementation of the Product Owner role depends heavily on the method or framework.

However, the application of a single development method or framework is seldom in practice. In fact, customized hybrid development approaches, where at least two frameworks/methods are combined and further tailored to better accommodate project needs, are state of the art. For a better understanding of the considered hybrid development environment for this thesis, this part provides the following definition: *A hybrid development environment describes a product development process, where multiple development approaches coexist. This can require agile software development methods / frameworks to be scaled, and / or tailored, and / or combined with traditional development approaches to its own context needs.*

Overall, what is missing is a combination of both: A consolidated description of Product Owner tasks, characteristics & structures and how they are applied in hybrid development environments.

To close this gap, the goal of this thesis has been stated as follows: analyze the Product Owner tasks, characteristics and structures to deduce recommendations to tailor the Product Owner role to hybrid development environments. Therefore, the here discussed methods and frameworks provide meaningful information to describe the Product Owner tasks, characteristics and structures in general. This knowledge provides a necessary basis to achieve the overall goal of this thesis. Therefore, the author of this thesis extracted descriptive phrases from this part as part of a first cycle coding method [86] to split the relevant information into segments for later stages of data analysis. The descriptive phrases of Product Owner tasks, characteristics & structures are summarized in Table 4.1.

**Table 4.1:** Descriptive Phrases of Product Owner Tasks, Characteristics & Structures of Part I

<b>TASKS</b>	<b>Source</b>
Represent customer needs	Scrum
Requirements elicitation	Scrum
On-Site Customer makes business decisions	XP
Coordinate sprint	Nexus
Clarify requirements	LeSS
Make decision on cost, schedule and content	LeSS
Accepts/reject the deliverables from team(s)	SAFe
Collaborate closely with management	SAFe
Collaborate closely with development team	SAFe
Manage / owns the backlog	Scrum, XP, Nexus, LeSS, SAFe
Prioritize backlog	Scrum, XP, SAFe
<b>CHARACTERISTICS</b>	<b>Source</b>
Leader of the process	Nexus
Leader of PO team (Overall PO)	LeSS
Decision Maker for assigned team (Area PO)	LeSS
<b>STRUCTURES</b>	<b>Source</b>
Represented by a single Person	Scrum, Nexus
Nexus team (Product Owner, Scrum Master, Nexus Integration Team Members)	Nexus
Group of people rather than a single person	LeSS
PO team (PO, APO and maybe others to clarify requirements)	LeSS
Work with product management	SAFe
Product Owner can have up to 2 development teams	SAFe
Fan-out-model (1 Product Manager with up to 4 Product Owners)	SAFe

The descriptive phrases are used again in Chapter 13.1 to incorporate and conclude the knowledge to deduce recommendations to support the tailoring of the Product Owner role to hybrid development environments.

## Part II

# Structuring the Area of Research





---

<b>5</b>	<b>Related Work on the Product Owner Role in Industry</b>	<b>25</b>
5.1	Tasks and Characteristics . . . . .	<b>25</b>
5.1.1	Tasks . . . . .	25
5.1.2	Characteristics . . . . .	26
5.1.3	Reference Model of Tasks & Characteristics . . . . .	27
5.2	Product Owner Teams – Terminology and Structures . . . . .	<b>28</b>
<b>6</b>	<b>Identifying Research Directions – A Mapping Study on Product Owners in Industry</b>	<b>31</b>
6.1	Research Objective . . . . .	<b>31</b>
6.2	Research Design . . . . .	<b>31</b>
6.2.1	Research Questions . . . . .	32
6.2.2	Research Method . . . . .	33
6.2.3	Execution . . . . .	34
6.2.4	Data Extraction . . . . .	35
6.3	Results . . . . .	<b>36</b>
6.3.1	Addressed Research Topics . . . . .	37
6.3.2	Results from Addressed Research Topics . . . . .	39
6.3.3	Consideration of Environment in Research . . . . .	42
6.4	Identified Research Directions for this Thesis . . . . .	<b>43</b>
<b>7</b>	<b>Part II Summary</b>	<b>45</b>

---



# 5

## Related Work on the Product Owner Role in Industry

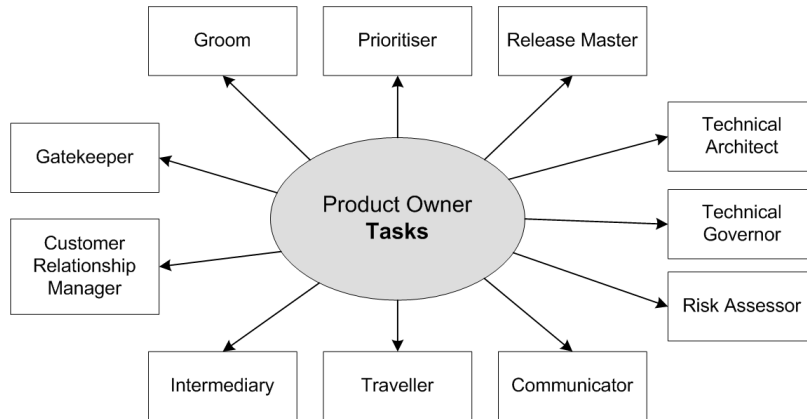
As reflected in Chapter 3 customized hybrid development approaches are state of the art. Meanwhile, each framework or method that is used to create these hybrid development approaches provides an individual description of the Product Owner role. Within the past years, some research has been conducted to better understand the implementation of the Product Owner role in practice. An overview of the terms used for the various Product Owner roles as well as study results on the actual tasks and characteristics of this role (in practice) are presented in this chapter.

### 5.1 Tasks and Characteristics

As described in previous sections, it remained unclear *how* the Product Owner can achieve the goal of maximizing the value of the product. The following summarizes related work on the topics of Product Owner tasks, characteristics and Product Owner team structures in large companies.

#### 5.1.1 Tasks

Bass et al. [6, 8] provide insights in the tasks and activities of Product Owners in large companies – and asserts that the combination of the identified tasks helps to tailor the software process [6]. Over the course of 3 years, they performed qualitative studies where they observed and interviewed 55 practitioners in 10 companies [8]. Initial results were presented in 2015 by Bass [6] and reported an identification of nine Product Owner team functions: groom, prioritiser, release master, technical architect, technical governor, communicator, traveller, intermediary and risk assessor. In 2018, Bass et al. [8] reported 8 distinct Product Owner activities [8] whereas 6 activities are in conformity with the results of 2015 and two new activities (gatekeeper and customer relationship manager) have been identified. All activities are represented in a Product Owner taxonomy in Figure 5.1.



**Figure 5.1:** Product Owner taxonomy (based on [6, 8])

The **Groom** clarifies the details of product backlog items and their respective acceptance criteria.

The **Prioritiser** selects requirements that bring highest value or benefit.

The **Release Master** manages and approves release plans.

The **Technical Architect** designs, implements and disseminates a reference architecture between the Scrum Teams.

The **Technical Governor** provides technical governance framework to project teams in order to ensure usage of common tools and technologies for the project.

The **Risk Assessor** evaluates technical complexity and potential shortcomings in the development teams' skills and capabilities.

The **Communicator** transfers knowledge between onshore and offshore sites.

The **Traveller** gathers an understanding of a client's needs by spending time onshore at customer sites.

The **Intermediary** acts as an interface between senior roles, and the team, to disseminate domain knowledge.

The **Gatekeeper** determines feature or story completeness for inclusion in a release.

The **Customer Relationship Manager** provides technical support to customers, assists with site preparation and product installation, and does product training.

### 5.1.2 Characteristics

Pilcher [79] attempted to generate a practical guide that enables new Product Owners to apply agile product management techniques effectively in Scrum. Furthermore, he describes five desirable characteristics of Product Owners, addresses common mistakes when applying this role and suggests a team of Product Owners when it comes to scale this role to large projects. The described characteristics are as follows:

**(1) Communicator & Negotiator** The Product Owner communicates with and aligns different parties including customers, users, development and engineering, marketing, sales, service, operations, and management.

**(2) Visionary & Doer** The Product Owner envisions the final product and sees it through to completion. This includes requirements description, closely collaborating with the team, accepting or rejecting work results, and steering the projects by tracking and forecasting its progress.

**(3) Leader & Team player** The Product Owner is responsible for the product's success, pro-

vides guidance for everyone involved and makes tough decisions. He needs to be a team player, rely on close collaboration with other Scrum Team members, yet has no formal authority over them.

(4) **Available & Qualified** Being a Product Owner is usually a full-time job. Project’s progress suffers when the Product Owner is overworked. Being adequately qualified usually requires an intimate understanding of the customer and the market.

(5) **Empowered & Committed** An empowered Product Owner is essential to bring the product to life. The Product Owner must have the proper decision-making authority – from finding the right team members to deciding which functionality is delivered as part of the release.

### 5.1.3 Reference Model of Tasks & Characteristics

Based on the initial results on the Product Owner tasks by Bass [6] in 2015 and the described characteristics by Pichler [79], Unger-Windeler et al. [103] proposed a reference model (see Figure 5.2). As Pichler [79] and Bass [6] do not rate the entities, all of them are considered as equally important. However, some of the characteristics are seen more related to certain tasks (e.g. Communicator & Negotiator), while others are required on the full range (e.g. Visionary & Doer).

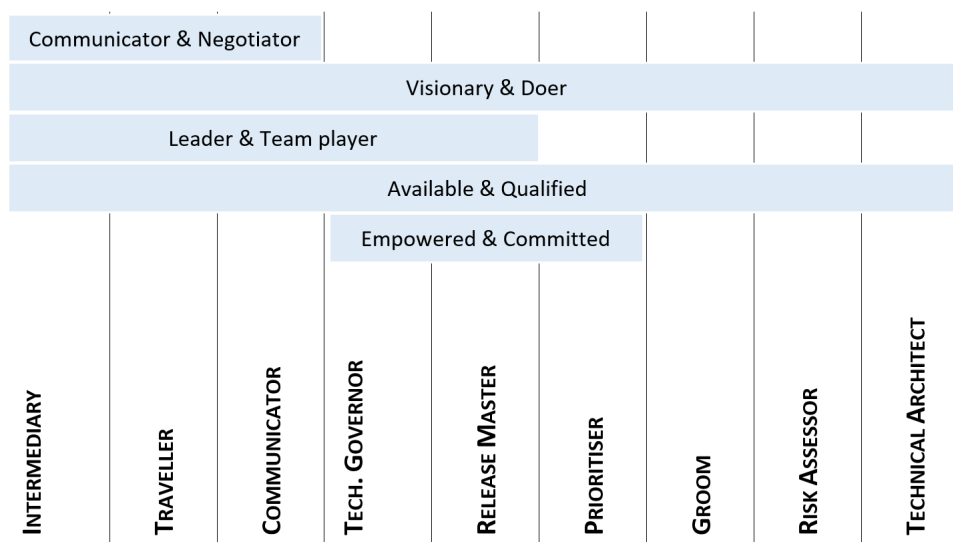


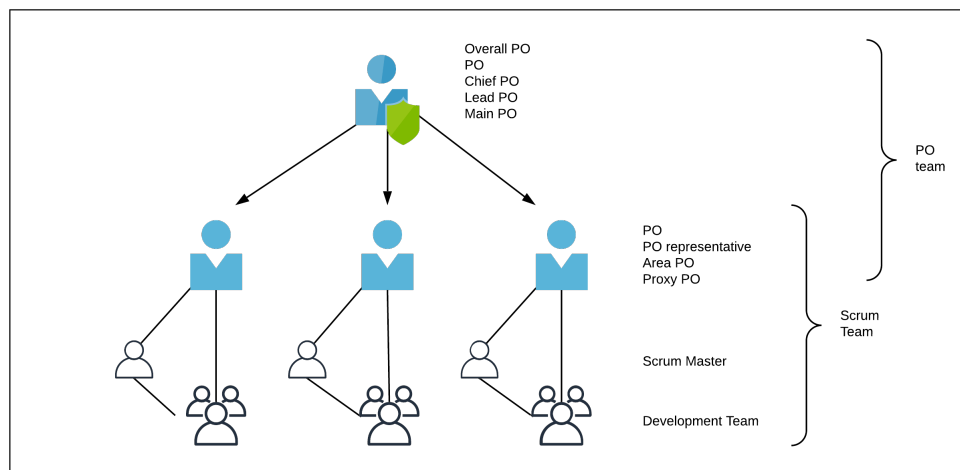
Figure 5.2: Reference Model Tasks & Characteristics [103]

The reference model was applied in a case study in the oil and gas industry to check whether the characteristics and functions of Product Owners are also present in this domain. Preliminary results indicate that the tasks differ and that not only the size of the company, but the organizational structure might have an impact on the Product Owner role as well. The reference model and the results of the case study have been published in [103] and presented at the *International Conference on Product-Focused Software Process Improvement (PROFES)*.

## 5.2 Product Owner Teams – Terminology and Structures

According to related literature, the Product Owner has to perform a variety of multiple tasks and hence, has to be a certain character [6, 8, 79]. Especially in large-scale environments the breadth of functions extend beyond the scope and skill set of a single individual [6]. To close this gap, the concept of Product Owner teams has been introduced by Larman and Vodde [62] (see Section 2.2.2). Within that concept tasks and responsibilities can be shared among members of this group and has been identified as a promising scaling solution (see Section 6.3.2).

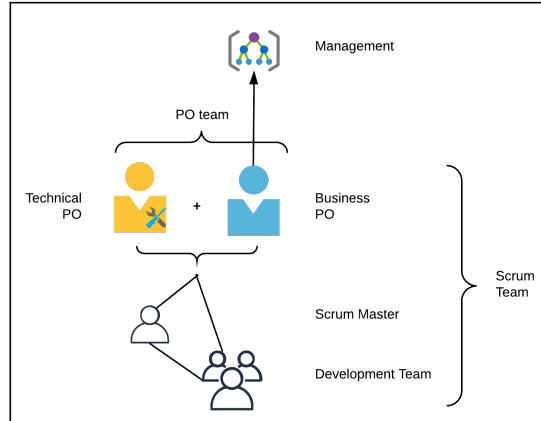
The concept of Product Owner teams has been addressed by a couple of researchers. While a few have coined individual terms for the (different) Product Owner role(s), other have just quoted these terms. As Schwaber [90] has been one of the first, who has defined different levels of the Product Owner role in 2007, his expression – overall Product Owner – is used as the base-term in the following. Others who have coined the terms, basically scaled the term *Product Owner* up or down. For instance, the term *Product Owner* means an overall Product Owner and a scaled down variant is named as *Area Product Owner* or *PO representative* [60, 61, 62]. These scaled down variants have a supporting function for the overall Product Owner. Concurrently, others have scaled up the term Product Owner to "Lead Product Owner" [27] "Chief Product Owner" [20, 79], whereas their supporting Product Owners are simply called Product Owners. Others again used an up-scaled version "Main Product Owner" together with down-scaled versions "Area / Proxy Product Owner" [75]. However, they all share a hierarchical concept. For this thesis, the author concluded all of the terms and concepts above and named it '*The hierarchical Product Owner Team*'. The structure of a hierarchical Product Owner Team is represented in Figure 5.3.



**Figure 5.3:** The hierarchical Product Owner Team

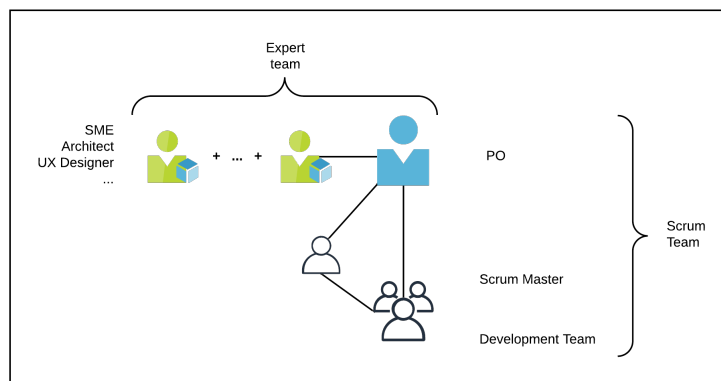
A non-hierarchical Product Owner team structure has been identified by Bass [6]. Here, the team is split into "Business Product Owner(s)" and "Technical Product Owner(s)" which share the identified tasks from Section 5.1. While the "Business Product Owner" mainly focuses on the gathering and prioritizing of requirements, the "Technical Product Owner" focuses on the software production practices such as coding standards and architecture. For this thesis, the author named this concept '*The non-hierarchical Product Owner Team*'. The structure is

represented in Figure 5.4.



**Figure 5.4:** The non-hierarchical Product Owner Team

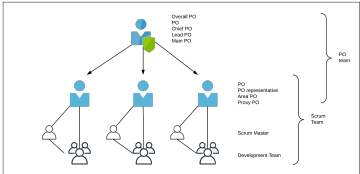
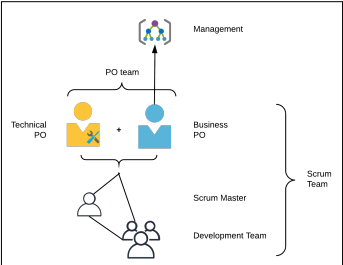
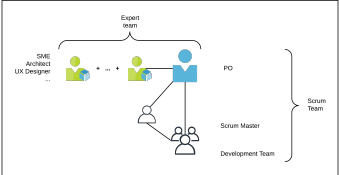
Another combination is introduced by the "Product Owner" together with other "Specialized Roles" (e.g. Subject Matter Experts, Architects, UX Designers) [25, 64, 106]. Here, the specialized roles do not share any responsibilities with the Product Owner, but they support the Product Owner by sharing their specialized knowledge. For this thesis, the author named this concept '*The Expert Team*'. The basic structure is represented in Figure 5.5.



**Figure 5.5:** The Expert Team with Specialized Roles

For a better understanding of the described structures and used terminology, an overview is presented in Table 5.1.

**Table 5.1:** Product Owner Team Structures and Terminology

Team Structure	Terminology	Description	Source
<p data-bbox="236 383 630 409">Hierarchical Product Owner team</p> 	Overall PO & PO	The <i>overall PO</i> is <b>supported by</b> the <i>PO(s)</i>	[90]
	Chief PO & PO	The <i>chief PO</i> is <b>supported by</b> the <i>PO(s)</i>	[20, 79]
	Lead PO & PO	The <i>lead PO</i> is <b>supported by</b> the <i>PO(s)</i>	[27]
	Main PO & Area / Proxy PO	The <i>main PO</i> is <b>supported by</b> the <i>Area / Proxy PO(s)</i>	[75]
	PO & PO representative / Area PO	The <i>PO</i> is <b>supported by</b> the <i>PO representative(s)</i> or <i>Area PO(s)</i>	[60, 61, 62]
<p data-bbox="209 956 657 983">Non-hierarchical Product Owner team</p> 	Business PO & Technical PO	The <i>business PO</i> works together with the <i>technical PO</i>	[6]
<p data-bbox="357 1292 504 1319">Expert team</p> 	PO & Specialized Role (e.g. Subject Matter Expert, Architect, UX Designer)	The <i>PO</i> works together with the <i>Specialized Role</i> . This team structure can be applied in a hierarchical as well as non-hierarchical Product Owner team structure.	[25, 64, 106]



# 6

## Identifying Research Directions – A Mapping Study on Product Owners in Industry

### 6.1 Research Objective

The previous chapters 2 and 5 highlight the various descriptions of the Product Owner role and provides an insight on the actual implementation of this role in practice – yet, a conclusive bigger picture of the studies and reports on this issue is missing. To fill this gap and to be able to move forward in the right direction with this thesis, a systematic mapping study was conducted by the author of this thesis (and others<sup>1</sup>). The findings structure the research area of *Product Owners in industry* in terms of research topics and applied research methods. In total, 30 contributions were identified that addressing seven research topics and generated consolidated answers for each of them. While some of those topics provide congruent results, others point to gaps in current research. The results of this study have been published in [104] and presented at the *International Conference on Software and Systems Process (ICSSP)* as well as in the *Journal of Software: Evolution and Process* [105].

### 6.2 Research Design

To achieve an overview of aspects regarding the role of Product Owners that have already been addressed by researchers, a mapping study has been conducted. This method primarily enables to structure a research area and to identify gaps and possibilities for future research [77, 78]. In addition, a mapping study allows to gain a wide overview of the research area [48]. The research process described by Petersen et al. [77] served as a basis for this research and will be described in this section. However, this process has been modified by adding the step of snowballing search as visualized in Figure 6.1. By including this step, a more comprehensive list of relevant papers has been achieved.

---

<sup>1</sup>Jil Kluender and Kurt Schneider

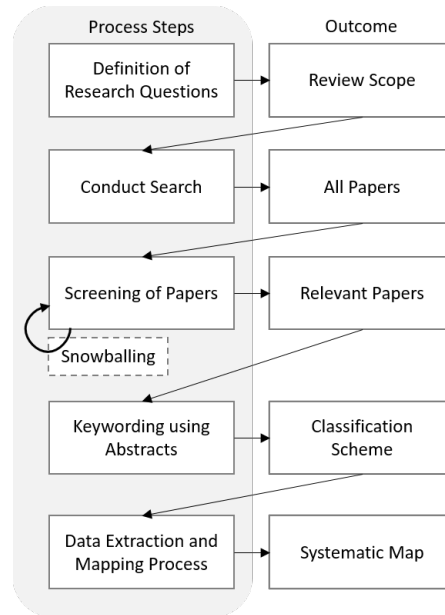


Figure 6.1: Modified Research Process

### 6.2.1 Research Questions

In contrast to systematic reviews where a very specific goal has to be formulated, the research questions in mapping studies are more general as they aim to discover research trends [78]. The main research question of this mapping study is:

**Research question (RQ):**

**Which aspects of the role of Product Owners and On-site Customers in industry have been subject to research?**

In particular, we are interested in answering the following subquestions:

*Research question 1: What is the current research on the Product Owner role?*

We view this question in two parts:

*RQ 1.1: What topics regarding the Product Owner role in industry are addressed in research?*

*RQ 1.2: What research method was applied to investigate these topics?*

Answers to these questions help us to classify and structure the research in this area.

*Research question 2: What insights about Product Owners are presented in research?*

To get the maximum value out of this research, we preserve and summarize the knowledge shared in the considered publications. Therefore, we structure the content based on the addressed research topics identified in research question 1.

*Research question 3: What external circumstances of the Product Owners' environment have been analyzed in research?*

While practitioners have an increased interest in tailoring agile software methods to large-scale offshore enterprise development programs [1, 62, 63], a co-located development team with

approximately nine members is ideal to work with [92]. Hence, we were interested in whether these external circumstances are considered in current research. In particular, we want to analyze current literature with respect to the following aspects which are part of the *environment*.

RQ 3.1: *What organizational structures are considered in literature?*

RQ 3.2: *What company sizes are considered in literature?*

RQ 3.3: *What team dimensions are considered in literature?*

RQ 3.4: *What team locations are considered in literature?*

## 6.2.2 Research Method

This section describes the process followed to conduct the search according to Petersen et al. [77].

**Defining the search string** The keywords were identified using PICO (Population, Intervention, Comparison, Outcomes) [48]. The PICO criteria were developed to identify keywords and formulate search strings from research questions [78].

**Population:** Population may refer to a specific software engineering role or a category of software engineering [78]. In our context, the populations are defined by *Product Owners and On-site Customers in industry*.

**Intervention:** In software engineering, intervention may refer to a software methodology [78]. In the context of this study, we investigate on the intervention of *two agile software development methods: Scrum and eXtreme Programming*.

**Comparison:** This is the software engineering methodology the intervention is compared to [48]. We do not compare two methodologies, but we rather compare findings of current research with regards to research topics addressed in literature (RQ1), the key findings (RQ2) and the environment (RQ3).

**Outcomes:** Outcomes should relate important factors such as reduced production costs or reduced time to market [48]. In our context, we expect measurable results in terms of *research gaps or saturation*.

Combining these considerations, we identified the keywords *Product Owner, Industry* and *Agile*. For these three keywords, we used the synonyms and abbreviations shown in Table 6.1.

**Table 6.1:** Overview search keywords

Search keyword 1	"Product Owner(s)"
Abbreviation	"PO(s)"
Related keywords	"On-Site Customer(s)" "Product Manager(s)"
Search keyword 2	"Industry" / "Industries"
Related keywords	"Organization(s)" "Practice"
Search keyword 3	"Agile"
Related keywords	"Scrum" "eXtreme Programming"
Abbreviation	"XP"

The keywords were used to formulate the search string:

*("product owner" OR "product owners" OR "PO" OR "POs" OR "product manager" OR "product managers" OR "on-site customer" OR "on-site customers" )*  
**AND** *("industry" OR "industries" OR "organization" OR "organizations" OR "practice")*  
**AND** *("agile" OR "scrum" OR "extreme programming" OR "xp")*

**Selection of Sources** Seven sources were selected for the mapping study: ACM Digital Libraries, Springer Link, Science Direct, IEEE Xplore, Google Scholar, Wiley Online Library and Scopus. With these sources, a comprehensive search was conducted [77]. Although Scopus covers IEEE Xplore and Elsevier, the two databases were included to verify the quality of the results. Based on the selected sources, the search string was adapted to the specific needs of each search engine.

**Study Selection** The study selection was processed in four iterations. We excluded studies and publications which are not relevant for our mapping study. To make this decision more objective, we defined the inclusion and exclusion criteria as summarized in Table 6.2.

**Table 6.2:** Inclusion and Exclusion Criteria

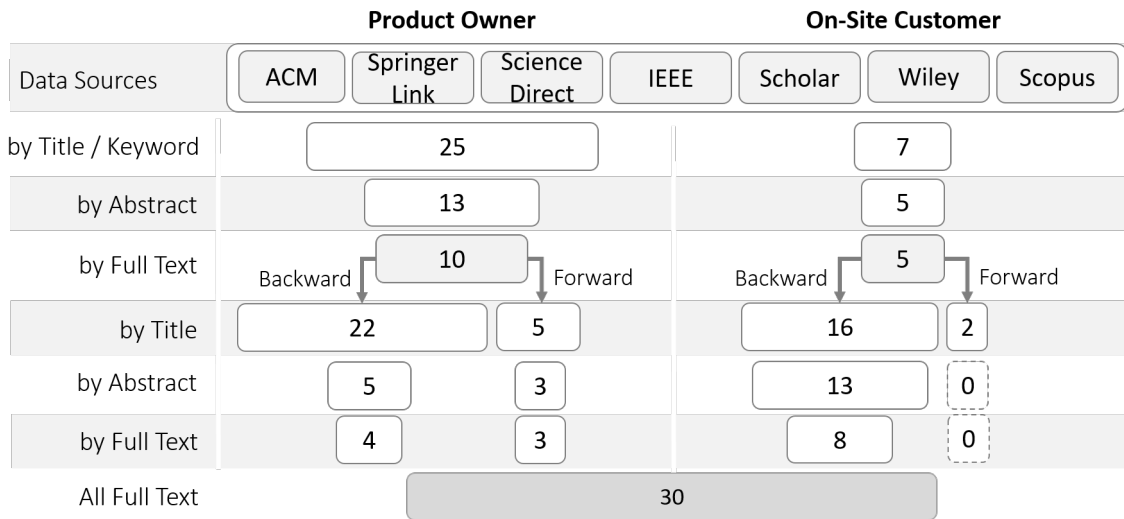
Inclusion	Paper discusses Product Owner in general Paper discusses Product Owner in industry / organizations Paper is a peer-reviewed contribution to a conference or a journal Publication is a master- or PhD-thesis or a technical report
Exclusion	Paper has no accordance with Product Owner and On-site Customer Duplicated papers Paper is not accessible Paper was written before 2001 Paper is not written in English or German

In the first iteration, we included articles based on title and keywords. In the second iteration, we filtered by abstracts before we read the full article in the third iteration. The studies remaining after this iteration were used as starting set to for the backward and forward snowball search in the fourth iteration. These steps are visualized in Figure 6.1 and Figure 6.2.

### 6.2.3 Execution

We executed the mapping study as described in the previous section. An overview of the process and the number of papers is shown in Figure 6.2.

In the first iteration, we identified 25 papers addressing the Product Owner and 7 addressing the On-site Customer (without duplicates) based on their titles and keywords. In the second iteration, we filtered the papers with respect to the inclusion and exclusion criteria based on abstract and keywords. This led to an exclusion of 3 papers addressing the Product Owner. The review of the full text in the third iteration resulted in the preliminary set of 10 (Product Owner) and 5 (On-site Customer) papers. However, we used these sets as *starting set* to conduct



**Figure 6.2:** Search process and filtering steps.

the snowballing sampling. As described by Wohlin et al. [114], we considered the references of the papers and the papers in which at least one of these 15 papers is cited. We applied the same process of the iterations 1 to 3 with the set of papers found during snowballing, i.e., we filtered the papers with respect to the inclusion and exclusion criteria and considered the title, keywords, abstract as well as the full text in the according iterations. Eventually, this led to the final set of 10+7 (Product Owner) and 5+8 (On-site Customer) papers which provides an insight in the current research status. The full list of references can be found in Appendix A.

## 6.2.4 Data Extraction

To extract data from the identified studies, we developed the form shown in Table 6.3.

Each data extraction field has a data item, value and if applicable, is mapped to the corresponding research question. The extraction was performed by the author of this thesis and reviewed by another researcher by tracing back the information in the extraction form to the statements in each paper and checking their correctness. The extracted data of each item was tabulated and is visually presented in Section 6.3.

### 6.2.4.1 Threats to Validity

The outcome of our mapping study is biased by different factors. We will discuss the threats to internal, external, construct and conclusion validity in the following [114].

**Internal validity** As the study was mainly executed by one researcher, the decision on the inclusion or exclusion of a paper mainly depended on one opinion and hence was subjective. In order to reduce this bias, we formulated criteria for inclusion and exclusion of a paper. Additionally, the form represented in Table 6.3 objectified the data extraction process and was revisited by the second researcher. This retains a certain objectivity of the results.

**Table 6.3:** Data extraction form

DATA ITEM	VALUE	RQ
<i>General</i>		
Title	Name of the article	
Author Name	Set of names of authors	
Year of Publication	Calendar year	
Study ID	Integer	
<i>Content</i>		
Research Topic	What topics are addressed	RQ1.1
Research Method	What research method is applied	RQ1.2
Content	What is the content of the contribution	RQ2
External Circumstances	Boolean	RQ3
Organizational Structures	Integer & flat / matrix / top-down	RQ3.1
Company Size	Integer & small / medium / large	RQ3.2
Team Dimension	Integer & small / medium / large	RQ3.3
Team Location	Integer & distributed / co-located	RQ3.4
<i>Agile Methodology</i>		
Methodology	Scrum / XP / non or both	

**External validity** We cannot guarantee that we have found all papers to structure the research area completely. To mitigate this threat, we performed the snowballing step which led to an inclusion of 15 more papers.

**Construct validity** The construction of the mapping study depends on the definition of the research questions, the resulting search string as well as the selection of the sources. Although we used PICO to generate the keywords defining the search string, we cannot guarantee that we have considered all related keywords and synonyms. As the product manager role is often closely related with product ownership we added "Product Manager(s)" as a related keyword to mitigate this threat. Nonetheless, there may be other synonyms and a different or extended search string probably would have led to different results. However, the papers included in the analysis draw a broad picture of the current state of research.

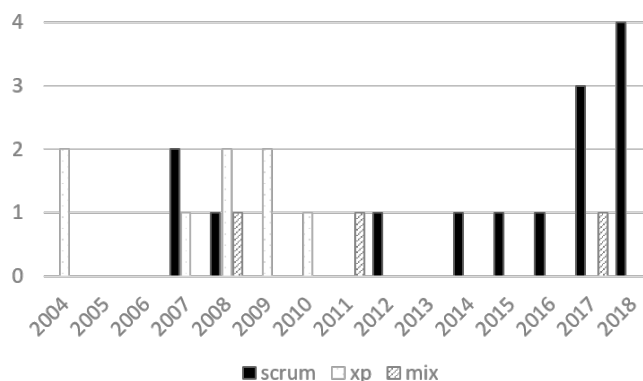
The results also depend on the selection of the sources. Some publications are found by more than one search engine while others are not. To reduce the threat of missing publications we conducted the search on seven data sources which are often used in literature reviews.

**Conclusion validity** Conclusion depends on the obtained data which is based on the construction and external validity. For our purpose – structuring the research area in regard to the Product Owner – as well as to identify gaps in current research, we are confident that the data were sufficient.

## 6.3 Results

As visualized in Figure 6.2, we considered a total amount of 30 papers which have been published over the last 15 years. There has been an increased interest in the matter of Product

Owner / On-site Customers in 2007, 2008 and 2009. However, the most attention was gained in 2017 and 2018 as can be seen in Figure 6.3.



**Figure 6.3:** Publications by year.

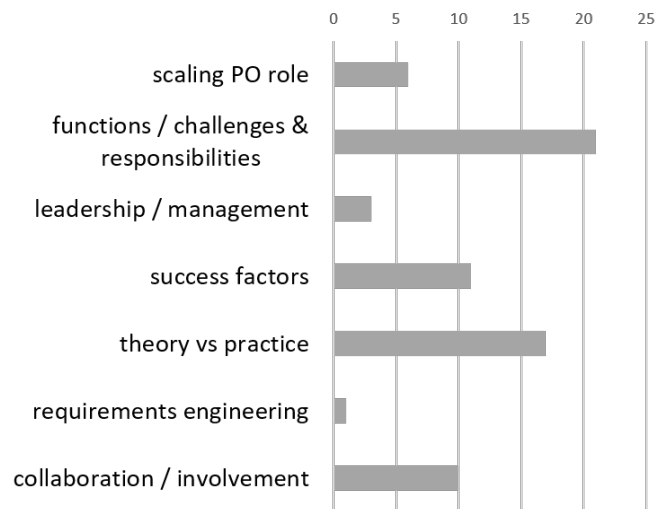
In the following subsections, we present our results according to the research questions. We present the extracted data based on the Table 6.3. The results from the mapping study are visually prepared.

### 6.3.1 Addressed Research Topics

The first research question aimed to provide an overview of the addressed research topics regarding the Product Owner role in industry and the research methods applied to answer them.

We categorized the addressed questions and identified seven topics. They are addressed in the context of Scrum, XP or for agile methodologies in general. Also, we identified seven applied research methods. Unfortunately, in three contributions we were not able to determine the research method as it has not been described sufficiently.

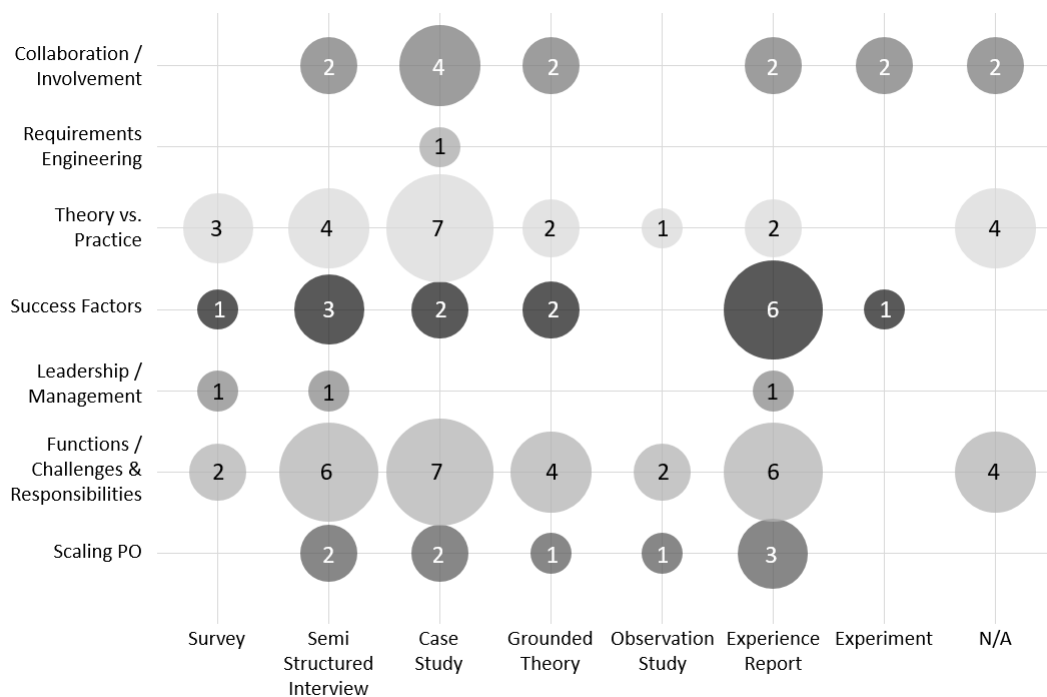
Most publications are addressing the *Functions / Challenges & Responsibilities* (21), compared *Theory vs. Practice* (17) and reported *Success Factors* (11) of Product Owners and On-site Customers. The least attention was paid to the topic of *Requirements Engineering* (1) practices of Product Owners / On-site Customers. An overview of the topics is visualized in Figure 6.4.



**Figure 6.4:** Research topics addressing the Product Owner / On-Site Customer Role in literature.

Most of the researchers collected their data with a case study (27%), conducted semi-structured interviews (18%) or shared their own experiences in an experience report (18%).

To get a comprehensive overview of what methods have been used to answer the research questions to the according topics, we mapped the results in Figure 6.5.



**Figure 6.5:** Systematic Map of Research Question 1.1 and 1.2.

Table 6.4 maps the references in regard to the addressed agile practices and research topics. It visualizes that the research topics 'scaling Product Owner role' and 'requirements engineering'



are considered in Scrum only, while 'theory vs. practice' as well as 'collaboration / involvement' mostly consider XP practices. Three contributions considered agile practices as a whole and did not specify any framework. They are summarized in the 'mix' category.

**Table 6.4:** Publications by Research Topic and Methodology

TOPIC	SCRUM	XP	Mix
Scaling PO Role	[6, 7, 25, 32, 64, 75]		
Functions / Challenges & Responsibilities	[6, 8, 25, 55, 81, 100, 103] [7, 64, 70, 74]	[29, 67, 72, 112] [28, 68, 69, 111, 113]	[37, 41]
Leadership / Management	[100]		[41, 94]
Success Factors	[6, 25, 64, 74, 75, 81]	[28, 69, 113, 116]	[37, 41]
Theory vs. Practice	[7, 33, 70, 74, 100, 103]	[29, 53, 67, 72, 112] [28, 66, 111, 113]	[94]
Requirements Engineering	[33]		
Collaboration / Involvement		[53, 68, 69, 113] [39, 72, 112, 116]	[37, 41]

### 6.3.2 Results from Addressed Research Topics

After having identified what research topics are addressed in current literature, we were interested in the answers they provide.

**1.) Scaling Product Owner role** When scaling the role of the Product Owner to large projects some publications (e.g., [6, 8, 25, 75]) report the concept of Product Owner teams as helpful. However, the roles and hierarchical structures within the Product Owner teams differ. While Bass et al. [6, 8] basically distinguish between technical and governance Product Owner roles, Paasivaara et al. [75] report hierarchical structures containing one Main / Chief Product Owner (CPO) and multiple Area Product Owners (APO) or Proxy Product Owners (PPO). They distinguish customer-focused and technically-focused Product Owner roles. And yet Croix and Easton [25] defined their Product Owner teams consisting of more diverse roles (such as customers, designers, analysts, security experts, and operations experts).

Rather than forming formal Product Owner teams, other researchers highlight the importance of a working communication structure within various roles to support the Product Owner: Gupta et al. [32] report good experiences in using "Obeya Wall" to communicate and define focus areas of Project Managers / Scrum Masters and Product Managers / Product Owners, while Lowery and Evans [64] propose one Product Owner for the bigger picture, who is supported by multiple in-team Subject Matter Experts (SMEs).

When it comes to smaller companies, Bass et al. [7] identified the concept of a Product Owner team as not feasible.

**Takeaway: In large-scale projects the Product Owner role is a group effort.**

**2.) Functions, Challenges & Responsibilities** When describing the tasks of Product Owners, some authors categorized them as a function or responsibility while others emphasized them as a challenge for the Product Owner. To improve readability, here we summarize them simply as activities. In total 21 different activities of Product Owners were mentioned in the considered literature. The most frequently named activity of Product Owners is *Communication* [6, 7, 8, 41, 70, 72, 81, 100, 103].

An overview of all identified activities and the references is presented in Table 6.5.

**Takeaway: Product Owner role is a communicator role.**

**Table 6.5:** Activities of Product Owners

Activities	Reference
Communication	[6, 7, 8, 41, 55, 70, 72, 81, 100, 103]
Acceptance tester	[7, 8, 67, 74, 113]
Customer relationship manager	[7, 8, 55, 70]
Writing user stories	[6, 8, 113]
Traveller	[6, 7, 8]
Intermediary person	[6, 7, 8]
Prioritize the backlog	[6, 7, 8]
Mastering the releases	[6, 7, 8]
Technical architect	[6, 8]
Technical governor	[6, 8]
Risk assessor	[6, 8]
Visionary	[41, 55]
Managing expectations	[29, 100]
Planning	[29, 100]
Gate keeper	[7]
Political advisor	[67]
Super secretary	[67]
Accountability	[41]
Teamwork	[70]
Expert trainer	[113]
Critical decision maker	[68]

**3.) Leadership / Management** The aspect of leadership and management in regard to the Product Owner was only considered in three publications. Shastri et al.[94] discovered that – in practice – the project manager is still in place in large projects, although this role should be replaced by the Scrum Master or Product Owner. Beside the formal role descriptions Sverrisdottir et al. [100] as well as Judy and Krummins-Beens [41] report that the understanding of the role and responsibility of the Product Owner is quite different between organizations but seldom in perfect conformance with the official Scrum method.

**Takeaway: The Product Owner management role has no generally accepted definition.**

**4.) Success Factors** The most frequently named success factors for Product Owners are the relationship between the Product Owner and the development team as well as the stakeholders. Although Koskela and Abrahamsson [53] identified that only 21% of the On-Site Customers' time was required for assisting the development team in the actual development work, having local Product Owner representatives [70, 75, 116] or a Subject Matter Expert [64, 113] on-site is considered as the main differentiator when it comes to clearly communicate responsibilities [25, 75] and priorities [75]. Working closely with the team establishes a partnership of trust and teamwork, which is considered as a success factor [41, 64, 74, 113].

**Takeaway: Relationship management is key.**

**5.) Theory vs. Practice** When researchers explicitly compared a Product Owner role to the description of another publication or analyzed real-world scenarios against theoretical definitions regarding the Product Owner role, we categorize this as 'theory vs. practice'. With regard to the topic of *functions, challenges & responsibilities*, for example, Bass et al. [7] compared their own findings of the Product Owner role in large enterprise settings [6] to the Product Owner role in small companies. Unger-Windeler and Klünder [103] compared the tasks described by Bass et al. [6] to the tasks of Product Owners in a system development environment. Sverrisdottir et al. [100] compared general descriptions of the Product Owner role to actual role description in industry. In regard to the topic of *leadership and management* in agile software development, Shastri et al. [94] analyzed to what extent the Project Manager role is still encountered in the agile industry although it is officially replaced by the Scrum Master or Product Owner role.

However, regardless of the topic the result of every comparison was always the same: it did not match. A possible explanation for this is that the settings of the two objects of comparison were not equal. Which in turn would be an indication for the importance of the environment to properly describe the Product Owner role in industry.

**Takeaway: No Product Owner role is like another.**

**6.) Requirements Engineering** We only found one publication that discussed the Product Owner role related to Requirements Engineering. Heikkila et al. [33] describe the requirements flow from strategy to release in a large-scale agile development environment and described the definition of the Product Owner role as insufficient and, thus, as problematic for the process. However, this publication does not provide insights in the requirements engineering practices of Product Owners.

**Takeaway: Real insights in Product Owners requirements engineering practices are absent.**

**7.) Collaboration / Involvement** The results regarding the collaboration / involvement of Product Owners or On-site Customers are closely related to the *success factors*. Hoda et al. [37] studied the impact of insufficient customer involvement on self-organizing agile teams. They identified problems in gathering and clarifying requirements, problems in prioritizing requirements, problems in securing feedback, loss of productivity, and in extreme cases, business loss. Supporting this, Wojciechowski et al. [116] report that On-site Customer practice has substantial positive influence on quality of communication and speed of software production.

Hence, adequate involvement and collaboration is necessary. However, although it is stated that most of the Product Owner’s time on-site is idle [53, 72] the absence of a Product Owner role is identified as cause for lack of involvement [37]. Hence, the collaboration needs to be designed more efficiently [112]

A solution for this is provided by Williams et al. [113]. They report extreme success as they have representatives on-site for a while in order to establish a close relationship between the customer and the development team so that the collaboration continues even though the customer is absent again. In turn, these reports go along with the success factor of relationship management.

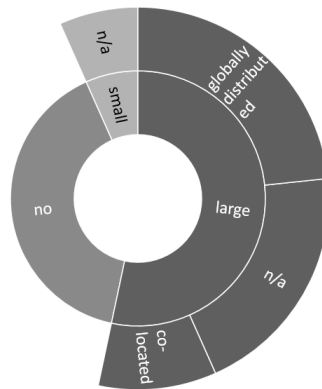
**Takeaway: The better the relationship, the better the collaboration.**

### 6.3.3 Consideration of Environment in Research

The last research question aims to provide an overview of the consideration rate of the Product Owners environment in terms of team dimension and location as well as company size and its organizational structures. Surprisingly, the external circumstances for a Product Owner are barely considered.

#### 6.3.3.1 Teams

Overall, 60% of the publications (18 out of 30) mention the dimensions of the team the Product Owner / On-site Customer is working with. Out of this, 16 publications mention large teams. While 7 of them consider globally distributed teams, only 3 report of co-located teams. The remaining 6 publications do not mention the location of the teams at all. An overview is presented in Figure 6.6.



**Figure 6.6:** Publications consider team dimensions.

#### 6.3.3.2 Company

The recognition of the Product Owner’s external circumstances in terms of the company and its organization is sparse in current literature as only 33% (10 out of 30) consider these factors at all. While as much as 27% mention the size of the company (7 large, 1 small), only 17% mention the organizational structure (2 top-down, 2 flat, 1 matrix) of the Product Owner’s environment. The references are mapped in regard to the addressed organizational structure

and the company size in Table 6.6.

**Table 6.6:** Publications consider external circumstances

Organization Size	Top- Down	Flat	Matrix	N/A
small		[7]		
medium				
large	[103]		[32]	[6, 8, 33, 55, 75]
N/A	[29]	[41]		[25, 28, 33, 37, 39, 53, 64, 66, 67, 68, 69, 70, 72, 74, 81, 94, 100, 111, 112, 113, 116]

## 6.4 Identified Research Directions for this Thesis

The contribution of this systematic mapping study is the identification of the seven research topics: (1) *scaling the PO role*, (2) *functions, challenges & responsibilities*, (3) *leadership / management*, (4) *success factors*, (5) *theory vs. practice*, (6) *requirements engineering*, (7) *collaboration / involvement* along with the consolidated answers for each of them.

While some of those topics provide congruent results, others point to gaps in current research. So is the Product Owner role in large-scale projects clearly defined as a group effort and communication skills are the most important skills of Product Owners and On-site Customers. However, questions regarding the requirements engineering practices, leadership and management responsibilities of Product Owners as well as the collaboration between Product Owners and (traditional) management roles remain unanswered.

Hence, to achieve the overall goal of this thesis, the identified research directions of

- Communication activities of Product Owners in a hybrid environment
- Requirements Engineering activities of Product Owners in a hybrid environment
- Collaboration between Product Owners and (traditional) management roles in a hybrid environment

are addressed in the following part of this thesis.



# 7

## Part II Summary

This part summarizes related work on the Product Owner tasks & characteristics and identifies the research direction for this thesis. The latter has been achieved by a systematic mapping study (conducted by the author of this thesis and others [104]) that structured the research area of Product Owners in industry and identified gaps in research. Concurrently, these gaps need to be closed for a thorough description of the Product Owner role in hybrid environments. Hence, the identified research directions for this thesis, are as follows: (1) communication activities of Product Owners in a hybrid environment, (2) requirements engineering activities of Product Owners in a hybrid environment, (3) collaboration between Product Owners and (traditional) management roles in a hybrid environment. These three research directions are addressed in Part III.

Besides the Product Owner tasks & characteristics, the considered literature in this part provides meaningful information to understand the state of the practice of Product Owner structures. Especially in large and hybrid projects, the Product Owner role is considered as a group effort and is realized in three different Product Owner team structures. This knowledge provides a necessary basis to achieve the overall goal of this thesis. Therefore, the author of this thesis summarized the relevant information of Product Owner tasks, characteristics & structures in Table 7.1 and prepared them for later stages of data analysis. This was achieved by initial coding [86]. As a first cycle method, the author of this thesis split the relevant information (identified in Section 6) into coded segments. Chapter 5 already provided codes for Product Owner tasks (by Bass [6, 8]) and characteristics (by Pichler [79]) which have therefore simply been reused. Some codes evolved from Chapter 6 have been similar to the already peer-reviewed codes presented in Chapter 5. In this case, the author mapped them immediately. The resulting segments of Product Owner tasks, characteristics & structures of this part are summarized in Table 7.1.

**Table 7.1:** Coded Segments of Product Owner Tasks, Characteristics & Structures of Part II

<b>TASKS</b>	<b>Chapter</b>	<b>Source</b>
Groom	5, 6	[6, 8, 113]
Prioritiser	5	[6, 8]
Release Master	5, 6	[6, 8, 29, 100]
Technical Architect	5	[6]
Technical Governor	5	[6]
Communicator	5, 6	[6, 7, 8, 41, 55, 70, 72, 81, 100, 103]
Traveller	5	[6, 8]
Intermediary	5	[6, 8]
Risk Assessor	5	[6]
Gatekeeper	5	[8]
Customer Relationship Manager	5	[8]
Acceptance tester	6	[7, 8, 67, 74, 113]
Customer relationship manager	6	[7, 8, 55, 70]
Managing expectations	6	[29, 100]
Political advisor	6	[67]
Super secretary	6	[67]
Visionary	6	[41, 55]
Accountability	6	[41]
Teamwork	6	[70]
Expert trainer	6	[113]
Critical decision maker	6	[68]
<b>CHARACTERISTICS</b>	<b>Chapter</b>	<b>Source</b>
Communicator & Negotiator	5	[79]
Visionary & Doer	5	[79]
Leader & Team player (including People Person)	5	[79]
Available & Qualified	5	[79]
Empowered & Committed	5	[79]
<b>STRUCTURES</b>	<b>Chapter</b>	<b>Source</b>
Hierarchical Product Owner team (An overall PO is supported by another PO)	5	[20, 62, 75, 79, 90]
Non-hierarchical Product Owner team (A business PO works together with a technical PO)	5	[6]
Expert team (A PO works together with a Specialized Role)	5	[25, 64, 106]

The coded segments are used again in Chapter 13.1 to incorporate and conclude the knowledge to deduce recommendations to support the tailoring of the Product Owner role to hybrid development environments.



## Part III

# State of the Practice: Product Owners in Hybrid Environments



---

<b>8</b>	<b>Product Owners Communication Activities in a Hybrid Environment</b>	<b>51</b>
8.1	Methodology . . . . .	<b>51</b>
8.1.1	Research Questions . . . . .	51
8.1.2	Research Site . . . . .	52
8.1.3	Data Collection . . . . .	53
8.1.4	Data Analysis . . . . .	53
8.2	Product Owners Communication Activities . . . . .	<b>55</b>
8.3	Quantification of Communication Activities . . . . .	<b>56</b>
8.4	Product Owners Communication Partners . . . . .	<b>59</b>
<b>9</b>	<b>Requirements Flow and Collaboration in a Hybrid Environment</b>	<b>61</b>
9.1	Related Work . . . . .	<b>61</b>
9.2	Methodology . . . . .	<b>62</b>
9.2.1	Research Question . . . . .	62
9.2.2	Research Site . . . . .	62
9.2.3	Data Collection . . . . .	63
9.2.4	Interviews . . . . .	65
9.2.5	Data Analysis . . . . .	65
9.2.6	Threats to Validity . . . . .	66
9.3	Requirements Flow in a Hybrid Environment . . . . .	<b>66</b>
9.4	Involved Roles in a Hybrid Requirements Engineering Process . . . . .	<b>74</b>
9.5	Product Owner Tasks within a Hybrid Requirements Engineering Process . . . . .	<b>75</b>
9.6	Collaboration between Product Owners and Traditional Management Roles . . . . .	<b>76</b>
<b>10</b>	<b>Expectations on the Product Owner Role in a Hybrid Environment</b>	<b>79</b>
10.1	Methodology . . . . .	<b>79</b>
10.1.1	Context . . . . .	79
10.1.2	Research Questions . . . . .	80
10.1.3	Participant Selection . . . . .	80
10.1.4	Questionnaire Creation and Pilot Tests . . . . .	81
10.1.5	Data Collection and Instrument . . . . .	81
10.1.6	Data Analysis . . . . .	81
10.1.7	Participants . . . . .	81
10.1.8	Limitations . . . . .	82
10.2	Expectations on Tasks . . . . .	<b>83</b>
10.2.1	Attributions of Tasks . . . . .	85
10.3	Expectations on Characteristics . . . . .	<b>86</b>
<b>11</b>	<b>Part III Summary</b>	<b>87</b>

---



# 8

## Product Owners Communication Activities in a Hybrid Environment

The objective of this chapter is to contribute knowledge to the area of the identified research direction of *communication activities of Product Owners in a hybrid environment* (see Chapter 7) in order to achieve the goal of this thesis: to analyze Product Owner tasks, characteristics and structures. Therefore, we (the author of this thesis and others<sup>1</sup>) collected both quantitative and qualitative data on Product Owners communication activities in a case study. The results of this study have been published in [83] and the *Journal of Software: Evolution and Process* [105].

### 8.1 Methodology

#### 8.1.1 Research Questions

To reach the objective of this research, the following research questions have been phrased:

*Research Question 1: What kind of communication activities does a Product Owner engage during a sprint?*

The answer to this question should help to grasp the actual communication activities beyond the regular Scrum / Nexus meetings.

*Research Question 2: How much time does a Product Owner spend on these communication activities?*

The results of Chapter 6 allows the conclusion that the Product Owner communicates a lot. However, current research does not provide any quantitative information regarding the communication activities. By answering this question, we would like to close this gap.

*Research Question 3: With whom does a Product Owner collaborate in these activities?*

As results of Chapter 6 state that the relationship and, hence, collaboration with others is key – it remains unknown whom these other roles are. Although the answer to this question will

---

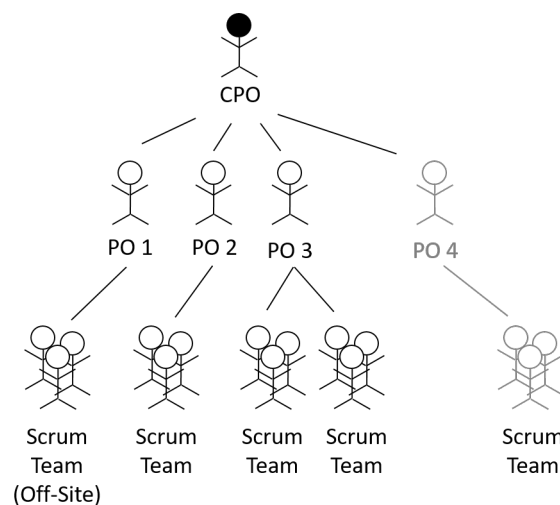
<sup>1</sup>Jil Kluender, Timothy M. Reuscher and Kurt Schneider

not identify the most important collaboration partners, it provides a first insight about the roles Product Owners actually communicate and hence interact with. Note that we are not interested in identifying the roles the Product Owners *should* collaborate with. We look at the state of “as-is” rather than on the state “to-be”.

### 8.1.2 Research Site

The case study was conducted in a large-scale project in the context of a systems-development environment at *Baker Hughes, a GE Company (BHGE)*. The company combines capabilities across the full value chain of oil and gas activities – including the development of digital solutions combining hardware technologies with software products. While hardware engineering has always been one of the company’s core businesses, software engineering is relatively new to them. In daily business, BHGE develops safety-critical systems based on reliable software. While the software is developed with a rather agile development approach, the overall product development is stage-gate managed. Hence, the Product Owner needs to communicate and negotiate with all stakeholders – including the end user of the overall product, leaders of other departments that are involved in the system development as well as the Scrum Teams.

Due to the large-scale of the project, the software group is following a slightly tailored Nexus approach as described by Schwaber [89] and created a distributed product ownership team (this team can be seen as equivalent to a Product Owner team). This team is similar to the case described by Paasivaara et al. [75]. It contains a Chief Product Owner (CPO) and 4 Proxy Product Owners<sup>2</sup> (PO) as shown in Figure 8.1.



**Figure 8.1:** PO team structure

While the CPO, PO1, PO2 and PO3 are located in Germany, PO4 is located in India. Each Product Owner has assigned at least one full development team including a Scrum Master and is responsible for up to three features within a release cycle of 3 months. While the POs 2, 3 and 4 are co-located with their teams, PO1 works with a team located off-site in Germany. PO3 has two teams assigned. For this research, we collected data from PO1, PO2 and PO3.

<sup>2</sup>In order to improve readability and to be in alignment with the naming in the real-world, we refer to the Proxy Product Owner only as Product Owner from here on.

### 8.1.3 Data Collection

In the first quarter of 2019, one researcher observed each of the three Product Owners located in Germany for the duration of an entire sprint (2 weeks). This resulted in an overall duration of observance of 6 weeks and three different sprints. To collect qualitative as well as quantitative data, we used two techniques concurrently: shadowing and the Goal-Question-Metric approach, which are described in the following.

**Shadowing** Shadowing is a research technique that was developed to uncover the activities as well as perspectives of a real person in the real-time context of an organization [71]. This data-generation method is beneficial, whenever the unit of analysis includes an individual as well as his network or organizational contexts. It gains a rich, dense and comprehensive data set that provides a detailed, first-hand and multidimensional picture of the role and tasks of the person being shadowed [71].

While shadowing, the researcher shadowed the respective Product Owner every minute of the day. He had his desk in the same room as the Product Owner, attended a meeting if the Product Owner attended that meeting, was forwarded the calendar and all incoming as well as outgoing emails to/from the Product Owner. Additionally, the researcher and Product Owner discussed the content of the meetings and emails. The researcher used a spreadsheet to track the observed communication activities including the name of the conversation partners, their role, the time and the duration of the conversation, whether it was formal or informal and the content discussed, followed by how the Product Owner processed the information.

This method collects large amounts of data, which can become difficult to analyze with respect to a specific goal [54]. To avoid this, we followed a goal-centered approach to assure that we did not miss any meaningful data.

**Goal-Question-Metric** Goal-Question-Metric (GQM) is a goal-oriented framework [54]. By applying this framework, questions and metrics are tailored to provide meaningful data that are relevant to draw proper conclusions and eventually reach the predefined goal. While the goals are defined on a conceptual level, the questions make the goals operational and the metrics map reality to comparable values and hence, makes them measurable. To achieve this, we generated a GQM plan according to Koziol [54] where we phrased the goal, questions and metrics in a hierarchical structure. An overview is presented in Table 8.1.

The researcher used a spreadsheet to manually collect the measured data for the metrics M1 to M7. A difficulty was to decide whether a particular interaction should be counted for one of these. To gain a consistent data set, we counted every verbal interaction that discussed subject-specific content. Hence, private conversations were excluded.

### 8.1.4 Data Analysis

We analyzed the gained data based on the applied techniques of shadowing and GQM.

**Shadowing** To analyze the gathered data, we used the technique of *FLOW Maps*. A FLOW Map is a specialized FLOW model [88] that provides a notation to visualize participants, activities, documents, and information flows within a project. The resulting FLOW Map for each Product Owner identified the communication activities as well as the involved roles.

**Table 8.1:** GQM Plan

GOAL	PURPOSE ISSUE OBJECT VIEWPOINT CONTEXT	Understand communication activities Product Owners objective large-scale project in system-development environment
Question	Q1	To what extent is the PO engaged in formal meetings?
Metric	M1 M2	Quantity per day Minutes per day
Question	Q2	To what extent is the PO engaged in informal meetings?
Metric	M3 M4 M5 M6	Quantity per day Minutes per day Percentage quantity $((M3 / M3 + M1)*100)$ Percentage minutes $((M4 / M4 + M2)*100)$
Question	Q3	How often does a PO talk to each role?
Metric	M7	Quantity per day

**Goal-Question-Metric** We defined the metrics by the goal and the question in a top-down manner before measuring. After measuring, we used the quantitative data to answer the question and eventually reach the goal bottom-up. Therefore, we used the method of descriptive statistics.

#### 8.1.4.1 Threats to Validity

According to Yin [120], there are three tests for evaluating the quality of our case study: construct validity, external validity and reliability.

**Construct validity** The validity of the construction can be ensured by using multiple sources of evidence. This has been achieved through the shadowing of the Product Owners in three different sprints. Additionally, we used the GQM framework to track the number of interactions and involved roles rather than just interviewing Product Owners and relying on their input solely.

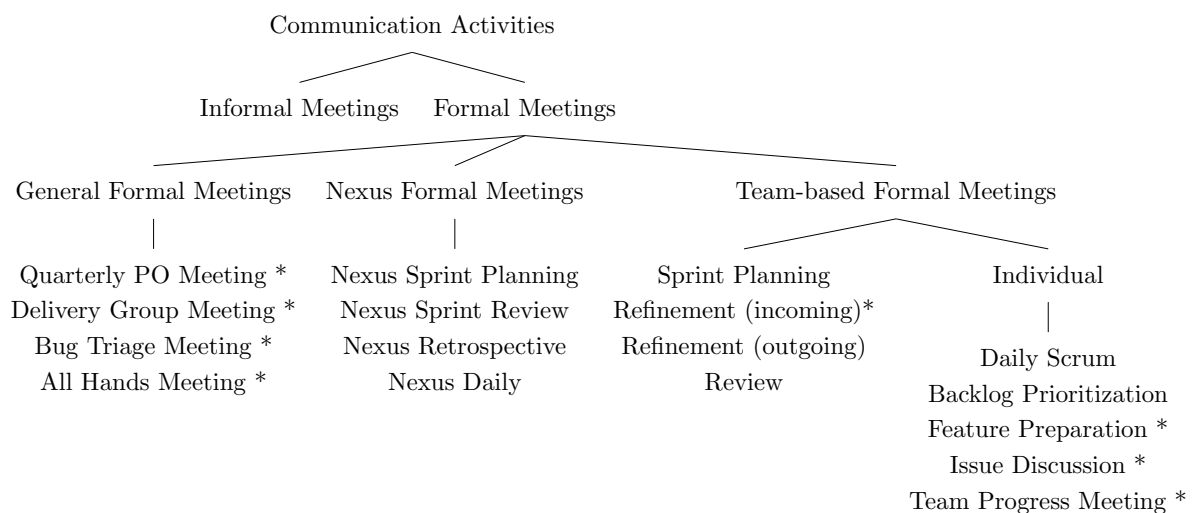
**External validity** The external validity can be achieved through replication of studies. In our study, we included insights from three different Product Owners who work in the same company. Hence, the results must not be over-generalized and may not be correct for Product Owners in another domain or in a company which is smaller/larger than BHGE. Consequently, replications of our study are required to draw more reliable conclusions.

**Reliability** We developed reliability through the use of a predefined spreadsheet, where we tracked the communication regarding the predefined metrics. This helps to ensure consistency in the data collection.



## 8.2 Product Owners Communication Activities

To answer RQ 1: *What communication activities does a Product Owner engage during a sprint?*, we categorized the communication activities as meetings. Furthermore, we distinguished between formal and informal meetings. Whenever a conversation between the shadowed Product Owner and another person took place, although it was not planned and hence had no agenda, we categorized it as an informal meeting. Furthermore, we grouped the formal meetings based on the number of attending Product Owners as well as the purpose of the meeting. Consequently we identified four categories of communication activities: one informal meeting category and three formal meeting categories. In addition, we mapped the actual meeting titles. The classification of the meetings is summarized in Figure 8.2.



**Figure 8.2:** Meeting Categories. The meeting titles marked with an asterisk (\*) identify that these meetings are beyond the regular Nexus meetings.

### 8.2.0.1 Informal Meetings

The informal meetings are mainly perceived as short interruptions in the office, where the Scrum Master or a developer had a short question to the respective Product Owner. These interruptions were face-to-face, via chat or telephone. Each of the Product Owners was engaged in such informal meetings.

### 8.2.0.2 Formal Meetings

**General Formal Meetings** The *General Formal Meetings* are neither official Nexus nor team-based meetings. However, all of the identified meetings in this group were attended by all of the Product Owners at the same time and served the purpose of synchronizing and aligning their work. We identified the title of these meetings as: *Quarterly PO Meeting*, *Delivery Group Meeting*, *Bug Triage Meeting* and *All Hands Meetings*.

**Nexus Formal Meetings** As the Software Group is following a slightly tailored Nexus approach, we identified some of the applied official Nexus events and hence, labeled them *Nexus*

*Formal Meetings.* These meetings serve the purpose of streamlining the agile process of large-scale Scrum. All three Product Owners attend the Nexus formal meetings at the same time and were identified as: *Nexus Sprint Planning*, *Nexus Sprint Review*, *Nexus Retrospective* and *Nexus Daily*.

**Team-based Formal Meetings** The remaining meetings were attended by only one individual Product Owner and discussed team-based topics. Thus, we called them *Team-based Formal Meetings*. However, although all of these meetings were called differently by each of the Product Owners, most of the meetings have the same purpose and were identified as follows: *Sprint Planning*, *Refinement (incoming)*, *Refinement (outgoing)*, *Review*. While they basically match the Nexus events, they take place on the team level. Additionally, we identified different flows of information in the individual refinement meetings.

The *Refinement (incoming)* meetings describe the flow of information from the SME (Subject Matter Expert) to the Product Owner. Hence, this meeting aims to refine the requirements so that the Product Owner understands them properly. This communication activity could be compared to the requirements analysis phase in the requirements engineering process. On the other hand, the *Refinement (outgoing)* meetings describe the flow of information (analyzed requirements) from the Product Owner to the Scrum Team.

Furthermore, we identified some slight differences on the level of the team-based formal meetings. Here, the individual Product Owner added some more meetings to support the work of their team. We labeled them as *Individual*. Examples for these meetings are the *Daily Scrum*: here, the Product Owners attend them either regularly, sporadically or not at all; *Backlog Prioritization*: here, one Product Owner conducts this tasks with the team, while others do this on the Nexus level.

**Finding:** During a sprint (2 weeks), a Product Owner is engaged in informal and formal meetings. The informal meetings serve the purpose of **solving issues for the team**. The formal meetings serve the purpose of **synchronizing and aligning the work of the Product Owners, streamlining the agile process of large-scale Scrum and discuss team-based topics and provide the team with the necessary requirements for the product under development**.

## 8.3 Quantification of Communication Activities

To answer RQ 2: *How much time does a Product Owner spend on these activities?*, we quantified the identified formal and informal meetings from Section 8.2 as follows.

### 8.3.0.1 Formal Meetings

Despite the qualitative analysis method to identify formal meetings, we were able to extract quantitative data due to the inclusion of the GQM approach.

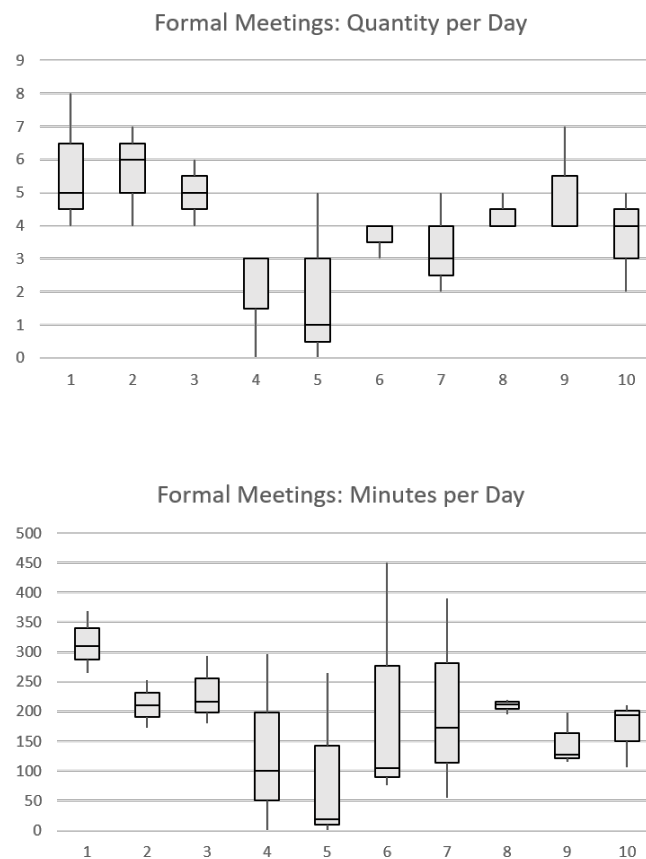
With Question Q1 and its respective metrics M1 and M2 (see GQM plan in Table 8.1), we assessed the quantity of formal meetings per day for each day of the sprint. As represented in Figure 8.3, the quantity depends on the day of the sprint cycle.

On day 1, 2, 3 the sprint starts. Here, formal communication occupies almost all days. While the Product Owners typically have 5 to 6 meetings per day in median, they spend between 200

to 300 minutes in median (3.3 to 5 hours) with this kind of communication per day.

On day 9 and 10 the sprint ends. Here, the amount of formal meetings is quite high (4 in median), although they only take up to 130 to 190 minutes in median (2.2 to 3.2 hours). In the meantime formal communication was happening as well, but less time consuming than at the start and end of the sprint.

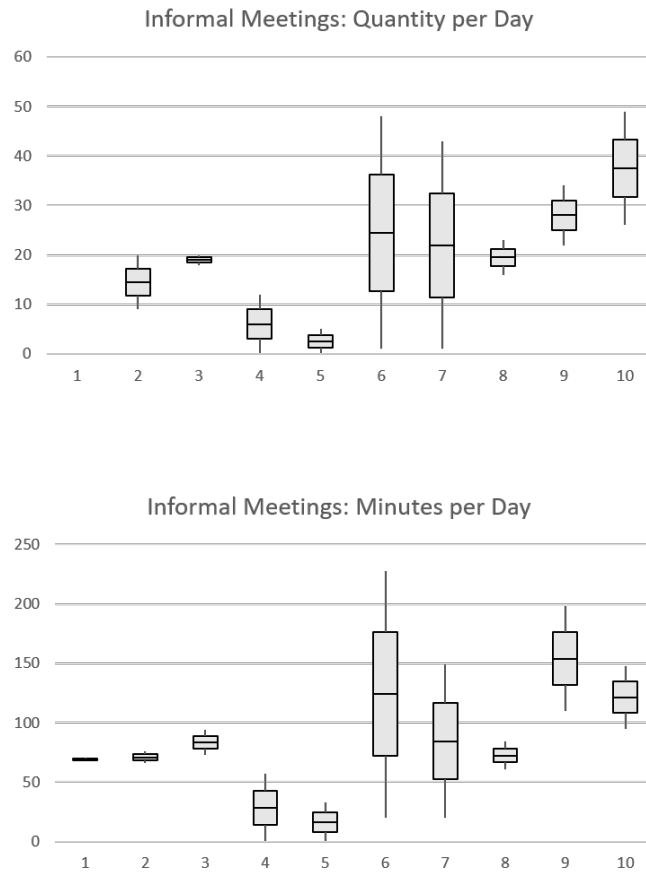
According to the results of RQ 4, these results identify that during a sprint Product Owners spend 32 hours to synchronize their work, streamline the agile process and work with their teams. Considering that the Product Owners work 70 hours per sprint (35 hours per week), formal communication takes up 45% of their time.



**Figure 8.3:** Quantification of Formal Meetings

### 8.3.0.2 Informal Meetings

Question Q2 and its respective metrics M3 and M4 (see GQM plan in Table 8.1), allowed to quantify informal meetings for each day of the sprint. The results represented in Figure 8.4 can be related to the results from Q1. While on days 1,2 and 3, where formal communication is high, informal communication is not very present. However, in the middle of the sprint (starting from day 6), the Product Owner gets involved in more informal communication. On day 9 and 10, where the sprint ends, the Product Owners communicate about 30 to 40 times informally. On day 9 the informal communication takes up the most time (150 minutes per day).



**Figure 8.4:** Quantification of Informal Meetings

Furthermore, the metrics M5 and M6 allowed us to assess the time spent in meetings percentage-wise. Summarizing, we can say that although 68% of the Product Owners meetings are informal (M5), only 31% of their overall time spent in meetings are spent on informal ones (M6).

According to the results of RQ 1, these results identify that during a sprint Product Owners spend 14 hours to solve upcoming issues addressed by their teams. Considering that the Product Owners work 70 hours per sprint (35 hours per week), informal communication takes up 20% of their time.

**Finding:** During a sprint (2 weeks), Product Owner spends 65% of their entire time in formal and informal meetings. 45% of the time is spent to synchronize their work, streamline the agile process and work with their teams (formal meetings). 20% of the time is spent to solve issues for their teams (informal meetings).

## 8.4 Product Owners Communication Partners

Our data provide qualitative as well as quantitative results to answer RQ 3: *What roles are involved in these communication activities?*

### 8.4.0.1 Qualitative Results

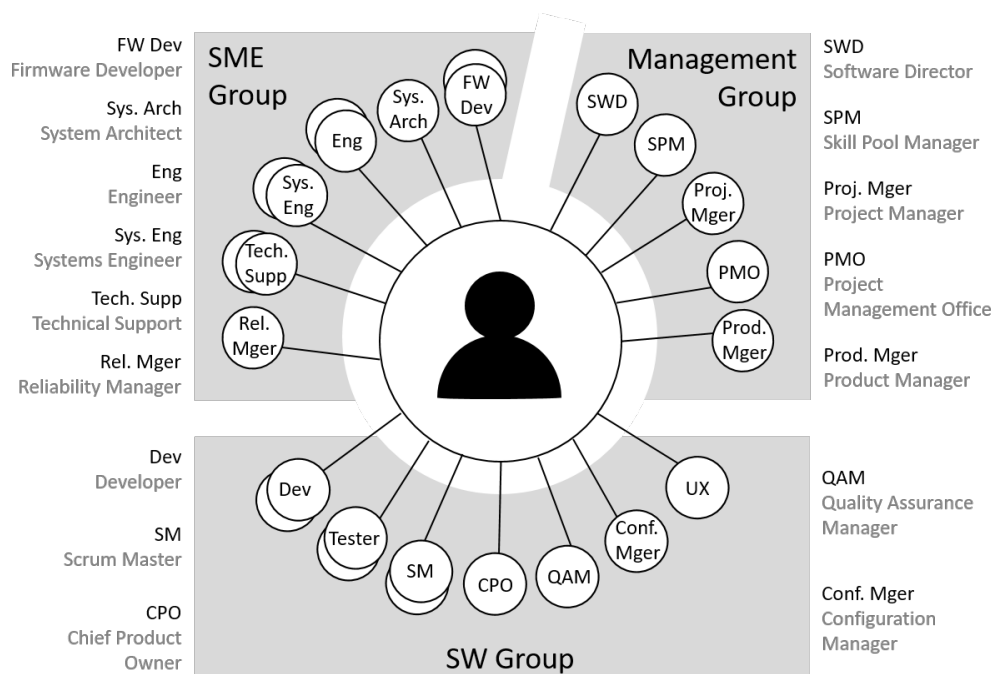
The qualitative analysis of the notes taken during the shadowing revealed that the Product Owner interacts with a total number of 18 different roles from three different departments: Management, Engineering / Operations (we combined these two groups and refer to it from now on as "Subject Matter Expert (SME) Group") and Software. These three groups can be seen as individual stakeholder groups within a large-scale project in the context of a systems-development environment. The 18 different roles are:

**Software Group:** Developer, Tester, Scrum Master, Chief Product Owner, Quality Assurance Manager, Configuration Manager, UI/UX Designer

**Management Group:** Software Director, Skill Pool Manager, Project Manager, Project Management Office, Product Manager

**SME Group:** Firmware Developer, System Architect, Engineer, System Engineer, Technical Support, Reliability Manager

Figure 8.5 depicts the roles a Product Owner communicates with. Some of these roles are represented by multiple individuals such as the Scrum master role. Roles represented by multiple individuals are represented by a twin-circle in Figure 8.5.



**Figure 8.5:** Product Owner Network. Roles represented by multiple individuals are represented by a twin-circle.

### 8.4.0.2 Quantitative Results

With the predefined Questions and Metrics (see Table 8.1), we captured the number of interactions with each role.

Table 8.2 summarizes the results and shows that most communication happens among the Product Owners, respectively with other roles from the Software Group. The left column summarizes the ranking of the contacted roles. However, the roles of the ranking 2 to 4 are considered as the Scrum Team. Within the Scrum Team the Product Owners communicate more often to the roles of the developers and testers than to the Scrum Master. The ranking of the groups identified that Product Owners communicate most with members of the software group (rank # 1), second most with members of the management group (rank # 2) and least often with members of the subject matter expert group (rank # 3).

**Table 8.2:** Interactions over the course of a sprint

	Role	PO1	PO2	PO3	$\Sigma$	Rank
Software Group	PO	64	65	103	232	1
	Developer	28	56	105	189	2
	Tester	23	47	43	113	3
	Scrum Master	16	44	46	106	4
	CPO	21	12	19	52	5
	Quality Assurance Manager	17	13	13	43	6
	Config. Manager	4	18	5	27	7
	UI/UX Designer	1	2	9	12	11
				774	<b>1</b>	
Management	SW Director	1	13	7	21	8
	Skill Pool Manager	4	11	4	19	9
	Project Manager	0	10	3	13	10
	Project Management Office	0	2	0	8	14
	Product Manager	0	1	0	1	16
				62	<b>2</b>	
SME Group	Firmware Developer	0	18	9	27	7
	System Architect	11	5	6	13	11
	Engineer	3	3	3	9	12
	System Engineer	0	1	5	6	13
	Tech Support	0	0	2	2	15
	Reliability Manager	1	0	0	1	16
				58	<b>3</b>	
	$\Sigma$ Interactions	191	321	382		
	Network Size (Roles)	14	17	15		

**Finding:** During a sprint (2 weeks), a Product Owner gets in contact with 15 different roles, while some of the roles are represented by multiple persons. The most frequently contacted roles are members of the software group (rank # 1), the management group (rank # 2) and members of the subject matter expert group (rank # 3).

# 9

## Requirements Flow and Collaboration in a Hybrid Environment

The objective of this chapter is to contribute knowledge to the area of the identified research direction of *requirements engineering activities of Product Owners in a hybrid environment* as well as on *collaboration between Product Owners and (traditional) management roles in a hybrid environment* (see Chapter 7) in order to achieve the goal of this thesis: to analyze Product Owner tasks, characteristics and structures. Therefore, we (the author of this thesis and others<sup>1</sup>) conducted a case-study to provide an overview of how requirements are managed in a large interdisciplinary project, what roles are involved in this process and what tasks are performed by the individual roles (and the Product Owner role in particular). The results of this study have been published in [82].

### 9.1 Related Work

A similar research was approached by Heikkilae et al. [33] in 2017. In a case study at Ericsson, they conducted 43 interviews with practitioners to describe the requirements flow from strategy to release. They report that the requirements management was done in three different processes, where each process followed its own process model, purpose and planning approach. So was the release management plan-driven, feature development process was conducted continuously and the implementation process was agile. Within this, they identified the roles of the product manager, portfolio manager, product owner, technical specialists and developer who have to collaborate on the different stages.

Overall, they state that this mixed approach provided some benefits as well as problems and highlighted the issue of "defining the product owner role" in organizations where the higher level planning processes are not agile.

**Distinction to this research** This research follows a similar approach and identifies the roles & responsibilities in the requirements engineering process. However, the data will be analyzed with a focus on the role of the Product Owner role.

---

<sup>1</sup>Tanita Ressler

## 9.2 Methodology

### 9.2.1 Research Question

To reach the objective of this research, the following research questions were phrased:

Research Question 1: *How are the requirements managed in a large interdisciplinary project?*

Research Question 2: *What roles are involved in the requirements management process?*

Research Question 3: *What activities are performed by the Product Owner within the requirement management process?*

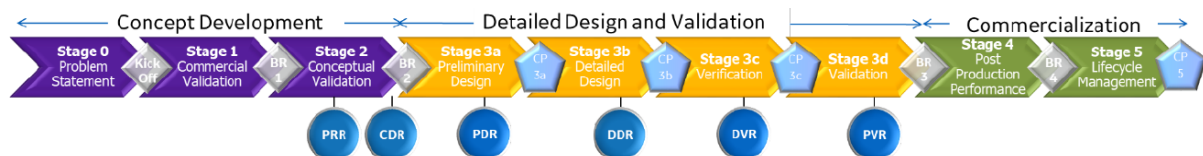
Research Question 4: *How do Product Owners and traditional management roles collaborate on the requirements?*

### 9.2.2 Research Site

The case study was conducted in a large-scale project in the context of a systems-development environment at *Baker Hughes, a GE Company (BHGE)*. The company combines capabilities across the full value chain of oil and gas activities – including the development of digital solutions combining hardware technologies with software products. While hardware engineering has always been one of the company’s core businesses, software engineering is relatively new to them. In daily business, BHGE develops safety-critical systems based on reliable software. While the software is developed with a rather agile development approach, the overall product development is stage-gate managed.

**Stage-Gate Process / Hybrid Product Development Process** The company follows a slightly tailored stage-gate product development process by Robert G. Cooper [21]. The stage-gate process divides the overall product development into individual *stages*. After every stage, a so called *gate* needs to be passed. To pass these gates, certain quality aspects must be met. Only after passing this gate, the next stage can be approached. The gates are usually assessed by upper managers which are empowered enough to approve budget and resources to the stages and included projects.

The stage-gate process applied at Baker Hughes has been slightly tailored to the needs of the company and divided the stages in three areas: *concept development* (phase 0 to 2), *detailed design and validation* (phase 3) and *commercialization* (phase 4 and 5) (Baker Hughes,[3]<sup>2</sup>) and is represented in Figure 9.1.



**Figure 9.1:** Stage-Gate-Process at Baker Hughes (Baker Hughes,[3]) )

<sup>2</sup>Source derived from the intranet (not publicly available) of Baker Hughes



The stages are assessed by six gates that are named *Kick-Off*, *Business Review* (BR) or *Checkpoints* (CP). Furthermore, stage 2, 3a, 3b, 3c and 3d are enhanced by technical reviews: *Product Requirement Reviews* (PRR), *Conceptual Design Review* (CDR), *Preliminary Design Review* (PDR), *Detailed Design Review* (DDR), *Preliminary Verification Review* (PVR), *Detailed Verification Review* (DVR).

Within the stages 3a to 3d (of the *detailed design phase*) agile methods are applied, while all other stages are traditional. This illustrates the hybrid character of the product development process at Baker Hughes, meanwhile, it also states that the agile development is compressed into an overall traditional development approach and hence represents the definition of a hybrid development environment as stated in Section 3.3.

Furthermore, due to the large-scale of the project, the software group is following a slightly tailored Nexus approach as described by Schwaber [89] and created a distributed product ownership teams. The Product Owner team structure as well as communication activities of this role at the research site has been elaborated in the previous Section 8.

### 9.2.3 Data Collection

The study focused on the product development of novel drilling tools that are heavily software dependent. First, appropriate interview partners were selected, before the FLOW method was used to gather relevant information from the participants.

#### 9.2.3.1 FLOW

FLOW is an established method to analyze the communication and hence, analyze the information flow in software projects. It is a systematic analysis of information flows to detect lacks or anomalies in communication structures [88]. For data collection, FLOW-Interviews are conducted with members of the project [47]. Therefore, the FLOW method provides an Elicitation Package containing an interview guide as well as a survey sheet [97]. The survey sheet provides a template, where the researcher can enter the information gained from the interviewee during the interview. With this, FLOW-Interviews result in a FLOW diagram that visualizes the communication activities, the involved participants as well as the used tools and types of information flow and storage. Here, FLOW distinguishes between two types of information flows and stores: solid and fluid [95]. As solid information are usually captured in written documents [88] fluid information are undocumented meetings, informal face-to-face communication and implicit knowledge [97]. The FLOW syntax is presented in Figure 9.2.

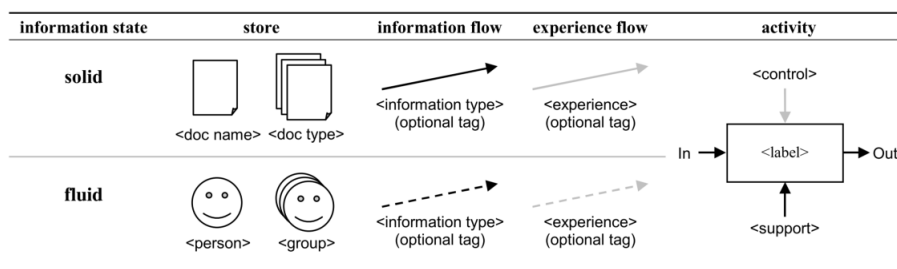


Figure 9.2: Basic FLOW syntax by Schneider et al. [88]

### 9.2.3.2 Participant selection

To select the appropriate participants, a kick-off meeting was held. Here the Project Manager, the Director of the Software Group as well as a Systems Engineer (SE) was invited to outline the process broadly and point to the right interview partners. They highlighted identified the following roles to cover the entire requirements engineering process within the hybrid product development process:

**Product Manager** is responsible for the management of the entire product, the requirements as well as the budget and schedule. He is the main point of contact to the end users and analyses their business needs.

**System Engineer** is responsible to validate the requirements of the entire system. They have a vast overview of all elements and requirements of the product.

**Chief Product Owner** is responsible to share the vision of the end product and builds the main interface between the software and the management group.

**Product Owner** is responsible for the elicitation & validation (refinement) and distribution of the requirements, as well as the review and approval / disapproval of the developed increment against these requirements. Due to the tailored Nexus approach, the Product Owner is also responsible for managing the sprint process and reporting the status of the development (see results from Section 8).

The kick-off meeting identified that stakeholders from three different groups should be considered: the management group, the engineering group and the software (SW) group. The Project Manager itself is considered to be part of the management group but mainly acts as a technical lead for all involved hardware teams of the engineering group. This makes him responsible for the technical realization as well as the quality of the product, but he is not involved in the requirements engineering process and hence, has not been considered as an interview partner for this study. Same applies to the Director of the software group.

An overview of the considered roles for the interview is presented in Table 9.1.

**Table 9.1:** Roles and Responsibilities of considered Interview Partners at Baker Hughes

	<b>Role</b>	<b>Basic Job Description</b>	<b>Interviewed?</b>
Group Management	Product Manager	business analyst, provides requirements and manages budget and schedule of the product	yes
	Project Manager	technical leader of the product	no
Eng. Group	System Engineer	validation of system related requirements	yes
Software Group	Director	responsible for process and entire software group but not on requirements level	no
	Chief Product Owner	share the vision of the end product and builds main interface between management and software group	yes
	Product Owner	is responsible for refinement and distribution of requirements as well as managing the nexus process and reporting the status	yes

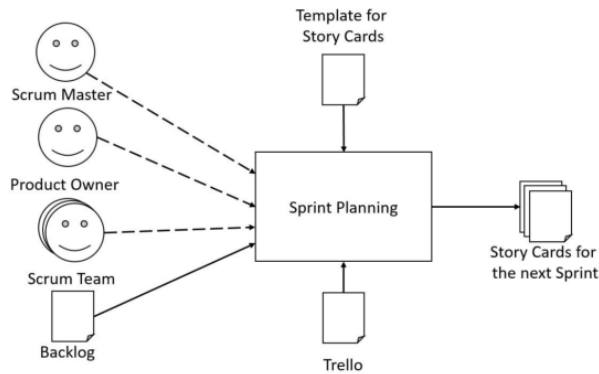
### 9.2.4 Interviews

A slightly modified FLOW-Interview was conducted with the identified interview partners. At least one representative from each involved department was interviewed. Each interview lasted 45 to 60 minutes and was recorded and transcribed in bullet point form. In accordance with the FLOW method [96], the adjusted interview guide (see Appendix B) was followed and the predefined survey sheet was filled during the interview.

With this, for each interviewee the main tasks and collaboration efforts within the requirements management process were addressed.

### 9.2.5 Data Analysis

First steps of the data analysis were conducted during the interview already. This is because the survey sheet was filled during the interview in collaboration with the interviewee. Here, the involved roles, controlling as well as supporting instances and the outcome for each requirement activity are documented in as many isolated FLOW diagrams as activities have been discovered. An exemplary FLOW diagram of the activity "Sprint Planning" is represented in Figure 9.3.



**Figure 9.3:** Exemplary isolated FLOW-Diagram

With this, multiple isolated FLOW diagrams resulted from each interview. Immediately after the interview, the records along with the isolated FLOW diagrams were reviewed. After every interview, the individual FLOW diagrams were combined with the previously generated ones and a consolidated overview of the entire process as well as the collaboration mechanisms was achieved eventually.

### 9.2.6 Threats to Validity

According to Yin [120], there are three tests for evaluating the quality of the case study: construct validity, external validity and reliability.

**Construct validity:** The validity of the construction can be ensured by using multiple sources of evidence. This has been achieved by interviewing four different people on the same workflow. This provided an overall picture from four different viewpoints that all coincided.

**External validity:** The external validity can be achieved through replication of studies. Although the study was conducted in only one company with a highly specific setup, the study itself was quite similar to a previously conducted study by Heikkilae et al. [33]. Although the setups were differed, the overall results coincided. Also, the results are in accordance with the conducted case study at the same research site (see Section 8). However, the results must not be over-generalized and may not be correct for Product Owners in another domain or in a company which is smaller/larger than the case study. Consequently, replications of this study are required to draw more reliable conclusions.

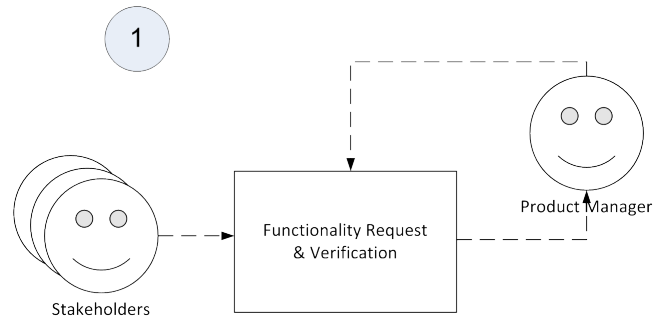
**Reliability:** Reliability was developed through the use of the the Elicitation Package provided by the FLOW method with a predefined interview guide as well as a survey sheet. This ensured consistency in the data collection.

## 9.3 Requirements Flow in a Hybrid Environment

At *Baker Hughes*, the requirements engineering process to develop a new functionality into a product, involves three different groups: management group, engineering group, software group. Overall, twelve activities within the requirements management flow have been identified as follows:

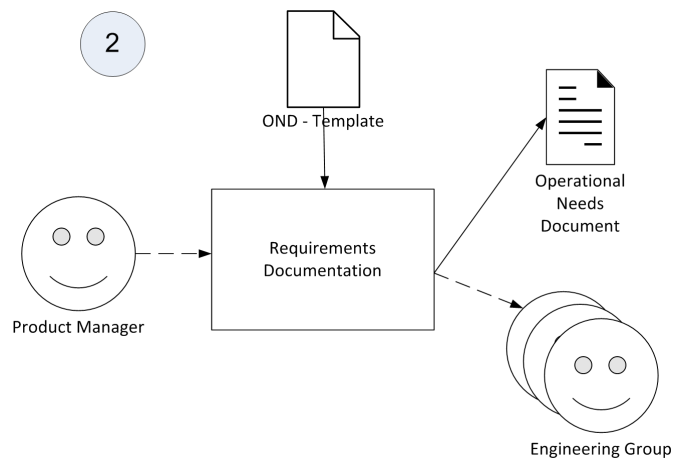
1. The **Product Manager (PM)** gets requests for certain functionalities from several kind of stakeholders (end-users or engineers – depending on the part of the product). The Product

Manager verifies the requests against the overall relevance of the business. See Figure 9.4.



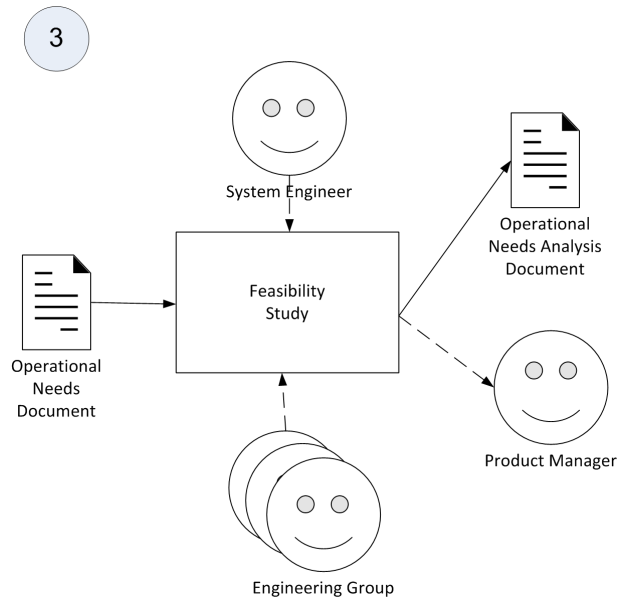
**Figure 9.4:** Product Manager: Verification of Request

**2.** If the **PM** verified a need for this functionality, he documents the request as a set of requirements in an *Operational Needs Document (OND)*. This document is created based on a template. The resulting OND will be handed over to the Engineering Group. See Figure 9.5.



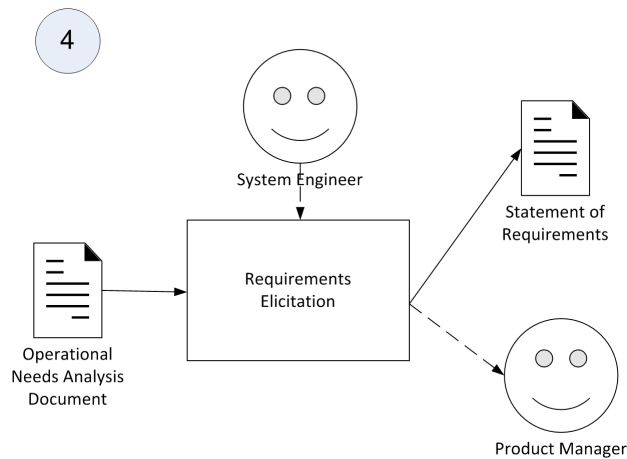
**Figure 9.5:** Product Manager: Documentation of Requirements

**3.** The **Engineering Group** verifies these requirements and conducts a feasibility study. This is controlled by a System Engineer. The results are documented in the *Operational Needs Analysis Document (ONAD)* and communicated to the Product Manager. See Figure 9.6.



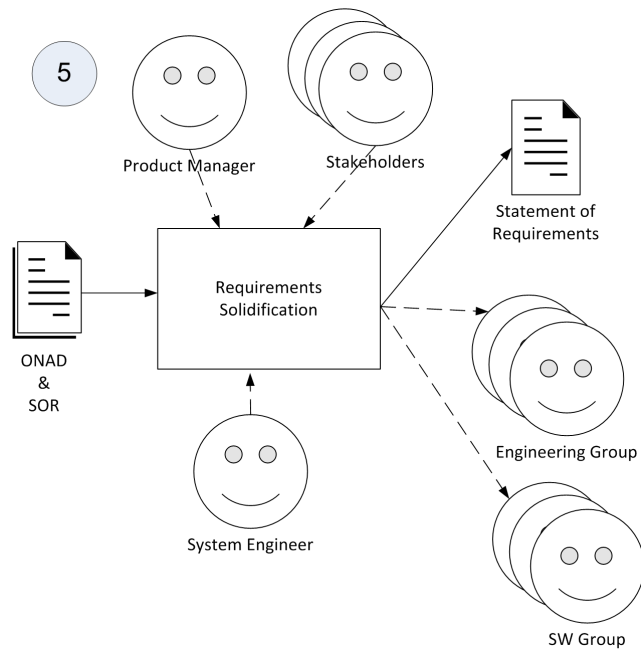
**Figure 9.6:** Engineering Group: Conduct feasibility study

4. The ONAD is used for further requirements elicitation from the hardware perspective, which will be specified and documented in the *Statement of Requirements (SOR)*. The results of the feasibility study and the resulting SOR are taken back to the PM for validation. See Figure 9.7.



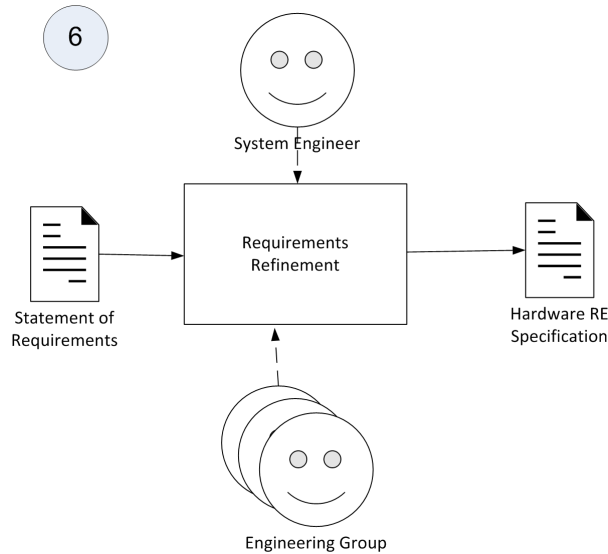
**Figure 9.7:** Engineering Group: Requirements elicitation

5. The ONAD as well as the SOR are discussed by the **PM and the System Engineers**. For verification purposes, the PM discusses them with the Stakeholders as well. When the stakeholder agrees to the refined requirements, they are documented in an updated version of the *Statement of Requirements (SOR)*. From then on the SOR is considered as the final set of requirements and is distributed to both, the SME Group as well as to the Software Group. Every further change of these requirements require the consolidation of the PM. See Figure 9.8.



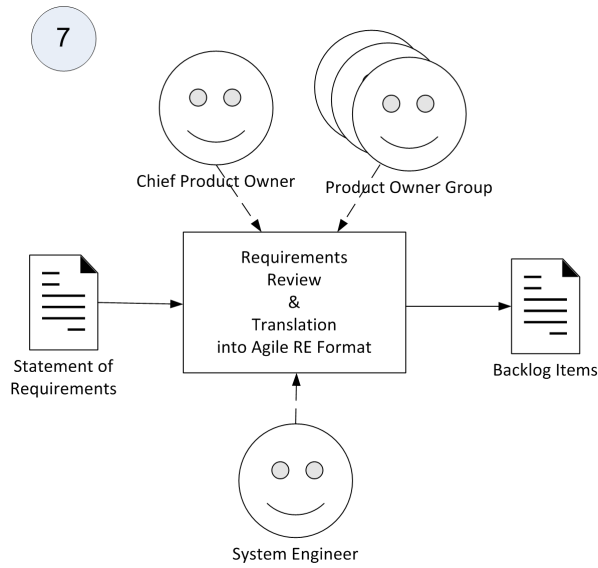
**Figure 9.8:** PM: Requirements solidification

6. The final version of the SOR will be further analyzed by the Engineering Group to generate the *Hardware RE Specification* document. See Figure 9.9.



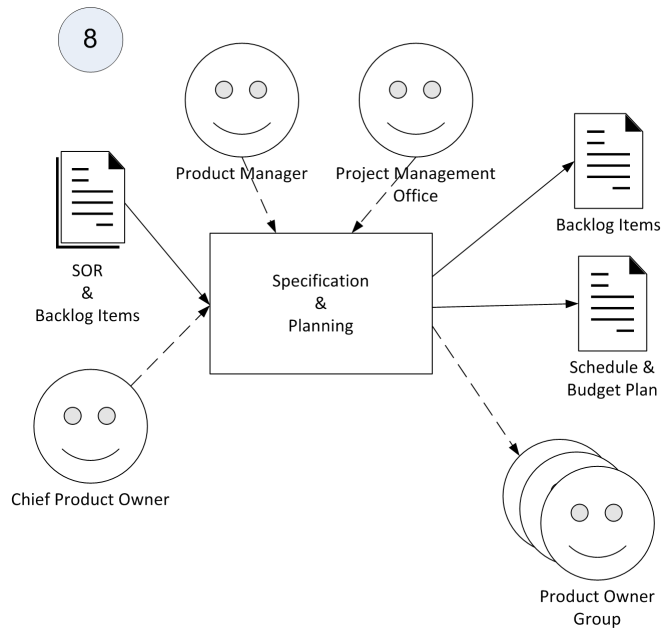
**Figure 9.9:** Engineering Group: SOR refinement

7. The **Software Group** gets the final version of the SOR. The Product Owners as well as the Chief Product Owner review the requirements. The System Engineer supports this activity. The Product Owner translates the content from the SOR into the Product Backlog, where the requirements are stored in agile RE compatible formats as *Epics* and *User Stories*. The Product Backlog is accessible and used by the entire software group to assign work items and trace the requirements. See Figure 9.10.



**Figure 9.10:** SW Group: SOR review

8. The generated Epics and User Stories are further specified by the CPO in collaboration with the management group (Product Manager, Project Management Office) to clarify expectations as well as for planning purposes. The results are documented in the agile RE tool and communicated with the respective Product Owners. See Figure 9.11.

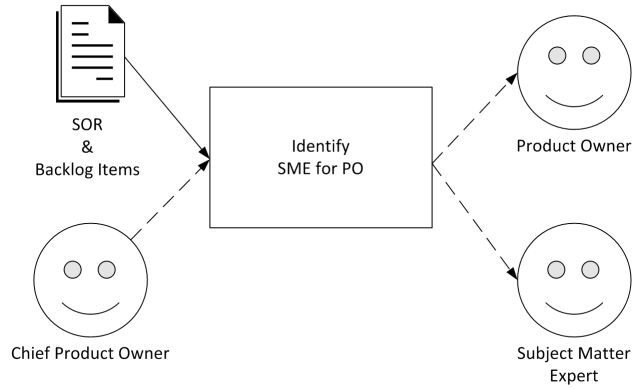


**Figure 9.11:** SW Group: planning

9. Based on the information on each Product Backlog Item (PBI), the CPO identifies Subject Matter Experts that should assist the respective Product Owner with clarification of the requirements. See Figure 9.12.



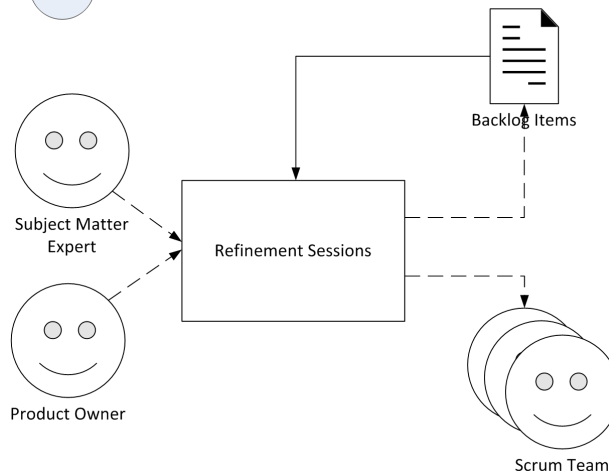
9



**Figure 9.12:** SW Group: Identify SMEs

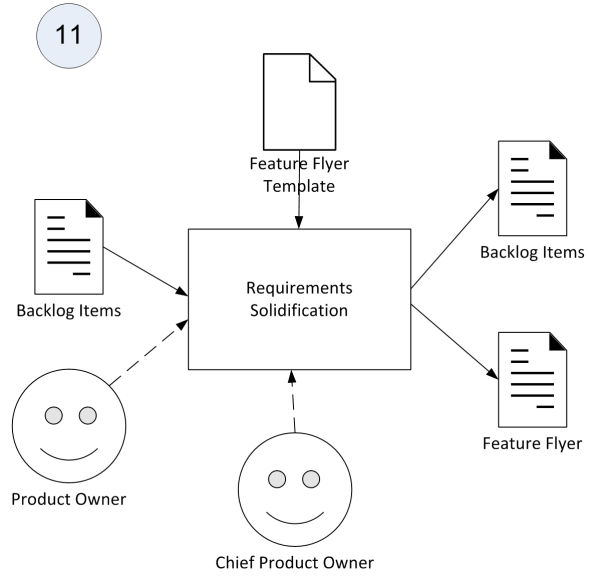
10. Each **Product Owner** gets together with the identified Subject Matter Expert by the CPO to refine the requirements. The gathered information are stored in the Backlog and are refined iterative. The sessions are sometimes shared with the Scrum Team. See Figure 9.13.

10



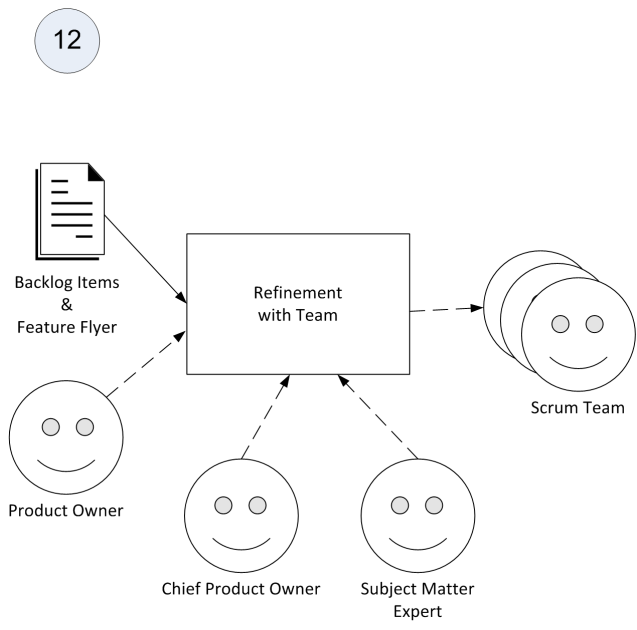
**Figure 9.13:** SW Group: PO refinement with SME

11. The gained knowledge from the refinement session with the SME is used by the Product Owner to update the Backlog Items / User Stories. As an enhancement of the User Story format, each Product Owner also generates a Feature Flyer. This Feature Flyer is based on a template and has to be approved by the Chief Product Owner. See Figure 9.14.



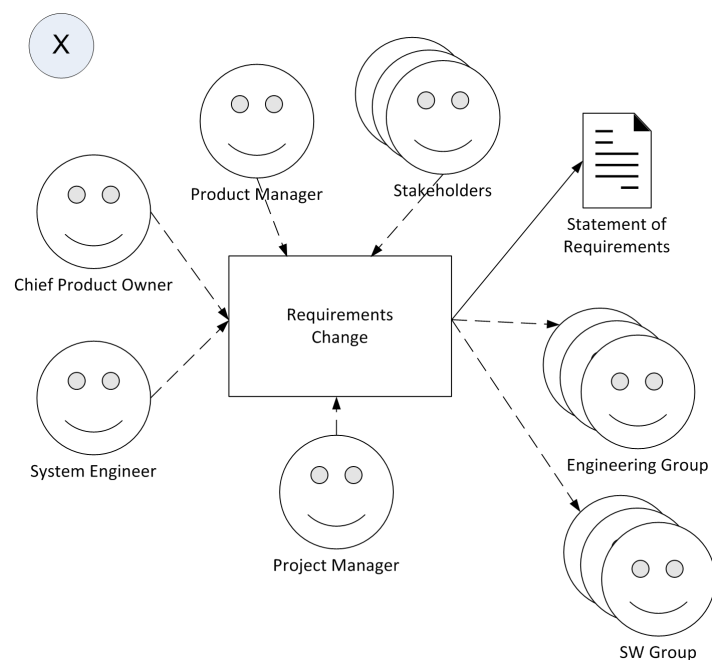
**Figure 9.14:** SW Group: Requirements solidification

12. Each Product Owner uses the Feature Flyer and regular Backlog Items to discuss the requirements for implementation with the Scrum Team. The Subject Matter Expert as well as the Chief Product Owner may support this activity occasionally to provide further clarification for the team. See Figure 9.15.



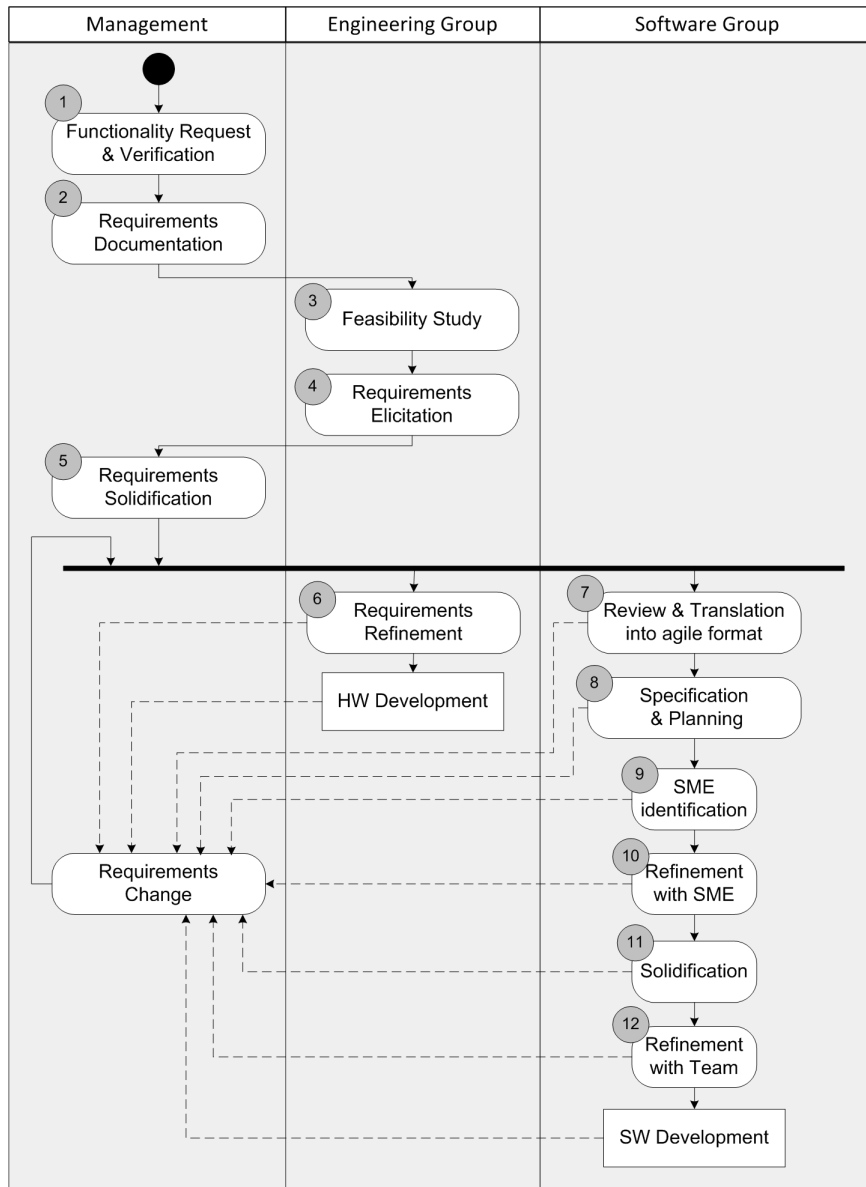
**Figure 9.15:** SW Group: PO refinement with Scrum Team

**X.** If, at any point of time, the Software Group or the SME Group identifies that changes are required in the SOR, the Chief Product Owner or the System Engineer raises the concern to the Product Manager. The Product Manager controls the changes of the requirements (in accordance with the Stakeholders) and is responsible for documenting the changes in the SOR. The Project Leader supports this activity. The results are distributed to the software and engineering group. See Figure 9.16.



**Figure 9.16:** PM: Requirements Change

Figure 9.17 provides a **consolidated overview** of the requirements management process. Although FLOW diagrams generally do not support a chronological representation of the activities, the author of this thesis created one to emphasize the flow of the requirements as the entire development process is highly traditional due to the stage-gate-process.



**Figure 9.17:** Combined and Chronological FLOW diagram

## 9.4 Involved Roles in a Hybrid Requirements Engineering Process

The following eight different roles are involved in this hybrid requirements engineering process: Product Manager, Project Manager, Project Management Office, System Engineer, Engineering group members, Chief Product Owner, Product Owner, Scrum Team members. An overview of the roles mapped to their activities is presented in Table 9.2.

**Table 9.2:** Roles and Activities within hybrid RE process at Baker Hughes

	Role	Activity	Number
Management	Product Manager	Functionality request & verification	1
		Requirements documentation	2
		Requirements solidification	5
Specification & Planning		8	
Requirements change		X	
	Project Manager	Requirements change	X
	Project Management Office	Specification & Planning	8
Eng. Group	System Engineer	Feasibility study	3
		Requirements elicitation	4
		Requirements solidification	5
		Requirements refinement	6
		Review & Translation of requirements	7
		Requirements change	X
	Engineering group members	Feasibility study	3
		Requirements refinement	6
Software Group	Chief Product Owner	Review & Translation of requirements	7
		Specification & Planning	8
		Identify SME	9
		Requirements change	X
	Product Owner	Review & Translation of requirements	7
		Refinement with SME	10
		Requirements solidification	11
		Refinement with Scrum Team	12
	Scrum Team members	Refinement session with Scrum Team	12

## 9.5 Product Owner Tasks within a Hybrid Requirements Engineering Process

In order to provide meaningful information regarding the Product Owners activities within the requirement management process, the term *Product Owner* needs to be distinguished, as it can be scaled up or down (see Section 5.2). In this case study, the term Product Owner refers to the supporting type of Product Owners, while the term Chief Product Owner is used to identify the overall Product Owner. According to this, the answer to this research question is split.

**The Chief Product Owner** describes his main activities as

- reviewing & translating the requirements from the traditional requirements document SOR (Statement of Requirements) into an agile compatible format (Epic, Feature, User Story) in the Backlog (activity 7).
- specifying the requirements from a software development perspective and generate a development plan accordingly (activity 8).
- identifying the right Subject Matter Expert to support the Product Owner in the refinement of the Features and User Stories (activity 9).

Furthermore, the CPO supports the activities 10,11,12 of the **Product Owner** and is involved whenever requirements are changing (activity X).

**The Product Owner** describes his main activities as

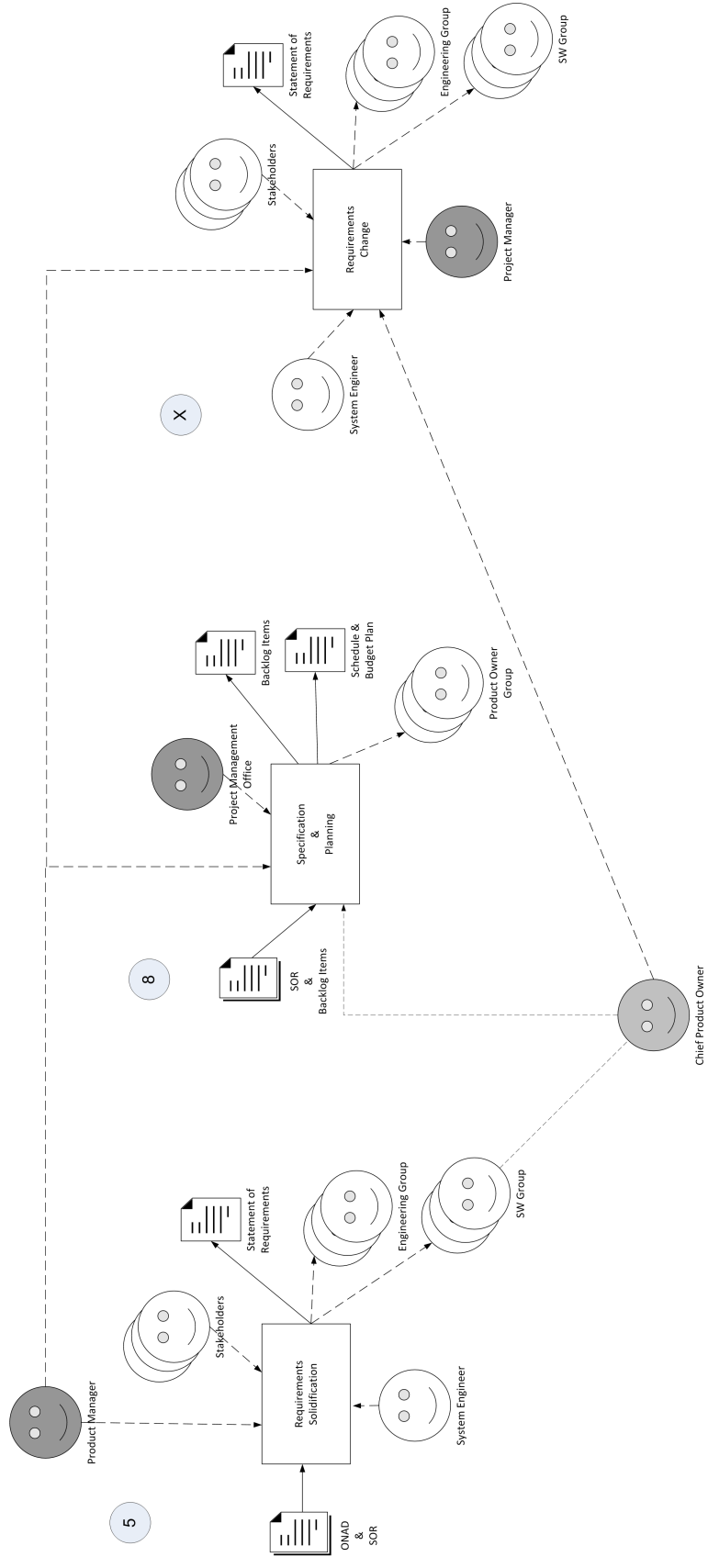
- refining the requirements from the Backlog with the respective Subject Matter Expert (activity 10).
- solidifying the refined requirements in so called Feature Flyers (activity 11).
- communicating the requirements to the development team for implementation (activity 12).

## 9.6 Collaboration between Product Owners and Traditional Management Roles

As this question is again not clearly defined due to terminology issues, the answer to this question is split as well.

The **Chief Product Owner** is the first representative of the software group who gets to know about upcoming requirements by getting the SOR from the Product Manager. As the traditional format of the requirements documentation (SOR) is not suitable for the agile way of working in the software group, he has to translate them into Epics, Features and User Stories in the Backlog. As the overall traditional product development process requires budgeting and scheduling of the development, he has to commit to a development plan. To do so, he collaborates with the traditional management roles of the Product Manager and the Project Management Office. Whenever any of the solidified requirements from the SOR needs to be changed, the Chief Product Owner collaborates with the traditional management roles of the Product Manager and the Project Management Office. The combined FLOW diagram in Figure 9.18 illustrates the relationships.

**The Product Owner** collaborates with no traditional management roles directly.



**Figure 9.18: Collaboration FLOW of CPO and traditional management roles**





# 10

## Expectations on the Product Owner Role in a Hybrid Environment

The objective of this chapter is to contribute knowledge to the area of the identified research direction of *collaboration between Product Owners and (traditional) management roles in a hybrid environment* (see Chapter 7), as well as to analyze Product Owner tasks, characteristics and structures. However, to do so, we (the author of this thesis and others<sup>1</sup>) did not analyze the state of practice but rather the expectations on the Product Owner tasks, characteristics and structures. Therefore, an online survey has been conducted to gather feedback from various roles within a hybrid development team. The results of this study have been published in [106] and presented at the international *Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*.

### 10.1 Methodology

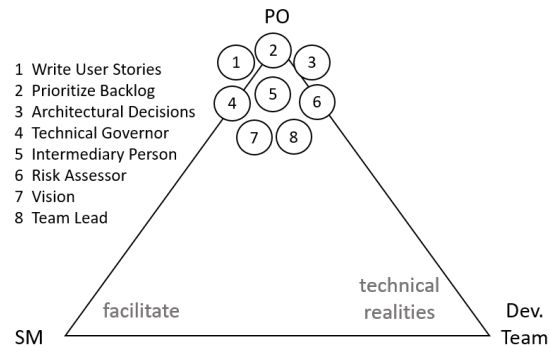
#### 10.1.1 Context

This study has been conducted in early 2017 and hence is primarily based on the identified tasks by Bass [6] at that time<sup>2</sup>. To visualize the identified tasks of Product Owners in relation to the overall Scrum Team, we extended the basic representation of the Scrum Team (See Figure 2.1 in Section 2.1.1) as depicted in Figure 10.1. Hence, the research questions and, thus, the results are all based on this.

---

<sup>1</sup>Kurt Schneider

<sup>2</sup>Bass et al. [8] have published an updated version of their results in 2018 as described in Chapter 5



**Figure 10.1:** Extended representation of a Scrum Team

### 10.1.2 Research Questions

To reach the objective of this research, the following research questions have been phrased:

*Research Question 1: Which tasks does the Scrum Team expect to be performed by the Product Owner?*

While related work reviews the implemented state of this role [6, 8, 55, 75, 103], we were interested in the Scrum Teams perspective and asked which of the tasks they expect to be performed by the PO or by any other role within the organization. Hence, we asked who should write user stories, prioritize the backlog, make architectural decisions, be the technical governor, act as an intermediary person, do the risk assessment and communicate the vision of the end-product (according to the described tasks in Chapter 5). Although the role of a team lead is obsolete when following the Scrum framework – the role of a leader is mentioned anyhow [79]. Therefore, we asked the team whether there should be a leader of their Scrum Team at all – and if so, who should take that role?

We excluded the questions about communicating and travelling as we see them more as a means to the end rather than actual tasks that could be assigned to a specific role.

*Research Question 2: How would the Scrum Team characterize the role of a perfect Product Owner?*

Here, we address the described characteristics of Product Owners in industry (according to the described characteristics in Chapter 5). With this question, we were interested in how Scrum Team members would characterize this role.

We did not want to point the participants in any direction, hence we provided a free-text field and asked for their own words.

### 10.1.3 Participant Selection

Our research question addresses the expectations of the Scrum Team on the Product Owner role. However, as this research is directly linked to our previous research approach, consequently, our target group consists of all Scrum Team members of the respective software group in the Oil & Gas industry at *Baker Hughes, a GE Company (BHGE)*.

We directly contacted 156 people belonging to the technical software department at BHGE via email. Besides the roles of the Development Team and the Scrum Master, this included the roles of Directors, Managers, UX Leads as well as Testers.

Since we were not able to exactly control who participates in the survey, we asked the participants at the beginning to select their role (Product Owner, Scrum Master, Developer, UX Designer, Tester, Architect, Manager, Other[included a free-text field]). With this, we could get an idea of the participating roles. Although all of the participants belong to the same group, they work in different projects, Scrum Teams and are located in various countries.

#### 10.1.4 Questionnaire Creation and Pilot Tests

We designed the survey by following established guidelines [40, 84, 87]. To ease answering and analyzing the survey, we mainly used multiple choice questions. Most of them contained the "Other" option to allow participants to specify missing options in free-text. Most parts of the survey were descriptive, but it also contained some exploratory parts, as on the expectations of leadership of a Scrum Team or the characteristics of the PO role.

Based on the software groups organization, we knew that the participants are located in India, Germany, USA and the Netherlands. Considering this as well as the fact that the companies business language is English, we created an online version of the survey in English.

To ensure that the survey questions were comprehensible and valid with respect to the study construct, we went through three rounds of pre-tests and refined the survey. In each pre-test, a software professional completed the survey and we discussed how it could be improved.

#### 10.1.5 Data Collection and Instrument

In early 2018, we conducted the survey implemented in LimeSurvey. We directly contacted 156 people belonging to the technical software department at BHGE via an email sent through the director of this group. Having the director as the sender of the email should have the effect that as many employees as possible follow his request to participate in the survey. The survey was open for 4 weeks. A friendly reminder was sent to the target group after 2 weeks - to encourage everyone again to take the time to participate.

#### 10.1.6 Data Analysis

All but the question regarding the role of the participant were optional, to avoid forcing them to answer. Multiple choice questions that contained free-text in the "Other" option were coded manually for analysis [87].

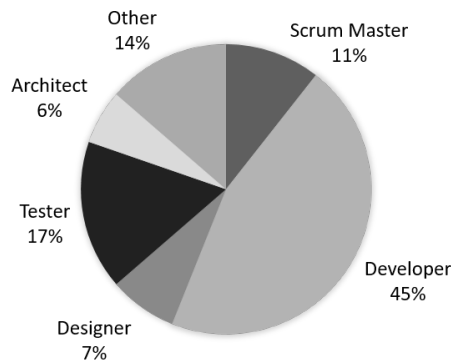
#### 10.1.7 Participants

In total, 84 of 156 employees answered the survey. 66 finished the survey. The majority of the respondents, 25 (38%) are located in India. 18 (27%) work in Germany, 13 (20%) in the USA and 10 (15%) in the Netherlands.

Although we sent the survey to all roles, the majority of the respondents (45%) described their role as Developer. The remaining roles are as follows: 17% Tester, 11% Scrum Master, 8% UX Designer, 6% Architect, 14% described their role as "Other" (including: Technical Writer, Service Manager, Product Manager, Designer/Developer/Junior Architect, Software Configuration Manager, Director, Requirement Analyst, UX Lead).<sup>3</sup> See Figure 10.2

---

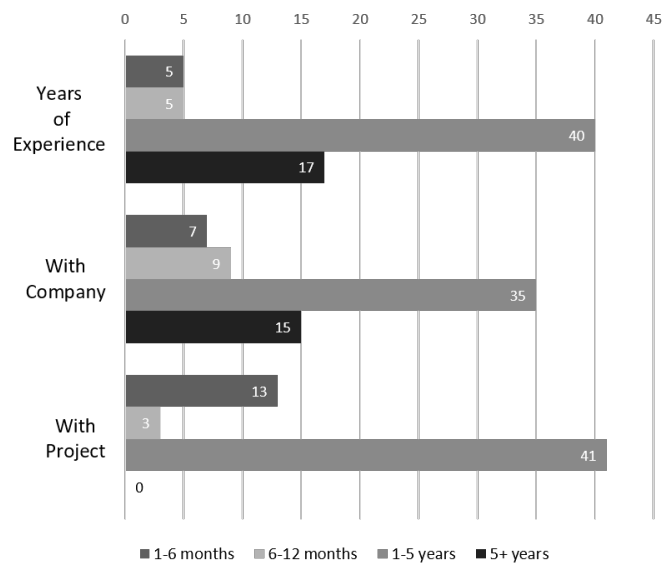
<sup>3</sup>Since all of these roles work very closely with the PO and had an interest as well as an opinion on this matter, we included them in this study. Hence, whenever we refer to the Scrum Team, we include these roles.



**Figure 10.2:** Roles of Participants

The majority (60%) of the respondents have 1-5 years of experience in their described role. While 23% the respondents were affiliated with the company for more than five years, 53% were with the company for one to five years. 62% were working in the respective projects for one to five years.

See Figure 10.3



**Figure 10.3:** Years of Experience in the respective role

### 10.1.8 Limitations

This section discusses the threats to validity.

The primary threat to construct validity is the single use of a survey. This causes a mono-method bias as all collected data are based on a single source. The respondents rationales and thoughts behind their answers remain unknown. Hence, there is little control over the quality of responses.

The survey was anonymous and none of the questions touch upon sensitive topics. However, the results of some questions might be influenced by social desirability bias, e.g., by stating that

the they would expect the Product Owner to do what he actually does, versus what the respondents would really expect him to do. However, the option of choosing multiple answers in the confirmatory parts of the survey, makes us think that this threat is under control. Additionally, as this survey was intentionally asking for the individuals expectations, there were no right or wrong answers.

Furthermore, we only checked for previously identified tasks rather than leaving the answer of potential tasks open to the respondents. Hence, it is possible that some tasks were not listed, but probably expected by the respondents.

An important threat to internal validity is the instrumentation. The time that it takes to complete the survey is crucial. Although the target group was directly encouraged by the management to take the time to answer this survey, they were not willing to spend too much time with the survey or answer redundant questions. To mitigate the risk of survey fatigue, we went through three rounds of pre-tests to design the survey accordingly.

The validity of the conclusions regarding the characterization of the product owner role might be affected subjectively, since the author performed the coding and analysis mainly on her own. Although the study has been conducted only in one company, the results are in alignment with other study results and hence might be applicable to other similarly operating industries as well.

## 10.2 Expectations on Tasks

The Scrum Team does not expect the Product Owner to perform all of the tasks alone. There are some tasks, the Scrum Team would assign to the PO solely, while there are others which the team expects the Product Owner to work closely with other roles or even not to be involved at all. An overview of this result is represented in Table 10.1.

**Table 10.1:** Expected roles on tasks by the Scrum Team

Role	Task
PO	Write User Stories Prioritize Backlog
PO & Management	Communicate Vision of End-Product
PO & Scrum Master	Act as an Intermediary Person
All Roles	Risk Assessment
Architect	Make Architectural Decisions Act as a Technical Governor
Scrum Master	Lead of the Scrum Team

A more detailed representation of the results is displayed in Figure 10.4 and is further described below.

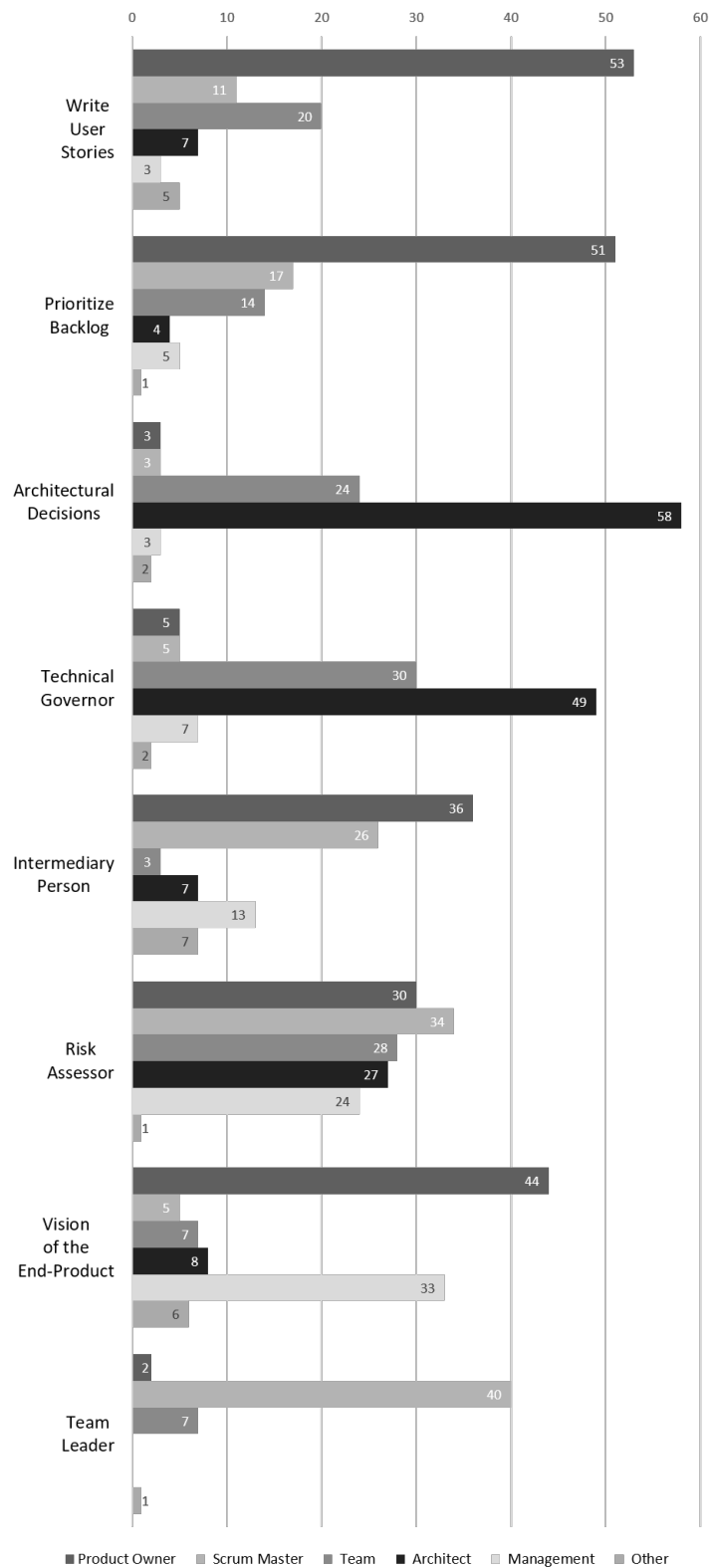


Figure 10.4: Expectations on Tasks

**Write User Stories** 80% of the respondents expect the Product Owner to write user stories.

**Prioritize Backlog** Prioritizing the backlog is also expected to be done primarily by the Product Owner (77%).

**Make Architectural Decisions** Architectural decisions are expected to be done solely by the Architect (88%) and the development team (36%). Neither the Product Owner nor the Scrum Master should have a stake in this.

**Technical Governance** Similar to the task of 'architectural decisions', the Scrum Team expect the Architect (74%) and the development team (45%) to govern technical aspects.

**Intermediary Person** The role of an intermediary person who brings together all stakeholders is expected to be done by both roles the Product Owner (55%) as well as the Scrum Master (39%).

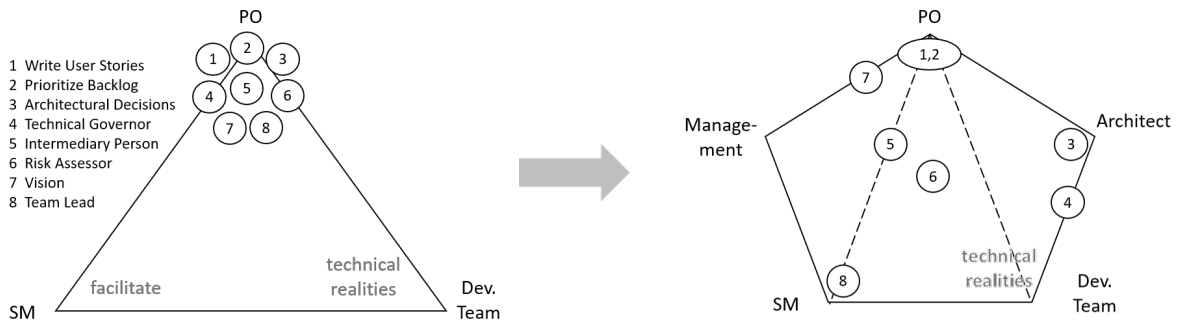
**Risk Assessor** The expectations on who should do the risk assessment is spread on all roles. While 45% expect the Product Owner to do it, 52% see the Scrum Master, 42% see the team and 41% the architect and 36% the management to be involved in this task.

**Communicating the vision of the end-product** The communication of the vision of the end-product is expected to be done primarily by two roles: the Product Owner (67%) and the Management (50%).

**Team Leader** First, we asked the Scrum Team whether they think there should be a leader of their team. 53% responded with yes, 35% responded no and 12% did not answer that question. However, the participants that would like to have a Scrum Team lead, expect the Scrum Master(69%) or a team member (20%) to take over that responsibility. Only 6% of the respondents would assign the Product Owner to this role.

### 10.2.1 Attributions of Tasks

To further support the tailoring of the Scrum roles in industry, we propose to further extend the Scrum Team triangle 10.1 by two more corners: the corner of the management and the corner of the architect. Additionally, we identified that specialized roles such as the User Experience Designer (UX Designer) or the Quality Assurance Specialist (QA) is expected to be involved in certain tasks as well. However, there are no significant numbers on a certain tasks. Based on our findings, these two roles are important in large projects and help to support the product owner role. Additionally, we placed the tasks in respect of the responsible roles to visualize the expected shared responsibilities among the roles. See Figure 10.5.



**Figure 10.5:** Extended Scrum Team triangle

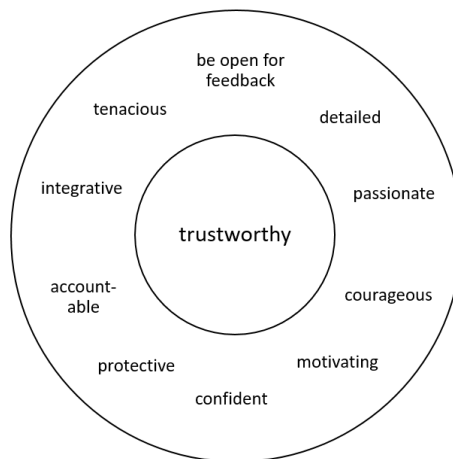
### 10.3 Expectations on Characteristics

Eventually, we did not only ask about the specific tasks, but we want to give the Scrum Team members the opportunity to describe characteristics a perfect Product Owner should have.

We asked the participants to describe the role in their own words and hence provided a free-text field. As this is always a hurdle for participants, only 27 (41%) put in an answer. However, we coded the responses manually and identified the following ten attributes, that the Scrum Team used to describe the perfect Product Owner: be open for feedback, detailed, passionate, courageous, tenacious, motivating, confident, protective, accountable, integrative.

Overall, we condense these attributes to *trustworthy*.

A visual representation of this result can be seen in Figure 10.6.



**Figure 10.6:** Attributes of a perfect Product Owner



# 11

## Part III Summary

To achieve the overall goal of this thesis, Chapter 7 identified three research directions: (1) communication activities of Product Owners in a hybrid environment, (2) requirements engineering activities of Product Owners in a hybrid environment and (3) collaboration between Product Owners and (traditional) management roles in a hybrid environment, which have been addressed in this part of the thesis.

Two novel approach are to be highlighted here: one is the identification of performed tasks by the two distinct roles of Chief Product Owners and (supporting) Product Owners in an industry setting. The other is the reversed approach of not assessing the actual implementation of the Product Owner role, but to survey the expectations of the Scrum Team members on this role.

However, each study provides meaningful information on the state of the practice of Product Owner tasks, characteristics & structures in hybrid development environments. As this knowledge is necessary to achieve the overall goal of this thesis, the author of this thesis summarized the relevant information in Table 11.1 and prepared them for later stages of data analysis. Therefore, the author of this thesis extracted descriptive phrases from the gained data as part of a first cycle coding method [86] to split the relevant information into segments or mapped them to already existing codes provided in Chapter 7. Furthermore, whenever possible, the results are distinguished by the roles of the Chief / Overall Product Owners and the so-called supporting Product Owners. The descriptive phrases of Product Owner tasks, characteristics and structures are summarized in Table 11.1.

**Table 11.1:** Descriptive Phrases of Product Owner Tasks, Characteristics & Structures of Part III

<b>TASKS</b>	<b>Chapter</b>
<b>All Product Owners</b>	
PO interacts with Subject Matter Experts and Product Management rather than actual end-users	8, 9
Establish formalized communication due to interfaces to distributed teams and hardware department	9
PO should not make any architectural decisions	10
POs should write user stories and prioritize the backlog solely	10
<b>Chief / Overall Product Owner</b>	
Reviewing & translating the requirements from traditional requirements documents into agile compatible formats (Epic, Feature, User Story)	9
Specifying the requirements from a software development perspective and generate a development plan accordingly	9
Identifying the right Subject Matter Expert to support the PO in the refinement of the Features and User Stories	9
Supporting the PO in all their tasks	9
Release planning	9
<b>(Supporting) Product Owner</b>	
Refining the requirements from the Backlog with the respective Subject Matter Expert	8, 9
Solidifying the refined requirements in so called Feature Flyers	9
Communicating the requirements to the development team for implementation	8, 9
Communication and alignment with the other POs is a time consuming effort	8
POs spend half of their time coordinating the sprint	8
<b>CHARACTERISTICS</b>	
PO are not considered as leaders of the Scrum Team	10
CPO is considered as the leader of the software group	9
POs act as a proxy between management and other departments to share and communicate a vision. They do not invent their own vision.	8, 9, 10
POs should be trustworthy	10
POs should be communicators, as they communicate with up to 15 different roles per sprint	8, 9
<b>STRUCTURES</b>	
PO teams state of the practice in large-scaled / hybrid project	8 9, 10
Teams consists of overall PO & supporting PO, business PO & technical PO, POs & expert roles	8, 9
Establishing a so-called Feature Team that consists of POs and hardware expert roles to focus on requirements engineering tasks	9
Only the CPO engages with (upper) management	9
POs collaborate with PMO for budget reporting	8
Close collaboration with management is expected	10

The descriptive phrases are used again in Chapter 13.1 to incorporate and conclude the knowledge to deduce recommendations to support the tailoring of the Product Owner role to hybrid development environments.

## Part IV

# The HyPrO Expert System



---

<b>12 Type of Research Artifact</b>	<b>93</b>
<b>13 Model</b>	<b>95</b>
13.1 Knowledge-base: Product Owner . . . . .	95
13.1.1 Product Owner Tasks . . . . .	96
13.1.2 Product Owner Characteristics . . . . .	97
13.1.3 Product Owner Structures . . . . .	98
13.2 Knowledge-base: Software Process Tailoring Criteria . . . . .	102
13.2.1 Influencing Factors on Product Owners . . . . .	102
13.2.2 Action Items . . . . .	104
13.3 Resulting Catalog . . . . .	105
13.4 Inference Engine . . . . .	106
13.4.1 Decision Tree . . . . .	108
<b>14 Instance</b>	<b>113</b>
14.1 User Interface . . . . .	113
14.1.1 Measuring Scales . . . . .	115
14.1.2 Radar Chart . . . . .	115
14.1.3 Content Box . . . . .	116
14.1.4 Interaction Flow . . . . .	119
14.2 Calculation of Results . . . . .	119
14.2.1 Value Arrays . . . . .	120
14.2.2 Text Arrays . . . . .	120
14.3 Key Design Decisions . . . . .	121
<b>15 Validation</b>	<b>123</b>
15.1 Comparative Case Study . . . . .	123
15.1.1 Test Design . . . . .	123
15.1.2 Data Collection . . . . .	125
15.1.3 Data Analysis . . . . .	125
15.2 Results . . . . .	129
15.2.1 HyPrO Expert System vs. Human Experts . . . . .	130
15.2.2 Scenario 1 . . . . .	131
15.2.3 Scenario 2 . . . . .	133
15.2.4 Scenario 3 . . . . .	135
15.2.5 Test of Hypothesis . . . . .	136
15.2.6 Threats to Validity . . . . .	136

---



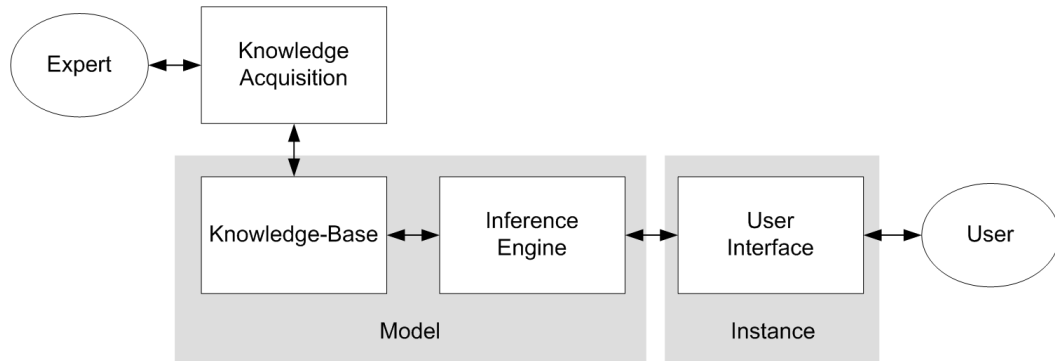
# 12

## Type of Research Artifact

According to the research approach described in Section 1.3, the applied design science research in this thesis results in an artifact which aims to "support operations, management, analysis, and decisionmaking functions in an organization" (Davis and Olson 1985, p. 6, as cited in [36]). It represents the *Design Cycle* of this project and iterates between its design and its evaluation to constantly improve the design [35]. In this thesis, the resulting artifact is the so-called *HyPrO Expert System*, which was built to address the unsolved problem of compressing expert knowledge to support the tailoring of the Product Owner role to hybrid development environments. To achieve this, it is necessary to understand that the design activities are both a process as well as a product [34, 36]. According to Hevner et al. [34, 36], there are two processes (build and evaluate) and four products (constructs, models, methods, and instantiations) of design science. The *HyPrO Expert System* itself represents the product, whereas the research process was to build and evaluate the expert system. In this case, the product contains a model and an instance. The model is generated to understand the real world and to explore the effects of changes in it [36]. The instance is generated to provide evidence, that the artifact is suited to its intended purpose [36].

The generated artifact in this thesis is an expert system. An expert system is a computer program that emulates a human expert in a certain domain [31, 98] to solve a problem or to support the decision-making process [73]. An expert system bundles the knowledge of a certain domain and, thus, is also called *knowledge-based system* [42, 98]. The benefit of expert systems against human experts is that they provide more consistent results, in a shorter time frame when assessing complex tasks [98]. With this, expert systems are a suitable artifact that result from the application of the design science research approach, as they both share the same goal: support the decision making functions, by providing more consistent results, in a shorter time frame, to support the tailoring of the Product Owner role to hybrid development environments.

According to Styczynski et al. [98], expert systems contains four components: *Knowledge-Base*, *Knowledge Acquisition System*, *Inference Engine* and the *User-Interface* as presented in Fig 12.1.



**Figure 12.1:** Components of an expert system

The *Knowledge-Base* bundles the knowledge as facts and builds the foundation of the expert system [31, 42, 98]. It acquires its knowledge by the manual input provided by a human expert via the *Knowledge Acquisition* component. The *Inference Engine* contains rules on how the individual facts are related to each other [31] and the *User-Interface* builds the representation of the knowledge and enables a human user to access the knowledge [98].

However, O’Keeffe [73] highlights that ”ES is simultaneously a piece of software and a model” [73, p.7][13, 99].

To avoid confusion, from here on, the terms are defined as follows: the model describes the knowledge-base and the inference engine, the instance describes the piece of software which includes the user-interface. The process of generating the model, the instance and the overall evaluation of the the resulting *HyPrO-Expert System* are described in the following chapters.



# 13

## Model

The here described model is a product resulting from the research design approach after Hevner et al. [36]. According to the defined terms in Chapter 12, the model contains the knowledge-base and the inference engine. While the knowledge-base builds the foundation of the expert system, the inference engine contains rules on how the individual facts from the knowledge-base are related to each other [31, 42]. To be in accordance with Hevner et al. [36], the author constantly iterated the knowledge-base and the instance during the development of the model.

In this thesis, the knowledge-base is represented by a catalog that incorporates knowledge as facts from the two rigors of the (1) Product Owner Tasks, Characteristics and Structures and the (2) *Software Process Tailoring Criteria*.

To reach to the incorporation of knowledge, constant comparison [24] was applied to identify common rationals of the individual Product Owner tasks, characteristics & structures (resulting from the research of this thesis) and the criteria that tailor the software process (knowledge provided by other researchers).

The inference engine is represented by the related decision tree and truth tables of the defined criteria and the Product Owner peculiarities.

### 13.1 Knowledge-base: Product Owner

While the individual parts of this thesis contribute the resulted codes and phrases for the Product Owner tasks, characteristics and structures, this part further concludes them to reduce duplicates. Therefore, constant comparison [24] was applied. Within this process, the author of this thesis compared the codes to each other to carve out differences as well as similarities until the codes were saturated. Furthermore, the author of this thesis constantly compared the knowledge from the rigor of *Software Process Tailoring Criteria* to filter the most relevant Product Owner tasks, characteristics and structures, to support the tailoring of the Product Owner role to hybrid development environments.

For each of the Product Owners peculiarities (tasks, characteristics, structures), a consolidated set of codes and descriptions was generated by the author with this process. They are presented in this section.

### 13.1.1 Product Owner Tasks

A list of Product Owner tasks has been generated throughout the individual parts of this thesis (Part I, II, III). Overall, 30 tasks have been identified. Although the results of Chapter 9 divided the tasks to the different level of Product Owners (Chief Product Owners and supporting Product Owners), the tasks will not further divided by the level of the Product Owner. This is due to a limited generalizability of the results. However, constant comparison [24] as well as the comparison with knowledge from the rigor of *Software Process Tailoring Criteria* further concluded the considerable tasks and resulted in a list of 13 relevant tasks of a Product Owner in a hybrid environment. Table 13.1 provides an overview of the identified tasks.

**Table 13.1:** Relevant Product Owner Tasks in Hybrid Environments

<b>Task</b>	<b>Description</b>	<b>Source</b>
Communicator	transfer knowledge between stakeholders and development team(s) sites	[6, 7, 8, 41, 55, 70, 72, 81, 100, 103, 104, 105]
Groom	clarify the details of product backlog items, and their respective acceptance criteria	[6, 8, 103, 104, 113]
Prioritizer	select requirements that bring highest value or benefit	[6, 8, 103, 104]
Traveller	travel to the development sites to spend time with the various teams for knowledge sharing	[6, 8, 103, 104]
Intermediary Person	act as an interface between senior roles and the team, to disseminate domain knowledge	[6, 8, 103, 104]
Risk Assessor	assess risk of new feature development	[6, 103, 104]
Gatekeeper	determine feature or story completeness for inclusion in a release	[8, 104]
Acceptance Tester	accept / reject new increments	[7, 8, 67, 74, 104, 113]
Customer Relationship Manager	provide technical support to customers, assists with site preparation and product installation, and do product training	[7, 8, 55, 70, 104]
Expectation Manager	manage expectations of various stakeholders in terms of feasibility and timing of new features	[29, 100, 104]
Super Secretary	perform multiple activities beyond the PO role, e.g. taking over Scrum Master activities	[67, 104]
Visionary	envision the product and share the vision to shape the product	[41, 55, 104]
Decision Maker	make decisions to solve issues during the development process	[68, 104]

### 13.1.2 Product Owner Characteristics

The relevant characteristics have been initially identified by Pichler [79]. Over the course of this thesis, the author conducted some studies that checked on these characteristics [82, 103, 105, 106]. Although the results generally support the characteristics by Pichler, some descriptions of the individual characteristics have been adjusted to fit to hybrid development environments. Table 13.2 provides an overview of the relevant characteristics of a Product Owner in a hybrid environment according to this thesis.

**Table 13.2:** Relevant Product Owner Characteristics in Hybrid Environments

Characteristic	Description	Source
Communicator & Negotiator	Communicate with and align different parties and departments, such as customers, users, development and engineering, marketing, sales, service, operations, and management.	[79, 103, 105, 106]
Visionary & Doer	Envision the final product and see it through to completion. This includes requirements description, closely collaborating with the team, accepting or rejecting work results, and steering the projects by tracking and forecasting its progress. Hereby, the vision of the final product does not come from the Product Owner itself – he rather act as a proxy between all parties and communicates management decisions/visions.	[79, 82, 103, 106]
Leader & Team player	Take responsible for the product’s success, provides guidance for everyone involved and makes tough decisions. As he has no formal authority over the team, he is not perceived as the leader of the Scrum Team. However, as he relies on close collaboration with other Scrum Team members, he should be trustworthy and needs to be a team player.	[79, 103, 106]
Available & Qualified	Being a Product Owner is usually a full-time job, where on average 65% of the time is spent in meetings. Project’s progress suffers when the Product Owner is overworked. Being adequately qualified usually requires an intimate understanding of the customer and the market.	[79, 103, 105, 106]
Empowered & Committed	The Product Owner should bring the product to life. He must have the proper decision-making authority – from finding the right team members to deciding which functionality is delivered as part of the release. In hybrid environments this authority is often minimized as a consensus have to be made among multiple stakeholders. Also, the level of empowerment depends on the level of the Product Owner (Chief / Overall PO vs. supporting PO).	[79, 82, 103]
Manager <sup>a</sup>	The Product Owner should have a strong emphasize on budget and time constrains to successfully conclude the project.	Chapter 15

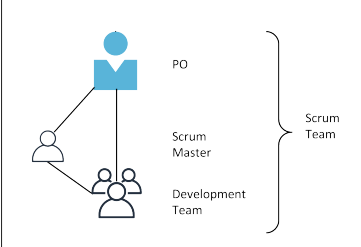
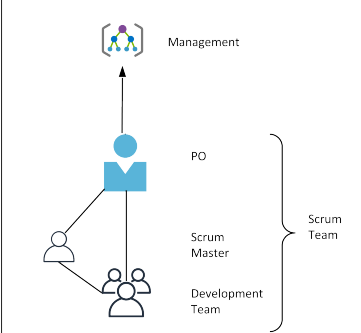
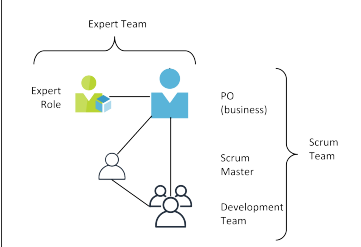
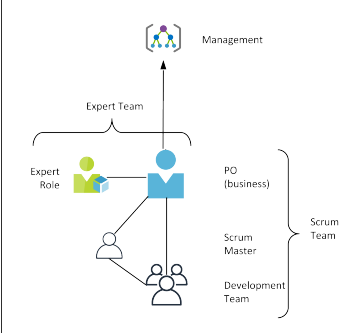
<sup>a</sup>This characteristic resulted from the comparative case study to validate the knowledge-base of the HyPrO Expert System. It was not listed in the knowledge-base by the time of the case study and hence, has not been considered in the validation.

### 13.1.3 Product Owner Structures

Besides the regular Product Owner description from Section 2, there have been some Product Owner team structures reported to scale the agile method by various researchers (see Section 5.2). This knowledge, combined with the gained insights from the conducted studies by the author of this thesis [104, 106], resulted in the identification of three types of Product Owner structures: Single Product Owner, non-hierarchical Product Owner Teams and hierarchical Product Owner Teams. As each of the three structures provides four sub-structures, a total number of 12 Product Owner structure variants have been identified by the research of this thesis. The 12 variants are described in the following:

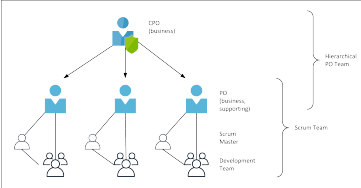
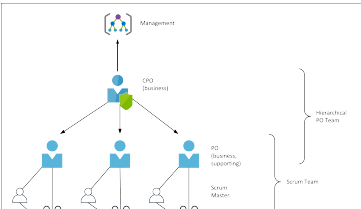
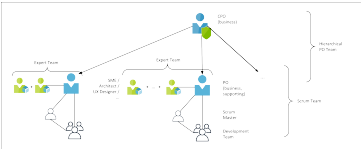
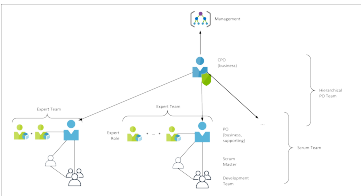
**Single Product Owner** Even in a hybrid environment the Product Owner role does not need to be scaled up necessarily. However, a single Product Owner role can be structured in different ways. According to Sections 5.2 and 10, Product Owners should be teamed up with expert roles and/or emphasize on the collaboration with management roles. Table 13.3 provides an overview of the different structures of a single Product Owner in a hybrid environment according to this thesis.

**Table 13.3:** Single Product Owner Structures for Hybrid Environments

Structure	Description
<p><b>Single</b></p> 	<p>A single Product Owner should do the job. No further management support is required.</p>
<p><b>Single PO with emphasize on management collaboration</b></p> 	<p>A single Product Owner should do the job. The PO should engage with the management.</p>
<p><b>Single PO paired with Expert</b></p> 	<p>A single Product Owner should do the job. The PO should consult an expert for his specialized knowledge. However, this expert does not share any responsibilities with the PO. No further management support is required.</p>
<p><b>Single PO paired with Expert and emphasize on management collaboration</b></p> 	<p>A single Product Owner should do the job. The PO should engage more with the management and consult an expert for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, this expert does not share any responsibilities with the PO. The PO should engage with the management.</p>

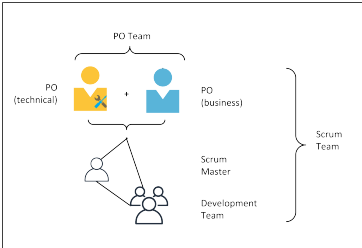
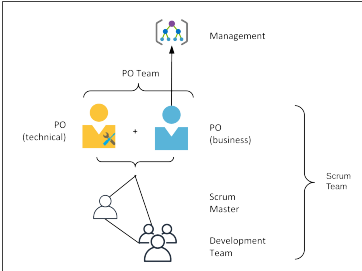
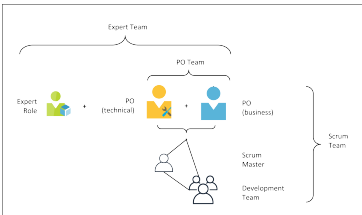
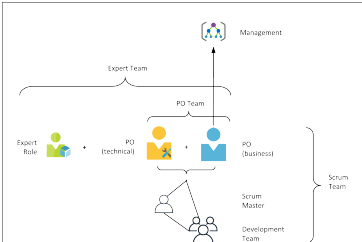
**Hierarchical Product Owner Teams** As stated in Section 5.2, Product Owner teams are state of the practice in hybrid environments. With multiple Product Owners, often tasks and responsibilities are distributed on two levels of Product Owners: the Chief / Overall Product Owner and supporting Product Owners. According to Sections 5.2 and 10, Product Owners should be teamed up with expert roles and/or emphasize on the collaboration with management roles. Table 13.4 provides an overview of the different hierarchical Product Owner Team structures in a hybrid environment according to this thesis.

**Table 13.4:** Hierarchical Product Owner Team Structures for Hybrid Environments

Structure	Description
<p><b>Hierarchical PO team with CPO</b></p> 	<p>A hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business) is recommended. The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Teams(s). No further management support is required.</p>
<p><b>Hierarchical PO team with CPO emphasizes on management collaboration</b></p> 	<p>A hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business) is recommended. The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Teams(s). The CPO should engage with management.</p>
<p><b>Hierarchical PO team with CPO paired with Expert</b></p> 	<p>A hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business) is recommended. The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Teams(s). Additionally, the POs should consult an expert for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain at the CPO and PO solely. No further management support is required.</p>
<p><b>Hierarchical PO team with CPO paired with Exp.and emphasize on mgmt collaboration</b></p> 	<p>A hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business) is recommended. The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Teams(s). Additionally, the POs should consult an expert for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain at the CPO and PO solely. The CPO should engage with management.</p>

**Non-Hierarchical Product Owner Teams** As stated in Section 5.2, Product Owner teams do not need to have to be organized in a hierarchical structure. So can Product Owner teams be non-hierarchical by distributing their work in the areas of business-oriented tasks and technical-oriented tasks [6]. Furthermore, according to Sections 5.2 and 10, Product Owners should be teamed up with expert roles and/or emphasize on the collaboration with management roles. Table 13.5 provides an overview of the different peculiarities of non-hierarchical Product Owner Team structures in a hybrid environment according to this thesis.

**Table 13.5:** Non-Hierarchical Product Owner Team Structures for Hybrid Environments

Structure	Description
<p><b>Basic PO Team</b></p> 	<p>A non-hierarchical PO Team. One PO is responsible for the business requirements, while the other is responsible for technical governance (e.g. architecture). No further management support is required.</p>
<p><b>Basic PO Team with emphasize on management collaboration</b></p> 	<p>A non-hierarchical PO Team is recommended. The PO (business) is responsible for the business requirements, while the other PO (technical) is responsible for technical governance (e.g. architecture). The PO (business) should engage with the management.</p>
<p><b>Basic PO Team paired with Expert</b></p> 	<p>A non-hierarchical PO Team is recommended. PO (business) is responsible for the business requirements, while the other PO (technical) is responsible for technical governance (e.g. architecture). Additionally, an expert should be consulted for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain by the PO Team (business and technical solely). No further management support is required.</p>
<p><b>Basic PO Team paired with Expert and emphasize on management collaboration</b></p> 	<p>A non-hierarchical PO Team is recommended. PO (business) is responsible for the business requirements, while the other PO (technical) is responsible for technical governance (e.g. architecture). Additionally, an expert should be consulted for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain by the PO Team (business and technical solely). The PO (business) should engage with the management.</p>

## 13.2 Knowledge-base: Software Process Tailoring Criteria

As stated in Section 3.2, software process tailoring criteria have been subject to research for quite some time already. Whereas multiple approaches provided meaningful information, the systematic literature study conducted by Kalus and Kuhrmann [43] on this topic provides a good overview of the results.

They present an overall collection of 49 tailoring criteria which are clustered into four categories: Team, Internal Environment, External Environment, Objective [43]. Furthermore, they provided 20 actions that can be used to tailor the process and hence, to overcome the risk of the failure of the project [43].

However, to create the necessary knowledge-base for this thesis, the two rigors of (1) *Product Owner Tasks, Characteristics and Structures* and the (2) *Software Process Tailoring Criteria* had to be incorporated. To reach to the incorporation of knowledge, constant comparison [24] was applied.

With this method, the author of this thesis identified relevant factors and action items that have an influence on the Product Owner role in hybrid development environments.

### 13.2.1 Influencing Factors on Product Owners

With the method of constant comparison [24], the author of this thesis identified 14 factors that influence and hence, require a tailoring of the Product Owner. These 14 factors are a subset of the 49 software process tailoring criteria provided by Kalus and Kuhrmann [43]. The identified factors as well as their rationales are listed by their categories below.

#### 13.2.1.1 Team

**Team Size** The team size is a key criteria for tailoring the Product Owner role as well as selecting the right software process [18]. This is because it is an indicator for both, team coordination [117] as well as numbers of required Product Owners [12, 26, 64, 69, 72]. As soon as the team needs to be split in multiple Scrum Teams, multiple Product Owners should be available and coordinated, which in return, requires a more formalized communication [105].

**Team Distribution** Distributed teams require a Product Owner (representative) to be on-site with the development team to ensure proper communication of the responsibilities and priorities [39, 53, 64, 69, 72, 92]. As this adds up the number of Product Owners, they again, need to be coordinated which requires a more formalized communication [105] as well as travelling of the Product Owners to the various sites [12, 26, 78].

#### 13.2.1.2 Internal Environment

**Clear Project Proposal** A clear project proposal contains goals and requirements [109, 110, 121] that are essential for the project's success. The less clear the goals and the requirements are, the Product Owner has to emphasize more on requirements engineering practices, get the sign-off from management [106] and / or has to create his own vision. All this needs to be properly documented in the backlog [12, 26, 53] and tested by the Product Owner.



**Management Commitment** Management commitment is required to ensure a shared goal and support when it comes to solve problems quickly and to make project progress decisions [109, 110]. The commitment of the management has an influence on the Product Owner tasks, as he is perceived as the link between management and development teams [106].

**Financial Control** Financial control has an impact on the Product Owner with respect to the project budget. A huge budget usually implies a more formalized process with more traditional documentation and more frequent reporting to the project management office. Hence, the Product Owner has to establish a formalized communication with the PMO [106]. It also requires more emphasize on planning [43].

### 13.2.1.3 External Environment

**Number of Stakeholders** The higher the number of involved stakeholders the more the Product Owner has to act as an intermediary person to accommodate all interests [6, 8]. This increases the size of the network of the Product Owner and consequently increases the number of formal interactions [105]. Furthermore, more time is required to negotiate all needs and requirements [43, 79]. To handle the negotiation and communication the Product Owner needs to be qualified to oversee all requirements [79], distinguish and prioritize them, evaluate the risks, manage expectations [6, 8, 29, 100]. Also a high number of stakeholders might require a Product Owner team to handle multiple areas simultaneously.

**Stakeholder / User Availability** Similar to the availability of the top management, the availability of stakeholders is a success factor [43]. As the Product Owners in such an environment rather act as a proxy [82, 105, 106], they rely on the stakeholders feedback and sign-off. This requires regular (and probably even more formalized) communication [105].

**Requirements Stability** The stability of requirements influences the strategy for requirements elicitation, architecting the solution, its implementation and tests [43]. Hence, they are critical for the projects success. The Product Owner needs to be able to assess the stability of the requirements to estimate potential changes. This requires a solid domain knowledge or to be teamed up with a domain expert. He also needs to emphasize on requirements elicitation and validation in case of volatile requirements.

### 13.2.1.4 Objective

**Complexity** The more complex the project is, the more the Product Owner has to emphasize on formalized communication, creating prototypes [43], requirements engineering, architecture or front-end issues [105]. As this might be too much for a single individual, the Product Owner role should be scaled to a Product Owner team so that responsibilities can be shared [105]. Also experts for the individual areas of expertise should be consulted [106].

**Legacy System** A legacy system usually impacts the requirements engineering process [43]. The less documentation is available, the more communication is required to analyze the legacy system. Therefore, Product Owners should either be very familiar with the legacy system or team up with someone who is. To ensure compatibility, the Product Owner should also consult an expert (architect) [106].

**Criticality** Safety & security usually cause a comprehensive documentation of the project [43]. The more safety critical the product is, the more quality standards needs to be considered. The Product Owner needs to be trained on this.

**Hardware Development** If hardware development is also part of a project, the Product Owner becomes an interface between the hardware group and the software group [105]. If the hardware follows a more traditional development approach, the Product Owner needs to align the work and emphasize on planning [43]. Furthermore, the Product Owner should either be keen on hardware development or be teamed up with an expert. Also a Product Owner team could be established to share responsibilities [106].

**User Interface** The Product Owner should be teamed up with an expert (UX Designer / Front-end Designer) [106].

**System Integration Test** A required system integration test needs to be prepared properly. This requires the Product Owner to emphasize on planning and setting up a test integration team [43].

### 13.2.2 Action Items

In addition to the software process tailoring criteria, Kalus and Kuhrmann [43] provided 20 action items from which 14 have been identified as relevant for Product Owners in hybrid development environments. The action items are clustered into four categories: Stakeholder-related, Project life cycle, Project organization, Knowledge building/preservation [43]. The relevant action items for this thesis are clustered, labeled and described in Table 13.6.

**Table 13.6:** Relevant Action Items for Product Owners

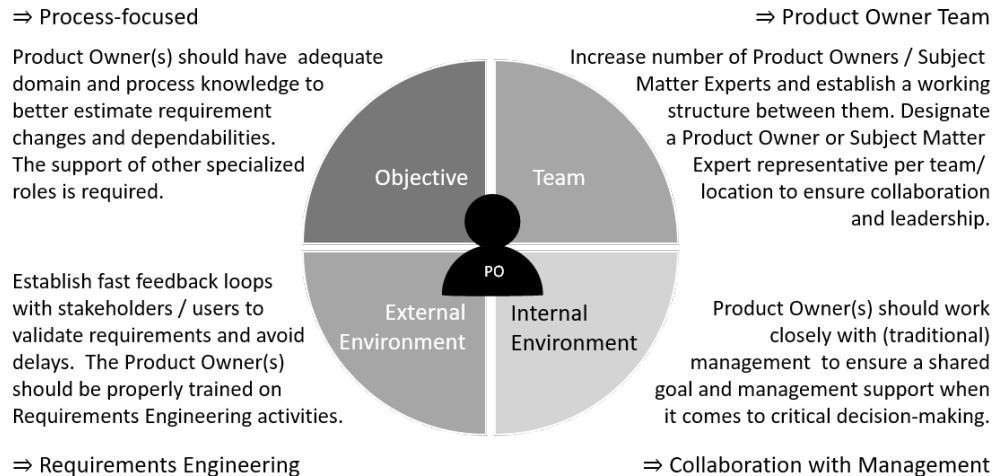
Stakeholder-related	ACI AUI AMI	intensify customer involvement intensify end user involvement, e.g. ui testing ensure management involvement
Project Life Cycle	ARE AAR AIT AFI AFL	put emphasis on requirements engineering put emphasis on system architecture put emphasis on integration and test put emphasis on financial project management, e.g. collaborate with PMO put emphasis on fast feedback loops
Project Organization	ADF ACP ATI	expand project documentation, e.g (agile) RE tool formalize project communication pattern, e.g. schedule regular meetings beyond the used framework select appropriate tools w.r.t. the process's weight
Knowledge-related	AIW ATT AKM	intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops provide training, e.g., role-specific training provide knowledge management infrastructure

### 13.3 Resulting Catalog

The incorporation of the two rigors of the (1) *Product Owner Tasks, Characteristics & Structures* and the (2) *Software Process Tailoring Criteria* result in a catalog (the full catalog can be found in Appendix C) which combines the following:

- **Influencing factors** on Product Owners along with their rationales (presented in Section 13.2.1),
- Respective **implications** on the Product Owner role according to the research conducted over the course of this thesis
- Mapping of the **corresponding Product Owner tasks, characteristics & structures and action items** (presented in Section 13.1 and 13.2.2)

The catalog provides knowledge to deduce two level of recommendations: individual recommendations for each factor and high-level recommendations for each category. The high-level recommendations contain a consolidated recommendation on the structure of the Product Owner. So implies the category *Team*, that the Product Owner role should be scaled, the category *Internal Environment*, that the Product Owner(s) should collaborate with the management, the category *External Environment*, that the Product Owner(s) should emphasize on Requirements Engineering activities and the category *Objective*, that the Product Owner(s) should focus on the process. This is depicted in Figure 13.1.



**Figure 13.1:** Category based high-level recommendations

Although both level of recommendations are important contributions to support the tailoring of the Product Owner role to hybrid development environments, the catalog alone is not sufficient to serve the overall goal of this thesis. This is because, whenever multiple factors play together – which is usually the case in a hybrid environment – dependencies become too complex and a reasonable recommendation for an isolated factor might not work in the combination with other factors. To overcome this, rules need to be defined on how the individual facts from the knowledge-base are related to each other [31, 42], which is described in the following section.

## 13.4 Inference Engine

The inference engine contains rules on how the individual facts from the knowledge-base are related to each other [31, 42]. These rules are necessary to achieve the overall goal of this thesis: to deduce recommendations to tailor the Product Owner role to hybrid development environments.

By the use of logical expressions, the *HyPrO Expert System* can deduce recommendations for the Product Owner tasks, characteristics & structures from **a set** of *influencing factors on Product Owners* – rather than just from one factor at a time, as this is the case in the catalog (see Section 13.3).

Therefore, Formula 13.1 applies:

$$\{P1, P2, \dots, Pk\} \vdash C \quad (13.1)$$

where:

$P$  = influencing Factor on Product Owner

$C$  = Clauses of relevant Action Items (Section 13.2.2) + Product Owner Tasks, Characteristics, Structures (Section 13.1)

In this thesis, a set of 14  $P$ s (influencing factors on Product Owners) has been identified. For the *HyPrO Expert System* the  $P$ s are defined as in Table 13.7.

**Table 13.7:** Defined  $P$ s for the *HyPrO Expert System*

Category	Factor	$P$
Team	Team Size	T.1
	Team Distribution	T.2
Internal Environment	Clear Project Proposal	IE.1
	Management Commitment	IE.2
	Financial Control	IE.3
External Environment	Number of Stakeholders	EE.1
	Stakeholder / User Availability	EE.2
	Requirements Stability	EE.3
Objective	Complexity	O.1
	Legacy System	O.2
	Criticality	O.3
	Hardware Development	O.4
	User Interface	O.5
	System Integration Test	O.6

For the recommendations ( $C$ s) a set of 45 relevant Action Items + Product Owner Tasks, Characteristics & Structures has been identified in Section 13.1 and 13.2. This set is summarized in Table 13.8.

**Table 13.8:** Set of relevant Action Items and Product Owner Peculiarities for the *HyPrO Expert System*

	Category	Clause
Action Items	Stakeholder-related	intensify customer involvement (ACI) intensify end user involvement, e.g. ui testing (AUI) ensure management involvement (AMI)
	Project Life Cycle	put emphasis on requirements engineering (ARE) put emphasis on system architecture (AAR) put emphasis on integration and test (AIT) put emphasis on financial project management (AFI) put emphasis on fast feedback loops, e.g., continuous delivery and deployment, on-site-customer (AFL)
	ProjectOrganization	expand project documentation, e.g., formalized documentation using templates (ADF) formalize project communication pattern (ACP) select appropriate tools w.r.t. the process's weight (ATI)
	Knowledge-related	intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops (AIW) provide trainings, e.g., role-specific trainings (ATT) provide knowledge management infrastructure (AKM)
Product Owner Peculiarities	Tasks	Communicator Groom Prioritiser Intermediary RiskAssesor Gatekeeper Acceptance Tester Customer Relationship Manager Expectation Manager Super Secretary Visionary Decision Maker
	Characteristics	Communicator & Negotiator (C&N) Visionary & Doer (V&D) Leader & Team player (L&T) Available & Qualified (A&Q) Empowered & Committed (E&C)
	Structure	<b>Single Product Owner</b> Single PO Single PO + mgmt Single PO + exp Single PO + exp + mgmt  <b>Non-Hierarchical Product Owner Teams</b> Basic PO Team

*Continued on next page*

Table 13.8 – Continued from previous page

	Category	Clause
	Structure	Basic PO Team + mgmt Basic PO Team + exp Basic PO Team + exp + mgmt  <b>Hierarchical Product Owner Teams</b> Hierarchical PO team with CPO Hierarchical PO team with CPO+ mgmt Hierarchical PO team with CPO + exp Hierarchical PO team with CPO + exp + mgmt

For this thesis, a classification of the *Ps* is required to determine the particular hybrid development environment and thus, to deduce the adequate recommendations on the Product Owner role. Therefore, a decision tree was used.

### 13.4.1 Decision Tree

A decision tree was used as it is suitable to classify data items based on their features [42]. A decision tree contains of two types of nodes: the internal node and the leaf node [42]. The internal node splits into different branches according to the values that the corresponding attribute can take [42]. The leaf node classifies the branch [42].

#### 13.4.1.1 Values and Classifications

For the *HyPro Expert System*, two classes (branches) have been defined: (1) no tailoring required, (2) tailoring required.

To classify the **factors individually**, applicable values and thresholds have been defined for each factor. Table 13.9 represents an overview of the defined values and thresholds for each factor as well a for the categories.

Therefore, following rule applies:

$$P_{val} \geq Threshold \implies C \tag{13.2}$$

where:

$P_{val}$  = Value of Factor

$C$  = Clauses of relevant Action Items (Section 13.2.2) + Product Owner Tasks, Characteristics, Structures (Section 13.1)

The assigned values are based on individual rationals, which can be reviewed in Appendix E.

**Table 13.9:** Defined Values & Classifications per Factor (P)

Category	<i>P</i>	Values	Threshold	Classification
Team Category	<b>T.all</b>	<b>7</b>	<b>3.5</b>	$< Threshold = (1)$ no tailoring required $\geq Threshold = (2)$ tailoring required
	T.1	3	1.5	
	T.2	4	2	
Internal Environment Category	<b>IE.all</b>	<b>15</b>	<b>7.5</b>	
	IE.1	5	2.5	
	IE.2	5	2.5	
	IE.3	5	2.5	
External Environment Category	<b>EE.all</b>	<b>14</b>	<b>7</b>	
	EE.1	4	2	
	EE.2	5	2.5	
	EE.3	5	2.5	
Objective Category	<b>O.all</b>	<b>18</b>	<b>9</b>	
	O.1	5	2.5	
	O.2	2	1	
	O.3	5	2.5	
	O.4	2	1	
	O.5	2	1	
	O.6	2	1	

For example, the factor *Team Size* (T.1) provides the following three options (values):  $< 10$  ( $< Threshold$ );  $11 - 30$  ( $= Threshold$ );  $> 30$  ( $> Threshold$ ). These three options derive from the knowledge that a Scrum Team should not have more than 9 team members per team [92] and a Product Owner should not work with more than 2 Scrum Teams at a time [23]. This leads to the following implications (the implied *Cs* are based on the catalog that can be found in Appendix C):

**If** the team size is  $< 10$  team members ( $P_{val} < Threshold$ ),  
**then** the entire team could be formed in one Scrum Team.  
 $\implies$  Implication: No tailoring is required: No *C* implies.

**If** the team size is  $11 - 30$  team members ( $P_{val} = Threshold$ ),  
**then** the team might be split up into more than one Scrum Team.  
 $\implies$  Implication:  $C = ADF, ACP$  (Action Items) + Communicator (PO Tasks); C&N, A&Q, L&T (PO Characteristics); Team (PO Structure)

**If** the team size is  $> 30$  team members ( $P_{val} > Threshold$ ),  
**then** the team has to be split in more than two Scrum Teams.  
 $\implies$  Implication:  $C = ADF, ACP$  (Action Items) + Communicator, Traveller (PO Tasks); C&N, A&Q, L&T (PO Characteristics); Team (PO Structure)

For the 14 factors( $P$ s), the author of this thesis defined 54 values. To deduce adequate recommendations for hybrid development environments, multiple – or even – all possible combinations of factors should be considered at a time. With the identified number of 54 values and 14 factors, a consideration of all possible combination of factors would lead to  $\frac{54^{14}}{14} \approx 4^{14}$  (268435456) combinations and hence, would require  $\approx 4^{14}$  different recommendations. With such a high number, the *HyPrO Expert System* would become too complex and the resulting recommendations would not be distinguishable enough to provide added-value.

To reduce the complexity, the *HyPrO Expert System* only considers the combinations of factors within each category and the combination of the categories itself. To classify the **categories**, the values of the respective factors are summed up and compared to the defined threshold of the category. With this, the following numbers of combinations and hence, clauses of relevant Action Items + Product Owner Tasks, Characteristics, Structures ( $C$ s) are achieved:

Team (2 Factors) =  $2^2 = 4$  combinations = 4  $C$ s.

Internal Environment (3 Factors) =  $2^3 = 8$  combinations = 8  $C$ s.

External Environment (3 Factors) =  $2^3 = 8$  combinations = 8  $C$ s.

Objective (6 Factors) =  $2^6 = 64$  combinations = 64  $C$ s.

Overall (4 Categories) =  $2^4 = 16$  combinations = 16  $C$ s.

The resulting  $C$ s have been generated by a logical disjunction of the factors as well as a logical disjunction of the categories. Exemplary, the resulting truth table for category *Internal Environment* is presented in Table 13.10. All truth tables can be found in Appendix D.

**Table 13.10:** Truth Table for Category *Internal Environment*

			IE1	IE2	IE3	IE1 ∨ IE2	IE1 ∨ IE3	IE2 ∨ IE3	IE1 ∨ IE2 ∨ IE3
Action Items	Stakehol.-related	ACI							
		AUI							
		AMI	1	1		1	1	1	1
	Project Life Cycle	ARE	1			1	1		1
		AAR							
		AIT							
		AFI			1		1	1	1
	Project Orga	AFL							
		ADF							
		ACP			1		1	1	1
	Knowl.-related	ATI							
		AIW							
		ATT							
		AKM							

*Continued on next page*



Table 13.10 – Continued from previous page

						IE1 ∨ IE2	IE1 ∨ IE3	IE2 ∨ IE3	IE1 ∨ IE2 ∨ IE3
			IE1	IE2	IE3				
Product Owner Peculiarities	Tasks	Communicator							
		Groom	1			1	1		1
		Prioritizer							
		Travel							
		Intermediary							
		Risk Assessor							
		Gate Keeper							
		Acceptance Tester	1			1	1		1
		Customer Rel. Manager		1		1		1	1
		Expectation Manager							
		Super Secretary							
		Visionary	1			1	1		1
		Critical Dec. Maker							
	Charact.	C&N			1		1	1	1
		V&D	1			1	1		1
		L&T	1	1		1	1	1	1
		A&Q							
		E&C	1			1	1		1
	Structure	single							
		PO Team	1	1		1	1	1	1
		Expert Team	1	1		1	1	1	1

A decision tree result for each category individually, as well as for the combination of the categories. The decision tree visualizes the different recommendations (*Cs*) based on the classification of the individual factors. Exemplary, the resulting decision tree for category *Internal Environment* is depicted in Figure 13.2.

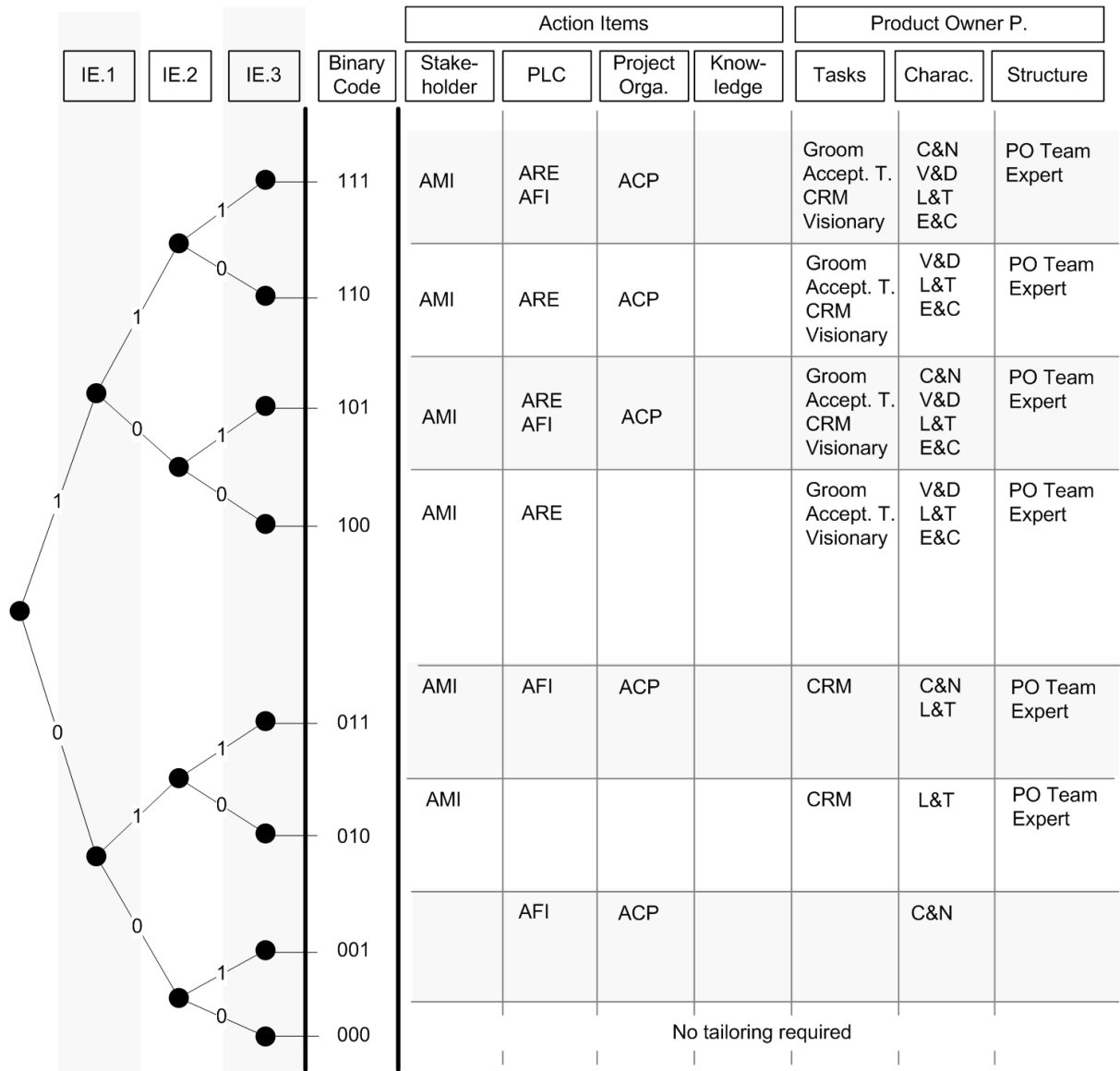


Figure 13.2: Decision Tree for Category Internal Environment

# 14

## Instance

The resulting product from this thesis contains a model and an instance. Whereas the model is generated to understand the real world and to explore the effects of changes in it [36], the instance is generated to provide evidence, that the artifact is suited to its intended purpose [36]. To provide such evidence, the author of this thesis developed the *HyPrO Expert System* as a proof of concepts (POC) in a web application.

The intention of this POC-web application is to visualize the model and make the results easily accessible for users. Based on the input of the user, the web application calculates the results and presents recommendations on how to tailor the Product Owner role according to the particular hybrid environment. In accordance to the applied design science research by Hevner et al. [36], the author constantly iterated the design and knowledge-base during the development of the web-application.

This chapter describes the User Interface elements of the *HyPrO Expert System* POC-web application, its general work flow and the calculation of the results. Furthermore, it discusses its key design decisions.

### 14.1 User Interface

This section describes the User Interface of the *HyPrO Expert System*, its work flow and elements. Figure 14.1 presents the user interface of the *HyPrO Expert System*. The numbers indicate the three sections: (1) Factors, (2) Radar Chart, (3) Content Box.

## HyPrO Model

To get recommendations on how to tailor your Hybrid Product Owner - please rate the statements below.

### Team Factors

#### T1

Size of the overall team



#### T2

Team(s) location



### Internal Environmental Factors

#### IE1

The initial project proposal was clear and concrete



#### IE2

Management is committed to the project goals and available to the project



#### IE3

Project is financially controlled



### External Environmental Factors

#### EE 1

Number of involved stakeholders



#### EE 2

Stakeholders / users are available



#### EE 3

Requirements are stable



### Objective Factors

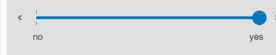
#### O1

Project is complex



#### O2

Project needs to consider a legacy system



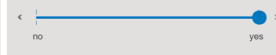
#### O3

Criticality of product



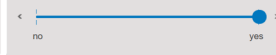
#### O4

Project contains hardware development as well



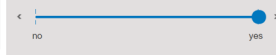
#### O5

Project has a front-end component

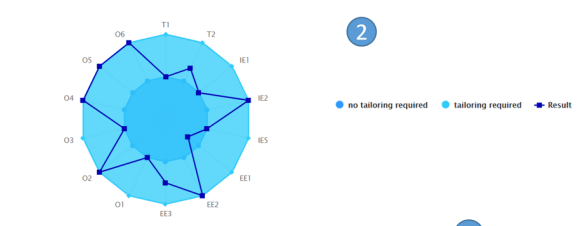


#### O6

Project requires a system integration test



## HyPrO Chart



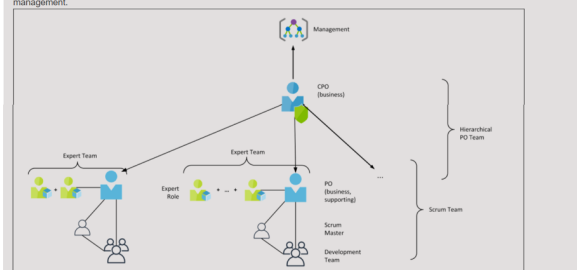
### Overview

Team Internal Environment External Environment Objective

- #### Recommendations for Tailoring
- Increase number of Product Owners / Subject Matter Experts and establish a working structure between them. Designated Product Owner or Subject Matter Expert representatives per team / location ensures collaboration and leadership. [Team]
  - Product Owner(s) should work closely with (traditional) management to ensure a shared goal and management support when it comes to critical decision-making. [Internal Environment]
  - Establish fast feedback loops with stakeholders / users to validate requirements and avoid delays. The Product Owner(s) should be properly trained on Requirements Engineering activities. [External Environment]
  - Product Owner(s) should have adequate domain and process knowledge to better estimate requirement changes and dependencies. The support of other specialized roles is required. [Objective]

#### Recommended Structure

Form a hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business). The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Team(s). Additionally, the POs should consult an expert for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain by the CPO and PO solely. The CPO should engage with management.



#### Recommended Action Items

- intensify customer involvement
- intensify end user involvement, e.g. ui testing
- ensure management involvement
- put emphasis on requirements engineering
- put emphasis on system architecture
- put emphasis on integration and test
- put emphasis on financial project management, e.g. collaborate with PMO
- put emphasis on fast feedback loops,
- expand project documentation, e.g (agile) RE tool
- formalize project communication pattern, e.g. schedule regular meetings beyond the used framework
- select appropriate tools w.r.t. the process's weight
- intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops
- provide trainings, e.g., role-specific trainings
- provide knowledge management infrastructure

Figure 14.1: HyPrO Expert System User Interface

### 14.1.1 Measuring Scales

The measuring scales are represented in Point 1 in Figure 14.1. The *HyPrO Expert System* uses measuring scales to get information about the particular hybrid environment of the user, to assess whether the Product Owner needs to be tailored to this environment. Therefore, the identified influencing factors on Product Owners (described in 13.2.1) are grouped and presented by their categories. Every factor has a statement that the user should rate to assign the according value. The values are defined as described in Section 13.4.1 and in Appendix E.

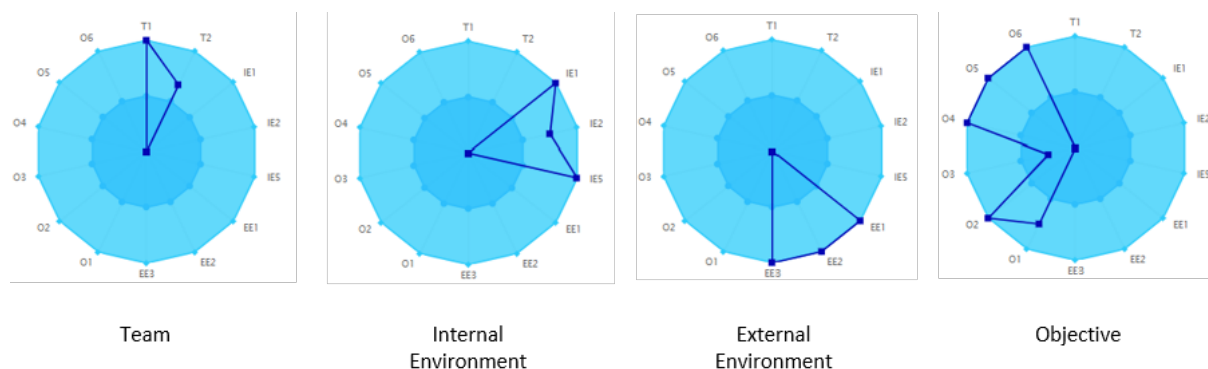
While some of the scales use likert-items [2] to let the user rate the statement, others simply distinguish between *yes / no* or provide actual numeric values. The scales are based on rationals how each value impacts the Product Owner role. Also they are similar to the ones that [16, 58, 119] have used for their peer-reviewed models.

Even though the scales work like a multiple-point scale, they are presented as sliders. This is to accommodate a consistent interface for the user. Also, the slider positions are arranged consistently; slider position on the left of the slider bar indicates that no tailoring is required, slider position on the right of the slider bar indicates that tailoring is required for this factor. The decision whether tailoring is required is based on pre-defined thresholds by the author of this thesis. A detailed explanation of the thresholds and the respective calculation of the results is presented in Section 14.2.

### 14.1.2 Radar Chart

The radar charts are represented in Point 2 in Figure 14.1. Radar charts are a form of graph that allows a visual comparison between multiple aspects of a situation. In this case, the *HyPrO Radar Chart* visually represents the classification of the influencing factors. Every factor that is placed within the inner, dark blue area is classified as *no tailoring required*, every factor that is placed within the outer, light blue area is classified as *tailoring required*.

It also allows to the user to easily assess and compare the classification of the different categories. Figure 14.2 depicts the class of 'tailoring required' for each category.

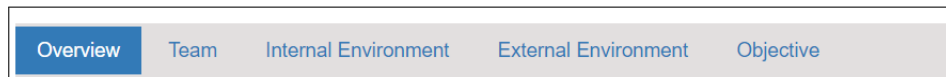


**Figure 14.2:** Comparison among Categories

The *HyPrO Radar Chart* is based on the 'Home-Ground-Chart' by Boehm and Turner [16]. The Home Ground charts indicate visually whether a more light-weight or heavier-based approach should be applied. The indication depends on the values on the axis. Assigned values towards the center of the graph indicate a light-weight approach. Assigned values towards the periphery of the graph indicate a heavier-weight approach. The *HyPrO Radar Chart* indicates that the Product Owner needs to be tailored when the values are assigned toward the periphery of the graph and vice versa.

### 14.1.3 Content Box

The content box (represented in Point 3 in Figure 14.1) contains multiple tabs with individual tab content. The content is the representation of the knowledge gained in this thesis as described in Section 13.4.1. With this, the knowledge is structured in an overview tab as well as tabs for the individual categories.



**Figure 14.3:** Tab Row

#### 14.1.3.1 Tab: Overview

This tab presents the recommendations based on the input of all categories (as described in Section 13.4.1). To provide the user a better perception of the information, the recommendations are split into the following parts: Recommendations for Tailoring, Recommended Structure, Recommended Action Items. Exemplary, the content of the overview tab is presented in Figure 14.4.

Overview Team Internal Environment External Environment Objective

### Recommendations for Tailoring

- Increase number of Product Owners / Subject Matter Experts and establish a working structure between them. Designated Product Owner or Subject Matter Expert representatives per team / location ensures collaboration and leadership. [Team]
- Product Owner(s) should work closely with (traditional) management to ensure a shared goal and management support when it comes to critical decision-making. [Internal Environment]
- Establish fast feedback loops with stakeholders / users to validate requirements and avoid delays. The Product Owner(s) should be properly trained on Requirements Engineering activities. [External Environment]
- Product Owner(s) should have adequate domain and process knowledge to better estimate requirement changes and dependencies. The support of other specialized roles is required. [Objective]

### Recommended Structure

Form a hierarchical Product Owner Team with a CPO (business) and a supporting set of POs (business). The CPO is responsible for the overall backlog, while (supporting) POs are responsible for certain areas of the backlog. Also, the CPO is not assigned to a specific Scrum Team, while the POs have their dedicated Scrum Teams(s). Additionally, the POs should consult an expert for specialized knowledge (e.g. Subject Matter Expert, UX Designer). However, the responsibilities remain by the CPO and PO solely. The CPO should engage with management.

### Recommended Action Items

- intensify customer involvement
- intensify end user involvement, e.g. ui testing
- ensure management involvement
- put emphasis on requirements engineering
- put emphasis on system architecture
- put emphasis on integration and test
- put emphasis on financial project management, e.g. collaborate with PMO
- put emphasis on fast feedback loops,
- expand project documentation, e.g (agile) RE tool
- formalize project communication pattern, e.g. schedule regular meetings beyond the used framework
- select appropriate tools w.r.t. the process's weight
- intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops
- provide trainings, e.g., role-specific trainings
- provide knowledge management infrastructure

Figure 14.4: Tab 'Overview' Content

**Recommendations for Tailoring** For every category that has been classified as equal or above to the defined threshold ( $\geq Threshold$ ), a bullet point is listed with the respective high-level recommendations that have been deduced from the catalog (Section 13.3).

**Recommended Structure** This area explains the recommended structure that results from the input of all categories by the user. This area represents the knowledge of the Product Owner *Structure* that have been gained with this thesis. It presents an image that depicts the concept of the recommended structure along with a short explanation. The structure is highly dependable

on the consideration of all categories. This is why the structure is presented in the overview tab solely.

**Recommended Action Items** This area lists all action items that result from the input of all categories by the user. The action items are not phrased in full sentences so that they are easy to understand by the user.

### 14.1.3.2 Tabs by Category

The category tabs represent the categories of the influencing factors. The content of each tab provides detailed recommendations on the selected factor and is split into the following parts: Recommended set of skills the Product Owner(s) should have, Recommended tasks the Product Owner(s) should emphasize on, Recommendations on how this could be achieved. Exemplary, the content of a category tab is presented in Figure 14.5.

Overview Team Internal Environment External Environment **Objective**

Detailed Recommendations for Objective Factors

- **Product Owner(s) should have adequate domain and process knowledge to better estimate requirement changes and dependabilities. The support of other specialized roles is required. [Objective]**

The Product Owner(s) should have the following skills:

- 🗨️ **Communicator & Negotiator**  
Communicate with and align different parties including customers, users, development and engineering, marketing, sales, service, operations, and management.
- 👑 **Leader & Team player**  
Take responsible for the product's success, provides guidance for everyone involved and makes tough decisions. He needs to be a team player, rely on close collaboration with other Scrum team members, yet has no formal authority over them.
- 💎 **Available & Qualified**  
Being a Product Owner is usually a full-time job. Project's progress suffers when the PO is overworked. Being adequately qualified usually requires an intimate understanding of the customer and the market.

The Product Owner(s) should emphasize on the following tasks:

- transfer knowledge between stakeholders and development team(s) sites. [COMMUNICATOR]
- act as an interface between senior roles and the team, to disseminate domain knowledge. [INTERMEDIARY]
- determine feature or story completeness for inclusion in a release. [GATEKEEPER]
- accept /reject new increments. [ACCEPTANCE TESTER]
- manage expectations of various stakeholders in terms of feasibility and timing of new features. [EXPECTATIONS MANAGER]
- perform multiple activities beyond the PO role, e.g. taking over scrum master activities. [SUPER SECRETARY]

This can be achieved by the following:

- put emphasis on requirements engineering
- put emphasis on system architecture
- put emphasis on integration and test
- put emphasis on fast feedback loops,
- expand project documentation, e.g (agile) RE tool
- formalize project communication pattern, e.g. schedule regular meetings beyond the used framework
- select appropriate tools w.r.t. the process's weight
- intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops
- provide trainings, e.g., role-specific trainings
- provide knowledge management infrastructure

Figure 14.5: Tab Content of Category:Objective



**Recommended set of skills** This area represents the knowledge of the Product Owner *Characteristics* that have been gained with this thesis. It lists all mapped characteristics along with a short explanation.

**Recommended tasks** This area represents the knowledge of the Product Owner *Tasks* that have been gained with this thesis. It lists all mapped tasks along with a short explanation.

**Recommendations on how this could be achieved** This area represents the knowledge of the *Action Items* and lists all mapped action items along with a short explanation.

#### 14.1.4 Interaction Flow

The interaction flow of the *HyPrO Expert System* can be described as follows:

1. The default setting of the sliders are set in a way that no factor is above the threshold. This implies that no tailoring of the Product Owner is required. The structure of a single Product Owner is presented.
2. The user assesses the current state of the project by rating the statements of each category.
3. A function within the *HyPrO* web application gets the values from the sliders and compares them to the predefined threshold (described in Table 13.9 and Section 14.2.1).
4. The chart updates immediately and visually represents the classification of the factor.
5. The tabs present the recommendations based on the input in accordance to the decision tree and truth tables.

## 14.2 Calculation of Results

Based on the input of the user, the POC-web application calculates the results and presents recommendations on how to tailor the Product Owner role according to that particular hybrid environment. Therefore, a function has been used in the POC-web application, to evaluate the input and return the respective recommendation. Figure 14.6 depicts the function.

```
function eval(texts,values){
  thr = 50;
  v = 0;
  for(var i=0; i<values.length; i++){
    if(values[i]>=thr){
      v=v+Math.pow(2,i);
    }
  }
  return texts[v];
}
```

**Figure 14.6:** Function to evaluate the input and return recommendation

The function can be explained as follows:

```

function 'eval', input: array of scaled values and array of texts

set thr (threshold) to 50
set v to 0 // value that determines the texts later on
go through each item in the array "values"
if the value in the array "values" is above the threshold, increase v by 2 to the power of the
current position

```

The loop is performing a conversion of the values into a binary number (1=equal or above threshold, 0=below threshold) and follows it up with a binary to decimal result (in the least significant bit first domain). The following examples show the conversion:

```

input: [60,0,0] → [1, 0, 0] → v = (20) + (01) + (02) = 1
input: [60,60,0] → [1, 1, 0] → v = (20) + (21) + (02) = 3
input: [60,60,60] → [1, 1, 1] → v = (20) + (21) + (22) = 7

```

The return of the function is the *v*-st element of the text array.

### 14.2.1 Value Arrays

The values that are defined for each factor (*P*) of the *HyPrO Expert System* (see Table 13.9) are stored in value arrays.

While the values within the model range from 2 to 5 options (see Section 13.4.1), the values of the factors needed to be scaled within the instance of the model due to computational reasons. In the function 'eval', the threshold is set to 50. Therefore, the value of each factor has been scaled to a range from 0 – 100 with its predefined numbers of options.

Also, as stated in Section 14.1 sliders are not sliders but a multiple-point scale. With this, the user can only decide between certain values and hence set the value either below, right on, or above the threshold.

The returned recommendations (aka text modules) are based on the values. Only if a value is **equal or above** threshold, tailoring is required and the respective text for this slider is displayed (see definition in Table 13.9).

### 14.2.2 Text Arrays

As stated in Rule 13.2 (defined in Section 13.4.1), the value of each factor compared to the threshold implies the respective clauses of Action Items + Product Owner Tasks, Characteristics, Structures. Within the instance of the POC-web application of the *HyPrO Expert System*, this requires a collection of text modules than can be returned. The text modules are a representation of the model (described in Chapter 13) and thus, a representation of the two rigors of the Product Owner and Software-Process-Tailoring Criteria knowledge-bases. The individual text modules are stored as elements in a text array and are returned by the function 'eval' described above.

### 14.3 Key Design Decisions

This section discusses key design decision that have been made for the *HyPrO Expert System*.

**Why an Expert System?** The decision to build an expert system is based on the fact that it supports the intended research goal as well as the applied design science research approach (as addressed in Section 12).

**Limitations of the *HyPrO Expert System*** The *HyPrO Expert System* is realized as a proof of concept web application. This means the *HyPrO Expert System* does not have an external relational-database which can be queried with rule-based logic. This is because the main focus of this thesis should be on the gain of knowledge rather than the implementation of a full expert system based on formal logic (e.g. with PROLOG). The *HyPrO Expert System* is a means to an end to make the developed knowledge-base measurable. However, the *HyPrO Expert System* contains rules on how the individual facts from the knowledge-base are related to each other and uses logical expressions to deduce recommendations to support the tailoring of the Product Owner role to hybrid development environments (as addressed in Section 13.4).

**Validity of Results** The calculation of the resulting recommendations is based on the input values (provided by the user) and the pre-defined threshold. As the validity of the results impacts the overall validity of the *HyPrO Expert System*, the author of this thesis reduced the error-proness of the input-options by providing multi-point scales instead of sliders (addressed in Section 14.1). Furthermore, the input values are converted into binary numbers which categorize the values into above / below threshold. This makes the results more reliable and less prone to errors.



# 15

## Validation

Based on the applied design research approach described in Section 1.3, the resulting artifact needs to be returned into the environment (Relevance Cycle) to be validated. In the case of this thesis, the resulting artifact is the *HyPrO Expert System* which is based on the gained and consolidated knowledge that resulted from this thesis. Furthermore, the validation of the *HyPrO Expert System* is needed to support or refute the hypothesis of this thesis. As this hypothesis states that *a knowledge-base expert system, which provides equal or more comprehensive recommendations than human experts, would support the tailoring of the Product Owner role to hybrid development environments* (see Section 1.2), it requires the results of the *HyPrO Expert System* to be compared to human experts.

However, a reported issue in the validation of expert systems is that the results of the *HyPrO Expert Systems* could not only be evaluated by experts [73]. This is due to the fact of the so called 'super-human fallacy' [17] where human are biased (pro and against) when they know that the problem solution was generated by a computer. To bypass this, as well as to provide evidence to support the hypothesis of this thesis, a comparative case study was conducted.

The details of the validation are discussed in the following sections.

### 15.1 Comparative Case Study

To validate expert systems, test cases are an accepted method [73]. Here, test cases are solved by human experts and are run through the system. Eventually, the output solutions are compared. For this thesis, a comparative case study [85, 120] with three test cases (scenarios) and four human experts has been conducted. The three different scenarios cover a representative set of test cases [31] and each case was repeated four times to compare alternative descriptions and explanations [120].

#### 15.1.1 Test Design

To validate the knowledge-base of the *HyPrO Expert System*, a set of test problems needs to be setup upon which the competence of human and the system can be measured [31]. This test setup should be representative for the domain while not being too complex [31, 73].

Therefore, the author of this thesis generated three fictional scenarios to measure the competence of the system and compare it to human experts. Here, human experts have been asked to provide a strategy to tailor the Product Owner role to three fictional scenarios. A non-expert ran the test case through the *HyPrO Expert System* to provide a strategy to tailor the Product Owner to the same fictional scenarios. Eventually, the author of this thesis compared all results. The expert system is considered as valid when its strategy is rated at least as competent as the strategy provided by the human experts.

**Test Case Scenarios** The three test case scenarios were all fictional and intended to cover the full range of recommendations to tailor the Product Owner role to the environment. The information were grouped by the categories of the influencing factors on the Product Owner. Table 15.1 represents the scenarios.

**Table 15.1:** Test Case Scenarios

Factor	Scenario 1	Scenario 2	Scenario 3
T.1	Group of 10 developers	Group of 10 developers	Group of >30 developers
T.2	Co-located in one location	Co-located in one location	Distributed globally
IE.1	The initial project was proposal was clear	The initial project was proposal was not clear	The initial project was proposal was clear and concrete
IE.2	Management is committed to project goals but not available	Management is not fully committed and only available rarely	Management is not fully committed and only available rarely
IE.3	Project is not strongly financially controlled	Project is strongly financially controlled	Project is strongly financially controlled
EE.1	Number of stakeholders is high	Only few stakeholders are involved	Only few stakeholders are involved
EE.2	Neither the stakeholders nor the users are available for testing purposes	Neither the stakeholders nor the users are available to discuss requirements	The stakeholders as well as the users are available for testing purposes
EE.3	Requirements are stable	Requirements are not stable	Requirements are stable
O.1	Project is rather simple	Project is rather complex	Project is not considerably complex
O.2	Project does not need to consider a legacy system	Project does not need to consider a legacy system	Project needs to consider a legacy system
O.3	Criticality of developed product comfortably low (some money would be lost, but no lives are in danger)	Criticality of developed product is in lower mid-range (money would be lost, but no lives are in danger)	Criticality of developed product is in mid-range (some money would be lost, but no lives are in danger)
O.4	Project contains hardware development	Project contains hardware development	Project contains hardware development
O.5	Project has huge front-end component	Project has only as small front-end component	Project has only as small front-end component
O.6	No system integration tests are required	No system integration tests are required	System integration tests are required

## 15.1.2 Data Collection

To collect meaningful human expertise, appropriate experts were selected. In an interview-like situation, the experts were asked to assess the scenarios and provide recommendations on how to tailor the Product Owner to the particular scenario.

### 15.1.2.1 Human Experts

The author of this thesis selected the following roles to assess the scenarios: Product Owner, Quality Manager, Director Technical Software, Project Lead. The roles have been chosen as they all are responsible to implement the Product Owner role in a hybrid environment, which goes along with the goal of this thesis.

The participating Product Owner has 5+ years of experience in the area of hybrid software development at Baker Hughes and holds a doctoral degree in Software Engineering. The Quality Manager has 3+ years of experience in streamlining the process of hybrid software development at Baker Hughes and more than 10 years of experience in managing software project in other companies. The Director has 3+ years of experience in leading the technical software department at Baker Hughes and more than 20 years of experience in managing software project in other companies. The participating Project Lead has 12+ years of experience in leading technical projects at Baker Hughes and holds a doctoral degree in Mechanical Engineering.

### 15.1.2.2 Execution

All experts assessed the three scenarios individually. Therefore, the author of this thesis met the experts in person in a conference room at a Baker Hughes facility. Each meeting took between 1 and 1.5 hours and followed the same procedure: First, the author explained that this assessment is to validate the results of this thesis, wherefore the personal expertise of the expert would be needed, that there are no right or wrong answers, that all recommendations are anonymized. Second, the experts were handed the written scenarios (they are listed in Table 15.1 and can be found in Appendix F) and asked to assess the project with the overall goal to provide recommendations on how the Product Owner should be tailored to this environment so that the project risks will be minimized.

During the assessment, the author tried to be as unobtrusive as possible to neither impact nor falsify the results. However, to get the experts share their thoughts, some questions needed to be phrased or answered. This is further discussed in section 15.2.6.

## 15.1.3 Data Analysis

To validate the *HyPrO Expert System* and to support / refute the hypothesis of this thesis, the gained data needed to be analyzed. However, the analysis is difficult in this case. While the most common statistical measure for assessing the reliability of an expert system would be the measurement of agreement (e.g. Cohen [19]), this method is not suitable in this case. This is due to the fact that within the measure of agreement, the *HyPrO Expert System* could only be rated as good as human experts – but not better. However, this would be required to support the hypothesis of this thesis and hence, to validate the gained knowledge-base.

To overcome this difficulty, the author of this thesis developed a method to rate the performance of each individual (including the *HyPrO Expert System*) and to compare the performance

of the *HyPrO Expert System* to the group of human experts.

### 15.1.3.1 Preprocessing

To validate the gained knowledge, the qualitative data needed to be preprocessed into a compatible format. The preprocessing contained two steps: first, code and cluster the information gained from the experts into a set of recommendations and second, cluster the results provided by the *HyPrO Expert System* into the same structure the expert results were clustered. The second step altered the structure of the results by the *HyPrO Expert System*. With this, they do not represent the exact structure described in Chapter 14. Therefore, it was necessary to make the result provided by the system compatible to the results provided by the human experts, while the content of the recommendations should not be changed.

To get a better overview of the results and to analyze the data further, the resulting recommendations (preprocessed data) are arranged and represented in separate tables [115] for each scenario (see Tables 15.2, 15.4, 15.6), whereas each table contains the following categories: General Recommendations, Product Owner Structure, Product Owner Characteristics. The category 'General Recommendations' is similar but not equal to the so far addressed Product Owner Tasks.

### 15.1.3.2 Performance Evaluation

The preprocessed data (described in previous section) allowed the author to compare the recommendations (results) provided by all participants (including the *HyPrO Expert System*) to each other and thus, to rate their individual performance.

The rating is based on a score that is assessed for each individual and each category (General Recommendations, Product Owner Structure, Product Owner Characteristics). The scores can range between  $-100$  and  $+100$  points. The method is explained in the following:

For each category, within the resulting tables of recommendations (see Tables 15.2, 15.4, 15.6), a binary field results. Figure 15.1 represents an exemplary field.

$$A(c,e) = \begin{array}{c} \downarrow c \\ \begin{array}{c|ccccc} & \xrightarrow{e} & & & & \\ & \text{Exp. 1} & \text{Exp. 2} & \text{Exp. 3} & \text{Exp. 4} & \text{HyPrO} \\ \text{Code1} & 0 & 0 & 0 & 0 & 1 \\ \text{Code2} & 0 & 1 & 1 & 1 & 1 \\ \text{Code3} & 1 & 0 & 1 & 0 & 1 \end{array} \end{array}$$

**Figure 15.1:** Binary Field (exemplary)



Here, each recommendation gets a weight factor (quantifier) assigned according to how often this recommendation was provided by all other experts (the *HyPrO Expert System* is included in this calculation). The weight factor is calculated by Formula 15.1.

$$W(c, e) = \sum_{i=1}^N A(c, i) - A(c, e) \quad (15.1)$$

where:

$c$  = coded recommendation;

$e$  = expert;

$N$  = Number of experts;

$i$  = identification of element;

$A$  = Answer;

Figure 15.2 depicts the corresponding values based on Formula 15.1:

		$\xrightarrow{\quad e \quad}$					
		Exp. 1	Exp. 2	Exp. 3	Exp. 4	HyPrO	
$W(c,e) =$	$c$	Code1	1	1	1	1	0
		Code2	4	3	3	3	3
	↓	Code3	2	3	2	3	2

**Figure 15.2:** Assigned Weight Factors (exemplary)

With this weight factor, an interim score for each expert is calculated by Formula 15.2.

$$S(c, e) = A(c, e) \cdot W(c, e) \quad (15.2)$$

where:

$c$  = coded recommendation;

$e$  = expert;

$A$  = Answer;

$W$  = Weight factor;

Figure 15.3 depicts the corresponding values based on Formula 15.2:

		$\xrightarrow{\quad e \quad}$					
		Exp. 1	Exp. 2	Exp. 3	Exp. 4	HyPrO	
$S(c,e) =$	$c$	Code1	0	0	0	0	0
		Code2	0	3	3	3	3
	↓	Code3	2	0	2	0	2

**Figure 15.3:** Interim Values (exemplary)

Whenever a participant missed to provide a recommendation that was provided by others, the participant gets as many penalty points assigned as identified by the weight factor. Therefore, a

penalty-rule was established. However, to avoid that the *HyPrO Expert System* outperforms the human experts just by the sheer amount of recommendations, an auxiliary function has been defined as follows: only if more than one expert (including *HyPrO*) provided this recommendation, then everyone who did not provide this recommendation gets as many penalty points as identified by the weight factor. The auxiliary function 15.3 serves the calculation of the penalty points in Formula 15.4 accordingly.

$$G(x) = \begin{cases} 1 \forall x > 1 \\ 0 \forall x \leq 1 \end{cases} \quad (15.3)$$

$$P(c, e) = (W(c, e) - S(c, e)) \cdot G(W(c, e)) \quad (15.4)$$

where:

$c$  = coded recommendation;

$e$  = expert;

$W$  = Weight factor;

$S$  = Score;

$G$  = Auxiliary function;

Figure 15.4 depicts the corresponding values based on Formula 15.4:

		$\xrightarrow{\quad e \quad}$				
		Exp. 1	Exp. 2	Exp. 3	Exp. 4	HyPrO
$P(c,e) =$	$c \downarrow$	Code1	0	0	0	0
		Code2	4	0	0	0
		Code3	0	3	0	3

**Figure 15.4:** Interim Values with Penalty Points (exemplary)

The penalty-rule helps to assess the quality of each recommendation as it values common recommendations more than individual ideas. With the above, the resulting score of each participant can be calculated by Formula 15.5:

$$R(e) = \frac{\sum_{i=1}^M (S(i, e) - P(i, e))}{\sum_{i=1}^M W(i, e)} \cdot 100 \quad (15.5)$$

where:

$e$  = expert;

$i$  = identification of element;

$M$  = Number of coded recommendations;

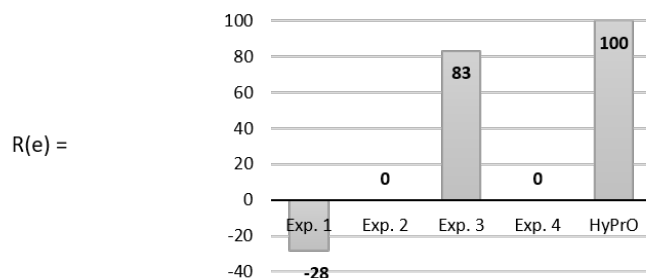
$W$  = Weight factor;

$S$  = Score;

$P$  = Penalty points;

The resulting scores are scaled to a minimum score of  $-100$  and a maximum score of  $+100$ . The resulting scoring points allows to rate the performance of each participant (including the *HyPrO Expert System*).

Figure 15.5 depicts the resulting scores based on Formula 15.5:



**Figure 15.5:** Resulting Scores per Participant (exemplary)

However, to put the individual performances in relation to each other and to compare the performance of the *HyPrO Expert System* against the group of human experts eventually, the resulting score of the *HyPrO Expert System* is compared to the median resulting score of all human experts.

The results are presented in the following section.

## 15.2 Results

This section presents the results obtained in the comparative case study to provide evidence to support the hypothesis for this thesis, validate the *HyPrO Expert System* and thus, the knowledge gained within this thesis.

In order to classify the *HyPrO Expert System* as 'valid', the author of this thesis defined the following:

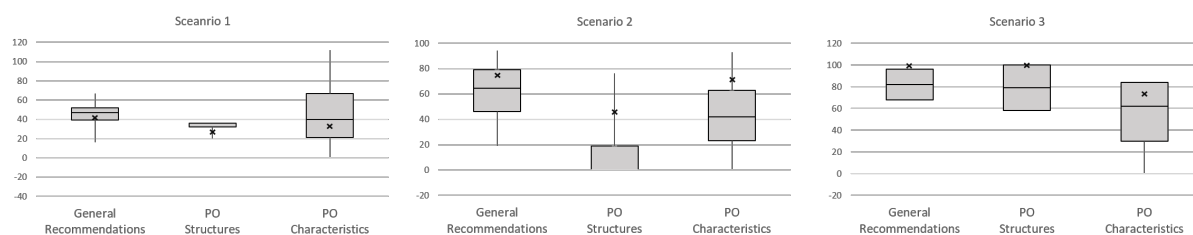
The system is rated as **valid** when the scoring points (performance) of the *HyPrO Expert System* is within or above the range of the scores of the human expert group.

The *HyPrO Expert System* is classified as **overall valid**, when the system is rated as valid in each scenario.

To be rated as **valid for a scenario**, the system needs to be rated as valid in at least 2 out of 3 cases.

## 15.2.1 HyPrO Expert System vs. Human Experts

Overall, the *HyPrO Expert System* surpassed the group of human experts by providing equal or more comprehensive results in 7 out of 9 cases (3 scenarios, each with 3 categories). Figure 15.6 depicts the scores of the human experts (box plots) and the individual score of the *HyPrO Expert System*.



**Figure 15.6:** Scores of all human experts versus the *HyPrO Expert System*

The distribution of the scoring points (performance) among the human experts is graphically represented by the boxes in Figure 15.6.

The boxes mark the minimum range for the scoring points of the *HyPrO Expert System* to be classified as 'valid'. Thus, each score of the *HyPrO Expert System* that is within the box is classified as *equal to human experts* and hence, classified as 'valid'. Consequently, every scoring point of the *HyPrO Expert System* that is above the box, is classified as *better than human experts* and hence, classified as 'valid'. Every scoring point of the *HyPrO Expert System* that is below the box, is classified as *neither equal, nor better than the human expert* and hence, classified as 'not valid'.

As depicted in Figure 15.6, the *HyPrO Expert System* performed within that range – or above – in every case but one. The only case where the *HyPrO Expert System* performed not as good as the human experts is the category of 'Product Owner Structures' in scenario 3.

According to the definition of 'valid' above, the following results:

The *HyPrO Expert System* is rated as **OVERALL VALID**.

## 15.2.2 Scenario 1

Table 15.2 presents the coded and clustered recommendations for each category and marks who provided the recommendation.

**Table 15.2:** Scenario 1: Recommendations provided by the human experts and *HyPrO Expert System*

Recommendation	Expert 1	Expert 2	Expert 3	Expert 4	<i>HyPrO ES</i>
General Recommendations					
intensify stakeholder involvement	x	x	x		x
intensify mgmt involvement	x	x	x		
pair with experts	x	x	x	x	x
emphasize on RE				x	x
emphasize on domain knowledge					x
formalize project communication pattern beyond current framework					x
emphasize on integration test	x	x	x	x	x
create and share vision of new product		x		x	
Product Owner Structure					
Single Product Owner	x	x		x	
Two Product Owners			x		x
Lead Product Owner			x		
Business Product Owner					x
Technical Product Owner					x
Team Up with Expert Roles (e.g. UX, SME, Architect)	x		x	x	x
Product Owner Characteristics					
Communicator	x	x		x	x
Manager			x	x	
Leader				x	x
Visionary		x		x	
Empowered & Committed					x
Available & Qualified	x	x	x	x	x

According to the applied method, the values as shown in Table 15.3 resulted. In this scenario, the human experts performed better than the *HyPrO Expert System* in 2 out of 3 categories. However, as depicted in Figure 15.6 the score of the *HyPrO Expert System* is within the range of distributed scores of the human experts in 2 out of 3 categories. With this, the recommendations provided by the *HyPrO Expert System* are at least equal to the recommendations provided by the human experts. With this result, the *HyPrO Expert System* is rated as valid for this scenario.

**Table 15.3:** Scenario 1: Values to assess performance

	Recommendations	PO Structure	PO Characteristics	Overall
Expert 1	47	36	7	
Expert 2	67	36	31	
Expert 3	47	20	-21	
Expert 4	16	36	91	
<b>Median Score of Human Experts</b>	<b>47</b>	<b>36</b>	<b>19</b>	<b>102</b>
<b>Score of HyPrO ES</b>	<b>41</b>	<b>22</b>	<b>33</b>	<b>96</b>

The greatest distribution of scoring points (performance) among the human experts was identified in the category of **Product Owner Characteristics**. This is not surprising, as such characteristics are hard to define anyways. However, the recommendations of the *HyPrO Expert System* did not deviate much from the overall consensus of the human expert group. What is noticeable, is that two human experts mentioned that the Product Owner should be much of a manager. This characteristics has not been mentioned in any of the considered literature so far and hence, was not in the *HyPrO knowledge-base* by the time of the case study<sup>1</sup>.

Although the distribution of scoring points (performance) among the human experts in the category of the **Product Owner Structure** is quite low, the actual recommendations differ from each other. Here, the human experts split on whether it should be one or two Product Owners. However, they mainly stated that it depends on the number of Scrum Teams. While most (3 out of 4) said it would be best to keep the number of Product Owners as small as possible to keep the collaboration effort low, one expert and the *HyPrO Expert System* recommended to have two Product Owners. However, the expert recommended to organize them in a hierarchy where one serves as the lead and focuses on the organizational overhead while the other focuses on the team. Yet, the *HyPrO Expert System* recommended to organize the two Product Owners in a non-hierarchical way, where one serves as a business Product Owner and the other focuses on technical aspects. However, that the Product Owner should be teamed up with experts roles, like a UX Designer, was recommended 4 out of 5 times (including *HyPrO*).

In the category of **General Recommendations**, different from the human experts, the *HyPrO Expert System* did not recommend to intensify the involvement of the management. This is due to the fact that the scenario stated that the management is committed already. However, 3 out of 4 human experts still highlighted how important it is to involve management constantly. Only one other human expert mentioned that the lack of management availability can be evened-out by the Product Owner itself. Another remarkably difference is that the *HyPrO Expert System* recommended to formalize the communication pattern beyond the currently used framework. This recommendation was given due to the fact that the project is strongly financially controlled. This might require a formalized reporting structure to the Project Management Office or other traditional roles.

---

<sup>1</sup>Due to the fact that the characteristic 'manager' has been named in all scenarios by two human experts, this characteristic was listed in the knowledge-base after the case study.

### 15.2.3 Scenario 2

Table 15.4 presents the coded and clustered recommendations for each category and marks who provided that recommendation.

**Table 15.4:** Scenario 2: Recommendations provided by the human experts and *HyPrO Expert System*

Recommendation	Expert 1	Expert 2	Expert 3	Expert 4	<i>HyPrO ES</i>
General Recommendations					
intensify stakeholder involvement	x	x	x	x	x
intensify mgmt involvement	x	x	x	x	x
emphasize on RE	x	x		x	x
emphasize on domain knowledge		x			x
create and share vision of new product		x		x	
formalize project communication pattern beyond current framework					x
emphasize on integration test	x	x	x	x	x
Product Owner Structure					
Single Product Owner		x		x	
Two Product Owners	x		x		x
Lead Product Owner	x		x		
Business Product Owner	x	x			x
Technical Product Owner	x			x	x
Team Up with Expert Roles (e.g. UX, SME, Architect)	x	x	x	x	x
Product Owner Characteristics					
Communicator	x	x	x	x	x
Manager			x	x	
Leader			x	x	x
Visionary		x		x	x
Empowered & Committed					x
Available & Qualified	x	x	x	x	x

According to the applied method, the values as shown in Table 15.5 resulted. In this scenario, the *HyPrO Expert System* performed better than the human experts in 3 out of 3 categories. As depicted in Figure 15.6 the score of the *HyPrO Expert System* is either within the range of distributed scores of the human experts or above. In any case, it is above the median score of the human expert group.. With this, the recommendations provided by the *HyPrO Expert System* exceed the recommendations provided by the human experts. With this result, the *HyPrO Expert System* is rated as valid for scenario 2.

**Table 15.5:** Scenario 2: Values to assess performance

	Recommendations	PO Structure	PO Characteristics	Overall
Expert 1	55	69	0	
Expert 2	94	-7	31	
Expert 3	19	-7	-53	
Expert 4	74	-7	93	
<b>Median Score of Human Experts</b>	<b>64.5</b>	<b>-7</b>	<b>15.5</b>	<b>73</b>
<b>Score of HyPrO ES</b>	<b>78</b>	<b>43</b>	<b>71</b>	<b>192</b>

In this scenario, the deviation rate among the human experts is higher than in scenario 1. So did the human experts disagreed most on the PO Structure and Characteristics.

Due to a similar setup to scenario 1, the recommendations for the **Product Owner Structure** did not differ much from each other. The only difference is, that expert 1 switched to a hierarchical setup of the Product Owners. However, the described version of the structure is still different from the recommendations provided by expert 3. Expert 1 recommends to define a Lead Product Owner on top of the combination of a business and technical oriented Product Owner team. Also, in this scenario all participants recommended to the Product Owner be teamed up with experts roles, like a UX Designer.

Similar as in scenario 1, the Product Owner was **characterized** as a manager but also as a leader. Although, the characteristic of being *empowered & committed* was not mentioned by a human expert, the management-characteristics could be interpreted as such. However, to not falsify the results, this interpretation is not reflected in Table 15.4.

In the category of **General Recommendations**, the *HyPrO Expert System* also recommended to intensify the involvement of the management. However, formalizing the communication structure was still not mentioned by the human experts.

Overall, the scores of the *HyPrO Expert System* identify that the system provides more stable recommendations..



### 15.2.4 Scenario 3

Table 15.6 presents the coded and clustered recommendations for each category and marks who provided that recommendation.

**Table 15.6:** Scenario 3: Recommendations provided by the human experts and *HyPrO Expert System*

Recommendation	Expert 1	Expert 2	Expert 3	Expert 4	<i>HyPrO ES</i>
General Recommendations					
intensify stakeholder involvement	x	x	x	x	x
intensify mgmt involvement	x	x	x		x
pair with experts	x		x	x	x
regular travel activities for all involved parties	x	x	x	x	x
emphasize on RE					x
ensure shared vision and understanding of requirements across all levels	x	x	x	x	x
formal communication	x	x	x	x	x
emphasize on Legacy System	x	x	x	x	x
Product Owner Structure					
Multiple Product Owners	x	x	x	x	x
Lead / Chief Product Owner	x	x	x	x	x
Lead / Chief Product Owner to communicate with mgmt	x	x	x		x
PO at each location	x	x	x	x	x
Team Up with Expert Roles (e.g. UX, SME, Architect)	x		x	x	x
Product Owner Characteristics					
Communicator	x	x	x	x	x
Manager			x	x	
Leader		x	x	x	x
Visionary				x	x
Empowered & Committed			x		x
Available & Qualified	x	x	x	x	x

According to the applied method, the values as shown in Table 15.7 resulted. In this scenario, the *HyPrO Expert System* performed better than the human experts in 3 out of 3 categories. As depicted in Figure 15.6 the score of the *HyPrO Expert System* is either within the range of distributed scores of the human experts or above. In any case, it is above the median score of the human expert group. With this, the recommendations provided by the *HyPrO Expert System* exceed the recommendations provided by the human experts. With this result, the *HyPrO Expert System* is rated as valid for scenario 3.

**Table 15.7:** Scenario 3: Values to assess performance

	Recommendations	PO Structure	PO Characteristics	Overall
Expert 1	96	100	-11	
Expert 2	68	58	29	
Expert 3	96	100	73	
Expert 4	68	58	73	
<b>Median Score of Human Experts</b>	<b>82</b>	<b>79</b>	<b>51</b>	<b>212</b>
<b>Score of HyPrO ES</b>	<b>100</b>	<b>100</b>	<b>73</b>	<b>273</b>

As in Scenario 1 and 2, the greatest distribution of scoring points (performance) among the human experts was identified in the category of **PO Characteristics**. While the Product Owner was again described to be a *manager*, in this scenario more experts described the Product Owner as a *leader*. Also, one expert explicitly mentioned the importance of autonomy and empowerment of the Product Owner team in this setup.

In the category of **General Recommendations**, all experts (including the *HyPrO Expert System*) made the same recommendations. Only the aspect of involving the management was not considered as important by one expert. However, this expert highlighted that he would expect the Lead Product Owner to act as a manager.

In this scenario, all experts recommended to **structure** the Product Owners in a hierarchical Product Owner team.

### 15.2.5 Test of Hypothesis

According to Seaman [93], "a hypothesis cannot be proven, it can only be supported or refuted" [93, p.13]. The results presented above build up evidence to support the hypothesis for this thesis. The hypothesis states that *a knowledge-base expert system, which provides equal or more comprehensive recommendations than human experts, would support the tailoring of the Product Owner role to hybrid development environments* (see Section 1.2). The results of the comparative case study identified that the recommendations of the *HyPrO Expert System* surpassed the recommendations of the human experts in 7 out of 9 cases in terms of the rating system provided by the method described in Section 15.1.3. In the other two cases, the recommendations of the *HyPrO Expert System* were rated as equal to the recommendations of the human experts. Consequently, the hypothesis is supported.

### 15.2.6 Threats to Validity

According to Runeson and Hoest [85] four aspects of validity should be considered when reporting a case study in software engineering: construct validity, internal validity, external validity and reliability.

**Construct Validity** The construct validity determines how well the used variables fit to the intended purpose of measurement [85].

In the validation approach of the *HyPrO Expert System*, the variables are the following: the defined scenarios which were used to measure the performance of the participants as well as the coding and clustering of the resulting qualitative data to bring them into a compatible format. To mitigate the risks that comes with the variable of the scenarios, the author constructed three scenarios to have multiple sources of evidence.

**Internal Validity** The internal validity is concerned with causal relationships of the considered factors [85].

In this case, the considered factors that threatens the internal validity is mainly the selection of human experts. Although they all work in the same company, they have different backgrounds and vary in their years of experiences.

**External Validity** The external validity addresses to what extend the results can be generalized and to what extend the results might be of interest to others[85].

In this case study, the *HyPrO Expert System* and hence, its generated knowledge-base, was validated. With this, the comprised knowledge within that knowledge-base can be rated as general valid for the intended purpose of supporting the tailoring of the Product Owner role in hybrid environments. Consequently, the results might be of interest for other researchers.

**Reliability** The reliability addresses how the results of this study depend on the specific researcher [85].

In the validation approach of the *HyPrO Expert System*, the author of this thesis met with the human experts in person and let them assess the scenarios in an interview-like situation. This needs to be considered as a risk to the reliability. However, to mitigate this risk, the author did not share any details of test design with the human experts. Hence, they did not know whether if or what the author expected them to say. Furthermore, the author did not provide any additional information to the scenarios, while they were assessed.



**Part V**

**Conclusion**



---

<b>16 Contributions of Thesis</b>	<b>143</b>
16.1 Theoretical Work . . . . .	<b>143</b>
16.2 Empirical Work . . . . .	<b>144</b>
16.3 Constructive Work . . . . .	<b>144</b>
<b>17 Conclusion</b>	<b>147</b>
<b>18 Limitations and Future Work</b>	<b>149</b>

---





# 16

## Contributions of Thesis

The main contribution of this thesis is the gained knowledge in the area of Product Owners in hybrid development environments. The gained knowledge was made accessible through the design, development and validation of the *HyPrO Expert System*. With this, the author of this thesis addressed the unsolved problem of compressing expert knowledge to support the tailoring of the Product Owners to hybrid development environments. However, the main contribution is a sum of intermediate steps of theoretical and empirical work which results are reflected in the *HyPrO Expert System*.

### 16.1 Theoretical Work

The conducted theoretical work structured the research area of Product Owners in Industry by summarizing existing knowledge as well as identifying gaps. Eventually, further gained empirical results were combined with existing knowledge from the area of *Software Process Tailoring*. The main contributions for this category of work are as follows:

- The construction of a **reference model of tasks and characteristics of Product Owners** that indicates overlapping attributes from both aspects. The results were published in [103].
- A **mapping study on Product Owners in industries** that summarized knowledge of this topic and identified gaps which further defined the research direction of this thesis. The results were published in [104, 105].
- The generation of the **Catalog of tailoring criteria and implications on the Product Owner in hybrid environments**. Therefore, existing knowledge of *Software Process Tailoring* was reviewed, filtered and merged with basic knowledge as well as newly gained results on the topic of the Product Owner in hybrid environments.
- Extension of the **Knowledge-base of Product Owners in hybrid development environments** by identifying a new Product Owner characteristic (Manager), 13 relevant

Product Owner tasks and 3 distinct Product Owner Team structures in hybrid development environments.

## 16.2 Empirical Work

The conducted empirical work identified the topic of Product Owners in hybrid environments as problematic and hence, defined the motivation of this thesis. Further conducted case studies provided new knowledge on the topic of Product Owners in hybrid environments and extended this knowledge area.

- **A case study on Product Owner tasks and characteristics** in a hybrid environment at Baker Hughes. The preliminary results indicate that previously identified tasks and characteristics of this role are not directly applicable to a hybrid development environment. The results were published in [103].
- **A survey study on Product Owner tasks and characteristics** in a hybrid environment to assess the expectations of the Scrum Team on this role. The results revealed that in a hybrid environment the Product Owner is not perceived as a leader of the team, but should work closely with the management as well as other specialized roles. The results were published in [106].
- **A case study on communication activities of Product Owners in a hybrid environment.** Here, qualitative as well as quantitative data were collected to further define the most frequently named task of Product Owners: communication. The results were published in [105].
- **A case study on requirements flow and collaboration in a hybrid environment.** This study revealed step by step how the requirements are handled from the idea to the development and how the involved departments are handling it. The results have been published in [82].
- The **validation study of the developed *HyPrO Expert System***. The study compared the recommendations to tailor the Product Owner role to hybrid environments provided by human experts to the recommendations provided by the *HyPrO Expert System*. The results rated the *HyPrO Expert System* as valid.

## 16.3 Constructive Work

This section describes the practical aspects of the design and development of the *HyPrO Expert System*. It represents the research artifact resulting from the design science approach by Hevner et al. [36] which, in the case of this thesis, contains a model, its rules and an instance.

- The generation of the (HyPrO) **model**. This step required three steps. First, the consolidation of basic knowledge and newly gained information on the Product Owner tasks, characteristics and structure. Second, the identification of respective influencing factors on the Product Owner role in a hybrid environment and, third, the consolidation of both in a catalog that addresses the factors and presents high-level recommendations on the

Product Owner role. With this, two independent knowledge-bases have been generated as well as a consolidated catalog.

- The definition of **rules** on how the individual facts from the knowledge-base are related to each other. By use of logical expressions, the *HyPrO Expert System* can deduce recommendations for the *Product Owner tasks, characteristics & structures* from **a set of influencing factors on Product Owners** rather than just from one isolated factor. From a set of 14 influencing factors an overall number of 84 different recommendations can be generated.
- The **instantiation** of the *HyPrO Expert System* as a proof of concept web application. The intention of this application is to visualize the model and make the results easily accessible for users. Based on the input of the user, the web application calculates the results and presents recommendations on how to tailor the product owner to that particular hybrid environment.



# 17

## Conclusion

The main goal of this thesis was to support the tailoring of the Product Owner role by analyzing Product Owner tasks, characteristics and structures to deduce recommendations to tailor the Product Owner role to hybrid development environments.

Therefore, the Product Owner tasks, characteristics and structures have been analyzed in a number of case studies and results of previous research has been concluded in a mapping study. This resulted in the identification of 13 frequently conducted tasks, 6 favorable characteristics and 12 structures of Product Owners. The deduction of recommendations is based on the combination of the identified Product Owner tasks, characteristics & structures as well as existing knowledge of software process tailoring and its action items. This combination was chosen as software process tailoring is directly related to customized hybrid environments. Therefore, the existing knowledge has been reviewed and was narrowed down to a set of 14 influencing factors on the Product Owner and 14 respective action items.

This results in an overall number of 84 distinct recommendations. These recommendations are based on the individual project situation, which can be assessed with the help of a rating system of the influencing factors.

The rating system and the output of the respective recommendation are instantiated in the proof of concept web application named *HyPrO Expert System*. With this, the results of this thesis are made easily accessible for everyone.

In Section 1.2 the author hypothesized that an expert system would support the tailoring of the Product Owner role to hybrid development environments equally to or even better than human experts at this point of time. A conducted comparative case study confirmed the hypothesis and rated the *HyPrO Expert System* as overall valid. According to the thesis goals, the *HyPrO Expert System* supports the tailoring of the Product Owner to hybrid development environments. This makes the results of this thesis and hence, the *HyPrO Expert System* applicable in different domains, such as the oil & gas industry, the automotive domain, regulated domains, small and big companies and pure software as well as system development environments.



# 18

## Limitations and Future Work

To conclude this thesis, this chapter addresses its limitations and potential future work.

**Application of Tailored Product Owner Roles in Industry** The results of this thesis support the tailoring of the Product Owner role to hybrid environments. However, the recommendations are only based on the underlying knowledge-base. Although the recommendations have been validated, they have not been applied in real settings. Hence, it remains unknown to what extent the industry benefits from the actual application of a tailored Product Owner role according to the *HyPrO Expert System*.

*Resulting implication for future work:* Various versions of tailored Product Owner roles should be applied in industry to identify and measure actual benefits.

**Continuous Extension of Knowledge-Base** The generated knowledge-bases within this thesis are based on related results from various researchers within the past two centuries. However, the awareness of importance of hybrid development environments has improved significantly only since the results of the HELENA study [58] have been published in 2018. Hence, it is to be expected that more research in this field is to come. Here it is important that newly gained information flow into the knowledge-base.

*Resulting implication for future work:* To provide an easy flow of information a respective infrastructure would be needed. Hence, this aspect would provide two potentials for future work: one is the continuous research in this area and selection of relevant results to extend the knowledge-base of Product Owners in hybrid environments. The other would be to establish an easy to use infrastructure to extend such a knowledge-base accordingly.

**Enhanced Development of the *HyPrO Expert System*** The *HyPrO Expert System* has been developed as a proof of concept web application. This type of development was suitable for this thesis but left room for improvement. If it should be used in a broader environment, the expert system should be enhanced with a data bank which stores the knowledge and a separate inference engine. Or even further, a state of the art artificial intelligence system should be developed.

*Resulting implication for future work:* A high quality artificial intelligence should provide a learning algorithm, which learns from "success stories" where the Product Owner peculiarities are described within the context of hybrid environments whereupon the system would constantly update the recommendations.



**Part VI**  
**Appendix**



---

<b>A</b>	<b>Systematic Mapping Study Reference List</b>	<b>155</b>
A.1	Startset Product Owner . . . . .	155
A.2	Snowball Backward Product Owner . . . . .	156
A.3	Snowball Forward Product Owner . . . . .	156
A.4	Startset On-site Customer . . . . .	156
A.5	Snowball Backward On-site Customer . . . . .	157
<b>B</b>	<b>Requirements Engineering Process - Interview Questions</b>	<b>159</b>
<b>C</b>	<b>Catalog</b>	<b>163</b>
C.1	Category: Team . . . . .	163
C.1.1	Factor: Team Size [T.1] . . . . .	163
C.1.2	Factor: Team Distribution [T.2] . . . . .	164
C.1.3	High-level Recommendations for Category: Team . . . . .	164
C.2	Category: Internal Environment . . . . .	165
C.2.1	Factor: Clear Project Proposal [IE.1] . . . . .	165
C.2.2	Factor: Management Commitment [IE.2] . . . . .	165
C.2.3	Factor: Financial Control [IE.3] . . . . .	166
C.2.4	High-level Recommendations for Category: Internal Environment . . . . .	167
C.3	Category: External Environment . . . . .	167
C.3.1	Factor: Number of Stakeholders [EE.1] . . . . .	167
C.3.2	Factor: Stakeholder / User Availability[EE.2] . . . . .	168
C.3.3	Factor: Requirements Stability [EE.3] . . . . .	168
C.3.4	High-level Recommendations for Category: External Environment . . . . .	169
C.4	Category: Objective . . . . .	169
C.4.1	Factor: Complexity [O.1] . . . . .	169
C.4.2	Factor: Legacy System [O.2] . . . . .	170
C.4.3	Factor: Criticality [O.3] . . . . .	170
C.4.4	Factor: Hardware Development [O.4] . . . . .	171
C.4.5	Factor: User Interface [O.5] . . . . .	172
C.4.6	Factor: System Integration Test [O.6] . . . . .	172
C.4.7	High-level Recommendations for Category: Objective . . . . .	173
<b>D</b>	<b>Truth Tables</b>	<b>175</b>
D.1	Truth Table for Category: Team . . . . .	176
D.2	Truth Table for Category: Internal Environment . . . . .	177
D.3	Truth Table for Category: External Environment . . . . .	178
D.4	Truth Table for Category: Objective . . . . .	179
D.5	Combined Categories Truth Table . . . . .	184
<b>E</b>	<b>Values &amp; Scales</b>	<b>187</b>

<b>F Test Case Scenarios</b>	<b>189</b>
<b>G Publications</b>	<b>193</b>
<b>List of Figures</b>	<b>195</b>
<b>List of Tables</b>	<b>197</b>
<b>Glossary</b>	<b>199</b>
<b>Bibliography</b>	<b>201</b>
<b>Curriculum Vitae</b>	<b>211</b>

---



# Systematic Mapping Study Reference List

This appendix lists the considered publications for the systematic mapping study (described in Section 6) ordered by the steps of the additional snowballing search.

## A.1 Startset Product Owner

1. J. M. Bass. How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, 20(6):1525–1557, 2015.
2. J. M. Bass, S. Beecham, M. A. Razzak, C. N. Canna, and J. Noll. An empirical study of the product owner role in scrum. In I. Crnkovic, M. Chaudron, M. Chechik, and M. Harman, editors, *Proceedings of the 40th International Conference on Software Engineering Companion Proceedings - ICSE '18*, pages 123–124, New York, New York, USA. ACM Press, 2018.isbn: 9781450356633.
3. R. K. Gupta, S. Jain, and B. Singh. Challenges in scaling up a globally distributed legacy product. In M. Paasivaara, D. Smite, and R. Evaristo, editors, *Proceedings of the 13th Conference on Global Software Engineering - ICGSE '18*, pages 77–81, New York, New York, USA. ACM Press, 2018. isbn: 9781450357173.
4. V. T. Heikkilä, M. Paasivaara, C. Lassenius, D. Damian, and C. Engblom. Managing the requirements flow from strategy to release in large-scale agile development: a case study at ericsson. *Empirical Software Engineering*, 22(6):2892–2936, 2017.issn: 1573-7616.doi:10.1007/s10664-016-9491-z.
5. S. Kristinsdottir, M. Larusdottir, and A. Cajander. 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, and F. Kis, editors, *Human-Centered and Error-Resilient Systems Development*, pages 3–16, Cham. Springer International Publishing, 2016.isbn:978-3-319-44902-9.
6. M. Paasivaara, V. T. Heikkilä, and C. Lassenius. Experiences in scaling the product owner role in large-scale globally distributed scrum. In *2012 IEEE Seventh International Conference on Global Software Engineering*, pages 174–178. IEEE, 8/27/2012 - 8/30/2012.isbn: 978-1-4673-2357-4.

7. D. Raithatha. Making the whole product agile – a product owners perspective. In *International Conference on Extreme Programming and Agile Processes in Software Engineering*, pages 184–187. Springer, 2007.
8. Y. Shastri, R. Hoda, and R. Amor. Does the "project manager" still exist in agile software development projects?:57–64, 2016.
9. H. S. Sverrisdottir, H. T. Ingason, and H. I. Jonasson. The role of the product owner in scrum – comparison between theory and practices. *Procedia-Social and Behavioral Sciences*, 119:257–267, 2014.
10. C. Unger-Windeler and J. Klünder. On the tasks and characteristics of product owners: a case study in the oil and gas industry. In *International Conference on Product-Focused Software Process Improvement*, pages 3–11. Springer, 2018.

## A.2 Snowball Backward Product Owner

1. A. d. S. Croix and A. Easton. The product owner team. In *Agile 2008 Conference*, pages 274–279. IEEE, 8/4/2008 - 8/8/2008. isbn: 978-0-7695-3321-6.
2. R. Hoda, J. Noble, and S. Marshall. The impact of inadequate customer collaboration on self-organizing agile teams. *Information and Software Technology*, 53(5):521–534, 2011. issn: 09505849. doi:10.1016/j.infsof.2010.10.009.
3. K. H. Judy and I. Krumins-Beens. Great scrums need great product owners: unbounded collaboration and collective product ownership. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, page 462. IEEE, 1/7/2008 - 1/10/2008.
4. M. Lowery and M. Evans. Scaling product ownership. In *AGILE 2007 (AGILE 2007)*, pages 328–333. IEEE, 8/13/2007 - 8/17/2007. isbn: 0-7695-2872-4.

## A.3 Snowball Forward Product Owner

1. J. M. Bass, S. Beecham, J. Noll, and Razzak. All hands to the pumps: the product owner role in small companies. lero technical report: 20171.
2. G. Matturro, F. Cordovées, and M. Solari. Role of product owner from the practitioner's perspective. an exploratory study. In *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*, pages 113–118. The Steering Committee of The World Congress in Computer Science, 2018.
3. S. Oomen, B. De Waal, A. Albertin, and P. Ravesteyn. How can scrum be succesful? competences of the scrum product owner. In *Proceedings of the 25th European Conference on Information Systems (ECIS)*, pages 57–64. AISeL, June 5-10 2017.

## A.4 Startset On-site Customer

1. J. Koskela and P. Abrahamsson. On-site customer in an xp project: empirical results from a case study. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and T. Dingsøyr, editors, *Software Process Improvement*, volume 3281 of *Lecture Notes in Computer Science*, pages 1–11, Berlin, Heidelberg. Springer Berlin Heidelberg, 2004. isbn:978-3-540-23725-9.

2. A. Martin, R. Biddle, and J. Noble. The xp customer team: a grounded theory. In 2009 Agile Conference, pages 57–64. IEEE, 8/24/2009 - 8/28/2009.isbn: 978-0-7695-3768-9.
3. S. Mohammadi, B. Nikkahan, and S. Sohrabi. An analytical survey of on-site customer practice in extreme programming. In International Symposium on Computer Science and its Applications, pages 1–6. IEEE, 10/13/2008 - 10/15/2008.isbn: 978-0-7695-3428-2.
4. X. Wang, Z. Wu, and M. Zhao. The relationship between developers and customers in agile methodology. In 2008 International Conference on Computer Science and Information Technology, pages 566–572. IEEE, 2008.
5. A. Wojciechowski, M. Wesolowski, and W. Complak. Experimental evaluation of 'on-site customer' xp practice on quality of software and team effectiveness. In R. Meersman, T. Dillon, and P. Herrero, editors, On the Move to Meaningful Internet Systems: OTM 2010 Workshops, volume 6428 of Lecture Notes in Computer Science, pages 269–278, Berlin, Heidelberg. Springer Berlin Heidelberg, 2010.isbn: 978-3-642-16960-1.

## A.5 Snowball Backward On-site Customer

1. C Farrell, R Narang, S Kapitan, and H Webber. Towards an effective onsite customer practice. In Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2002). Springer, 2002.
2. M. Finsterwalder. Does xp need a professional customer? In XP2001 Workshop on customer involvement. ACM, 2001.
3. D. Hussman. Coaching a customer team. In G. Goos, J. Hartmanis, J. van Leeuwen, M. Marchesi, and G. Succi, editors, Extreme Programming and Agile Processes in Software Engineering, volume 2675 of Lecture Notes in Computer Science, pages 254–260, Berlin, Heidelberg. Springer Berlin Heidelberg, 2003.isbn: 978-3-540-40215-2.
4. A. Martin, R. Biddle, and J. Noble. The xp customer role in practice: three studies. In Agile Development Conference, pages 42–54. IEEE, 22-26 June 2004.isbn: 0-7695-2248-3.
5. A. Martin, J. Noble, and R. Biddle. Being jane malkovich: a look into the world of an xp customer. In International Conference on Extreme Programming and Agile Processes in Software Engineering, pages 234–243. Springer, 2003.
6. A. M. Martin. The role of customers in extreme programming projects. PhD thesis, Victoria University of Wellington, 2009.
7. N. Wallace, P. Bailey, and N. Ashworth. Managing xp with multiple or remote customers. In Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2002). Springer, 2002.
8. M. Williams, J. Packlick, R. Bellubbi, and S. Coburn. How we made onsite customer work – an extreme success story. In AGILE 2007 (AGILE 2007), pages 334–338. IEEE, 8/13/2007 - 8/17/2007.isbn: 0-7695-2872-4.





# B

## Requirements Engineering Process - Interview Questions

This appendix presents the interview questions described in Section 9.2.4.

# Requirements Engineering Process - Interview Questions

## 1. Attendees

Name:

Date:

## 2. Introduction

- What are your roles at Baker Hughes? Especially in the requirements engineering process?
- What are your main tasks?
- Who is using your work results?
- What informations do you need for your work?

## 3. Elicitation

- How is a new requirement started?
- Do you have a standardized process? Do you have standards which you have to follow? (project level, software level)?
- When you talk first about the requirements? In which stage of the Stage-Gate-Process?
- Is there anyone particular responsible for a requirement? Who? Does that change over the time during the project?
- Who will forward the requirements? From top to bottom?
- Difference between stage and gate?
- Do you freeze the requirements in stage 2? No changes after that?
- What meetings happened here for the requirements?
- How is the initial backlog defined?
- Is there a stakeholder list?
- What guidelines does the company have regarding requirements? Hardware section: DDR(Detailed Design Review), TDR (Technical Design Review)? What are you know about equivalences on the software side?
- What tools are used to support the requirement process? Pro and con with these tools?
- Functional/Non-Functional Requirements are these collected in the process?
- Who is doing it?
- Who should do it?

## 4. Spezification/Documentation

- When and how are requirements documented during the project? When should it be done in your opinion?
- Do you have a standard document structure?
- How do you use the documents in the project?
- Do you think this is the right way to do it?
- What happens in the feature flyer? What is documented here? Is anyone reviewing it? Who should review it?
- Do you use the TFS? Do you often look into it and who has access?

- How much time do you spend on documenting requirements?
- Who is responsible for the requirements documentation?
- Is the document easy to change?

#### 5. Validation/Refinement

- What kind of validation is done for requirements in the project?
- What reviews do you have for the validation?
- When are the reviews? Are the reviews documented anyhow?
- Who is part of it?
- What meetings happened here for the requirements?
- Do you check the features? When and how are you doing this during the project?
- There is a feature quality checklist. Do you use it for checking the features?
- When do you see the feature again?
- Do you check if the feature meets your standards?
- What happened when the feature is not what you expected?
- How are change handled during the project?

#### 6. Communication

- Who you think you communicate the most with about features/requirements?
- Are there meetings about requirements?
- With who do you work together?
- Which meetings are important for that?
- How many percent of information are written or oral?
- Other actors, who are involved in the process?

#### 7. Problems&Suggestions

- Where do you see problems?
- Where do you see improvement?
- Suggestions?





# Catalog

This appendix presents the resulting catalog of the consolidated knowledge of the two rigors of (1) *Product Owner Tasks, Characteristics & Structures* and the (2) *Software Process Tailoring Criteria* as described in Section 13.3.

## C.1 Category: Team

### C.1.1 Factor: Team Size [T.1]

#### C.1.1.1 Rationales

The team size is a key criteria for tailoring the Product Owner role as well as selecting the right software process [18]. This is because it is an indicator for both, team coordination [117] as well as numbers of required Product Owners [12, 26, 64, 69, 72]. As soon as the team needs to be split in multiple scrum teams, multiple Product Owners should be available and coordinated, which in return, requires a more formalized communication [105].

#### C.1.1.2 Implications on Product Owner role

At least one Product Owner or Subject Matter Expert should be available per scrum team. Depending on how many scrum teams are formed, the number of Product Owners / Subject Matter Experts needs to be adjusted [104]. To coordinate all Product Owners / Subject Matter Experts, formalized communication needs to be established [105].

#### C.1.1.3 Mapping of Product Owner Peculiarities & Action Items

Table C.1: Mapped PO peculiarities to T.1

Tasks	Characteristics	Structure
Communication	C&N A&Q L&T	Team

**Table C.2:** Mapped Action Items to T.1

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
			ADF <sup>a</sup>
			ACP

<sup>a</sup>The description to the abbreviations can be found in Table 13.8.

## C.1.2 Factor: Team Distribution [T.2]

### C.1.2.1 Rationales

Distributed teams require a Product Owner (representative) to be on-site with the development team to ensure proper communication of the responsibilities and priorities [39, 53, 64, 69, 72, 92]. As this adds up the number of Product Owners, they again, need to be coordinated which requires a more formalized communication [105] as well as travelling of the Product Owners to the various sites [12, 26, 78].

### C.1.2.2 Implications on Product Owner role

A local representative of the Product Owner / Subject Matter Expert should be available on-site to communicate priorities and responsibilities [105]. To coordinate all Product Owners / Subject Matter Experts, formalized communication needs to be established and requires travelling to bring local representatives together [105]. If there is no local representative for each location, other representatives need to travel to the various sites on a regular basis [105].

### C.1.2.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.3:** Mapped PO peculiarities to T.2

Tasks	Characteristics	Structure
Communication	C&N	Team
Traveller	L&T	
	A&Q	

**Table C.4:** Mapped Action Items to T.2

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
			ADF
			ACP

## C.1.3 High-level Recommendations for Category: Team

Increase number of Product Owners / Subject Matter Experts and establish a working structure between them. Designated Product Owner or Subject Matter Expert representatives per team / location ensures collaboration and leadership.

## C.2 Category: Internal Environment

### C.2.1 Factor: Clear Project Proposal [IE.1]

#### C.2.1.1 Rationales

A clear project proposal contains goals and requirements [109, 110, 121] that are essential for the project's success. The less clear the goals and the requirements are, the Product Owner has to emphasize more on requirements engineering practices, get the sign-off from management [106] and/ or has to create his own vision. All this needs to be properly documented in the backlog [12, 26, 53] and tested by the Product Owner.

#### C.2.1.2 Implications on Product Owner role

The Product Owner needs to envision the end-product, communicates that vision and makes sure that it is in alignment with the managements vision of the product [103, 104]. The Product Owner has to groom the backlog [104]. This sometimes requires to translate the requirements from traditional requirements documents into agile compatible formats [82]. The Product Owner needs to test the outcome against these requirements and accepts or rejects them [104].

#### C.2.1.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.5:** Mapped PO peculiarities to IE.1

Tasks	Characteristics	Structure
Groom	V&D	Team
Acceptance Tester	L&T	Expert
Visionary	E&C	

**Table C.6:** Mapped Action Items to IE.1

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
AMI	ARE		

### C.2.2 Factor: Management Commitment [IE.2]

#### C.2.2.1 Rationales

Management commitment is required to ensure a shared goal and support when it comes to solve problems quickly and to make project progress decisions [109, 110]. The commitment of the management has an influence on the Product Owners tasks, as he is perceived as the link between management and development teams [106].

#### C.2.2.2 Implications on Product Owner role

Product Owner should work closely with management (this also involves close communication with the customer) to ensure a shared goal [106].

### C.2.2.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.7:** Mapped PO peculiarities to IE.2

Tasks	Characteristics	Structure
Customer Rel. Manager	L&T	Team Expert

**Table C.8:** Mapped Action Items to IE.2

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	AFI	ACP	

## C.2.3 Factor: Financial Control [IE.3]

### C.2.3.1 Rationales

Financial control has an impact on the Product Owner with respect to the project budget. A huge budget usually implies a more formalized process with more traditional documentation and more frequent reporting to the project management office. Hence, the Product Owner has to establish a formalized communication the Project Management Office [106]. It also requires more emphasize on planning [43].

### C.2.3.2 Implications on Product Owner role

The Product Owner has to consider development costs and establish a formalized communication with the Project Management Office [105].

### C.2.3.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.9:** Mapped PO peculiarities to IE.3

Tasks	Characteristics	Structure
	C&N	

**Table C.10:** Mapped Action Items to IE.3

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	AFI	ACP	



## C.2.4 High-level Recommendations for Category: Internal Environment

Product Owners should work closely with (traditional) management to ensure a shared goal and management support when it comes to critical decision-making.

## C.3 Category: External Environment

### C.3.1 Factor: Number of Stakeholders [EE.1]

#### C.3.1.1 Rationales

The higher the number of involved stakeholders the more the Product Owner has to act as an intermediary person to accommodate all interests [6, 8]. This increases the size of the network of the Product Owner and consequently increases the number of formal interactions [105]. Furthermore, more time is required to negotiate all needs and requirements [43, 79]. To handle the negotiation and communication the Product Owner needs to be qualified to oversee all requirements [79], as well as make distinctions and priorities, evaluate the risks, manage expectations [6, 8, 29, 100]. Also a high number of stakeholders might require a Product Owner team to handle multiple areas simultaneously.

#### C.3.1.2 Implications on Product Owner role

With a high number of involved stakeholders, the Product Owner has to act as an intermediary person and need to negotiate to accommodate all interests. This increases the size of the network of the Product Owner and consequently increases the number of formal interactions [105]. To lead the negotiation, the Product Owner needs to be qualified to oversee all requirements, prioritize them, evaluate the risks and manage expectations.

#### C.3.1.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.11:** Mapped PO peculiarities to EE.1

Tasks	Characteristics	Structure
Communication	C&N	Single PO
Prioritizer	A&Q	
Intermediary		
Expectation Manager		
Critical Decision Maker		

**Table C.12:** Mapped Action Items to EE.1

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
ACI		ADF	AIW
AUI		ACP	

### C.3.2 Factor: Stakeholder / User Availability[EE.2]

#### C.3.2.1 Rationales

Similar to the availability of the top management, the availability of stakeholders is a success factor [43]. As the Product Owners in such an environment rather act as a proxy [82, 105, 106], they rely on the stakeholders feedback and sign-off. This requires regular (and probably even more formalized) communication [105].

#### C.3.2.2 Implications on Product Owner role

The Product Owner relies on stakeholders feedback in terms of sign-off to ensure a shared goal. This requires communication on a regular basis.

#### C.3.2.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.13:** Mapped PO peculiarities to EE.2

Tasks	Characteristics	Structure
Communication Intermediary	C&N	

**Table C.14:** Mapped Action Items to EE.2

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	AFL		

### C.3.3 Factor: Requirements Stability [EE.3]

#### C.3.3.1 Rationales

The stability of requirements influence the strategy for requirements elicitation, architecting the solution, its implementation and tests [43]. Hence, they are critical for the projects success. The Product Owner needs to be able to assess the stability of the requirements to estimate potential changes. This requires a solid domain knowledge or to be teamed up with a domain expert. He also needs to emphasize on requirements elicitation and validation in case of volatile requirements.

#### C.3.3.2 Implications on Product Owner role

To ensure stable requirements, the Product Owner need to emphasize on requirements engineering practices, has to gain more domain knowledge (to be able to assess the quality / stability of the requirements), or should team up with a domain expert.

### C.3.3.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.15:** Mapped PO peculiarities to EE.3

Tasks	Characteristics	Structure
Communicator	C&N	
Gate Keeper	A&Q	
Acceptance Tester	E&C	
Critical Decision Maker		

**Table C.16:** Mapped Action Items to EE.3

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	ARE		
	AFL		

### C.3.4 High-level Recommendations for Category: External Environment

Establish close feedback loops with stakeholders / users to validate requirements and avoid delays. The Product Owner(s) should be properly trained on Requirements Engineering activities.

## C.4 Category: Objective

### C.4.1 Factor: Complexity [O.1]

#### C.4.1.1 Rationales

The more complex the project is, the more the Product Owner has to emphasize on formalized communication, creating prototypes [43], requirements engineering, architecture or front-end issues [105]. As this might be too much for a single individual, the Product Owner role should be scaled to a Product Owner team so that responsibilities can be shared [105]. Also experts for the individual areas of expertise should be consulted [106].

#### C.4.1.2 Implications on Product Owner role

To manage the increased effort of a complex project, the Product Owner role might be scaled up. This requires a Product Owner Team and / or the collaboration with other experts.

### C.4.1.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.17:** Mapped PO peculiarities to O.1

Tasks	Characteristics	Structure
Communication	C&N	Team
Intermediary	T&L	Expert
Super Secretary	A&Q	

**Table C.18:** Mapped Action Items to O.1

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	ARE	ADF	
	AAR	ACP	
	AFL		

## C.4.2 Factor: Legacy System [O.2]

### C.4.2.1 Rationales

A legacy system usually impacts the requirements engineering process [43]. The less documentation is available, the more communication is required to analyze the legacy system. Therefore, Product Owners should either be very familiar with the legacy system or team up with someone who is. To ensure compatibility, the Product Owner should also consult an expert (architect) [106].

### C.4.2.2 Implications on Product Owner role

The Product Owner should work with former project members of the legacy system to understand the system, provide/get specific training on this system and / or establish a knowledge management infrastructure.

### C.4.2.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.19:** Mapped PO peculiarities to O.2

Tasks	Characteristics	Structure
Communication	C&N	Expert
Intermediary	T&L	
Gate Keeper	A&Q	
Expectations Manager		

**Table C.20:** Mapped Action Items to O.2

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	ARE		ATT
	AAR		AKM
	AIT		

## C.4.3 Factor: Criticality [O.3]

### C.4.3.1 Rationales

Safety & security usually cause a comprehensive documentation of the project [43]. The more safety critical the product is, the more quality standards needs to be considered. The PO needs to be trained on this.

### C.4.3.2 Implications on Product Owner role

The Product Owner should get properly trained on quality standards.

### C.4.3.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.21:** Mapped PO peculiarities to O.3

Tasks	Characteristics	Structure
Communication Intermediary Acceptance Tester	A&Q	

**Table C.22:** Mapped Action Items to O.3

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
			AIW ATT AKM

## C.4.4 Factor: Hardware Development [O.4]

### C.4.4.1 Rationales

If hardware development is also part of a project, the Product Owner becomes an interface between the hardware group and the software group [105]. If the hardware follows a more traditional development approach, the Product Owner needs to align the work and emphasize on planning [43]. Furthermore, the Product Owner should either be keen on hardware development or be teamed up with an expert. Also a Product Owner team could be established to share responsibilities [106].

### C.4.4.2 Implications on Product Owner role

The Product Owner becomes an interface to other departments and needs to collaborate with them. For a complex project, a cross-functional-team could be established [82].

### C.4.4.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.23:** Mapped PO peculiarities to O.4

Tasks	Characteristics	Structure
Communication Intermediary Gate Keeper Expectations Manager	C&N T&L	Expert

**Table C.24:** Mapped Action Items to O.4

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	ARE AAR AIT		

## C.4.5 Factor: User Interface [O.5]

### C.4.5.1 Rationales

The Product Owner should be teamed up with an expert (UX Designer / Front-end Designer) [106].

### C.4.5.2 Implications on Product Owner role

The Product Owner should work closely with a specialized role (e.g. UX Designer / Front-end Designer).

### C.4.5.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.25:** Mapped PO peculiarities to O.5

Tasks	Characteristics	Structure
Communication	C&N	Expert
Intermediary	T&L	

**Table C.26:** Mapped Action Items to O.5

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related

## C.4.6 Factor: System Integration Test [O.6]

### C.4.6.1 Rationales

A required system integration test needs to be prepared properly. This requires the Product Owner to emphasize on planning and setting up a test integration team [43].

### C.4.6.2 Implications on Product Owner role

The Product Owner becomes an interface to other departments and needs to collaborate with them. For a complex project, a cross-functional-team could be established [82].

### C.4.6.3 Mapping of Product Owner Peculiarities & Action Items

**Table C.27:** Mapped PO peculiarities to O.6

Tasks	Characteristics	Structure
Communication	C&N	Team
Intermediary	T&L	Expert
Gate Keeper	A&Q	
Expectations Manager		

**Table C.28:** Mapped Action Items to O.6

Stakeholder-related	Project Life Cycle	Project Organization	Knowledge-related
	AAR AIT	ATI	

### C.4.7 High-level Recommendations for Category: Objective

Product Owner(s) should have adequate domain knowledge to better estimate requirement changes and dependencies. The support of other specialized roles is required.





# D

## Truth Tables

This appendix presents the truth tables by the related categories described in Section 13.4. The truth tables have their basis in the mapping of the Product Owner peculiarities and the action items to the individual factors as described in the catalog in Section 13.3 and Appendix C.

## D.1 Truth Table for Category: Team

**Table D.1:** Truth Table for Category:Team

			T1	T2	T1 ∨ T2
Action Items	Project Organization	ADF	1	1	1
		ACP	1	1	1
		ATI			
Product Owner Peculiarities	Tasks	Communicator	1	1	1
		Groom			
		Prioritizer			
		Travel		1	1
		Intermediary			
		Risk Assessor			
		Gate Keeper			
		Acceptance Tester			
		Customer Rel. Manager			
		Expectation Manager			
		Super Secretary			
		Visionary			
		Critical Decision Maker			
	Charact.	C&N	1	1	1
		V&D			
		L&T	1	1	1
		A&Q	1	1	1
		E&C			
	Structure	single			
		PO Team	1	1	1
		Expert Team			

## D.2 Truth Table for Category: Internal Environment

**Table D.2:** Truth Table for Category: Internal Environment

						IE1 ∨ IE2	IE1 ∨ IE3	IE2 ∨ IE3	IE1 ∨ IE2 ∨ IE3	
Action Items	Stakehol.-related	ACI								
		AUI								
		AMI	1	1		1	1	1	1	
	Project Life Cycle	ARE	1			1	1		1	
		AAR								
		AIT								
		AFI			1		1	1	1	
	Project Orga	AFL								
		ADF								
		ACP			1		1	1	1	
	Knowl.-related	ATI								
		AIW								
		ATT								
	Product Owner Peculiarities	Tasks	AKM							
			Communicator							
Groom			1			1	1		1	
Prioritizer										
Travel										
Intermediary										
Risk Assessor										
Gate Keeper										
Acceptance Tester			1			1	1		1	
Customer Rel. Manager				1		1		1	1	
Expectation Manager										
Super Secretary										
Visionary		1			1	1		1		
Critical Dec. Maker										
Charact.		C&N			1		1	1	1	
		V&D	1			1	1		1	
		L&T	1	1		1	1	1	1	
		A&Q								
		E&C	1			1	1		1	
Structure		single								
		PO Team	1	1		1	1	1	1	
	Expert Team	1	1		1	1	1	1		

### D.3 Truth Table for Category: External Environment

**Table D.3:** Truth Table for Category: External Environment

						EE1 ∨ EE2	EE1 ∨ EE3	EE2 ∨ EE3	EE1 ∨ EE2 ∨ EE3
			EE1	EE2	EE3	EE2	EE3	EE3	EE3
Action Items	Stakehol.-related	ACI	1			1	1		1
		AUI	1			1	1		1
		AMI							
	Project Life Cycle	ARE			1		1	1	1
		AAR							
		AIT							
		AFI							
	Project Orga	AFL		1	1	1	1	1	1
		ADF	1			1	1		1
		ACP	1			1	1		1
	Knowl.-related	ATI							
		AIW	1			1	1		1
ATT									
Product Owner Peculiarities	Tasks	AKM							
		Communicator	1	1	1	1	1	1	1
		Groom							
		Prioritizer	1			1	1		1
		Travel							
		Intermediary	1	1		1	1	1	1
		Risk Assessor	1			1	1		1
		Gate Keeper			1		1	1	1
		Acceptance Tester			1		1	1	1
		Customer Rel. Manager							
		Expectation Manager	1			1	1		1
		Super Secretay							
	Visionary								
	Critical Dec. Maker	1		1	1	1	1	1	
	Charact.	C&N	1	1	1	1	1	1	1
		V&D							
		L&T							
		A&Q	1		1	1	1	1	1
		E&C			1		1	1	1
	Structure	single	1			1	1	1	1
		PO Team							
Expert Team									

#### D.4 Truth Table for Category: Objective



















## Values & Scales

This appendix lists the values, scales and rationals of the identified influencing factors on the Product Owner role described in Section 13.4.1. The rationals result from the results of the research conducted by the author of this thesis. The scales have been used similarly in the listed references. This is highlighted as it marks the reliability of the scales from other well-known researchers in the field of Software Engineering.

**Table E.1:** Values, Scales & Rationales of influencing Factors

<i>P</i>	Value	Scale	Rationale	Similarly used by
T.1	3	micro 10 small 11-30 large 30	The bigger the team, the more likely it is that more Product Owner are required.	[16, 58]
T.2	4	co-located nationally regionally globally	The distribution of the teams identifies how many Product Owners are required.	[58]
IE.1	5	strongly agree agree neutral disagree strongly disagree	A blurry project proposal might lead to volatile requirements. Early indicator for actions on Requirements Engineering.	[119]
IE.2	5	strongly agree agree neutral disagree strongly disagree	The less the management is committed to the project, the more the Product Owner needs to engage with them.	[119]

*Continued on next page*

Table E.1 – *Continued from previous page*

<i>P</i>	Value	Scale	Rationale	Similarly used by
IE.3	5	strongly disagree	A strongly controlled project requires formalized communication. The PMO should be involved.	[119]
		disagree		
		neutral		
		agree		
		strongly agree		
EE.1	4	very few 1-5	More stakeholders might require multiple Product Owners and / or a Stakeholder team should be formed.	none
		few 5-10		
		many 10-15		
		a lot 15		
EE.2	5	strongly agree	The less available the stakeholders are, the more formalized communication and strict feedback loops are required.	[119]
		agree		
		neutral		
		disagree		
		strongly disagree		
EE.3	5	strongly agree	The less stable the requirements are, the more formalized procedures (e.g. Definition of Ready), close feedback loops or cross-functional teams are required.	[119]
		agree		
		neutral		
		disagree		
		strongly disagree		
O.1	5	strongly disagree	The more complex the project is, the more formalized communication and documentation is required. Might require prototyping, too.	[119]
		disagree		
		neutral		
		agree		
		strongly agree		
O.2	2	no	Requires to focus on compatible requirements, data migration efforts.	none
		yes		
O.3	5	comfort	The more critical the product is, the more stable requirements and testing is required.	[16]
		directionary funds		
		essential funds		
		single life		
O.4	2	many lives	Requires collaboration with other departments and integration tests.	none
		no		
O.5	2	yes	Requires collaboration with UX/UI experts and focus on user testing.	none
		no		
O.6	2	no	Requires integration strategies and skilled team members.	none
		yes		



## Test Case Scenarios

This appendix presents the scenarios handed to the participants of the comparative case study described in Section 15.1.2.

## Scenario 1

- A group of >30 developers
- Distributed to multiple locations across the globe

- The initial project proposal was clear and concrete
- Management is not fully committed and only available rarely
- Project is strongly financially controlled

- Only few stakeholders are involved
- The stakeholders as well as the users are available for testing purposes
- Requirements are stable

- Project is not considerably complex
- Project needs to consider a legacy system
- Criticality of developed product is in mid-range (some money would be lost, but no lives are in danger)
- Project contains hardware development
- Project has only a small front-end component
- System integration tests are required



## Scenario 2

- A group of 10 developers
- Co-located in one location

- The initial project proposal was not clear
- Management is not fully committed and only available rarely
- Project is strongly financially controlled

- Only a few stakeholders are involved
- Neither the stakeholders nor the users are available to discuss requirements
- Requirements are not stable

- Project is rather complex
- Project does not need to consider a legacy system
- Criticality of developed product is in lower mid-range (money would be lost, but no lives are in danger)
- Project contains hardware development
- Project has only as small front-end component
- No system integration tests are required

### Scenario 3

- A group of 10 developers
- Co-located in one location

- The initial project proposal was clear
- Management is committed to project goals but not available
- Project is not strongly financially controlled

- Number of stakeholders is high
- Neither the stakeholders nor the users are available for testing purposes
- Requirements are stable

- Project is rather simple
- Project does not need to consider a legacy system
- Criticality of developed product comfortably low (some money would be lost, but no lives are in danger)
- Project contains hardware development
- Project has huge front-end component
- No system integration tests are required



## Publications

This appendix presents the publications of the author of this thesis.

- C. Unger-Windeler and J. Kliünder. On the tasks and characteristics of product owners: a case study in the oil and gas industry. In *International Conference on Product-Focused Software Process Improvement*, pages 3–11. Springer, 2018. [103]
- C. Unger-Windeler, J. Kliünder, and K. Schneider. A mapping study on product owners in industry: identifying future research directions. In *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, pages 135–144. IEEE, 2019. [104]
- C. Unger-Windeler and K. Schneider. Expectations on the product owner role in systems engineering – a scrum team’s point of view. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 276–283. IEEE, 2019. [106]
- C. Unger-Windeler, J. A.-C. Kliünder, T. M. Reuscher, and K. Schneider. Are product owners communicators? a multi-method research approach to provide a more comprehensive picture of product owners in practice. *Journal of Software: Evolution and Process*, 2020. (accepted but not yet published) [105]
- O. Karras, C. Unger-Windeler, L. Glauer, and K. Schneider. Video as a by-product of digital prototyping: capturing the dynamic aspect of interaction. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, pages 118–124. IEEE, 2017. [46]
- J. Klünder, C. Unger-Windeler, F. Kortum, and K. Schneider. Team meetings and their relevance for the software development process over time. In *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 313–320. IEEE, 2017. [49]

- N. Prenner, C. Unger-Windeler, and K. Schneider. How are hybrid development approaches organized? – a systematic literature review. In *2020 IEEE International Conference on Software and System Processes (ICSSP)*. IEEE, 2020. [80]

# List of Figures

1.1	Three-cycle view of Design Science Research (based on [35]) . . . . .	3
2.1	Basic Representation of a Scrum Team[106] . . . . .	10
2.2	LeSS Product Owner Team and Feature Teams[62, p.265] . . . . .	12
2.3	Fan-out model for Product Manager, PO, and Agile teams [23] . . . . .	13
3.1	Home Ground Polar Chart (based on [16]) . . . . .	16
5.1	Product Owner taxonomy (based on [6, 8]) . . . . .	26
5.2	Reference Model Tasks & Characteristics [103] . . . . .	27
5.3	The hierarchical Product Owner Team . . . . .	28
5.4	The non-hierarchical Product Owner Team . . . . .	29
5.5	The Expert Team with Specialized Roles . . . . .	29
6.1	Modified Research Process . . . . .	32
6.2	Search process and filtering steps. . . . .	35
6.3	Publications by year. . . . .	37
6.4	Research topics addressing the Product Owner / On-Site Customer Role in literature. . . . .	38
6.5	Systematic Map of Research Question 1.1 and 1.2. . . . .	38
6.6	Publications consider team dimensions. . . . .	42
8.1	PO team structure . . . . .	52
8.2	Meeting Categories. The meeting titles marked with an asterisk (*) identify that these meetings are beyond the regular Nexus meetings. . . . .	55
8.3	Quantification of Formal Meetings . . . . .	57
8.4	Quantification of Informal Meetings . . . . .	58
8.5	Product Owner Network. Roles represented by multiple individuals are represented by a twin-circle. . . . .	59
9.1	Stage-Gate-Process at Baker Hughes (Baker Hughes,[3]) . . . . .	62
9.2	Basic FLOW syntax by Schneider et al. [88] . . . . .	63
9.3	Exemplary isolated FLOW-Diagram . . . . .	66
9.4	Product Manager: Verification of Request . . . . .	67
9.5	Product Manager: Documentation of Requirements . . . . .	67
9.6	Engineering Group: Conduct feasibility study . . . . .	68
9.7	Engineering Group: Requirements elicitation . . . . .	68
9.8	PM: Requirements solidification . . . . .	69
9.9	Engineering Group: SOR refinement . . . . .	69

9.10	SW Group: SOR review . . . . .	70
9.11	SW Group: planning . . . . .	70
9.12	SW Group: Identify SMEs . . . . .	71
9.13	SW Group: PO refinement with SME . . . . .	71
9.14	SW Group: Requirements solidification . . . . .	72
9.15	SW Group: PO refinement with Scrum Team . . . . .	72
9.16	PM: Requirements Change . . . . .	73
9.17	Combined and Chronological FLOW diagram . . . . .	74
9.18	Collaboration FLOW of CPO and traditional management roles . . . . .	77
10.1	Extended representation of a Scrum Team . . . . .	80
10.2	Roles of Participants . . . . .	82
10.3	Years of Experience in the respective role . . . . .	82
10.4	Expectations on Tasks . . . . .	84
10.5	Extended Scrum Team triangle . . . . .	86
10.6	Attributes of a perfect Product Owner . . . . .	86
12.1	Components of an expert system . . . . .	94
13.1	Category based high-level recommendations . . . . .	105
13.2	Decision Tree for Category Internal Environment . . . . .	112
14.1	<i>HyPrO Expert System</i> User Interface . . . . .	114
14.2	Comparison among Categories . . . . .	115
14.3	Tab Row . . . . .	116
14.4	Tab 'Overview' Content . . . . .	117
14.5	Tab Content of Category:Objective . . . . .	118
14.6	Function to evaluate the input and return recommendation . . . . .	119
15.1	Binary Field (exemplary) . . . . .	126
15.2	Assigned Weight Factors (exemplary) . . . . .	127
15.3	Interim Values (exemplary) . . . . .	127
15.4	Interim Values with Penalty Points (exemplary) . . . . .	128
15.5	Resulting Scores per Participant (exemplary) . . . . .	129
15.6	Scores of all human experts versus the <i>HyPrO Expert System</i> . . . . .	130

# List of Tables

4.1	Descriptive Phrases of Product Owner Tasks, Characteristics & Structures of Part I	20
5.1	Product Owner Team Structures and Terminology . . . . .	30
6.1	Overview search keywords . . . . .	33
6.2	Inclusion and Exclusion Criteria . . . . .	34
6.3	Data extraction form . . . . .	36
6.4	Publications by Research Topic and Methodology . . . . .	39
6.5	Activities of Product Owners . . . . .	40
6.6	Publications consider external circumstances . . . . .	43
7.1	Coded Segments of Product Owner Tasks, Characteristics & Structures of Part II	46
8.1	GQM Plan . . . . .	54
8.2	Interactions over the course of a sprint . . . . .	60
9.1	Roles and Responsibilities of considered Interview Partners at Baker Hughes . .	65
9.2	Roles and Activities within hybrid RE process at Baker Hughes . . . . .	75
10.1	Expected roles on tasks by the Scrum Team . . . . .	83
11.1	Descriptive Phrases of Product Owner Tasks, Characteristics & Structures of Part III . . . . .	88
13.1	Relevant Product Owner Tasks in Hybrid Environments . . . . .	96
13.2	Relevant Product Owner Characteristics in Hybrid Environments . . . . .	97
13.3	Single Product Owner Structures for Hybrid Environments . . . . .	99
13.4	Hierarchical Product Owner Team Structures for Hybrid Environments . . . . .	100
13.5	Non-Hierarchical Product Owner Team Structures for Hybrid Environments . . .	101
13.6	Relevant Action Items for Product Owners . . . . .	104
13.7	Defined Ps for the <i>HyPrO Expert System</i> . . . . .	106
13.8	Set of relevant Action Items and Product Owner Peculiarities for the <i>HyPrO Expert System</i> . . . . .	107
13.9	Defined Values & Classifications per Factor (P) . . . . .	109
13.10	Truth Table for Category <i>Internal Environment</i> . . . . .	110
15.1	Test Case Scenarios . . . . .	124
15.2	Scenario 1: Recommendations provided by the human experts and <i>HyPrO Expert System</i> . . . . .	131
15.3	Scenario 1: Values to assess performance . . . . .	132

15.4 Scenario 2: Recommendations provided by the human experts and <i>HyPrO Expert System</i> . . . . .	133
15.5 Scenario 2: Values to assess performance . . . . .	134
15.6 Scenario 3: Recommendations provided by the human experts and <i>HyPrO Expert System</i> . . . . .	135
15.7 Scenario 3: Values to assess performance . . . . .	136
C.1 Mapped PO peculiarities to T.1 . . . . .	163
C.2 Mapped Action Items to T.1 . . . . .	164
C.3 Mapped PO peculiarities to T.2 . . . . .	164
C.4 Mapped Action Items to T.2 . . . . .	164
C.5 Mapped PO peculiarities to IE.1 . . . . .	165
C.6 Mapped Action Items to IE.1 . . . . .	165
C.7 Mapped PO peculiarities to IE.2 . . . . .	166
C.8 Mapped Action Items to IE.2 . . . . .	166
C.9 Mapped PO peculiarities to IE.3 . . . . .	166
C.10 Mapped Action Items to IE.3 . . . . .	166
C.11 Mapped PO peculiarities to EE.1 . . . . .	167
C.12 Mapped Action Items to EE.1 . . . . .	167
C.13 Mapped PO peculiarities to EE.2 . . . . .	168
C.14 Mapped Action Items to EE.2 . . . . .	168
C.15 Mapped PO peculiarities to EE.3 . . . . .	169
C.16 Mapped Action Items to EE.3 . . . . .	169
C.17 Mapped PO peculiarities to O.1 . . . . .	169
C.18 Mapped Action Items to O.1 . . . . .	170
C.19 Mapped PO peculiarities to O.2 . . . . .	170
C.20 Mapped Action Items to O.2 . . . . .	170
C.21 Mapped PO peculiarities to O.3 . . . . .	171
C.22 Mapped Action Items to O.3 . . . . .	171
C.23 Mapped PO peculiarities to O.4 . . . . .	171
C.24 Mapped Action Items to O.4 . . . . .	172
C.25 Mapped PO peculiarities to O.5 . . . . .	172
C.26 Mapped Action Items to O.5 . . . . .	172
C.27 Mapped PO peculiarities to O.6 . . . . .	173
C.28 Mapped Action Items to O.6 . . . . .	173
D.1 Truth Table for Category:Team . . . . .	176
D.2 Truth Table for Category: Internal Environment . . . . .	177
D.3 Truth Table for Category: External Environment . . . . .	178
D.4 Truth Table for Category: Objective (O1-O5) . . . . .	180
D.5 Truth Table for O6 . . . . .	182
D.6 Truth Table for All Categories . . . . .	184
D.7 Truth Table for Structures for all Categories . . . . .	186
E.1 Values, Scales & Rationales of influencing Factors . . . . .	187



# Glossary

<b>APO</b>	Area Product Owner
<b>A&amp;Q</b>	Available & Qualified
<b>AI</b>	Action Item
<b>AAR</b>	Action Item: put emphasis on system architecture
<b>ACI</b>	Action Item: intensify customer involvement
<b>ACP</b>	Action Item: formalize project communication pattern, e.g. schedule regular meetings beyond the used framework
<b>ADF</b>	Action Item: expand project documentation, e.g (agile) RE tool
<b>AFI</b>	Action Item: put emphasis on financial project management, e.g. collaborate with PMO
<b>AFL</b>	Action Item: put emphasis on fast feedback loops
<b>AIT</b>	Action Item: put emphasis on integration and test
<b>AIW</b>	Action Item: intensify meetings/workshops, e.g., specific for certain stakeholder focus groups, shorten intervals between workshops
<b>AKM</b>	Action Item: provide knowledge management infrastructure
<b>AMI</b>	Action Item: ensure management involvement
<b>ARE</b>	Action Item: put emphasis on requirements engineering
<b>ATI</b>	Action Item: select appropriate tools w.r.t. the process's weight
<b>ATT</b>	Action Item: provide training, e.g., role-specific training
<b>AUI</b>	Action Item: intensify end user involvement, e.g. ui testing
<b>BHGE</b>	Baker Hughes, a GE Company
<b>CRM</b>	Customer Relationship Manager
<b>CDM</b>	Critical Decision Maker
<b>CDR</b>	Conceptual Design Review
<b>CP</b>	Checkpoints
<b>CPO</b>	Chief Product Owner
<b>C&amp;N</b>	Communicator & Negotiator
<b>DDR</b>	Detailed Design Review
<b>DVR</b>	Detailed Verification Review
<b>Exp</b>	Expert
<b>E&amp;C</b>	Empowered & Committed
<b>FDD</b>	Feature-Driven Development
<b>GQM</b>	Goal-Question-Metric
<b>HW</b>	Hardware
<b>HyPrO</b>	Hybrid Product Owner
<b>LeSS</b>	Large-Scale Scrum
<b>L&amp;T</b>	Leader & Team player
<b>Mger</b>	Manager
<b>Mgmt</b>	Management
<b>OSC</b>	On-Site Customer
<b>PBI</b>	Product Backlog Item
<b>PDR</b>	Preliminary Design Review
<b>PM</b>	Product Manager
<b>PMO</b>	Project Management Office
<b>PO</b>	Product Owner

<b>POC</b>	Proof of Concept
<b>PPO</b>	Proxy Product Owner
<b>PRR</b>	Product Requirement Review
<b>PVR</b>	Preliminary Verification Review
<b>RE</b>	Requirements Engineering
<b>Rel.</b>	Relationship
<b>RQ</b>	Research Question
<b>SAFe</b>	Scaled Agile Framework
<b>SME</b>	Subject Matter Expert
<b>SOR</b>	Specification of Requirements
<b>SW</b>	Software
<b>UI</b>	User Interface
<b>UX</b>	User Experience
<b>V&amp;D</b>	Visionary & Doer
<b>XP</b>	eXtreme Programming

# Bibliography

- [1] S. W. Ambler and M. Lines. *Disciplined agile delivery: A practitioner's guide to agile software delivery in the enterprise*. IBM Press, 2012.
- [2] P. Atteslander and J. Cromm. *Methoden der empirischen Sozialforschung*. ESV basics. Erich Schmidt, Berlin, 13., neu bearb. und erw. Aufl. Edition, 2010. ISBN: 9783503126187.
- [3] Baker Hughes. Internal document: product development process, 2018.
- [4] V. R. Basili and H. D. Rombach. The tame project: towards improvement-oriented software environments. *IEEE Transactions on Software Engineering*, 14(6):758–773, 1988. ISSN: 0098-5589. DOI: 10.1109/32.6156.
- [5] V. R. Basili and H. D. Rombach. Tailoring the software process to project goals and environments. In *Proceedings of the 9th international conference on Software Engineering*, pages 345–357. IEEE Computer Society Press, 1987.
- [6] J. M. Bass. How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, 20(6):1525–1557, 2015.
- [7] J. M. Bass, S. Beecham, J. Noll, and Razzak. All hands to the pumps: the product owner role in small companies. lero technical report: 2017.1.
- [8] J. M. Bass, S. Beecham, M. A. Razzak, C. N. Canna, and J. Noll. An empirical study of the product owner role in scrum. In *Proceedings of the 40th International Conference on Software Engineering: Companion Proceedings*, pages 123–124, 2018.
- [9] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Greening, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. Manifesto for agile software development. URL: <https://agilemanifesto.org/>, (Retrieved: Oct. 14th, 2019).
- [10] K. Beck. *extreme programming eXplained: Embrace change*. Addison-Wesley, Reading MA, 2000. ISBN: 0201616416.
- [11] K. Beck. *Extreme programming explained: Embrace change*. Addison-Wesley, Boston, MA, 2nd ed. Edition, 2005. ISBN: 9780321278654.
- [12] A. Begel and N. Nagappan. Usage and perceptions of agile software development in an industrial context: an exploratory study. In *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, pages 255–264. IEEE, 9/20/2007 - 9/21/2007. ISBN: 978-0-7695-2886-1.
- [13] K. L. Bellman. The modeling issues inherent in testing and evaluating knowledge-based systems. *Expert Systems with Applications*, 1(3):199–215, 1990.

- [14] B. Boehm and R. Turner. Management challenges to implementing agile processes in traditional development organizations. *IEEE Software*, 22(5):30–39, 2005. ISSN: 0740-7459. DOI: 10.1109/MS.2005.129.
- [15] B. Boehm and R. Turner. Using risk to balance agile and plan-driven methods. *Computer*, 36(6):57–66, 2003. ISSN: 0018-9162. DOI: 10.1109/MC.2003.1204376.
- [16] Boehm, Barry and Turner, Richard. *Balancing agility and discipline: A guide for the perplexed*. Addison-Wesley Professional, 2003.
- [17] B Chandrasekaran. On evaluating artificial intelligence systems for medical diagnosis. *AI magazine*, 4(2):34–34, 1983.
- [18] A. Cockburn. *Crystal clear: A human-powered methodology for small teams: A human-powered methodology for small teams*. Pearson Education, 2004.
- [19] J. Cohen. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.
- [20] M. Cohn. *Succeeding with Agile: Software Development using Scrum*. Addison-Wesley, 2009.
- [21] R. G. Cooper. Stage-gate systems: a new tool for managing new products. *Business Horizons*, 33:44–54, 1990.
- [22] R. G. Cooper et al. Winning at new products: accelerating the process from idea to launch, 2001.
- [23] © Scaled Agile, Inc. Product owner. URL: <https://www.scaledagileframework.com/product-owner/>, (Retrieved: Oct. 14th, 2019).
- [24] J. Corbin and A. Strauss. *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Sage publications, 2014.
- [25] A. d. S. Croix and A. Easton. The product owner team. In *Agile 2008 Conference*, pages 274–279. IEEE, 8/4/2008 - 8/8/2008. ISBN: 978-0-7695-3321-6. DOI: 10.1109/Agile.2008.94.
- [26] T. Dybå and T. Dingsøy. Empirical studies of agile software development: a systematic review. *Information and Software Technology*, 50(9-10):833–859, 2008. ISSN: 09505849. DOI: 10.1016/j.infsof.2008.01.006.
- [27] J. Eckstein. *Agile software development with distributed teams: Staying agile in a global world*. Addison-Wesley, 2013.
- [28] C Farell, R Narang, S Kapitan, and H Webber. Towards an effective onsite customer practice. In *Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2002)*. Springer, 2002.
- [29] M. Finsterwalder. Does xp need a professional customer? In *XP2001 Workshop on customer involvement*. ACM, 2001.
- [30] J. Grenning. Launching extreme programming at a process-intensive company. *IEEE Software*, 18(6):27–33, 2001.
- [31] P. D. Grogono, A. D. Preece, R. Shinghal, and C. Y. Suen. A review of expert systems evaluation techniques. In *Workshop on Validation and Verification of Knowledge-Based Systems*, pages 120–125, 1993.

- [32] R. K. Gupta, S. Jain, and B. Singh. Challenges in scaling up a globally distributed legacy product. In M. Paasivaara, D. Šmite, and R. Evaristo, editors, *Proceedings of the 13th Conference on Global Software Engineering - ICGSE '18*, pages 77–81, New York, New York, USA. ACM Press, 2018. ISBN: 9781450357173. DOI: 10.1145/3196369.3196389.
- [33] V. T. Heikkilä, M. Paasivaara, C. Lasssenius, D. Damian, and C. Engblom. Managing the requirements flow from strategy to release in large-scale agile development: a case study at ericsson. *Empirical Software Engineering*, 22(6):2892–2936, 2017. ISSN: 1573-7616. DOI: 10.1007/s10664-016-9491-z.
- [34] A. Hevner and S. Chatterjee. Design science research in information systems. In *Design research in information systems*, pages 9–22. Springer, 2010.
- [35] A. R. Hevner. A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4, 2007.
- [36] A. R. Hevner, S. T. March, J. Park, and S. Ram. Design science in information systems research. *MIS quarterly*:75–105, 2004.
- [37] R. Hoda, J. Noble, and S. Marshall. The impact of inadequate customer collaboration on self-organizing agile teams. *Information and Software Technology*, 53(5):521–534, 2011. ISSN: 09505849. DOI: 10.1016/j.infsof.2010.10.009.
- [38] P. Hohl, J. Münch, K. Schneider, and M. Stupperich. Forces that prevent agile adoption in the automotive domain. In *International Conference on Product-Focused Software Process Improvement*, pages 468–476. 2016.
- [39] D. Hussman. Coaching a customer team. In G. Goos, J. Hartmanis, J. van Leeuwen, M. Marchesi, and G. Succi, editors, *Extreme Programming and Agile Processes in Software Engineering*. Volume 2675, Lecture Notes in Computer Science, pages 254–260. Springer, Berlin, Heidelberg, 2003. ISBN: 978-3-540-40215-2. DOI: 10.1007/3-540-44870-5{\textunderscore}31.
- [40] R. Jacob, J. P. Décieux, and A. Heinz. *Umfrage: Einführung in die Methoden der Umfrageforschung*. 2013.
- [41] K. H. Judy and I. Krumins-Beens. Great scrums need great product owners: unbounded collaboration and collective product ownership. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, page 462. IEEE, 1/7/2008 - 1/10/2008. DOI: 10.1109/HICSS.2008.186.
- [42] J. Kacprzyk, L. C. Jain, C. Grosan, and A. Abraham. *Intelligent Systems*, volume 17. Springer Berlin Heidelberg, 2011. ISBN: 978-3-642-21003-7. DOI: 10.1007/978-3-642-21004-4.
- [43] G. Kalus and M. Kuhrmann. Criteria for software process tailoring: a systematic review. In *Proceedings of the 2013 International Conference on Software and System Process*, pages 171–180, 2013.
- [44] D. Karlstrom and P. Runeson. Combining agile methods with stage-gate project management. *IEEE software*, 22(3):43–49, 2005.
- [45] D. Karlström and P. Runeson. Integrating agile software development into stage-gate managed product development. *Empirical Software Engineering*, 11(2):203–225, 2006.

- [46] O. Karras, C. Unger-Windeler, L. Glauer, and K. Schneider. Video as a by-product of digital prototyping: capturing the dynamic aspect of interaction. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, pages 118–124. IEEE, 2017.
- [47] S. Kiesling, J. Klünder, D. Fischer, K. Schneider, and K. Fischbach. Applying social network analysis and centrality measures to improve information flow analysis. In *International Conference on Product-Focused Software Process Improvement*, pages 379–386. Springer, 2016.
- [48] B. Kitchenham and S. Charters. Guidelines for performing systematic literature reviews in software engineering. *Keele University EBSE Technical Report EBSE-2007-01*.
- [49] J. Kluender, C. Unger-Windeler, F. Kortum, and K. Schneider. Team meetings and their relevance for the software development process over time. In *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 313–320. IEEE, 2017.
- [50] J. Klünder, R. Hebig, P. Tell, M. Kuhrmann, J. Nakatumba-Nabende, R. Heldal, S. Krusche, M. Fazal-Baqaie, M. Felderer, M. F. G. Bocco, et al. Catching up with method and process practice: an industry-informed baseline for researchers. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice*, pages 255–264. IEEE Press, 2019.
- [51] J. Klünder, P. Hohl, and K. Schneider. Becoming agile while preserving software product lines: an agile transformation model for large companies. In *Proceedings of the 2018 International Conference on Software and System Process*, pages 1–10, 2018.
- [52] J. A.-C. Klünder, P. Hohl, N. Prenner, and K. Schneider. Transformation towards agile software product line engineering in large companies: a literature review. *Journal of Software: Evolution and Process*, 31(5):e2168, 2019.
- [53] J. Koskela and P. Abrahamsson. On-site customer in an xp project: empirical results from a case study. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and T. Dingsøyr, editors, *Software Process Improvement*, volume 3281 of *Lecture Notes in Computer Science*, pages 1–11, Berlin, Heidelberg. Springer, 2004. ISBN: 978-3-540-23725-9. DOI: 10.1007/978-3-540-30181-3{\textunderscore}1.
- [54] H. Koziolk. Goal, question, metric. In *Dependability metrics*, pages 39–42. Springer, 2008.
- [55] S. Kristinsdottir, M. Larusdottir, and Å. Cajander. 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, and F. Kis, editors, *Human-Centered and Error-Resilient Systems Development*, pages 3–16, Cham. Springer International Publishing, 2016. ISBN: 978-3-319-44902-9.
- [56] M. Kuhrmann. You can’t tailor what you haven’t modeled. In H. Zhang, L. Huang, and I. Richardson, editors, *Proceedings of the 2014 International Conference on Software and System Process - ICSSP 2014*, pages 189–190, New York, New York, USA. ACM Press, 2014. ISBN: 9781450327541. DOI: 10.1145/2600821.2600851.

- [57] M. Kuhrmann, P. Diebold, J. Münch, P. Tell, V. Garousi, M. Felderer, K. Trektene, F. McCaffery, O. Linssen, E. Hanser, et al. Hybrid software and system development in practice: waterfall, scrum, and beyond:30–39, 2017.
- [58] M. Kuhrmann, P. Tell, J. Klünder, R. Hebig, S. A. Licorish, and S. G. MacDonell. Helena stage 2 results. [online] DOI: 10.13140/RG.2.2.14807.52649, 2018.
- [59] M. Laanti, O. Salo, and P. Abrahamsson. Agile methods rapidly replacing traditional methods at nokia: a survey of opinions on agile transformation. *Information and Software Technology*, 53(3):276–290, 2011. ISSN: 09505849. DOI: 10.1016/j.infsof.2010.11.010.
- [60] C. Larman and B. Vodde. *Large-scale scrum: More with LeSS*. Addison-Wesley Professional, 2016.
- [61] C. Larman and B. Vodde. *Practices for scaling lean & agile development: Large, Multi-site, and Offshore Product Development with Large-Scale Scrum*. Addison-Wesley, Upper Saddle River NJ, 2010. ISBN: 9780321636409.
- [62] C. Larman and B. Vodde. *Scaling lean & agile development: thinking and organizational tools for large-scale Scrum*. Pearson Education India, 2008.
- [63] D. Leffingwell. *Scaling software agility: best practices for large enterprises*. Pearson Education, 2007.
- [64] M. Lowery and M. Evans. Scaling product ownership. In *AGILE 2007 (AGILE 2007)*, pages 328–333. IEEE, 8/13/2007 - 8/17/2007. ISBN: 0-7695-2872-4. DOI: 10.1109/AGILE.2007.51.
- [65] P. Manhart and K. Schneider. Breaking the ice for agile development of embedded software: an industry experience report. In *Proceedings. 26th International Conference on Software Engineering*, pages 378–386. 2004.
- [66] A. Martin, R. Biddle, and J. Noble. The xp customer role in practice: three studies. In *Agile Development Conference*, pages 42–54. IEEE, 22-26 June 2004. ISBN: 0-7695-2248-3. DOI: 10.1109/ADEV.2004.23.
- [67] A. Martin, R. Biddle, and J. Noble. The xp customer team: a grounded theory. In *2009 Agile Conference*, pages 57–64. IEEE, 8/24/2009 - 8/28/2009. ISBN: 978-0-7695-3768-9. DOI: 10.1109/AGILE.2009.70.
- [68] A. Martin, J. Noble, and R. Biddle. Being jane malkovich: a look into the world of an xp customer. In *International Conference on Extreme Programming and Agile Processes in Software Engineering*, pages 234–243. Springer, 2003.
- [69] A. M. Martin. *The role of customers in extreme programming projects*. PhD thesis, Victoria University of Wellington, 2009.
- [70] G. Matturro, F. Cordovés, and M. Solari. Role of product owner from the practitioner’s perspective. an exploratory study. In *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*, pages 113–118. Proceedings of the International Conference on Software Engineering Research and Practice, 2018.
- [71] S. McDonald. Studying actions in context: a qualitative shadowing method for organizational research. *Qualitative research*, 5(4):455–473, 2005.

- [72] S. Mohammadi, B. Nikkhahan, and S. Sohrabi. An analytical survey of on-site customer practice in extreme programming. In *International Symposium on Computer Science and its Applications*, pages 1–6, 10/13/2008 - 10/15/2008. ISBN: 978-0-7695-3428-2. DOI: 10.1109/CSA.2008.72.
- [73] R. M. O’Keefe and D. E. O’Leary. Expert system verification and validation: a survey and tutorial. *Artificial Intelligence Review*, 7(1):3–42, 1993.
- [74] S. Oomen, B. De Waal, A. Albertin, and P. Ravesteyn. How can scrum be succesful? competences of the scrum product owner.57–64, June 5-10 2017.
- [75] M. Paasivaara, V. T. Heikkila, and C. Lassenius. Experiences in scaling the product owner role in large-scale globally distributed scrum. In *2012 IEEE Seventh International Conference on Global Software Engineering*, pages 174–178. IEEE, 8/27/2012 - 8/30/2012. ISBN: 978-1-4673-2357-4. DOI: 10.1109/ICGSE.2012.41.
- [76] D. L. Parnas and P. C. Clements. A rational design process: how and why to fake it. *IEEE Transactions on Software Engineering*, SE-12(2):251–257, 1986. ISSN: 0098-5589. DOI: 10.1109/TSE.1986.6312940.
- [77] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Systematic mapping studies in software engineering. In *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*, pages 1–10. BCS, 2008.
- [78] K. Petersen, S. Vakkalanka, and L. Kuzniarz. Guidelines for conducting systematic mapping studies in software engineering: an update. *Information and Software Technology*, 64:1–18, 2015. ISSN: 09505849. DOI: 10.1016/j.infsof.2015.03.007.
- [79] R. Pichler. *Agile product management with Scrum: Creating products that customers love*. The Addison-Wesley signature series. Addison-Wesley, Upper Saddle River, N.J., 2010. ISBN: 9780321684172.
- [80] N. Prenner, C. Unger-Windeler, and K. Schneider. How are hybrid development approaches organized?-a systematic literature review. In *2020 IEEE International Conference on Software and System Processes (ICSSP)*. IEEE, 2020.
- [81] D. Raithatha. Making the whole product agile—a product owners perspective. In *International Conference on Extreme Programming and Agile Processes in Software Engineering*, pages 184–187. Springer, 2007.
- [82] T. Ressler. *Requirements Engineering Practices in Large-Scale Scrum: an explorative Study*. Masterthesis, Leibniz University Hannover, Software Engineering Group, 2019.
- [83] T. Reuscher. *Information Flow Analysis of Product Owners in Large-Scale Scrum*. Masterthesis, Leibniz University Hannover, Software Engineering Group, 2019.
- [84] C. Robson and K. McCartan. *Real world research: A resource for users of social research methods in applied settings*. 2016. ISBN: 9781119144854.
- [85] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009. ISSN: 1382-3256. DOI: 10.1007/s10664-008-9102-8.
- [86] J. Saldana. *An introduction to codes and coding. The coding manual for qualitative researchers*. Thousand Oaks, CA: Sage, 2009.
- [87] J. Saldana. *The Coding Manual for Qualitative Researchers*. Thousand Oaks, CA, 2013.



- [88] K. Schneider, K. Stapel, and E. Knauss. Beyond documents: visualizing informal communication. In *2008 Third International Workshop on Requirements Engineering Visualization (REV)*, pages 31–40. IEEE, 2008.
- [89] K. Schwaber. Nexus™ guide. the definitive guide to scaling scrum with nexus: the rules of the game. URL: <https://www.scrum.org/resources/nexus-guide>, (Downloaded: Oct. 14th, 2019).
- [90] K. Schwaber. *The enterprise and scrum*, 2007.
- [91] K. Schwaber and M. Beedle. *Agile software development with Scrum*. Series in agile software development. Prentice Hall, Upper Saddle River, NJ, 2002. ISBN: 9780130676344.
- [92] K. Schwaber and J. Sutherland. *The Scrum Guide™. The Definitive Guide to Scrum: The Rules of the Game*. 2017. URL: <https://www.scrumguides.org/index.html>, (Downloaded: Oct. 14th, 2019).
- [93] C. B. Seaman. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering*, 25(4):557–572, 1999.
- [94] Y. Shastri, R. Hoda, and R. Amor. Does the ”project manager” still exist in agile software development projects?:57–64, 2016.
- [95] K. Stapel, E. Knauss, and K. Schneider. Using flow to improve communication of requirements in globally distributed software projects. In *2009 Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills*, pages 5–14. IEEE, 2009.
- [96] K. Stapel and K. Schneider. Flow-methode-methodenbeschreibung zur anwendung von flow. *arXiv preprint:1202.5919*, 2012.
- [97] K. Stapel and K. Schneider. Managing knowledge on communication and information flow in global software projects. *Expert Systems*, 31(3):234–252, 2014.
- [98] Z. A. Styczynski, K. Rudion, and A. Naumann. *Einführung und Grundbegriffe der Expertensysteme*. Springer Berlin Heidelberg, 2017. ISBN: 978-3-662-53172-3. DOI: 10.1007/978-3-662-53172-3\_1.
- [99] C. Y. Suen, P. D. Grogono, R. Shinghal, and F. Coallier. Verifying, validating, and measuring the performance of expert systems. *Expert Systems with Applications*, 1(2):93–102, 1990.
- [100] H. S. Sverrisdottir, H. T. Ingason, and H. I. Jonasson. The role of the product owner in scrum-comparison between theory and practices. *Procedia - Social and Behavioral Sciences*, 119:257–267, 2014. ISSN: 18770428. DOI: 10.1016/j.sbspro.2014.03.030.
- [101] P. Tell, J. Klünder, S. Küpper, D. Raffo, S. G. MacDonell, J. Münch, D. Pfahl, O. Linssen, and M. Kuhrmann. What are hybrid development methods made of?: an evidence-based characterization. In *Proceedings of the International Conference on Software and System Processes*, pages 105–114. IEEE Press, 2019.
- [102] A. Tiwana and M. Keil. The one-minute risk assessment tool. *Communications of the ACM*, 47(11):73–77, 2004. ISSN: 00010782. DOI: 10.1145/1029496.1029497.
- [103] C. Unger-Windeler and J. Klünder. On the tasks and characteristics of product owners: a case study in the oil and gas industry. In *International Conference on Product-Focused Software Process Improvement*, pages 3–11. Springer, 2018.

- [104] C. Unger-Windeler, J. Klünder, and K. Schneider. A mapping study on product owners in industry: identifying future research directions. In *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, pages 135–144. IEEE, 2019.
- [105] C. Unger-Windeler, J. A.-C. Klünder, T. M. Reuscher, and K. Schneider. Are product owners communicators? a multi-method research approach to provide a more comprehensive picture of product owners in practice. *Journal of Software: Evolution and Process*, 2020.
- [106] C. Unger-Windeler and K. Schneider. Expectations on the product owner role in systems engineering—a scrum team’s point of view. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 276–283. IEEE, 2019.
- [107] VersionOne. 12th annual state of agile report. URL: <https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report>, (Retrieved: Oct. 14th, 2019).
- [108] VersionOne. 13th annual state of agile report, 2019. URL: Available from: <https://www.explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report>, (Retrieved: Oct. 14th, 2019).
- [109] L. Wallace and M. Keil. Software project risks and their effect on outcomes. *Communications of the ACM*, 47(4):68–73, 2004.
- [110] L. Wallace, M. Keil, and A. Rai. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. *Decision sciences*, 35(2):289–321, 2004.
- [111] N. Wallace, P. Bailey, and N. Ashworth. Managing xp with multiple or remote customers. In *Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2002)*. Springer, 2002.
- [112] X. Wang, Z. Wu, and M. Zhao. The relationship between developers and customers in agile methodology. In *2008 International Conference on Computer Science and Information Technology*, pages 566–572. IEEE, 2008.
- [113] M. Williams, J. Packlick, R. Bellubbi, and S. Coburn. How we made onsite customer work - an extreme success story. In *AGILE 2007 (AGILE 2007)*, pages 334–338. IEEE, 8/13/2007 - 8/17/2007. ISBN: 0-7695-2872-4. DOI: 10.1109/AGILE.2007.33.
- [114] C. Wohlin, M. Höst, and K. Henningsson. Empirical research methods in software engineering. In G. Goos, J. Hartmanis, J. van Leeuwen, R. Conradi, and A. I. Wang, editors, *Empirical Methods and Studies in Software Engineering*, pages 7–23. Springer, 2003. ISBN: 978-3-540-40672-3. DOI: 10.1007/978-3-540-45143-3<sup>2</sup>.
- [115] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.
- [116] A. Wojciechowski, M. Wesolowski, and W. Complak. Experimental evaluation of ‘on-site customer’ xp practice on quality of software and team effectiveness. In R. Meersman, T. Dillon, and P. Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2010 Workshops*, volume 6428 of *Lecture Notes in Computer Science*, pages 269–278. Springer, 2010. ISBN: 978-3-642-16960-1. DOI: 10.1007/978-3-642-16961-8<sup>45</sup>.

- [117] J. Wolfe and T. I. Chacko. Education team-size effects on business game performance and decision-making behaviors. *Decision Sciences*, 14(1):121–133, 1983.
- [118] P. Xu and B. Ramesh. Software process tailoring: an empirical investigation. *Journal of Management Information Systems*, 24(2):293–328, 2007. ISSN: 0742-1222. DOI: 10.2753/MIS0742-1222240211.
- [119] P. Xu and B. Ramesh. Using process tailoring to manage software development challenges. *IT Professional*, 10(4):39–45, 2008.
- [120] R. Yin. Case study research: design and methods, 4th edn sage publications. *Thousand Oaks*, 2009.
- [121] D. Zowghi and N. Nurmuliani. A study of the impact of requirements volatility on software project performance. In *Ninth Asia-Pacific Software Engineering Conference, 2002*. Pages 3–11. IEEE, 2002.



# Curriculum Vitae

## PERSONAL DETAILS

---

First Name	Carolin
Last Name	Unger-Windeler
Maiden Name	Unger
Date of Birth	February 4th, 1986
Place of Birth	Hannover

## EDUCATION

---

2008 – 2011	University of Applied Sciences Hannover, Bachelor of Arts, Technical Communication
2011 – 2014	Technische Universität Braunschweig, Master of Arts, Communication and Media Engineering
2013 – 2013	University of Iceland, Exchange Semester
2017 – 2020	Leibniz Universität Hannover, Doctorate, Software Engineering

## WORK EXPERIENCE

---

2010 – 2011	Volkswagen AG, Working Student (Public Relations), Wolfsburg, Germany
2011 – 2011	Volkswagen AG, Bachelor-Thesis: <i>On the way to become a top employer – Volkswagens internal media being put to the test</i> , Wolfsburg, Germany
2012 – 2013	Sports Center at Braunschweig University of Technology, Working Student (Web Development), Braunschweig, Germany
2012 – 2013	Institute of Operating Systems & Computer Network at Braunschweig University of Technology, Working Student (User Experience Design & Android Development), Braunschweig, Germany
2013 – 2014	IAV Automotive Engineering, Working Student (User Experience Design & E-Learning Development), Gifhorn, Germany
2013 – 2014	IAV Automotive Engineering, Master-Thesis: <i>Mobile Augmented Reality – Development of an Android-based learning application for the automotive industry</i> , Wolfsburg, Germany
2014 – 2016	Baker Hughes, User Experience Designer & Project Management, Houston, USA
2016 – 2017	Baker Hughes, User Experience Designer, Celle, Germany
2017 – 2020	Leibniz Universität Hannover, Research Associate, Hannover, Germany
2017 –	Baker Hughes, PhD Student & User Experience Designer, Celle, Germany