Identifying the Challenges of Applying "User Stories" in Practice and Exploring Possible Remedies: a Case Study

Anne-Claire de Vries
First supervisor: Dr. F. Dalpiaz, Utrecht University
Second supervisor: Prof. dr. S. Brinkkemper, Utrecht University
External supervisor: Edina Valkai, KPMG Digital Enablement

MSc Human-Computer Interaction
Utrecht University
Academic Year 2022-2023

7 November 2022

Contents

1 Introduction **1**

2 Background **2**
- 2.1 Agile Software Development 2
  - 2.1.1 Scrum 3
- 2.2 Requirements Engineering in ASD 4
  - 2.2.1 Agile RE benefits 5
  - 2.2.2 Agile RE challenges 6
- 2.3 User Stories 6

3 Related Work **7**
- 3.1 US Problems 7
  - 3.1.1 Ambiguity 8
  - 3.1.2 Collaboration 8
  - 3.1.3 System Design 8
- 3.2 Proposed US Solutions 9
  - 3.2.1 Ambiguity 9
  - 3.2.2 Collaboration 10
  - 3.2.3 System Design 10
- 3.3 Research gaps 11

4 Method **11**
- 4.1 Case Study Selection 12
  - 4.1.1 Project description 12
- 4.2 Canonical Action Research 13
- 4.3 Researcher–Client Agreement 13
- 4.4 Cyclical Process Model 14
- 4.5 Diagnosis 14
  - 4.5.1 Observation of work processes 15
  - 4.5.2 Interviews 15
  - 4.5.3 Data Analysis and Triangulation 16

5 Intervention Design **17**
- 5.1 Diagnosis Results 17
  - 5.1.1 The US format is applied inconsistently 17
  - 5.1.2 Issue types are used inconsistently. 18
  - 5.1.3 Minimal use of hierarchical relations 19
  - 5.1.4 Difficulty deciding what to include in a US 19
- 5.2 Action planning 20
- 5.3 Backlog Item Categorization Model 20
  - 5.3.1 Decision Tree 20
  - 5.3.2 Information Sheet 22
  - 5.3.3 Tutorial 22
- 5.4 Action taking 23

6 Evaluation **23**
6.1 Performance . . . 25
6.1.1 Effectiveness . . . 25
6.1.2 Efficiency . . . 25
6.2 Perceived Efficacy . . . 26
6.2.1 Exit Survey . . . 27
6.2.2 Exit Group Interview . . . 27

7 Results **27**
7.1 Effectiveness . . . 28
7.2 Efficiency . . . 29
7.3 Exit Survey . . . 29
7.3.1 Ease of Use . . . 29
7.3.2 Usefulness . . . 30
7.3.3 Intention to Use . . . 30

8 Discussion **30**
8.1 Conclusion . . . 31
8.2 Limitations . . . 32
8.2.1 Construct Validity . . . 32
8.2.2 Internal Validity . . . 32
8.2.3 External Validity . . . 33
8.2.4 Reliability . . . 33
8.3 Future Work . . . 33

A Examples of the diagnosed US issues **39**

B BICM Information Sheet **40**

1 Introduction

Since its introduction over two decades ago, Agile Software Development (ASD) has become an increasingly popular methodology used in the software development industry [47]. ASD is an iterative and incremental approach to software development, characterized by frequent software delivery and close communication within a team and with stakeholders [26]. ASD methodologies, such as Scrum, Kanban and eXtreme Programming (XP), have become popular thanks to their advantage over traditional software development, such as Waterfall, in their ability to better adapt to competitive threats, advances in software development technologies and increasing time-to-market pressures [7, 35]. This is due to ASD's more flexible and iterative processes for identifying and changing requirements, even in later stages of the development process [47].

Within ASD, user stories (USs) are by far the most widely used notation for expressing requirements, with over 90% of practitioners applying the US technique [55, 33]. USs are standardized descriptions of system features, as told from the end user's perspective. USs are especially suitable for ASD because they are expressed using comprehensible, natural language, which helps encourage effective communication within a team and with stakeholders [17, 2]. Moreover, USs help promote a shared understanding of the expected system goals and functions, which is beneficial to monitoring progress and identifying persistent problems [47, 2].

Several studies have contributed evidence that USs are indeed advantageous in improving productivity, software quality and faster delivery [2]. However, studies have also demonstrated that USs are vulnerable to problems to do with ambiguity, such as multiple interpretations and/or failure to capture complete requirements, collaboration and system design [1]. Moreover, a recent study suggests that only 15% of practitioners have a sound understanding of USs, which could suggest that agile software practitioners may not be using USs correctly [44].

Despite this, very few studies have investigated the application of USs in practice. As such, there is little insight into what problems practitioners might face regarding the use of USs and the consequences of these problems for the ASD process and its outcomes [2]. Specifically, most existing research has focused on developing solutions for US problems situated in narrowly defined contexts, failing to consider the multifaceted aspects of applying USs in practice. There is also a lack of validation and evaluation research, as a considerable proportion of proposed US solutions have not been tested in laboratory or real-world settings, which is necessary to facilitate the transfer of research outcomes to practice [2].

The aim of this research is to help address these research gaps by investigating the application of USs in a real-life ASD project, identifying problem areas and testing a possible remedy. Accordingly, the main research question this thesis aims to answer is:

MRQ: *What user story challenges can be observed in a real-life ASD project and how can these be remedied?*

To guide our research, we formulated the following sub-questions:

- RQ1: What types of of US problems have been addressed by the literature?
- RQ2: Which solutions have been proposed by the research community?
- RQ3: Which US challenges can be observed in practice?
- RQ4: How can an intervention be designed to help remedy these issues?
- RQ5: How effective is the intervention?

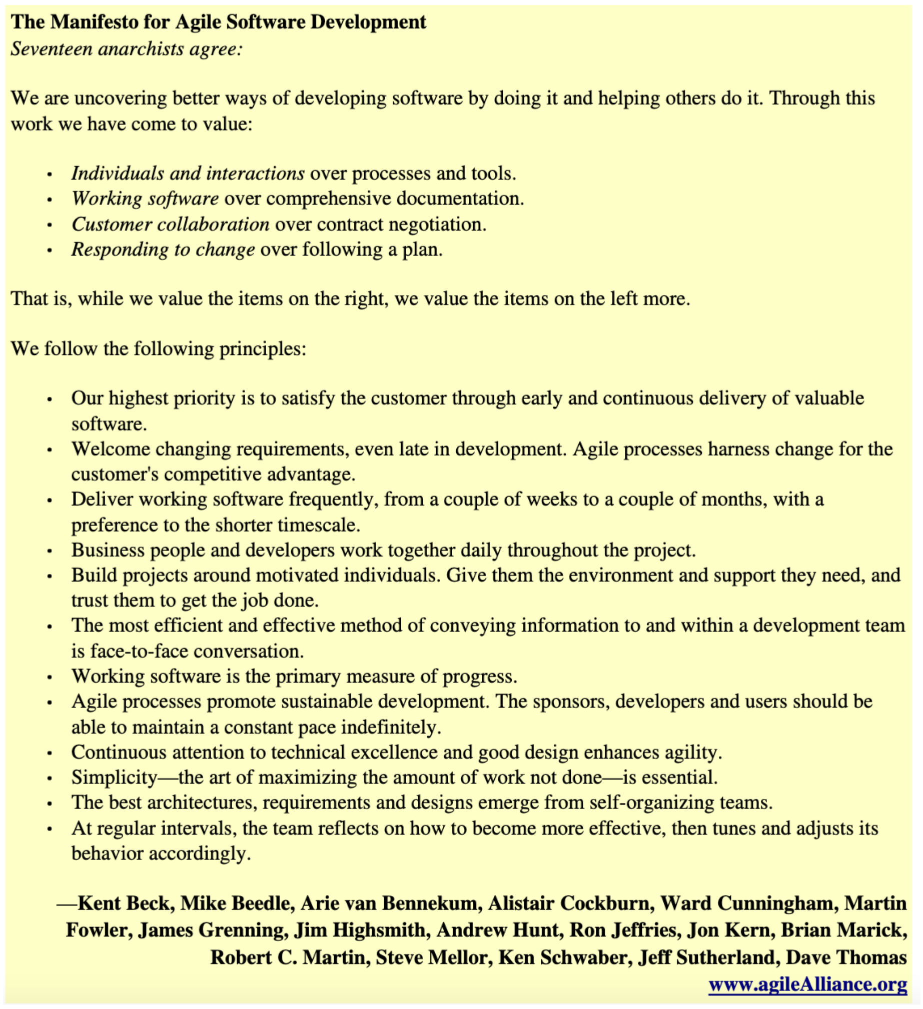
The remainder of this thesis is structured as follows. Section 2 provides background information regarding the key concepts addressed in this thesis. Section 3 discusses related work specific to USs. Section 4 presents the methodology used to investigate our case study. Section 5 presents the diagnosed US issues and describes how a subsequent intervention was planned, designed and implemented. Section 6 describes how the intervention was evaluated. Results of this evaluation are presented in Section 7. Finally, Section 8 presents our main conclusions, discusses threats to validity, and suggests avenues for future research.

2 Background

The purpose of this section is to provide readers with additional information regarding the key concepts in this thesis. First, we review the fundamentals of ASD and provide background on its most popular methodology, Scrum. Then, we discuss the role of requirements engineering in ASD and specifically, USs.

2.1 Agile Software Development

ASD emerged in 2001, when a group of software developers established the 'Agile Software Development Manifesto' (see Figure 1) to bring changes to the software engineering field [26]. The Manifesto describes four core values and twelve associated principles with which to approach software development. Since its publication, a number of ASD methodologies have been based off its values, including: eXtreme Programming (XP), Scrum, Lean Software Development, Feature-Driven Development (FDD), and Crystal Methodologies. Of these, Scrum has become the most popular, with over 80% of practitioners identifying it as the methodology they follow most closely [27]. Based on its predominance in the industry, this thesis chose to investigate an ASD project following the Scrum methodology.



The Manifesto for Agile Software Development
Seventeen anarchists agree:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- *Individuals and interactions* over processes and tools.
- *Working software* over comprehensive documentation.
- *Customer collaboration* over contract negotiation.
- *Responding to change* over following a plan.

That is, while we value the items on the right, we value the items on the left more.

We follow the following principles:

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

—Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas
www.agileAlliance.org

Figure 1: The Agile Manifesto. Source: [26]

2.1.1 Scrum

Instead of adhering to specific software development techniques, Scrum can be described as a lightweight ASD methodology aimed at helping a team generate value through adaptive solutions to complex problems [51]. As such, different processes and techniques can be employed in order to wrap around existing practices. However, Schwaber and Sutherland's guide to Scrum [51] suggests that at the least, Scrum requires a Scrum Master to foster an arrangement where:

1. A Product Owner (PO) orders the work for a complex problem into a

product backlog.

2. The Scrum team turns a selection of the work into an increment of value during a *sprint*.
3. The Scrum team and its stakeholders inspect the results and adjust for the next *sprint*.
4. Repeat.

The *product backlog* is one of the most important artefacts in a Scrum project and can be described as an emergent, ordered list of work needed to improve the system under development [51]. During a sprint planning, part of the product backlog is selected for a sprint backlog and developed during a so-called *sprint*. These sprints are fixed-length events, usually consisting of two weeks, during which the Scrum team works to produce a product increment of value. A Scrum project is thus made up of back-to-back sprints. Scrum also highlights the importance of close communication, which is why daily stand-up meetings are the primary method for communication. Figure 2 shows an example of the typical Scrum process.

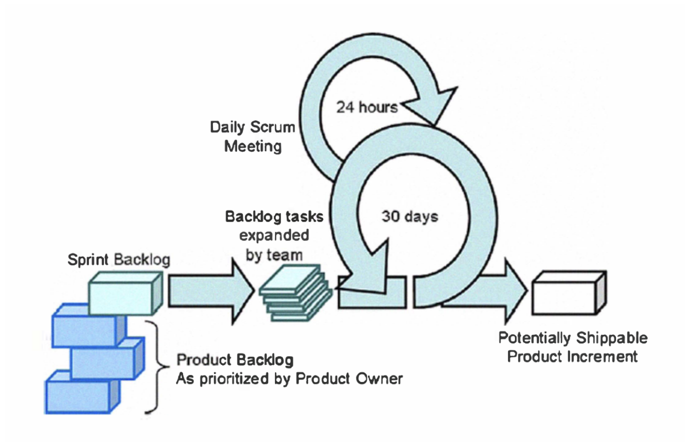

24 hours
Daily Scrum Meeting
Backlog tasks expanded by team
30 days
Sprint Backlog
Product Backlog
As prioritized by Product Owner
Potentially Shippable Product Increment

Figure 2: Example model of a typical Scrum process, a popular form of agile software development. Source: [50]

Requirements engineering in ASD, and thus Scrum, is fundamentally different from traditional software development methods, as we explain in the next section.

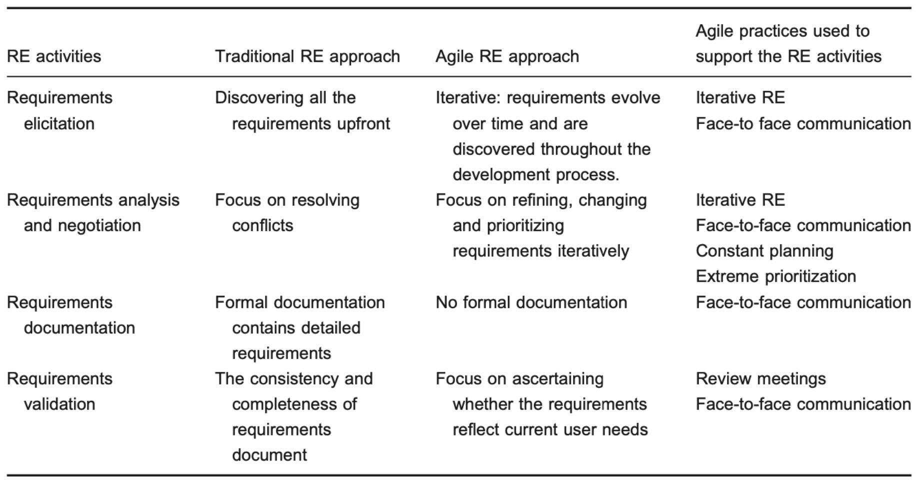
2.2 Requirements Engineering in ASD

Requirements Engineering (RE) is known to play a fundamental role in all sorts of software development processes, since incomplete, inadequate, inconsistent or ambiguous requirements are proven to have a critical impact on the resulting software's quality [53]. Specifically, 80 to 85 percent of project failures can be traced back to incorrect requirements [35].

RE has been defined by van Lamsweerde [53] as "a coordinated set of activities for exploring, evaluating, documenting, consolidating, revising and adapting the objectives, capabilities, qualities, constraints and assumptions that the system-to-be should meet based on problems raised by the system-as-is and opportunities provided by new technologies".

In traditional software development methods, such as Waterfall, the RE process often encompasses a sequential execution of requirements elicitation, analysis, negotiation, documentation and validation [29, 47]. Moreover, requirements are often managed by RE specialists in a phase separate from design and development and based on formal documentation.

In contrast, agile RE is characterized by frequent communication between team members and clients, customer involvement, a gradual defining of requirements and informal documentation [5]. Figure 3 shows an overview of traditional and agile approaches for RE activities.



RE activities	Traditional RE approach	Agile RE approach	Agile practices used to support the RE activities
Requirements elicitation	Discovering all the requirements upfront	Iterative: requirements evolve over time and are discovered throughout the development process.	Iterative RE Face-to face communication
Requirements analysis and negotiation	Focus on resolving conflicts	Focus on refining, changing and prioritizing requirements iteratively	Iterative RE Face-to-face communication Constant planning Extreme prioritization
Requirements documentation	Formal documentation contains detailed requirements	No formal documentation	Face-to-face communication
Requirements validation	The consistency and completeness of requirements document	Focus on ascertaining whether the requirements reflect current user needs	Review meetings Face-to-face communication

Figure 3: Traditional and agile approach for requirements engineering (RE) activities. Source: [47]

2.2.1 Agile RE benefits

Agile RE has been found to resolve several challenges posed by traditional RE [31].

First, communication issues posed by traditional RE are resolved by ASD's frequent face-to-face meetings, collocated teams, onsite customers and cross-functional teams [31, 5].

Second, the problem of overscoping is mitigated by ASD's one, continuous scope flow, gradual detailing and cross-functional teams [5].

Third, problems to do with requirements validation are resolved by requirement prioritisation in every iteration and prototyping [45], which provides customers with a blueprint of the product and therefore helps in validating the requirements [31, 47].

Fourth, problems to do with extensive requirements documentation are resolved by the use of USs, since USs provide to-the-point explanation of user demands [5, 8]. This, combined with frequent face-to-face communication in agile RE helps reduce the need for maintaining long documents.

Finally, the problem of rare customer involvement [8] is resolved by the agile practice of maintaining a close relationship with the customer, which may include having a customer representative onsite and customers playing a role in requirements prioritization, which ensures that customer goals will be met [45].

2.2.2 Agile RE challenges

On the other hand, several challenges have emerged with the agile RE approach [31].

First are problems with cost and schedule estimation, as it is difficult to develop accurate estimates of costs and schedules during early stages of ASD projects that are characterized by unstable problem domains, requirement volatility, and dynamic planning and design phases [47]. Furthermore, estimates are adjusted over time during the development process.

A second challenge has to do with architecture chosen by the development team during the early cycles becoming inappropriate or inadequate as newer requirements become known. Rework of the architecture may add significantly to project cost [47].

A third challenge has to do with the neglect of non-functional requirements. Due to minimal documentation practices and the use of USs, non-functional requirements are at risk of being ill defined or ignored during early development cycles. In ASD, there is a heavy focus on core functionality and issues related to scalability, maintainability, portability, safety or performance are often ignored [47, 31].

A final agile RE challenge concerns customer access and participation, since effectiveness of communication between the customer and team depends on customer availability, customer consensus and customer trust, particularly at the beginning of a project [47].

2.3 User Stories

USs are the most commonly used requirement notation method in ASD [55]. USs are especially suitable for agile RE because they are comprehensible, natural language descriptions of the system's requirements as told from the end-users perspective, which helps promote participatory design and shared understanding between stakeholders and developers [17, 31]. Moreover, USs support and encourage iterative development, as a team may start by defining broader stories, or epics, and disaggregate closer to development time [13].

The formulation of USs has been standardized using templates, the most common being the Connextra template, popularized by Cohn: "As a ⟨role⟩, I want ⟨goal⟩, so that ⟨benefit⟩" [17]. Here, the role describes the (type of) system user who wants the system to achieve or do something, the goal describes the action to be performed by the system in support of the user, and the benefit provides the rationale for this action for the user [2] An example of a US following the Connextra template is as follows:

"As a trainer, I want to delete one of my courses or events, so that I can better support changes in my schedule."

In practice, this template is often complemented with additional details, such as acceptance criteria, notes and effort estimations to help with the implementation of the requirement [17]. Typically, USs become more detailed depending on how close they are to implementation [14].

In this thesis, when we refer to USs, we refer to all elements relating to how practitioners apply the US technique, including the story template as well as any additional details.

3 Related Work

While much research has been devoted to agile RE as a whole, less research has focused specifically on the role of USs [2]. In this section, we discuss some of the published work related to USs, as based on the recently published literature review by Amna and Poels [2], who systematically searched all US studies and mapped them according to a multidimensional classification schema, covering research area, research problem, research outcome, research type and publication type.

In the following sections, we first look at what types of US problems have identified by the literature (RQ1). Then, we look at which kind of research outcomes, or solutions, have been proposed (RQ2). Finally, we highlight the current research gap in US research.

3.1 US Problems

Amna and Poels [2] identified twenty-two unique US issues investigated by the research, which they aggregated into three problem classes, namely: *ambiguity*, *collaboration* and *system design*. In the following sections, we elaborate on each of these problem classes and discuss some examples of specific issues US related to them.

3.1.1 Ambiguity

Roughly a quarter of all US studies have investigated problems related to ambiguity, which [1] define as: "problems related to the articulation of requirements as USs, which cause doubtful, vague, and multiple interpretations of these requirements". These problems may be caused by different uses of language to express requirements, limitations of the user story template, and/or differences in domain knowledge and experience. As such, four sub-classes relating to the problem of ambiguity have been identified: *vagueness* of requirements articulated as USs *inconsistencies* between USs, *insufficiency* of USs in capturing requirements, and *duplication* of requirements in USs [1].

Vagueness is a problem that occurs primarily during requirements elicitation and documentation activities and is a consequence of USs being a natural language RE artifact [1]. Vagueness may be attributed to unclear word choices in USs or the absence of a particular scope or reference that helps someone interpret a US.

Problems to do with inconsistency occur within a set of related USs if interpretations conflict, which may result in requirements being incomplete and systems being non-compliant [1]. Here, non-standard use of vocabulary has been identified as a source of inconsistency.

When a software system is large, complex or hardware-dependent, USs may not convey enough information for software design. This insufficiency of the US format may lead to the need for separate system requirements, or result in non-functional requirements being ignored [29].

Finally, in large-scale projects, duplicate USs may arise. This could be due to a large number of USs being generated, a lack of communication among team members or due to the speed of development imposed by the Scrum process. This is considered an issue due to the risk of unnecessary rework [1, 3].

3.1.2 Collaboration

Another quarter of studies are concerned with collaboration problems, which refer to a lack of effective collaboration within a team and/or between project stakeholders which can be traced back to the use of USs. Specifically, these studies have investigated the role of USs in human interaction during software development, with a focus on any shortcomings that USs may have in facilitating communication and collaboration [2].

Some examples of issues that fall under this problem class include: communication issues during US elicitation, project planning based on USs, and US estimation and prioritization conflicts [1].

3.1.3 System Design

Studies related to system design have focused on problems related to the impact that USs have on the quality of the system and its development.

Here, the main issues investigated are project management problems with resource estimation, planning, prioritization, and other types of analysis based on

USs. Studies have also addressed shortcomings of the US technique in capturing security/privacy constraints and non-functional requirements.

3.2 Proposed US Solutions

With regard to outcomes of US studies, Amna and Poels [2] found that most are solution-oriented (80%), with only a small part of the studies focusing on problem investigation or the explanation of observed phenomena (20%).

Furthermore research outcomes of the 186 US papers could be grouped according to six solution classes, namely: description, explanation, algorithm, model, prototype and framework.

In the following subsection, we provide some examples of solutions proposed by US research for each of the previously defined problem areas.

3.2.1 Ambiguity

To help mitigate problems to do with ambiguity, several different templates for writing USs have been formulated over the years. It is believed that despite Cohn's [17] 'original' US template addressing the WHO, WHAT and WHY, many practitioners still suffered from a lack of guidance in how to write effective USs, resulting in practitioners proposing their own solutions, causing numerous formal and informal US templates to appear [56]. For this reason, Wautelet *et al.* [56] listed and classified all existing US templates and proposed a unified model coupled with precise semantics with the goal to reduce communication issues between project stakeholders and simultaneously enhance the scalability of agile projects.

In addition to templates, several guidelines have been proposed for writing high-quality USs, such as the most well-known *INVEST* (Independent-Negotiable-Valuable-Estimable-Scalable-Testable) framework [54] and generic guidelines for ensuring quality in agile RE by Heck and Zaidman [28]. More recently, Lucassen et al., [39] proposed the Quality User Story (QUS) framework, which consists of 14 quality criteria that USs should strive to conform to [39].

A few recent studies have also explored the use of (automated) tools to help improve the quality of USs. For example, Dalpiaz *et al.*[19] tested an approach that combines Natural Language Processing (NLP) with Information Visualization (InfoVis) to help analysts automatically identify near-synonymy, homonymy and missing requirements in a list of USs.

There are also studies that have proposed the use of an algorithmic solution based on similarity metrics such as Cosine Similarity and Jaccard Index in order to help tackle the US duplication problems [3].

Moreover, to help USs convey more information, some studies have suggested extensions to the standard US technique consisting of a title, acceptance criteria and additional details [17]. For example, 'delivery stories', which complement USs with technical implications, effort estimation and associated risk [20], which is expected to help with requirement prioritization in larger projects.

Another example includes UserX Stories, which help remedy difficulties encountered by practitioners of incorporating UX aspects in software development [10]. Similarly, Moreno *et al.* [43] introduced the concept of usability stories, which address usability requirements related to a particular US. Finally, some studies have argued for the addition of personas to encourages a more in-depth understanding of the mental, emotional, and physical states of users interacting with the software system [32, 37].

3.2.2 Collaboration

Regarding problems to do with collaboration, some US studies have proposed solutions to help practitioners with requirements elicitation. Menkveld *et al.,* [41] have investigated crowd-based requirements elicitation, which promotes the active involvement of a large number of stakeholders in RE activities. Specifically, they proposed the CREUS method (i.e., Crowd-based Requirements Elicitation with User Stories). However, they found that CREUS cannot replace other elicitation techniques, not only because of the limited focus on quality concerns, but also because most ideas need substantial refinement.

Moreover, Lombriser *et al.,* [36] built an online gamified platform for requirements elicitation to help improve stakeholder engagement and ultimately performance in RE. Evaluation of their gamified requirements engineering model (GREM) provides promising initial empirical insights and suggests that competitive game elements may be advantageous for US elicitation.

Regarding US prioritization, Hudda *et al.* [30] proposed an approach that considers criteria from both the client's side as well as the developer's side, where previous methods considered criteria either from client side or developer side. However, their suggested approach requires continuous involvement of multiple clients, which is not always feasible.

3.2.3 System Design

First of all, many studies classified as being related to system design investigate RE activities that critically depend on high-quality US [2]. As such, some of the previously mentioned work proposing solutions to help remedy ambiguity are relevant to this problem area as well.

Some studies related to this problem area have suggested the use of conceptual models in the process of developing USs. Conceptual models are visual representations commonly used for understanding the domain of business functions and communicating with the stakeholders [18]. Conceptual models can be employed in requirements engineering to provide an overview for team members to understand the product domain, identify quasi-synonyms that may lead to misunderstandings, support model-driven engineering and analyze certain quality aspects, such as security and privacy [18].

Bragilovski *et al.* studied the process of deriving conceptual models from USs [6] by examining whether providing guidelines had an effect on the ability of humans to derive complete and valid conceptual models, which they tested in a two-factor, two-treatment controlled experiment with undergraduate students. Their results indicate that their guidelines improve the completeness and validity of the conceptual models in cases of medium complexity.

Moreover, Lucassen *et al.* [38] proposed the Visual Narrator, an automated approach based on natural language processing that extracts conceptual models from US requirements. Their evaluation showed positive accuracy results when USs are concise statements of the problem to solve and not lengthy descriptions of the solution.

3.3 Research gaps

Based on the related work reviewed for this thesis, several research gaps can be identified.

First, the number of studies that have investigated the application of USs in practice is relatively small, which is illustrated by the fact that most US research is solution-oriented (80%), with only a small part of the studies focusing on problem investigation or the explanation of observed phenomena (20%). Specifically, there is a technical focus in US research, meaning that more studies have looked into how to improve the user story technique than into understanding how to make better use of USs in RE activities [2]. As such, we believe there is a need to investigate the application of USs in practice to get insight into which issues regarding the application of US are actually being experienced by practitioners, which will help increase the practical relevance of research outcomes.

Moreover, there is a lack of validation and evaluation for US research, meaning that a considerable proportion of proposed solutions have not been tested in laboratory or real-world settings. [1]. As such, it is unknown for many of the proposed solutions what their effect is on quality of the RE process in ASD projects. We argue that there is a need for researchers to test their proposed solutions in a real-life context in order to transfer research results and increase its relevance.

4 Method

To gain insight into the challenges regarding the use of USs in practice and to test possible interventions, this research investigated a real-life ASD project applying the US format in their requirement notations.

Our case study followed the principles of Canonical Action Research (CAR) as proposed by Davidson [22]. This particular design was chosen based on this research's objective to obtain a intimate, holistic and detailed view on how USs are applied in a real-life, organizational context, as well as to help the organization improve their current situation.

In the following sections, we first discuss our procedure for selecting a case study and describe the chosen ASD project. Then, we elaborate on the principles of CAR and explain how they were used to guide the different phases of this research.

4.1 Case Study Selection

This thesis research was being hosted by KPMG Digital Enablement (DE), the software development arm of KPMG that helps clients with areas of digital solutions, ranging from advice and audit to design and development [23].

To ease accessibility to resources and to encourage cooperation and close-communication, purposive sampling was used to look among KPMG DE's ongoing projects for a potential case study. It should be noted, however, that external projects were regarded as a viable possibility should no suitable project be found within KPMG DE.

The goal was to find an active ASD project adhering to the Scrum framework, which self-reportedly made use of the US format in their requirement notations. Based on these requirements, two projects were considered as potential case studies. Ultimately, a collaborative project between Kennisnet and KPMG DE was selected due to its bigger team size and more traditional adherence to the Scrum framework.

4.1.1 Project description

The chosen project concerned the development of software product called "Entree Federatie" (EF). The project was a collaboration between KPMG DE and Kennisnet, a Dutch public organization that provides national ICT-infrastructure to the educational sector [25]. The goal of EF was to allow students and teachers easy access to educational services through so-called 'single sign on', obscuring the need of having to create different accounts for different educational services. The user-facing application used to enable this service was called 'Mijn Entree Federatie' (MEF).

The EF team consisted of eight members total (7M, 1F), the roles of which constituted a PO/technical specialist/Scrum Master whose main responsibility was maintaining the product backlog and facilitating team sessions, a Product Manager who kept in contact with stakeholders and ensured that the team kept working towards valuable increments, a Systems Architect concerned with how the requirements ought to be technically implemented, two developers who implemented the requirements, two UI/UX designers and a DevOps engineer.

The project closely adhered to the Scrum framework as shown in Figure 2. Each of their 2-week sprints started with a sprint planning, during which the PO presented the sprint backlog containing items prioritized for that particular sprint. During this session, team members also provided feedback and items were refined (i.e., adding additional notes/acceptance criteria) where necessary. The sprint planning also included an effort estimation activity, during which the team members assigned story points to each item using planning poker. At the end of the sprint planning, the team performed a sprint retrospective for the previous sprint, in which they noted highs/lows and gave the previous sprint an overall rating based on each team member's grade.

After this, the team began their work of implementing the backlog items selected for that sprint. The rest of the sprint was characterized by daily stand-up meetings.

The tools used to facilitate this project included Jira Software for project management, MS Teams for video conferencing, Miro for sprint retrospectives and Scrum Poker for effort estimation.

4.2 Canonical Action Research

Since the goal of this research was to investigate USs and test possible remedies in a real-life, organizational context, the methodology applied in this thesis was guided by the principles of canonical action research (CAR), as proposed by Davison [22].

Action Research (AR) is a case study research methodology aimed at diagnosing and solving organizational problems through intervention while at the same time contributing to scientific knowledge [22]. Its application has been especially dominant in the information systems (IS) domain, with numerous publications having made theoretical and applied contributions. However, AR has been criticized for its lack of guidelines and rigor [11]. Davison *et al.* [22] addressed these concerns by proposing a set of five principles and associated criteria to help assure the rigor and the relevance of Canonical Action Research (CAR), one of the most widely practised forms of AR.

CAR differentiates itself from other forms of AR due to its iterative and collaborative nature, which focuses on organizational development as well as the generation of knowledge [22]. The iterative process consists of carefully planned and executed cycles of activities, aimed at gaining an intimate view of a problem situation as well as to navigate possible solutions.

The five principles and associated criteria suggested by Davison [22] to improve the rigor and relevance of CAR are as follows:

1. Principle of the Researcher–Client Agreement (RCA);
2. Principle of the Cyclical Process Model (CPM);
3. Principle of Theory;
4. Principle of Change through Action;
5. Principle of Learning through Reflection.

The following sections elaborate on these principles and how they were applied in context of the current research.

4.3 Researcher–Client Agreement

According to CAR's first principle of Researcher-Client Agreement (RCA), it is necessary that client and researcher reach an agreement regarding mutual guarantees of behavior. A well-constructed RCA should provide a solid basis for building trust as well as promote a shared understanding of how CAR works and what its benefits and drawbacks may be for the organization [22]. Moreover, the researcher should share their goals, actions, possible interventions and outcomes in order to help promote a spirit of shared inquiry.

To this end, a presentation describing the current research's focus, motivation, intended actions, expected roles and responsibilities and data collection method was given to the EF team at the start of the case study. Additionally, an information sheet and corresponding consent form were provided to and signed by the project's PO, making explicit commitment to the project.

4.4 Cyclical Process Model

Following the RCA, our case study commenced. This phase was based on the Cyclical Process Model (CPM) [22], which describes a cycle consisting of five stages that should be executed in sequential fashion, namely (1) diagnosis (2) action planning (3) action taking (4) evaluation (5) reflection. Figure 4 shows the CAR process model.

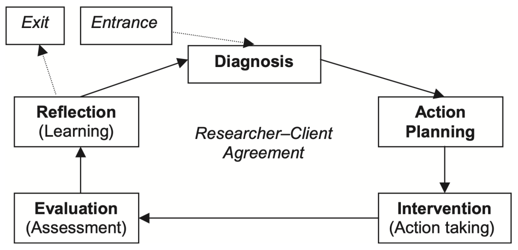


Figure 4: CAR process model. Source: [22]

In the following sections, we first describes our procedure for diagnosing potential US issues. For reasons to do with legibility, the list of US issues resulting from this diagnosis is provided in Section 5, along with the subsequent action planning and action taking. Evaluation is discussed in Section 6.

Typically, the reflection phase of the CPM is used to determine whether or not another research cycle repeating these steps is required. Due to time constraints, however, the current research could only complete one cycle.

4.5 Diagnosis

The aim of our diagnosis phase was twofold, (1) to obtain a thorough understanding of the team's current work processes and (2) to identify potential problem areas regarding their use of the US technique. As such, the following steps were formulated:

1. Observe work processes and artefacts related to the use of USs;
2. Interview team members;
3. Analyze data and triangulate;

These particular steps were chosen because they allowed for a holistic view on the situation, since this research did not assume a certain context of use of the US technique. Moreover, these steps considered both the team's view on the situation, as well as an independent diagnosis to find problems, confirm the nature of problems and/or determine their causes [22].

4.5.1 Observation of work processes

Before being able to identify problem areas, it was necessary to gain a thorough understanding of not only the product under development, but also the team's work processes, environment, tools and artefacts used. Since the Manifesto for Agile Software Development states that *"individuals and interactions should be valued over processes and tools"* [26] it was important to ensure that the EF team satisfied the core agile values, since this would help justify looking into ways of improving their documentation practices.

Gaining a thorough understanding of the project was done through means of several, informal talks with different team members, which were repeated until no new relevant information was gained.

After this, focus shifted to observing the team's application of USs. This included attending refinement sessions, sprint plannings and daily stand-up meetings, as well as viewing project artefacts, such as the product backlog. During the observed sessions, the team was instructed to continue their work as if the researcher was not present. To this effect, the researcher's microphone and camera were turned off. All off the attended sessions were held online using MS Teams and, with the participants' permission, were recorded for later reference. Moreover, approximately 100 backlog items from different sprints were randomly selected as a sample to be manually inspected by the researcher.

4.5.2 Interviews

Five EF team members were interviewed regarding their experience with USs, namely the Product Manager, Product Owner, one UI/UX designer and two developers. These participants were selected due to their self-proclaimed involvement with USs.

The interview technique was selected to give participants the chance to individually voice their experiences with USs. Moreover, interviews were considered a more suitable approach than other survey methods considering the small number of participants and the need for a rich investigation of content [34].

At the start of each interview, the research focus was explained and informed consent was gathered. Moreover, participants were reassured that the interview did not mean to test their knowledge about USs, or criticize how USs were used within their project. This was done to minimize the risk of experimenter demand.

The interviews were semi-structured, meaning that a set of basic questions was used to guide the interview, but that different follow-up questions were asked based on the participants' responses. Generally, the first part on the interview discussed each participant's role in the project, their experience with USs and the team's current application of USs, while the second part included questions more targeted towards US problem areas.

Interviews were conducted online via MS Teams and lasted half an hour on average. All interviews were audio-recorded and transcribed by the researcher. Original interview transcripts were in Dutch, meaning that any quotes in this thesis have been translated to English by the researcher for the purpose of legibility.

4.5.3 Data Analysis and Triangulation

During this step, the different types of collected data were analyzed with the goal of producing a list containing relevant US-related issues identified for this particular project.

For the interviews, a form of grounded theory (GT) [9] was used since it is considered especially well-suited for situations in which the researcher does not have preconceived ideas. Analysis of the transcripts consisted of open, axial, and selective coding. During open coding, groups of data were identified and labeled as either 'providing project context', 'current US practices', 'US issues' or 'otherwise relevant information'. During axial coding, pieces of text relating to the same code were clustered to search for similarities and differences. Finally, selective coding was used to further group data relating to US issues and formulate a set of final categories, which thus constituted US issues as uncovered through interviews.

With regard to data from observations, all of the sampled backlog items were manually inspected by the researcher and potential deviations from good practice were highlighted. This process was informed by the researcher's subjective knowledge regarding the application of US as based on the literature review performed in Section 3 as well as on gray literature, such as blogposts by US author Mike Cohn [12, 14, 13, 15] and US tutorials by Atlassian [57, 48]. The US issues identified during this inspection were also summarized under different categories an approach similar to that of the interview analysis.

Next, the list of US issues as based on interview data was triangulated with observations made in the product backlog and recorded sessions. For example, to validate the US issue "participants have difficulty deciding what to include in USs", which emerged from the interview, we looked at the product backlog and observed that there were indeed inconsistencies with regard to which information was included in USs. This was done to minimize the risk of experimenter demand and to add robustness to the identified US issues.

Based on this final step, our list containing relevant US issues was formulated. With this, we mean that the US issues formulated in this thesis were narrowed down to those believed to be worth remedying, as based on US literature.

The identified US issues are presented and discussed in Section 5.1.

5 Intervention Design

This next research phase pertained to the essential principle of CAR, which is to take action in order to change the current situation and its unsatisfactory conditions [22]. This step is characterized by an action planning phase based explicitly on the results from the diagnosis, followed by an implementation of the planned action and subsequent evaluation thereof.

As such, we first present the results from our problem diagnosis. Then, we discuss how an intervention was designed and implemented to help remedy the identified US issues.

Evaluation of the intervention is discussed in Section 6.

5.1 Diagnosis Results

The US-related issues diagnosed for the ASD project under investigation are summarized in Table 1. The following subsections describe our reasoning behind diagnosing each of these US issues.

Table 1: US-related issues identified for the EF project

Issue no.	**Issue description**
1	The US format is applied inconsistently.
2	Issue types are used inconsistently.
3	Limited use of requirements' hierarchical relations.
4	Difficulty deciding what to include in backlog items and when.

5.1.1 The US format is applied inconsistently

First of all, only a relatively small part of EF's Product Backlog consisted of 'traditional' USs following the "As a ⟨role⟩, I want ⟨goal⟩, so that ⟨benefit⟩" US template [17]. We believe this was due to the fact that the product under development was quite technical in nature, meaning there was a heavy focus on the back-end processes necessary to run the relatively small, user-facing front-end application. As a results, a large part of the requirements formulated by the team did not directly involve any end-users. Considering the fact that USs describe requirements from the end-user's perspective, the team stated that it felt "silly" to write these requirements in the US template, as illustrated by the following quote: *P1:"A large part of our system operates in the background, and for a lot of these processes, we actually don't want the user to notice that they are happening, and well: "as a user, I don't want to experience something" is not a very nice story, haha".*

Scrum alliance founder Mike Cohn acknowledges that not everything on a Scrum product backlog needs to be expressed using the US template [15]. However, unlike USs, there are no clear guidelines as to how to treat these other, non user-facing backlog items, such as technical work and/or knowledge acquisition [16]. This is illustrated by the following quote: *P3: "I think it would help if we also had a template for those other types of backlog items. You obviously have those very clear front-end stories that can follow the "as a...I want...so that..." template, which is very clear for those specific cases, but for more technical stories, I think it would help if we also had a template for those"*. As such, the team generally opted for a description of the work at hand accompanied by acceptance criteria. The following quote illustrates this: *P3: "We have quite a lot of technical items, for which you can simply use a few words to write down "this should work that way", and that also is what we do. In that regard, we have a lot of different backlog items, and definitely not all USs"*.

This, in itself, does not necessarily pose a problem, as one participant stated: *P3: "We have the luxury of being a very mature team that has worked together for a long time. Sometimes, even just half a word is enough to understand each other. There is also a lot of trust and open communication, which I think is very important within a team. If there are any uncertainties, people can immediately ask and resolve them"*.

However, we believe that the team's lack of guidance regarding what to include in non-USs has caused their product backlog to become unstructured, resulting in inconsistent use of the US-format when it should be applied (see Appendix A, Figure 9).

We consider this an issue for several reasons. First, although the team's maturity and close communication may obviate the need for more formal documentation and structure in their product backlog, Ramesh et al.,[47] argue that when there is a breakdown in communication caused by a variety of problems, including turnover of personnel, rapid changes to requirements and/or growing complexity of the system-under-design, the lack of consistent documentation may cause a variety of problems, including the ability to induct new members into the development team.

Second, by not applying the US template consistently, it is likely that the team is not able to enjoy its proven benefits [12].

Finally, use of a standardized structure, such as templates, has been found to improve overall work productivity and quality [40].

5.1.2 Issue types are used inconsistently.

Related to the previous issue, we observed that the team was inconsistent in their use of issue types.

Issue types can be used in requirement management tools such as Jira, Trello or Github to help users identify, categorize, and report different types of work in the product backlog. Use of these issue types also enables users to link requirements and make use of hierarchical levels, namely [57]:

- *Epic issues* represent high-level initiatives or significant deliverables that can have standard issues as child relations.

- *Standard issues* represent daily work. This may include Stories, Tasks and Bugs, but a team may also introduce their own standard issue types. These can have subtasks as child relations.
- *Subtask issues* help a team break standard issues into even smaller units of work. This is especially helpful if an issue requires multiple people working on it, or if a team underestimates the scope or complexity of their work.

The EF team had added Discussion and Improvement as standard issue types. However, they did not appear to have any clear system in place as to which issue types to assign to which pieces of work (see Appendix A, Figure 10).

We consider this an issue because we believe inconsistency in this area may also contribute to the US format being applied inconsistently and introduce an overall lack of structure in the product backlog.

5.1.3 Minimal use of hierarchical relations

Related to the previous issues, we observed that the team made little use of hierarchical levels (i.e., linking requirements using parent-child relations). The majority of the product backlog was made up of standard issue types (i.e., stories, bugs, tasks) linked to the same, large epic which represented the product as a whole (see Appendix A, Figure 11).

We consider this an issue due to its negative effect on requirements traceability, which has been recognised as an important factor for supporting various activities in the software system development process [52].

5.1.4 Difficulty deciding what to include in a US

This particular issue refers to team members' uncertainty as to which information to include in USs (and other issue types) and when. The following quote illustrates this: *P3: "Sometimes I have difficulty deciding what/what not to include in a US. Too much detail leads to really big USs that are difficult to read and discuss, but you also don't want them so concise that they lack important information"*.

Indeed, some USs observed in the product backlog were extremely large and detailed, containing the US description itself, along with elaborate background information, attachments, instruction on how to implement the story and acceptance criteria, (see Appendix A, Figure 12)

According to Cohn [14] and general US guidelines [54], the formulation of USs should become more detailed closer to implementation. When excessive detail is included, time spent adding the unnecessary detail is wasted, or developers may feel artificially constrained by the excessive detail. On the other hand, too little detail might cause confusion among team member as to what to build, causing team members to waste time asking questions or implement the wrong features.

5.2 Action planning

Next, potential interventions to help the team mitigate these identified US issues were explored.

We first looked at existing US literature, since interventions for which previous research has already provided valid theory are considered particularly appropriate [22]. As such, we referred to the literature mapping performed by Amna et al., [2], which provides an overview of all proposed US solutions. However, considering the research gap that motivated this study, we were unable to find any proposed solutions that were directly aligned with the specific US issues identified in this study. To the best of our knowledge, only issue no. 4: *difficulty deciding what to include in a US* is a problem for which several sources provide theoretical guidelines [54]. Therefore, we decided to design our own artefact to help mitigate issues no 1, 2 and 3.

Factors that were taken into consideration during this planning phase were that the intervention should not disrupt work processes and be feasible with regard to time, resources and scope. The following section explains our chosen intervention and motivation behind its design in more detail.

5.3 Backlog Item Categorization Model

The artefact created to help the team mitigate the diagnosed US issues is called the *Backlog Item Categorization Model (BICM)*, a theoretical framework consisting of (1) a decision tree (see Figure 5), and (2) and corresponding information sheet (see Appendix B, Figure 13).

The BICM helps inform the process of creating new product backlog items and/or modifying existing ones by having practitioners answer a few yes/no questions regarding the item at hand, which the decision tree helps categorize as belonging to one of six issue types. The corresponding information sheet provides additional details regarding these issue types, including a general description, suggested format, which information to include and its place within the hierarchical levels of the product backlog.

The motivation behind this design is that the decision tree ensures that similar backlog items are always treated similarly (e.g., new, user-facing features are always described using USs), thus helping mitigate issues no. 1: *The US format is applied inconsistently.* and no. 2: *Issue types are used inconsistently.* The information sheet is more aimed at issues no. 3: *minimal use of hierarchical relations.*

5.3.1 Decision Tree

To develop the decision tree, we first needed to decide which issue types to include as end points. To do so, we first identified the team's current use of issue types through means of observation. Then, we referred to online resources on the topic to identify good practices related to product backlog management, which included blog posts by Mike Cohn [16, 15, 13] and tutorials provided by

Atlassian [57]. Taking into account both the theory provided on the topic and the observations made in practice, we defined a set of six issue types we believed would address all of the team's needs, namely: Enabling Task, Epic, User Story, Feature-Driven Development (FDD) Story, Bug and Improvement.

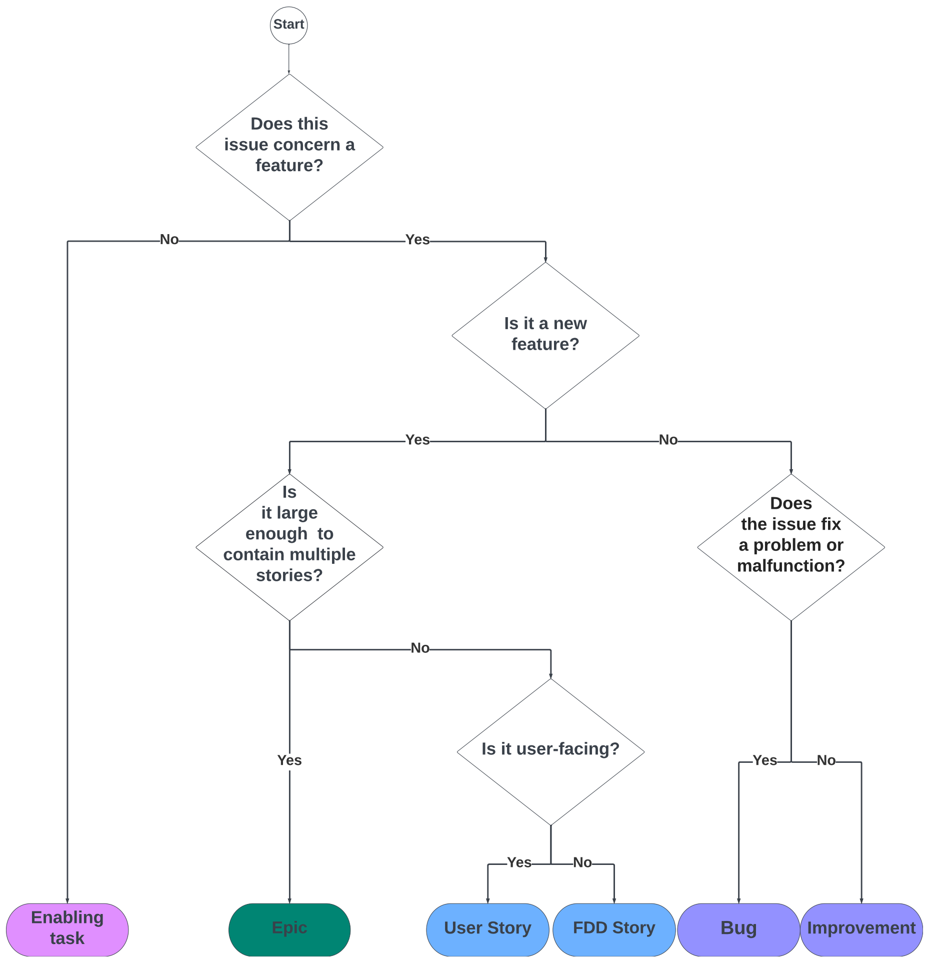

Start
Does this issue concern a feature?
No
Yes
Is it a new feature?
Yes
No
Is it large enough to contain multiple stories?
Does the issue fix a problem or malfunction?
No
Yes
Is it user-facing?
Yes
No
Yes
No
Enabling task
Epic
User Story
FDD Story
Bug
Improvement

Figure 5: Decision tree component of the Backlog Item Categorization Model (BICM).

Enabling Task. This category was included to acknowledge the fact that although teams primarily use the product backlog as a prioritized features list, it is often also used as a tool to manage other actions needed to support product development [15], which was the case for the EF team as well. Examples of enablers that may fall under this category are as follows [24]:

- Exploration enablers to denote research, prototyping, exploration of prospective solutions and evaluating alternatives.

- Architectural enablers to build the architectural runway, allowing for smoother and faster development.
- Infrastructure enablers to build, enhance, and automate the development, testing, and deployment environments, facilitating faster development, higher-quality testing, and a faster continuous delivery pipeline.
- Compliance enablers to managing specific compliance activities, including documentation, sign-offs, regulatory submissions and approvals.

Epic. This issue type was included to represent high-level initiatives or significant deliverables that are large enough to contain multiple stories.

User Story. This issue type was included to represent user-facing requirements that are not large enough to contain multiple stories.

FDD Story. This issue type was included to represent non user-facing requirements that are not large enough to contain multiple stories.

Bug. This issue type refers to those issue types that fix a problem or malfunction with an existing feature.

Improvement. This issue type refers to those issue types that improve an existing feature.

The goal was then find a minimum number of questions necessary to distinguish between these different issue types.

Note that themes and subtasks were intentionally excluded as issue types in the decision tree, as we considered themes to be top-level objectives which are not in themselves represented in the product backlog, while subtasks are considered the most granular pieces of work belonging to USs, FDD-stories, improvements and bugs, rather than being stand-alone backlog items.

5.3.2 Information Sheet

The second BICM component consists of an information sheet providing additional information regarding the issue types suggested by the decision tree (see Appendix B, Figure 13). The information sheet was designed to help promote an understanding of the hierarchy between requirements in the backlog, as well as provide guidelines for writing different product backlog issue types, such as which format to follow. The design of the information sheet was based on a combination of sources, including blog posts by Cohn and tutorials by Atlassian [15, 16, 57]

5.3.3 Tutorial

Considering the nature of the problems for this particular team, we decided to organize a tutorial session. This was motivated by the idea that a tutorial would enable us to both touch upon US theory as well as introduce the BICM. Moreover, we believed that for this pilot study it was beneficial to provide the team with the motivation behind the BICM's design, as we expected this would promote a deeper understanding of the artefact, which could prove useful in its evaluation.

The tutorial session lasted approximately 40 minutes and consisted of two parts. In the first part, we provided a recap of the current research (i.e., motivation) and its diagnosis results. Then, we discussed the US literature and theories used to inspire the BICM. Specifically, we explained that in practice, not everything on the Product Backlog needs to be a US and that the Product Backlog is generally made up of two categories: features and enablers, in which Epics, USs, FDD-stories, Bugs and Improvements constitute the features, while the supporting activities are considered enablers. Since this was the first time the term 'FDD-story' was introduced, we dedicated special attention to its suggested application and format. Next, we highlighted the importance of requirements traceability, which is promoted through the use of hierarchical levels. We then introduced the BICM and talked through its different components in relation to the previously mentioned theory.

The second part of the tutorial briefly referred to the INVEST- and QUS framework [54, 39] to help the team with the issue regarding US size. This part of the tutorial was omitted from evaluation.

The tutorial concluded with instructions regarding the subsequent experimental period. These instructions were purposefully kept simple, requesting only that the team consult the BICM in their work (i.e., when creating new backlog items or managing existing ones) from that point onward.

5.4 Action taking

Upon completing the tutorial session, each team member was sent a high-resolution image of the BICM, which they were requested to print out. The team then entered a 3-week experimental period in which they were instructed to consult the BICM in their work. The duration of this experimental period was determined by time constraints. Ideally, the experimental period would have spanned multiple sprints in order to collect more robust measures for the subsequent evaluation.

6 Evaluation

To evaluate the effect of introducing the BICM, we took inspiration from the Method Evaluation Model (MEM) [42], a theoretical model for validating information systems (IS) design methods (see Figure 6). This choice was inspired by the recent work by Berends and Dalpiaz [4], who also used the MEM as basis for their evaluation.

The MEM's authors argue that when evaluating the success of an IS method, both its Actual Efficacy (i.e., whether the method improves performance of the task) as well as its Perceived Efficacy (i.e., whether the method is perceived as being useful) need to be considered. The reason for this is that a method which improves performance but that is not used will have no effect on practices.

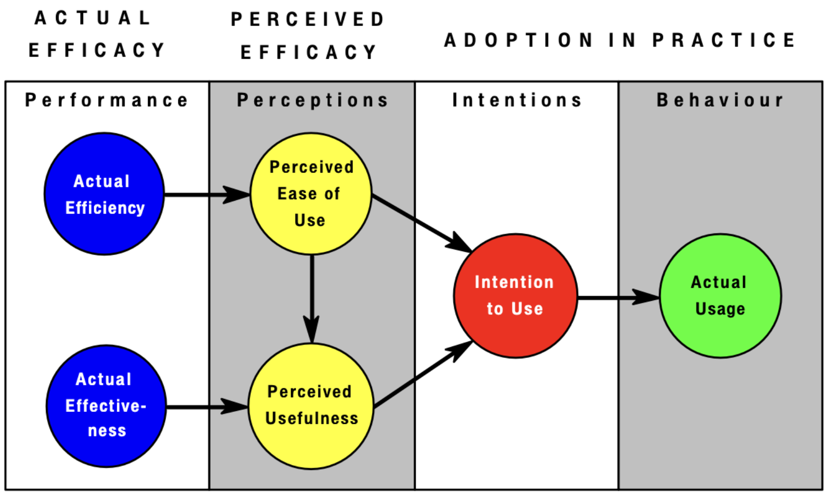


Figure 6: Method Evaluation Model used to guide our evaluation. Source: [42]

Similarly, a method which is used, but that does not improve performance is equally ineffective [42].

To evaluate a method's performance, the authors therefore suggest looking at measures related to efficiency (i.e., the effort required to complete a task) as well as measures related to effectiveness (i.e., the degree to which a method achieves its objectives). Regarding users' perception of a method, the authors suggest looking at its ease of use (i.e., the degree to which a person believes that using a method would cost effort), usefulness (i.e., the degree to which a person believes a particular method will be effective), users' intention to use and, if possible, actual usage.

There are no prescribed measures corresponding to each of these constructs, as application of the MEM may differ per method being evaluated. However, general guidelines dictate that efficiency be determined by a variety of input measures such as time, cost or effort, whilst effectiveness may be determined by evaluating the quantity and/or quality of a method's output [21]. Moreover, survey items from the Technology Acceptance Model (TAM) may be used as a basis for measuring perceived ease of use, usefulness and intention to use [21].

Due to our action research taking place in a real-life context with limited sample size and limited experimental control, our evaluation of the BICM is qualitative in nature. To strengthen our findings, we used a triangulation of methods, including observation, survey and interview.

The following sections describe how each of the MEM's constructs were considered in more detail.

6.1 Performance

To evaluate the BICM's performance, we considered its effectiveness and efficiency [42].

6.1.1 Effectiveness

With regard to effectiveness, we evaluated the degree to which the BICM achieved the following three objectives, chosen due to their direct alignment with the issues diagnosed in Section 5.1:

- O1:*"reduces inconsistent use of the US format" (issue no.1)*
- O2:*"reduces inconsistent use of Issue Types" (issue no.2)*
- O3:*"increases the use of hierarchical levels" (issue no.3)*

Here, one would have ideally taken samples from both the pre- and post-intervention product backlog and compared the number of violations relating to each dependent variable in a quantitative fashion. However, due to this study's short intervention period and thus small post-intervention sample, we did not consider this approach to be reliable. Instead, we relied on the researcher's subjective evaluation of the post-intervention product backlog, triangulated with interview data to draw tentative conclusions regarding the degree to which each objective was achieved.

6.1.2 Efficiency

Due to this study's short intervention period, along with the intervention taking place in a real-life environment with differing sprints and uncontrollable external factors, we could not reliably perform any quantitative measures with regard to time, rework and/or frequency of communication to determine the BICM's efficiency. We therefore chose to replace actual efficiency with perceived efficiency, thus measuring this construct using subjective participant data. This was done through means of the exit interview and by including a survey item asking team members to rate how the period after the intervention compared to the period before the intervention with regard to:

- time spent maintaining the Product Backlog
- team productivity

These items were measured using a 5-point Likert scale with opposing statements format and the additional option of 'Can't say'. The items were preceded by a short text reminding participants to consider the impact of the BICM, as opposed to external factors which may have influenced these variables during the intervention period.

6.2 Perceived Efficacy

To evaluate the BICM's perceived efficacy, we looked at the following three perception based variables:

1. Ease of Use. This variable was measured using the following six items:
    - (PEOU1). I found the procedure for applying the backlog categorization model complex and difficult to follow.
    - (PEOU2). Overall, I found the backlog item categorization model difficult to use.
    - (PEOU3). I found the method easy to learn.
    - (PEOU4). I found it difficult to apply the backlog item categorization model to our product backlog.
    - (PEOU5). I found the rules on how to use the backlog item categorization model clear and easy to understand.
    - (PEOU6). I am not confident that I am now competent to apply the backlog item categorization model in practice.
2. Usefulness. This was measured using the following eight items:
    - (PU1). I believe that use of the backlog item categorization model would reduce the effort required to maintain the product backlog.
    - (PU2). Maintaining the product backlog based on the backlog item categorization model would be more difficult for people to understand.
    - (PU3). The backlog item categorization model would make it easier for people to check whether backlog items are issued correctly.
    - (PU4). Overall, I found the backlog item categorization model to be useful.
    - (PU5). Using the backlog item categorization model would make it more difficult to maintain the product backlog.
    - (PU6). Overall, I think the backlog item categorization model does not provide an effective solution to problems to do with categorization, traceability and inconsistency.
    - (PU7). Overall, I think the use of the backlog item categorization model is an improvement to the previous way of working.
    - (PU8). Using the backlog item categorization model would make it easier to communicate with stakeholders.
3. Intention to Use. This was measured using the following two items:
    - (ITU1). I would definitely not use the backlog item categorization model to help maintain a product backlog.

- (ITU2). In the future, I intend to use the backlog item categorization model in preference to our previous way of working.

All of the items above were adapted from the TAM [21], with slight changes in wording to fit the context of the current research and reflect the objectives of the BICM. We also followed the TAM's negation of items. The TAM items from which these items were derived are shown in brackets before each item.

6.2.1 Exit Survey

The 3 items pertaining to perceived efficiency and 16 items measuring ease of use, usefulness and intention to use were combined into a single exit survey that was sent out via Google Forms at the end of the intervention period. Items were arranged in random order to reduce the risk of monotonous responses to questions relating to the same construct.

Analysis of the subsequent spreadsheet was performed in Excel. Considering this study's small sample size, responses to the exit survey were not intended for statistical analysis, but rather used to gauge whether the team's experience with the BICM could be considered positive or negative. Moreover, the survey provided an anonymous outlet for feedback, providing participants the opportunity to voice their experiences free of experimenter demand.

6.2.2 Exit Group Interview

Finally, we concluded the intervention period with a group interview. This was done to gather more in-depth data and to triangulate/contextualize findings based on the observations and exit survey.

At the start of the interview, participants were reassured that there was no right or wrong way to apply the BICM and that their use thereof was in no way being scrutinized.

The interview was semi-structured, with a set of basic questions used to guide the interview.

The interview was conducted online via MS Teams and lasted half an hour. The interview was audio-recorded with permission and transcribed and translated by the researcher.

7 Results

This section presents the results of our evaluation of the BICM, the intervention taken to help remedy this case study's identified US-issues. The following subsections discuss the results for each of the evaluative constructs defined in Section 6.

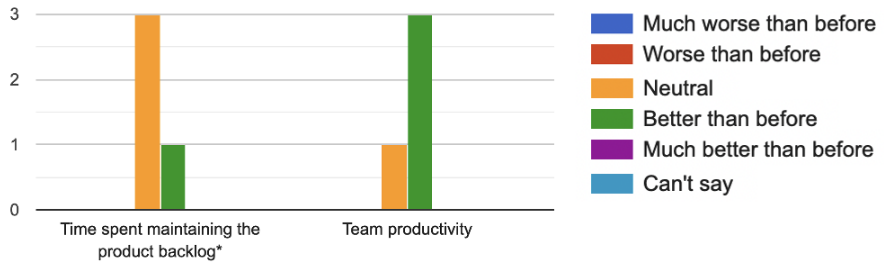


Figure 7: Responses to survey items related to efficiency. N=4.

7.1 Effectiveness

Regarding the degree to which the BICM achieved its objectives, clear changes in practice were observed upon inspection of the post-treatment Product Backlog items.

First of all, none of the user-facing requirements created during the intervention period showed inconsistencies with regard to use of the US format (O1), meaning that all requirements involving end-users now started with a descriptive title following the Connextra template: “As a ⟨role⟩, I want ⟨goal⟩, so that ⟨benefit⟩” [17]. Moreover, the observed non-USs, such as FDD-stories, bugs and improvements were also written using a consistent format, starting with a descriptive title following an FDD format as suggested by [15]: “⟨action⟩, the ⟨result⟩, ⟨by—for—of—to⟩ a(n) ⟨object⟩” [17]. These observations were in line with findings based on the group interview *P3: It (the BICM) was very applicable. We used it primarily on our new stories. We also applied it to our more functional stories, or FDD-stories, where we previously struggled a bit with what to write down.* An example of an FDD-story taken from the team's post-intervention backlog is as follows: *"Process dataset of the imported 'onderwijsbesturen' in order to update the API data"*.

Second, none of the post-intervention backlog items showed inconsistencies with regard to issue types (O2), meaning that similar items (e.g., user-facing requirements & improvements to existing features) were assigned similar issue types (e.g., 'Story' & 'Improvement').

Finally, we observed a change in the use of hierarchical levels (O3). Specifically, post-intervention product backlog items were no longer linked to one of few, large epics previously used to denote entire applications. Instead, the team had created new epics to link stories to specific components being worked on. This is confirmed by the team: *P3: "We used to use epic links to say something belonged to a certain application as a whole. We no longer do that, now you see real Epics, which can also be completed at some point. Before, when they referred to the entire application, which is never finished, the epics were also never finished. So I think that has been an important change"*.

7.2 Efficiency

Figure 7 shows the participants' responses to the survey items relating to efficiency.

With regard to time spent maintaining the product backlog, one participant rated the period after the intervention as being better than the period before the intervention, while other responses were neutral.

Regarding overall team productivity, three participants rating the period after the intervention as being better than before, while one participant was neutral.

This was reflected in the group interview as well, with the only direct mention of efficiency being as follows:" *P3: "Especially FDD-stories have become easier to write, to start with a sentence following a certain structure. It has made the process clearer.*

However, we argue that the following quotations also provide some indirect mentions of efficiency: *P3: "One of the things this (i.e., increase use of Epic links) has enabled us to do is add filters to the Backlog, meaning we can now search per component, which has helped me because it gives you insight regarding what needs to be done for a particular piece of work"* and *P4:"I also think it gives us more overview, because some stories actually touch different applications, and now we are able to see those multiple components in the story itself"*. Finally: *P5: "I can imagine that for diverse sprints containing multiple epics it has become more convenient. For me, it is nice to immediately be able to see the functional purpose of a story. It saves me having to bother P3 with unnecessary phone calls.*

Although these preliminary findings suggest a positive perceived effect on efficiency, we refrain from drawing any conclusions with regard to the BICM's actual efficiency due to the absence of quantitative metrics to triangulate these findings.

7.3 Exit Survey

Participants' survey answers relating to perceived ease of use, usefulness and intention to use are presented in Figure 8. This figure combines the questions to show an overview of all three constructs, as well as provide insight into individual questions.

7.3.1 Ease of Use

Based on the survey responses and exit interview, we conclude that participants perceived the BICM as being easy to use. All of the six survey items relating to this construct (see Figure 8, EOU1-EOU6) were answered in the positive. Interview data somewhat backs this finding, with one participant remarking *P1: "it (the BICM) was easily applicable"*.

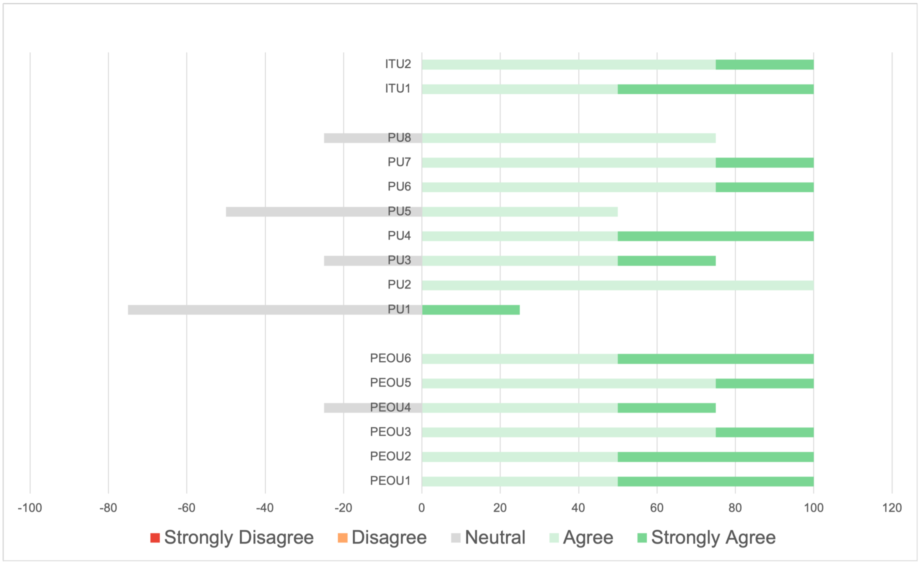


Figure 8: Results for the exit survey. N=4.

7.3.2 Usefulness

Overall, participants perceived the BICM as being useful, with none of the related survey items having been answered in the negative (see Figure 8, PU1-PU8). Only question PU1: "I believe that use of the Backlog Item Categorization Model would reduce the effort required to maintain the product backlog" received more neutral responses than positive ones. The team's perceived usefulness of the BICM is reflected in the interview data, as is illustrated by the quotes in Section 7.2.

7.3.3 Intention to Use

Participants indicated that they intended to use the BICM in their future work, with both survey results (see Figure 8, ITU1-IYU2) as well as interview data supporting this claim. Specifically, the product manager requested that the tutorial session be repeated for other ASD projects as well, so that they may also apply the BICM in their work.

8 Discussion

This section presents our main conclusions, discusses threats to validity, and suggests avenues for future research.

8.1 Conclusion

This study performed canonical action research on a real-life ASD project to investigate the challenges of applying user stories in practice. Subsequently, this research designed and evaluated the Backlog Item Categorization Model (BICM), a methodological framework specifically designed to help remedy some of the US issues identified in our case study.

Several sub-questions were formulated to help guide this research. With regard to RQ1: *What types of of US problems have been addressed by the literature?* , we found that existing US research can be classified according to three problem classes, namely ambiguity, collaboration and system design. However, very little of these studies have investigated US problems as observed in practice, which we argue is necessary to increase the relevance of US research.

With regard to RQ2: *Which solutions have been proposed by the research community?*, we found that solutions in the form of descriptions, explanations, algorithms, models, prototypes and frameworks have been proposed. However, here too, only a small part of US studies have focused on problem investigation or explanations of observed phenomena. Moreover, a small part of these solutions have been tested in real-world settings.

Regarding RQ3: *which US challenges can be observed in practice?*, the issues we observed in our case study included: (1) inconsistent use of the US format and (2) issue types, (3) limited use of hierarchical relations and (4) difficulty deciding what to include in different backlog items. We believe these issues may be explained by the fact that the ASD project under investigation was quite technical in nature. Although the predominant way for a Scrum team to express features is in the form of USs, practice shows that other items, such as technical work, bugs and knowledge acquisition activities also belong on the product backlog [15]. For projects that are more technical in nature, these "non-USs' may actually make up a relatively large part of the product backlog, as was the case for our case study. In such cases, we believe the lack of guidelines on when and how to write non-US product backlog items may cause a team to lack structure and become inconsistent. As such, this research highlights the need for more guidance on how to write other items that belong on the product backlog.

To find out how an intervention could be designed to help remedy these issues (RQ4), this study designed, implemented and evaluated the BICM, a methodological framework consisting of a decision tree and accompanying information sheet to help inform the process of creating and/or modifying product backlog items by having practitioners answer yes/no questions regarding the issue at hand, categorizing items as belonging to one of six issue types and providing suggestions regarding format and hierarchical relations.

To test our intervention's effectiveness (RQ5), we evaluated the BICM after a three week implementation period in practice. Results suggest that it was successful in achieving its objectives, with no observed inconsistencies regarding use of the US format and issue types and an increase in the use of hierarchical levels in the team's product backlog. Moreover, the BICM was perceived as being easy to use and useful and the team indicated that they intended to continue applying it in future work practices.

The US issues identified in this thesis are of practical relevance to the field of agile RE, as the number of studies investigating the practical application of US, problems that arise with their use and consequences of these problems for the ASD process is relatively small. Most existing US studies have focused on developing solutions for problems situated in narrowly defined ASD RE contexts, whereas this study attempted to investigate US issues in a holistic context. For example, our study considered USs in context of the tools used to facilitate their application, which, to the best of our knowledge, had not been done before. As such, this thesis offer novel insights into US challenges and paves the way for future research to increase the practical relevance of US research and solutions.

Moreover, this thesis contributes the BICM, which can be used by practitioners who experience similar issues as our case study.

Finally, this thesis addresses the lack of validation and evaluation of US research [2] by testing and evaluating the BICM in a real-life setting, which helps to facilitate the transfer of research results to practice.

8.2 Limitations

This thesis ought to be considered in light of several limitations. We discuss our limitations using the four types of validity suggested by Runeson and Höst for case study research [49], namely: construct, internal, external, and reliability.

8.2.1 Construct Validity

First of all, this thesis only completed one research cycle, which is a deviation from the Cyclical Process Model as proposed by Davison [22]. Typically, one would reflect on the outcomes of the intervention, followed by an explicit decision whether or not to proceed through another process cycle to move closer to the core of the organizational problem. Due to time constraints, we were unable to repeat another cycle after the BICM had been evaluated, which may have influenced the type of US issues we diagnosed in this research.

Moreover, we entered our investigation at a late stage in the project development cycle, which has undoubtedly influenced the types of US issues that we identified in this research. Although we directed several interview questions to the application of USs during earlier stages of the project, these findings relied on participants' memory and could not be triangulated with observations.

8.2.2 Internal Validity

This research faces the risk of experimenter demand, which refers to changes in behavior by participants due to cues about what constitutes appropriate behavior [58]. It is likely that the goals of this research were obvious to participants, which may influenced them to provide a positive evaluation of the BICM to favor the researcher. Although we tried to mitigate this threat by including the exit survey, our sample size was not large enough to guarantee a promise of anonymity.

Another factor to consider is that by bringing the team's US issues to light in the tutorial session, we might have constituted a change in behavior regardless of the BICM, making it less reliable to attribute the observed changes to our intervention.

Another important limitation is the sample size within our case study. Although sample size is generally less relevant in qualitative research [46], we believe that use of a larger sample size (i.e., more participants) could have contributed to a more robust evaluation of the BICM. For example, had our participants' responses not been in line with each other, we do not believe we could have drawn any conclusions based on the exit survey. However, a larger sample size may be difficult to achieve for this type of case study, since ASD projects typically do not consist of more than 8-10 people.

8.2.3 External Validity

Our use of a single case study to answer our research question poses a limitation with regard to generalizibility, as our results may be very context-specific. On the other hand, we believe that use of a single-case study enabled us to gain a more in-depth understanding of the identified problem areas. Moreover, we tried to increase the transferability of our study by designing an intervention that can be used by other teams experiencing issues of a similar nature.

8.2.4 Reliability

Finally, we acknowledge that there is a risk of researcher bias. Although we tried to formulate our list of US issues based on a triangulation of data sources, it is possible that biases, such as the researcher's prior knowledge regarding US issues, may have influenced our findings. As such, a triangulation across researchers would have been desirable.

Another limitation is that access to resources was limited for reasons to do with technical difficulties. Although we had permission to view and use all data produced by the team, we relied on the availability of team members to gain access to the product backlog. Specifically, team members had to asked to screen-share in order to view the product backlog in real-time or send samples of the product backlog. Although we do not think this has affected the outcome of our study, it would have been more reliable to have unlimited access to resources, since some details may have been overlooked.

8.3 Future Work

Future research can be conducted to help mitigate the limitations of this research.

First, we invite the RE community to perform similar case studies with the goal to provide additional insights into US issues experienced by practitioners and to help generalize the findings of this research.

Moreover, longitudinal case studies could be conducted to get a truly holistic overview of US issues that consider ASD projects from start to finish and all elements related to the application of USs.

Finally, we suggest further research to better evaluate the BICM. Specifically, we suggest a longer intervention period to study its long-term effects on practice, as well as evaluate its performance using more quantitative metrics. To evaluate actual effectiveness, for example, we suggest a statistical analysis of the number of pre- vs. post-intervention deviations from good practice in the product backlog. With regard to actual efficiency, studies could statistically compare the sum of story points completed in pre- vs. post-intervention sprints, or the the frequency of communication within a team as measured through comments.

Acknowledgements

I would like to thank my supervisor, Fabiano Dalpiaz, for his guidance and support in writing this thesis. I would also like to thank Edina Valkai, my supervisor within KPMG DE, as well all the employees from Kennisnet and KPMG who participated in and enabled this research.

References

[1] Anis R Amna and Geert Poels. "Ambiguity in User Stories: A Systematic Literature Review". In: *Information and Software Technology* (2022), p. 106824.

[2] Anis R Amna and Geert Poels. "Systematic Literature Mapping of User Story Research". In: *IEEE Access* (2022).

[3] R Barbosa, AEA Silva, and Regina Moraes. "Use of similarity measure to suggest the existence of duplicate user stories in the srum process". In: *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*. IEEE. 2016, pp. 2–5.

[4] Jasper Berends and Fabiano Dalpiaz. "Refining User Stories via Example Mapping: An Empirical Investigation". In: *2021 IEEE 29th International Requirements Engineering Conference (RE)*. IEEE. 2021, pp. 345–355.

[5] Elizabeth Bjarnason, Krzysztof Wnuk, and Björn Regnell. "A case study on benefits and side-effects of agile practices in large-scale requirements engineering". In: *proceedings of the 1st workshop on agile requirements engineering*. 2011, pp. 1–5.

[6] Maxim Bragilovski, Fabiano Dalpiaz, and Arnon Sturm. "Guided Derivation of Conceptual Models from User Stories: A Controlled Experiment". In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer. 2022, pp. 131–147.

[7] Lan Cao and Balasubramaniam Ramesh. "Agile requirements engineering practices: An empirical study". In: *IEEE software* 25.1 (2008), pp. 60–67.

[8] R Carlson, PJ Matuzic, and RL Simons. "Applying scrum to stabilize systems engineering execution". In: *CrossTalk* (2012), pp. 1–6.

[9] Kathy Charmaz. *Constructing grounded theory: A practical guide through qualitative analysis*. sage, 2006.

[10] Joelma Choma, Luciana AM Zaina, and Daniela Beraldo. "Userx story: incorporating ux aspects into user stories elaboration". In: *International Conference on Human-Computer Interaction*. Springer. 2016, pp. 131–140.

[11] Louis Cohen, Lawrence Manion, and Keith Morrison. "Action research". In: *Research methods in education*. Routledge, 2017, pp. 440–456.

[12] Mike Cohn. "Advantages of user stories for requirements". In: *InformIT Network, available at: http://www. informit. com/articles* (2004).

[13] Mike Cohn. *Epics, Features and User Stories*. URL: `https://www.mountaingoatsoftware.com/blog/stories-epics-and-themes`. (accessed: 05.09.2022).

[14] Mike Cohn. *How Detailed Should a User Story Be?* URL: `https://www.mountaingoatsoftware.com/blog/what-level-of-detail-should-be-captured-in-a-user-story`. (accessed: 05.09.2022).

[15] Mike Cohn. *Not Everything Needs to Be a User Story: Using FDD Features.* URL: `https://www.mountaingoatsoftware.com/blog/not-everything-needs-to-be-a-user-story-using-fdd-features`. (accessed: 05.09.2022).

[16] Mike Cohn. *Scrum Product Backlog.* URL: `https://www.mountaingoatsoftware.com/agile/scrum/scrum-tools/product-backlog`. (accessed: 05.09.2022).

[17] Mike Cohn. *User stories applied: For agile software development.* Addison-Wesley Professional, 2004.

[18] Fabiano Dalpiaz, Patrizia Gieske, and Arnon Sturm. "On deriving conceptual models from user requirements: An empirical study". In: *Information and Software Technology* 131 (2021), p. 106484.

[19] Fabiano Dalpiaz, Ivor van der Schalk, and Garm Lucassen. "Pinpointing ambiguity and incompleteness in requirements engineering via information visualization and NLP". In: *International Working Conference on Requirements Engineering: Foundation for Software Quality.* Springer. 2018, pp. 119–135.

[20] Maya Daneva et al. "Agile requirements prioritization in large-scale outsourced system projects: An empirical study". In: *Journal of systems and software* 86.5 (2013), pp. 1333–1353.

[21] Fred D Davis. "Perceived usefulness, perceived ease of use, and user acceptance of information technology". In: *MIS quarterly* (1989), pp. 319–340.

[22] Robert Davison, Maris G Martinsons, and Ned Kock. "Principles of canonical action research". In: *Information systems journal* 14.1 (2004), pp. 65–86.

[23] *Digital Enablement.* URL: `https://home.kpmg/nl/en/home/services/advisory/digital-transformation/digital-enablement.html`. (accessed: 1.03.2022).

[24] *Enablers.* URL: `https://www.scaledagileframework.com/enablers/`. (accessed: 25.09.2022).

[25] *Entree Federatie.* URL: `https://www.kennisnet.nl/entree-federatie/`. (accessed: 1.03.2022).

[26] Martin Fowler, Jim Highsmith, et al. "The agile manifesto". In: *Software development* 9.8 (2001), pp. 28–35.

[27] *Global State of Enterprise Agile in 2021.* 2021.

[28] Petra Heck and Andy Zaidman. "A quality framework for agile requirements: A practitioner's perspective". In: *arXiv preprint arXiv:1406.4692* (2014).

[29] Ville T Heikkilä et al. "A mapping study on requirements engineering in agile software development". In: *2015 41st Euromicro conference on software engineering and advanced applications.* IEEE. 2015, pp. 199–207.

[30] Shreeram Hudda, Ritika Mahajan, and Sarvesh Chopra. “Prioritization of User-Stories in Agile Environment”. In: *Indian Journal of Science and Technology* 9.10.17485 (2016).

[31] Irum Inayat et al. “A systematic literature review on agile requirements engineering practices and challenges”. In: *Computers in human behavior* 51 (2015), pp. 915–929.

[32] Pankaj Kamthan and Nazlie Shahmir. “Effective user stories are affective”. In: *International Conference on Ubiquitous Computing and Ambient Intelligence*. Springer. 2017, pp. 605–611.

[33] Mohamad Kassab. “The changing landscape of requirements engineering practices over the past decade”. In: *2015 IEEE Fifth International Workshop on Empirical Requirements Engineering (EmpiRE)*. IEEE. 2015, pp. 1–8.

[34] Nigel King, Christine Horrocks, and Joanna Brooks. *Interviews in qualitative research*. sage, 2018.

[35] Dean Leffingwell. *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional, 2010.

[36] Philipp Lombriser et al. “Gamified requirements engineering: model and experimentation”. In: *International Working conference on requirements engineering: foundation for software quality*. Springer. 2016, pp. 171–187.

[37] Larissa A Lopes et al. “Using UxD artefacts to support the writing of user stories: Findings of an empirical study with agile developers”. In: *Proceedings of the 19th International Conference on Agile Software Development: Companion*. 2018, pp. 1–4.

[38] Garm Lucassen et al. “Extracting conceptual models from user stories with Visual Narrator”. In: *Requirements Engineering* 22.3 (2017), pp. 339–358.

[39] Garm Lucassen et al. “Forging high-quality user stories: towards a discipline for agile requirements”. In: *2015 IEEE 23rd international requirements engineering conference (RE)*. IEEE. 2015, pp. 126–135.

[40] Garm Lucassen et al. “The use and effectiveness of user stories in practice”. In: *International working conference on requirements engineering: Foundation for software quality*. Springer. 2016, pp. 205–222.

[41] Abel Menkveld, Sjaak Brinkkemper, and Fabiano Dalpiaz. “User story writing in crowd requirements engineering: The case of a web application for sports tournament planning”. In: *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE. 2019, pp. 174–179.

[42] Daniel L Moody. “The method evaluation model: a theoretical model for validating information systems design methods”. In: (2003).

[43] Ana M Moreno and Agustín Yagüe. "Agile user stories enriched with usability". In: *International Conference on Agile Software Development.* Springer. 2012, pp. 168–176.

[44] Prabhat Pokharel and Pramesh Vaidya. "A Study of User Story in Practice". In: *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI).* IEEE. 2020, pp. 1–5.

[45] Zornitza Racheva, Maya Daneva, and Andrea Herrmann. "A conceptual model of client-driven agile requirements prioritization: Results of a case study". In: *Proceedings of the 2010 acm-ieee international symposium on empirical software engineering and measurement.* 2010, pp. 1–4.

[46] Paul Ralph et al. "Empirical standards for software engineering research". In: *arXiv preprint arXiv:2010.03525* (2020).

[47] Balasubramaniam Ramesh, Lan Cao, and Richard Baskerville. "Agile requirements engineering practices and challenges: an empirical study". In: *Information Systems Journal* 20.5 (2010), pp. 449–480.

[48] Max Rehkopf. *Stories, epics, and initiatives.* URL: `https://www.atlassian.com/agile/project-management/epics-stories-themes`. (accessed: 10.09.2022).

[49] Per Runeson and Martin Höst. "Guidelines for conducting and reporting case study research in software engineering". In: *Empirical software engineering* 14.2 (2009), pp. 131–164.

[50] Ken Schwaber. *Agile project management with Scrum.* Microsoft press, 2004.

[51] Ken Schwaber and Jeff Sutherland. "The Scrum Guide. 2020". In: *Accessed April* (2021).

[52] George Spanoudakis and Andrea Zisman. "Software traceability: a roadmap". In: *Handbook of software engineering and knowledge engineering: vol 3: recent advances.* World Scientific, 2005, pp. 395–428.

[53] Axel Van Lamsweerde. *Requirements engineering: From system goals to UML models to software.* Vol. 10. Chichester, UK: John Wiley & Sons, 2009.

[54] Bill Wake. "INVEST in good stories, and SMART tasks". In: *Retrieved December* 13 (2003), p. 2011.

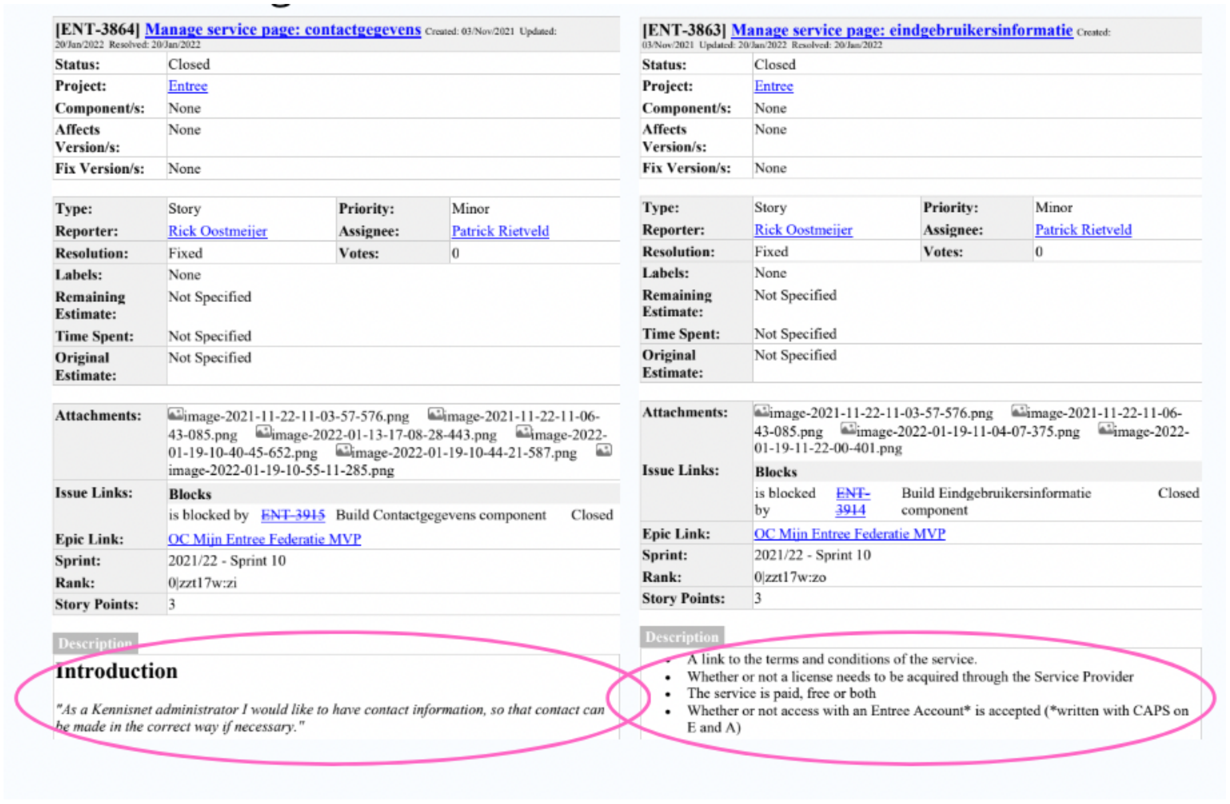
[55] Xinyu Wang et al. "The role of requirements engineering practices in agile development: an empirical study". In: *Requirements engineering.* Springer, 2014, pp. 195–209.

[56] Yves Wautelet et al. "Unifying and extending user story models". In: *International conference on advanced information systems engineering.* Springer. 2014, pp. 211–225.

[57] *What are issue types*. URL: `https://support.atlassian.com/jira-cloud-administration/docs/what-are-issue-types/`. (accessed: 10.09.2022).

[58] Daniel John Zizzo. “Experimenter demand effects in economic experiments”. In: *Experimental Economics* 13.1 (2010), pp. 75–98.

A Examples of the diagnosed US issues



[ENT-3864] Manage service page: contactgegevens Created: 03/Nov/2021 Updated: 20/Jan/2022 Resolved: 20/Jan/2022

Status:	Closed		
Project:	Entree		
Component/s:	None		
Affects Version/s:	None		
Fix Version/s:	None		
Type:	Story	**Priority:**	Minor
Reporter:	Rick Oostmeijer	**Assignee:**	Patrick Rietveld
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		
Attachments:	image-2021-11-22-11-03-57-576.png image-2021-11-22-11-06-43-085.png image-2022-01-13-17-08-28-443.png image-2022-01-19-10-40-45-652.png image-2022-01-19-10-44-21-587.png image-2022-01-19-10-55-11-285.png		
Issue Links:	**Blocks** is blocked by ~~ENT-3915~~ Build Contactgegevens component Closed		
Epic Link:	OC Mijn Entree Federatie MVP		
Sprint:	2021/22 - Sprint 10		
Rank:	0|zzt17w:zi		
Story Points:	3		

Description

Introduction

"As a Kennisnet administrator I would like to have contact information, so that contact can be made in the correct way if necessary."

[ENT-3863] Manage service page: eindgebruikersinformatie Created: 03/Nov/2021 Updated: 20/Jan/2022 Resolved: 20/Jan/2022

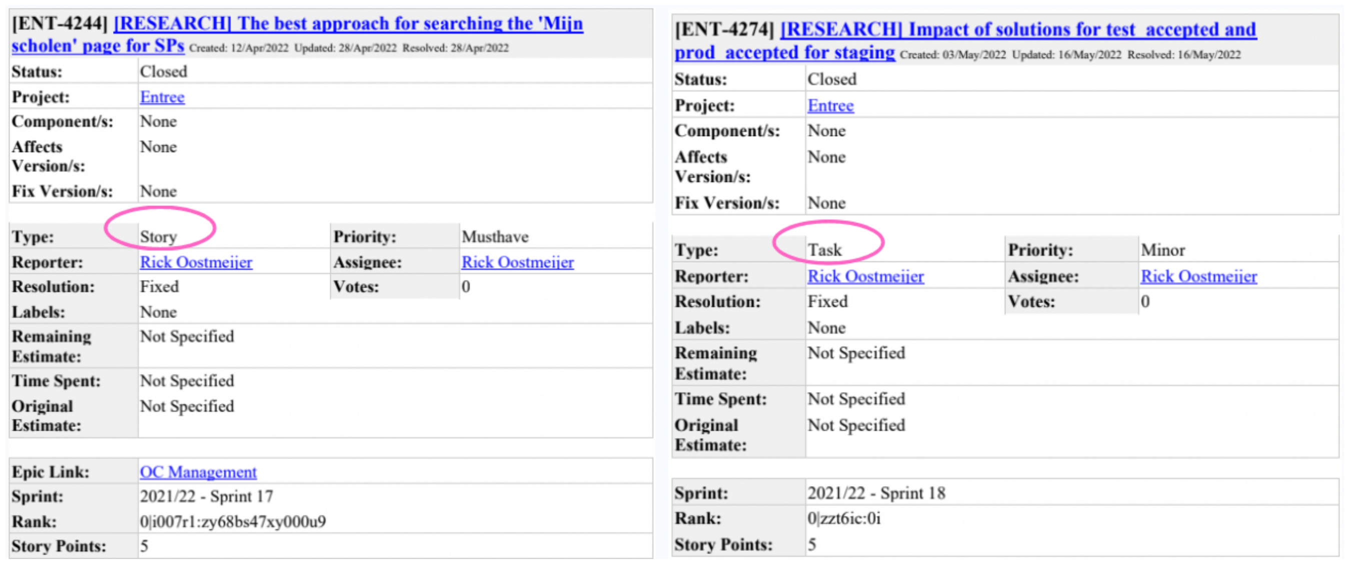
Status:	Closed		
Project:	Entree		
Component/s:	None		
Affects Version/s:	None		
Fix Version/s:	None		
Type:	Story	**Priority:**	Minor
Reporter:	Rick Oostmeijer	**Assignee:**	Patrick Rietveld
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		
Attachments:	image-2021-11-22-11-03-57-576.png image-2021-11-22-11-06-43-085.png image-2022-01-19-11-04-07-375.png image-2022-01-19-11-22-00-401.png		
Issue Links:	**Blocks** is blocked by ~~ENT-3914~~ Build Eindgebruikersinformatie component Closed		
Epic Link:	OC Mijn Entree Federatie MVP		
Sprint:	2021/22 - Sprint 10		
Rank:	0|zzt17w:zo		
Story Points:	3		

Description

- A link to the terms and conditions of the service.
- Whether or not a license needs to be acquired through the Service Provider
- The service is paid, free or both
- Whether or not access with an Entree Account* is accepted (*written with CAPS on E and A)

Figure 9: Inconsistent use of the US format in the team's product backlog.

B BICM Information Sheet



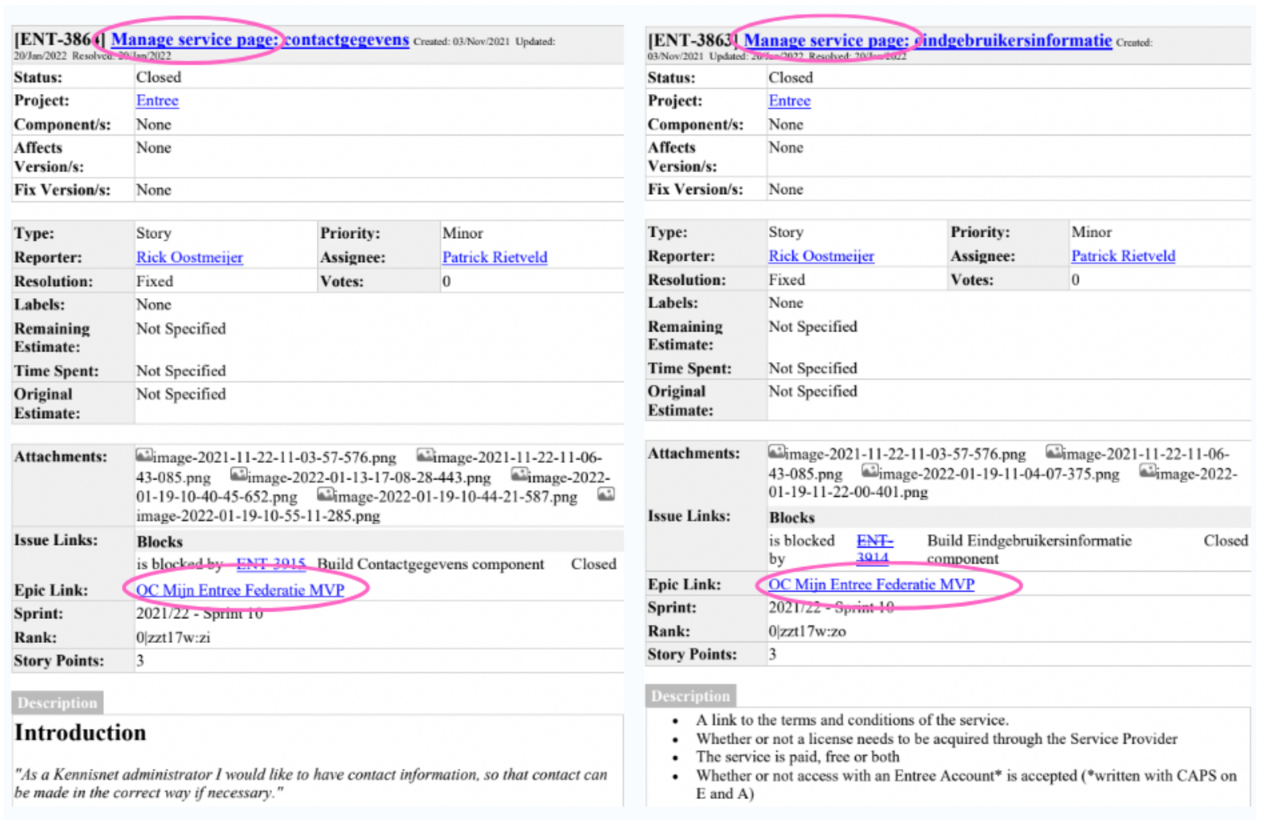
[ENT-4244] [RESEARCH] The best approach for searching the 'Mijn scholen' page for SPs Created: 12/Apr/2022 Updated: 28/Apr/2022 Resolved: 28/Apr/2022

Status:	Closed		
Project:	Entree		
Component/s:	None		
Affects Version/s:	None		
Fix Version/s:	None		
Type:	Story	**Priority:**	Musthave
Reporter:	Rick Oostmeijer	**Assignee:**	Rick Oostmeijer
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		
Epic Link:	OC Management		
Sprint:	2021/22 - Sprint 17		
Rank:	0|i007r1:zy68bs47xy000u9		
Story Points:	5		

[ENT-4274] [RESEARCH] Impact of solutions for test_accepted and prod_accepted for staging Created: 03/May/2022 Updated: 16/May/2022 Resolved: 16/May/2022

Status:	Closed		
Project:	Entree		
Component/s:	None		
Affects Version/s:	None		
Fix Version/s:	None		
Type:	Task	**Priority:**	Minor
Reporter:	Rick Oostmeijer	**Assignee:**	Rick Oostmeijer
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		
Sprint:	2021/22 - Sprint 18		
Rank:	0|zzt6ic:0i		
Story Points:	5		

Figure 10: Inconsistent use of issue types in the team's product backlog.



[ENT-3864] Manage service page: contactgegevens Created: 03/Nov/2021 Updated: 20/Jan/2022 Resolved: 20/Jan/2022

Status:	Closed
Project:	Entree
Component/s:	None
Affects Version/s:	None
Fix Version/s:	None

Type:	Story	**Priority:**	Minor
Reporter:	Rick Oostmeijer	**Assignee:**	Patrick Rietveld
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Attachments:	image-2021-11-22-11-03-57-576.png image-2021-11-22-11-06-43-085.png image-2022-01-13-17-08-28-443.png image-2022-01-19-10-40-45-652.png image-2022-01-19-10-44-21-587.png image-2022-01-19-10-55-11-285.png
Issue Links:	**Blocks** is blocked by ENT-3915 Build Contactgegevens component Closed
Epic Link:	OC Mijn Entree Federatie MVP
Sprint:	2021/22 - Sprint 10
Rank:	0|zzt17w:zi
Story Points:	3

Description

Introduction

"As a Kennisnet administrator I would like to have contact information, so that contact can be made in the correct way if necessary."

[ENT-3863] Manage service page: eindgebruikersinformatie Created: 03/Nov/2021 Updated: 20/Jan/2022 Resolved: 20/Jan/2022

Status:	Closed
Project:	Entree
Component/s:	None
Affects Version/s:	None
Fix Version/s:	None

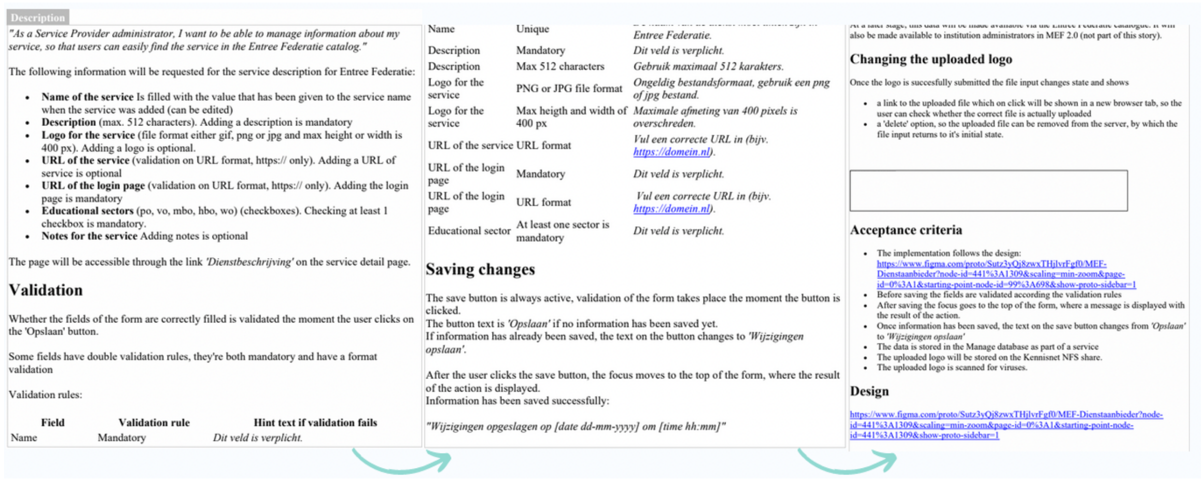
Type:	Story	**Priority:**	Minor
Reporter:	Rick Oostmeijer	**Assignee:**	Patrick Rietveld
Resolution:	Fixed	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Attachments:	image-2021-11-22-11-03-57-576.png image-2021-11-22-11-06-43-085.png image-2022-01-19-11-04-07-375.png image-2022-01-19-11-22-00-401.png
Issue Links:	**Blocks** is blocked by ENT-3914 Build Eindgebruikersinformatie component Closed
Epic Link:	OC Mijn Entree Federatie MVP
Sprint:	2021/22 - Sprint 10
Rank:	0|zzt17w:zo
Story Points:	3

Description

- A link to the terms and conditions of the service.
- Whether or not a license needs to be acquired through the Service Provider
- The service is paid, free or both
- Whether or not access with an Entree Account* is accepted (*written with CAPS on E and A)

Figure 11: Minimal use of hierarchy in the team's product backlog.



Description

"As a Service Provider administrator, I want to be able to manage information about my service, so that users can easily find the service in the Entree Federatie catalog."

The following information will be requested for the service description for Entree Federatie:

- **Name of the service** Is filled with the value that has been given to the service name when the service was added (can be edited)
- **Description** (max. 512 characters). Adding a description is mandatory
- **Logo for the service** (file format either gif, png or jpg and max height or width is 400 px). Adding a logo is optional.
- **URL of the service** (validation on URL format, https:// only). Adding a URL of service is optional
- **URL of the login page** (validation on URL format, https:// only). Adding the login page is mandatory
- **Educational sectors** (po, vo, mbo, hbo, wo) (checkboxes). Checking at least 1 checkbox is mandatory.
- **Notes for the service** Adding notes is optional

The page will be accessible through the link *'Dienstbeschrijving'* on the service detail page.

Validation

Whether the fields of the form are correctly filled is validated the moment the user clicks on the 'Opslaan' button.

Some fields have double validation rules, they're both mandatory and have a format validation

Validation rules:

Field	Validation rule	Hint text if validation fails
Name	Mandatory	*Dit veld is verplicht.*
Name	Unique	*[illegible] Entree Federatie.*
Description	Mandatory	*Dit veld is verplicht.*
Description	Max 512 characters	*Gebruik maximaal 512 karakters.*
Logo for the service	PNG or JPG file format	*Ongeldig bestandsformaat, gebruik een png of jpg bestand.*
Logo for the service	Max heigth and width of 400 px	*Maximale afmeting van 400 pixels is overschreden.*
URL of the service	URL format	*Vul een correcte URL in (bijv. https://domein.nl).*
URL of the login page	Mandatory	*Dit veld is verplicht.*
URL of the login page	URL format	*Vul een correcte URL in (bijv. https://domein.nl).*
Educational sector	At least one sector is mandatory	*Dit veld is verplicht.*

Saving changes

The save button is always active, validation of the form takes place the moment the button is clicked.
The button text is *'Opslaan'* if no information has been saved yet.
If information has already been saved, the text on the button changes to *'Wijzigingen opslaan'*.

After the user clicks the save button, the focus moves to the top of the form, where the result of the action is displayed.
Information has been saved successfully:

"Wijzigingen opgeslagen op [date dd-mm-yyyy] om [time hh:mm]"

At a later stage, this data will be made available via the Entree Federatie catalogue. It will also be made available to institution administrators in MEF 2.0 (not part of this story).

Changing the uploaded logo

Once the logo is succesfully submitted the file input changes state and shows

- a link to the uploaded file which on click will be shown in a new browser tab, so the user can check whether the correct file is actually uploaded
- a 'delete' option, so the uploaded file can be removed from the server, by which the file input returns to it's initial state.

Acceptance criteria

- The implementation follows the design: https://www.figma.com/proto/Sutz3yQj8zwxTHjlvrFgf0/MEF-Dienstaanbieder?node-id=441%3A1309&scaling=min-zoom&page-id=0%3A1&starting-point-node-id=99%3A698&show-proto-sidebar=1
- Before saving the fields are validated according the validation rules
- After saving the focus goes to the top of the form, where a message is displayed with the result of the action.
- Once information has been saved, the text on the save button changes from *'Opslaan'* to *'Wijzigingen opslaan'*
- The data is stored in the Manage database as part of a service
- The uploaded logo will be stored on the Kennisnet NFS share.
- The uploaded logo is scanned for viruses.

Design

https://www.figma.com/proto/Sutz3yQj8zwxTHjlvrFgf0/MEF-Dienstaanbieder?node-id=441%3A1309&scaling=min-zoom&page-id=0%3A1&starting-point-node-id=441%3A1309&show-proto-sidebar=1

Figure 12: Difficulty deciding what to include in a US in the team's product backlog

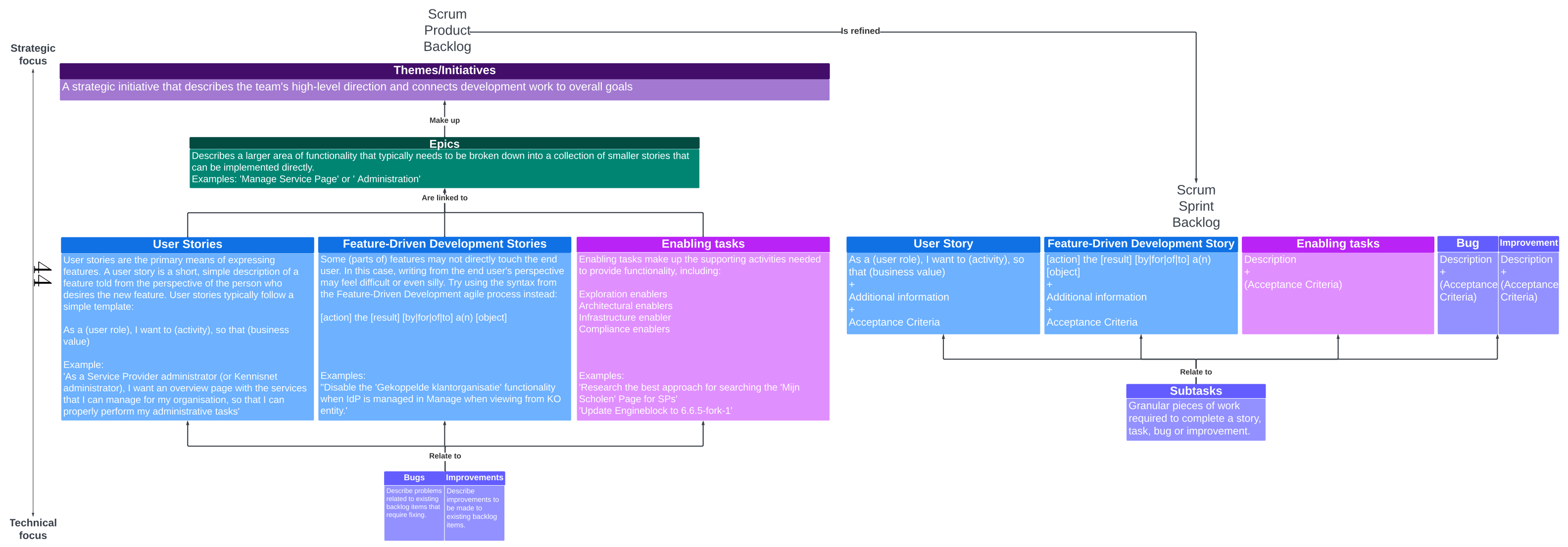

Strategic focus
Technical focus
Scrum Product Backlog
Is refined
Scrum Sprint Backlog
Themes/Initiatives
A strategic initiative that describes the team's high-level direction and connects development work to overall goals
Make up
Epics
Describes a larger area of functionality that typically needs to be broken down into a collection of smaller stories that can be implemented directly.
Examples: 'Manage Service Page' or ' Administration'
Are linked to
User Stories
User stories are the primary means of expressing features. A user story is a short, simple description of a feature told from the perspective of the person who desires the new feature. User stories typically follow a simple template:
As a (user role), I want to (activity), so that (business value)
Example:
'As a Service Provider administrator (or Kennisnet administrator), I want an overview page with the services that I can manage for my organisation, so that I can properly perform my administrative tasks'
Feature-Driven Development Stories
Some (parts of) features may not directly touch the end user. In this case, writing from the end user's perspective may feel difficult or even silly. Try using the syntax from the Feature-Driven Development agile process instead:
[action] the [result] [by|for|of|to] a(n) [object]
Examples:
"Disable the 'Gekoppelde klantorganisatie' functionality when IdP is managed in Manage when viewing from KO entity.'
Enabling tasks
Enabling tasks make up the supporting activities needed to provide functionality, including:
Exploration enablers
Architectural enablers
Infrastructure enabler
Compliance enablers
Examples:
'Research the best approach for searching the 'Mijn Scholen' Page for SPs'
'Update Engineblock to 6.6.5-fork-1'
Relate to
Bugs
Describe problems related to existing backlog items that require fixing.
Improvements
Describe improvements to be made to existing backlog items.
User Story
As a (user role), I want to (activity), so that (business value)
+
Additional information
+
Acceptance Criteria
Feature-Driven Development Story
[action] the [result] [by|for|of|to] a(n) [object]
+
Additional information
+
Acceptance Criteria
Enabling tasks
Description
+
(Acceptance Criteria)
Bug
Description
+
(Acceptance Criteria)
Improvement
Description
+
(Acceptance Criteria)
Relate to
Subtasks
Granular pieces of work required to complete a story, task, bug or improvement.

Figure 13: Information sheet component of the Backlog Item Categorization Model