SERIOUSLY AGILE AT SCALE

Creating a serious game for learning in large-scale agile transformation

Lappeenranta–Lahti University of Technology LUT

Master's Degree Programme in Software Product Management and Business

2023

Erno Viitanen

Examiner(s): Professor Maria Paasivaara

Associate Professor Sami Hyrynsalmi

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

LUT School of Engineering Science

Software Engineering

Erno Viitanen

Seriously Agile at Scale: Creating a serious game for learning in large-scale agile transformation

Master's thesis

2023

76 pages, 24 figures, 8 tables and 5 appendices

Examiner(s): Professor Maria Paasivaara and Associate Professor Sami Hyrynsalmi

Keywords: Serious Games, Agile transformation, Scaled Agile Framework

Organisations using previous-generation development methods see that agile development methods are an opportunity to enhance existing practices. However, introducing new development practices and changing existing processes can be challenging. The organisation may be reluctant to change already familiar development methods and practices due to organisational resistance, or the organisation may not have the necessary support from the management. Transformation to a previously unknown process also involves training for the members of the organisation, depending on the background of the participants. These training needs include topics like learning Agile methodologies or understanding the future framework to be used. To motivate participants and improve the effectiveness of training, researchers have suggested new ways of teaching, the use of serious games. They have received positive results on their effectiveness in learning environments.

This research focuses on the case company organisation and the understanding of agile software development practices in the organisation. The goal of this research is to show how the case company can utilize serious games for learning in the context of Scaled Agile Framework; how participants experience serious games as a learning method; and what parts of Scaled Agile Framework participants learn through serious games simulation.

This research was carried out in three phases. First, preliminary field research was conducted to determine the current understanding of Agile methodologies among the

people in the SAFe organisation. It was conducted using a survey questionnaire and there were 16 responses out of 40 participants. The insights from the field research were used in the second phase, a design science part of the research. In this phase a research artefact, a serious games simulation was designed and created. In the final phase, a simulation was carried out with participants from the SAFe organisation. There was a total of 7 volunteers participating in the simulation. After the simulation, a simulation survey was conducted to gather information about the simulation. It got 6 responses out of 7 participants and the results indicated that participants of the simulation learned some features of the Scaled Agile Framework, such as collaboration with other agile teams, and the importance of feedback in iterative development. The conclusion of this study is that serious games can be utilized to support the learning of software development processes, like the Scaled Agile Framework, through a playful simulation.

TIIVISTELMÄ

Lappeenrannan–Lahden teknillinen yliopisto LUT

LUT Teknis-luonnontieteellinen

Tietotekniikka

Erno Viitanen

Hyötypeleillä ketteräksi suuressa mittakaavassa: Hyötypelin luominen ja sen hyödyntäminen organisaation siirtyessä ketterään kehitykseen

Diplomityö

2023

76 sivua, 24 kuvaa, 8 taulukkoa ja 5 liitettä

Tarkastaja(t): Professori Maria Paasivaara ja Apulaisprofessori Sami Hyrynsalmi

Avainsanat: Hyötypelit, Ketterä ohjelmistokehitys, Laajamittainen ketteryys

Edellisen sukupolven kehitysmenetelmiä hyödyntävissä organisaatioissa nähdään uudet kehitysmenetelmät mahdollisuutena tehostaa nykyisiä toimintatapoja. Uusien toimintamallien käyttöönotto ja olemassa olevien prosessien uudistaminen voi kuitenkin olla haasteellista. Organisaatio voi olla haluton muuttamaan olemassa olevia käytäntöjä tai organisaatio ei välttämättä saa tarvittavaa tukea muutokselle johtoryhmältä. Ketterien kehitysmenetelmien käyttöönottoon liittyy myös uuden oppimista. Organisaation henkilöstölle tämä tarkoittaa käytännössä ketterän kehityksen menetelmien opiskelua sekä käyttöönotettavan viitekehyksen periaatteiden ymmärtämistä. Tutkijat ovat ehdottaneet hyötypelien hyödyntämistä opetusmenetelmänä henkilöiden motivoimiseksi sekä parantaakseen opetuksen tehokkuutta. Hyötypelien käyttö opetusmenetelmänä on tutkitusti tuottanut hyviä tuloksia oppimisen näkökulmasta.

Tämä diplomityö keskittyy case -yrityksessä olevaan organisaation ja organisaation ymmärrykseen ketterästä kehityksestä. Diplomityön tavoitteena on osoittaa, että miten hyötypelejä voidaan opetustarkoituksessa hyödyntää organisaation laajamittaisen ketterän kehityksen viitekehyksen käyttöönotossa; miten osallistujat kokevat hyötypelit opetusmenetelmänä; sekä mitä osia laajamittaisen ketterän kehityksen viitekehyksestä osallistujat oppivat hyötypeli -simulaation kautta.

Tämä tutkimus toteutettiin kolmessa vaiheessa. Ensimmäisessä vaiheessa toteutettiin alustava kenttätutkimus, jonka avulla selvitettiin SAFe-organisaatiossa työskentelevien henkilöiden nykyinen ymmärrys ketteristä menetelmistä. Tutkimus suoritettiin kyselylomakkeen avulla, joka tuotti vastauksia 16 kappaletta 40 osallistujasta.

Kenttätutkimuksesta saatuja tietoja käytettiin tutkimuksen toisessa vaiheessa, suunnittelutieteen osassa. Tässä vaiheessa suunniteltiin ja luotiin tutkimusartefakti, hyötypeleihin perustuva simulaatio. Tutkimuksen viimeisessä vaiheessa tätä simulaatiota testattiin SAFe-organisaatiossa olevien henkilöiden kanssa. Simulaatioon osallistui yhteensä 7 vapaaehtoista. Lopuksi toteutettiin vielä simulaatiokysely, jonka avulla kerättiin simulaation tulokset. Siihen saatiin vastauksia 6 kappaletta 7 osallistujasta. Nämä tulokset osoittivat, että simulaation osallistujat oppivat osan SAFe:n aihealueista, kuten työskentelystä muiden agile tiimien kanssa ja palautteen saamisen tärkeydestä iteratiivisessa kehityksessä. Tämän tutkimuksen lopputuloksena voidaan todeta, että hyötypelien avulla ohjelmistokehitysprosessien oppimista, kuten SAFe:a, voidaan tukea leikkimielisten simulaatioiden avulla.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Maria Paasivaara for guiding me through this research and helping me to complete my thesis. I would also like to thank all the people participating in the simulation of this research, especially my son Niko for building the simulation state with me.

I would also like to express my gratitude to my fellow students and teachers whom I had the pleasure of working with during my Master's degree.

Finally, I would like to thank my family, especially my wife Tiina, for supporting me through these years of studying while working at the same time. I could not have done it without you.

SYMBOLS AND ABBREVIATIONS

Abbreviations

SAFe	Scaled Agile Framework
RQ	Research Question
PI	Program Increment
IP	Innovation and Planning
ART	Agile Release Train
SoS	Scrum of Scrums
PO	Product Owner
I&A	Inspect and Adapt
LeSS	Large Scale Scrum
DAD	Disciplined Agile Delivery
LeanSAFE	Lean Scalable Agility for Engineering
DST	Don’t Starve Together

Table of contents

Appendices

Figures

Tables

1 Introduction

Agile development methods are nowadays almost taken for granted as a way of doing modern software development. Agile software development is a relatively new concept compared to the history of computer science and software development. Before agile development, software was designed and implemented according to legacy standards and practices, such as the waterfall model. For organisations that are still using previous-generation development methods, agile development methods are seen as an opportunity to enhance existing practices, increase software quality and enable higher customer satisfaction (Sidky 2007). Organisations using modern agile development methods, that have seen the benefits of agile software development, are now considering their transformation on a larger scale. A recent study by Putta et al. (2021) has shown that the main reason for adopting large-scale agile frameworks is the need to scale agile development to support more people and to remain competitive in the market.

1.1 The case organisation

CGI, the case company is a worldwide information technology and business consulting company. The company offerings include business consulting, systems integration, various services related to end-to-end, application and infrastructure solutions, digital transformation, cloud solutions and cybersecurity. Currently, the company employs around 90 000 people in 400 locations around the world (CGI 2023). The author of this paper was working for CGI at the time in one of CGI's own IP (intellectual property) solutions, CGI Profio360. The solution is a comprehensive ERP system for manufacturing and construction companies that have around 10 000 daily users (CGI Profio360 2023). The development of the solution started somewhere in the 1980s, a long time before agile development practices. From the solution organisation perspective, this meant that plan-driven development methods such as waterfall were used to develop the software solution.

The adoption of small-scale agile methods for CGI Profio360 organisation began in 2021 with the goal to improve the efficiency of software development. It was decided that the organisation would start with a single Scrum team. At the same time, some other solution

organisations inside CGI had already adopted or were starting to adopt the Scaled Agile Framework (SAFe), which drove its adoption in the CGI Profio360 organisation as well. The goal of the transformation was to scale an existing Scrum team into a solution-wide agile organisation. In August 2022, CGI Profio360 organisation started with their first Program Increment that lasted only 8 weeks, it was designed to be an increment devoted mainly to training and learning the Scaled Agile Framework. A group of 40 people were participating in the first Program Increment.

Introducing new development practices and changing already existing processes can be challenging. The organisation may be reluctant to change already familiar development methods and practices due to organisational resistance; or the organisation may not have the necessary support from the management (Dikert et al. 2016). Transformation to a previously unknown process also involves training for the members of the organisation, depending on the background of the participants. These training needs include topics like learning Agile methodologies and understanding the future development process framework that is planned to be taken into use. Some of these challenges were also apparent for the case organisation.

1.2 Serious games

Games are traditionally seen as fun and engaging for the players. The main purpose of games is to entertain people. Currently, the entertainment gaming industry is worth several hundred billion dollars. At the same time, the gaming industry has been criticized by several studies for making games too engaging, making them addictive in nature (Kuss and Griffiths 2012). Earlier research was mainly focused on the negative impacts of games, e.g., violent video games increase the aggressive thoughts of players (Connolly et al. 2012). Due to the popularity and engagement of games, alternative uses have been identified. Games that are designed for a purpose (e.g., have a learning goal) are called Serious Games. Several studies have shown that serious games can be useful as a learning method (Connolly et al. 2012).

This research focuses on the case company organisation and its transformation into large-scale agile software development. The goal of this research is to provide a solution for the case company organisation that utilizes serious games to support the learning of the Scaled

Agile Framework. This research provides insights into how participants of the study simulation experience serious games as a learning method and what they learn in the context of the Scaled Agile Framework.

2 Theoretical framework

This chapter describes the theoretical background for the research. The first section introduces different software development processes; the second section defines concepts related to learning in general. The final section focuses on serious games and their relation to learning.

2.1 Software development process

This section describes three (3) common software development processes that are used to develop software. Beginning from traditional, known as waterfall, then describing modern agile development methods and finalizing with large-scale agile methodology.

2.1.1 Traditional software development

Waterfall model is generally considered to be so called traditional software development process model. The first time it was documented in detail by Winston W. Royce in 1970. In this model the software is developed in a waterfall manner, starting from the top of the mountain with Requirements engineering. In this step the required functionalities are gathered and documented in detail for the next phase. After which the process continues with Design and implementation phase, where the documented requirements are transformed into detailed architectural design of the software solution. By utilizing the requirement and design documents the actual development work can be carried out. During the development phase, developers build the software using their skillsets and knowhow. During this phase, developers can also create and execute unit tests that validate the functionality of the software solution before the it is handed over to the next phase, the Testing phase. In this

phase the overall software system integration is tested for quality and functional verification purposes. The final phase of the waterfall model is Operations, where the software is documented for the end-users and deployed to live environment for the users to gain value from it (Royce 1987). Even though the waterfall was never mentioned in the original paper by Royce, the "waterfall" nature of the model can be seen in the referenced figure (Figure 1) of the original document.

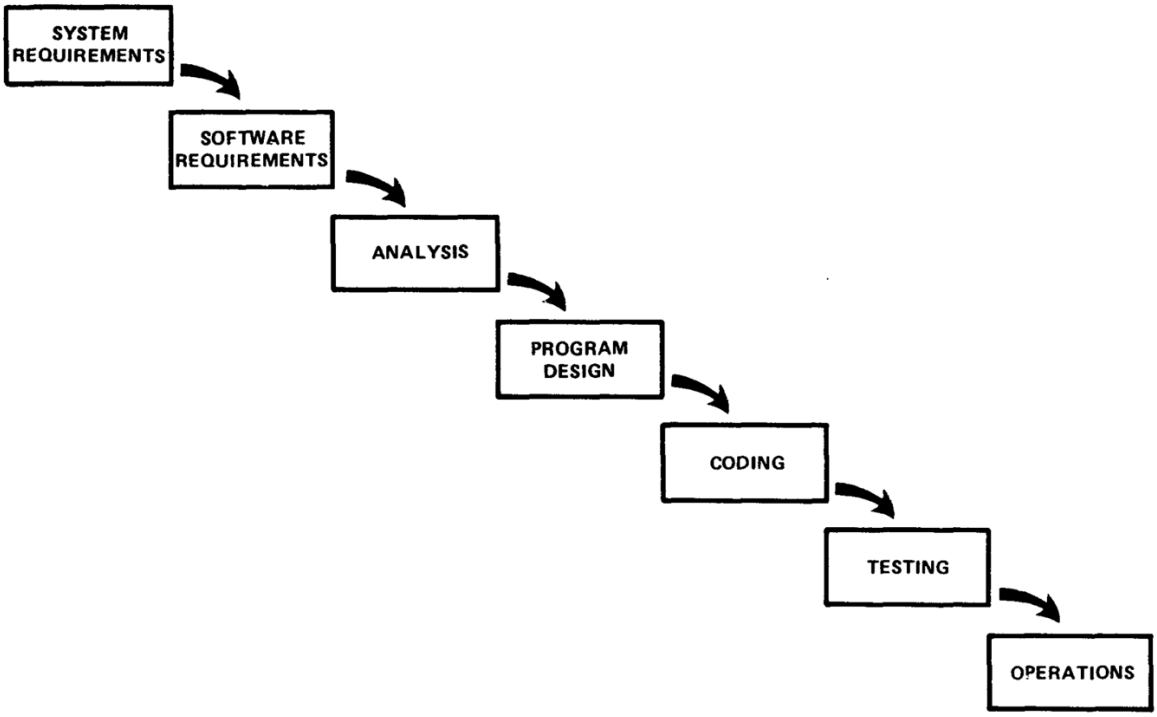


Figure 1. Traditional software development process model (Royce 1987).

Every phase of the waterfall model has a distinct objective that needs to be completed before the process can move from phase to the next one. Once the objective is completed, there is no way to return to the previous phase. The process-oriented nature of the waterfall model means that all requirements must be clearly known and defined before development can begin. There is no room for further changes once the process has moved beyond the initial requirements definition phase.

2.1.2 Agile software development

The strictness of the traditional software development method led to problems in which the software projects failed or took too long to complete. Alternative approaches that addressed these problems needed to be developed.

Manifesto for Agile software development was created and signed in 2001 (Fowler and Highsmith 2001). The manifesto defined central values on which the software development should be based on:

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

Fowler and Highsmith (2001) based the agile manifesto on 12 principles that were the driving factor of the Agile software development community. These agile manifesto founding principles and their relation to agile activities can be seen in table 1.

Table 1. 12 principles of agile manifesto

Principle	Agile activity
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	The iterative development cycle delivers value to the customer.
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	The agile approach embraces feedback and change.
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.	A rapid cycle of internal or external product delivery enables fast feedback.
Business people and developers work together daily throughout the project.	Frequent interaction between business and development ensures that requirements are iteratively agreed upon.
Build projects around motivated individuals, give them the environment and support they need and trust them to get the job done.	Managers need to trust the people who know the most about the situation and allow them to make the decisions.

The most efficient and effective method of conveying information with and within a development team is face-to-face conversation.	Elicit understanding by combining face-to-face conversation with enough documentation.
Working software is the primary measure of progress.	Iterative development provides milestones which are an accurate measure of progress.
Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.	Productivity, creativity, and alertness come from a healthy team that sustains a constant pace.
Continuous attention to technical excellence and good design enhances agility.	The software needs to be refactored to introduce new changes at a constant pace.
Simplicity—the art of maximizing the amount of work not done—is essential.	Simpler solutions allow easier changes.
The best architectures, requirements and designs emerge from self-organizing teams.	The best solutions come from within self-organizing teams that have the power to make decisions.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	Any agile team must refine and reflect to constantly improve its practices.

Scrum is an agile framework, and it is among the most popular and widely used method of agile software development (Rodríguez et al. 2018). The goal of scrum is to focus on delivering value to stakeholders. Scrum focuses on a self-organizing development team that can produce ideas into working software products. A Scrum team usually consists of one Scrum Master, one Product Owner and 2-7 Developers, depending on the project size. The Product Owner holds the product vision and is responsible for producing the product backlog which contains an ordered list of items that reflect what is needed to be done to improve the product. Scrum Master is helping the team to utilize scrum for developing the software product. A developer is identified as anyone on the team that is delivering work. Scrum is executed in development sprints that usually last 1-3 weeks. Before each sprint, a sprint planning meeting is used to define what to do and how to do it in the next sprint, producing a sprint backlog. During the sprint, the Scrum Master and Developers use daily meetings to synchronize, i.e., inspect that tasks are done, and upcoming tasks are adjusted. After each sprint, a sprint review is used to inspect the sprint outcome and to gain feedback from stakeholders. There is also a sprint retrospective after each sprint, which is used to improve

the process of building the product. The cycle is repeated by starting again with sprint planning in the next sprint (Agile Alliance, 2022).

2.1.3 Large-scale agile

There are numerous frameworks for scaling agile. What these frameworks have in common is their main objective, to scale existing, proven and widely adopted software industry team-based agile methodologies. Ebert and Paasivaara (2017) compared five of these frameworks in their work: Scrum of Scrums (SoS), Scaled Agile Framework (SAFe), Large-Scale Scrum (LeSS), Disciplined Agile Delivery (DAD), Lean Scalable Agility for Engineering (LeanSAFE). The selection of different frameworks for their work was based on surveys and industry usage. The most widely adopted framework for scaling purposes is Agile Scaled Framework (SAFe), even when some practitioners see it as too heavy, complex, and non-agile (Kittlaus and Fricker 2017, Ebert and Paasivaara 2017). This study focuses mainly on SAFe because the case organisation started their journey of adopting it.

To scale agile in a large organisation, SAFe has introduced a team of agile teams called Agile Release Train (ART). ART is powered by multiple agile teams that share a common goal to deliver one or more solutions in a value stream (Scaled Agile Framework 2022). One ART typically consists of 5-12 agile teams (around 50-125 people) that utilize a plan-commit-develop-deploy process to deliver value (Scaled Agile Framework 2022). Individual agile teams can choose any agile method or combination that works best for them, such as Scrum, Kanban, or XP.

Program increment (PI) is a timebox where ART delivers incremental value that typically lasts 8-12 weeks. The most common pattern for PI is four (4) Development Iterations and one (1) Innovation and Planning (IP) Iteration. There are two types of events in SAFe, ART events that include all ART participants, and Team events that include individual team members (Scaled Agile Framework 2022). Each Program Increment in SAFe starts with a PI Planning ART event, where ART participants are presented with the business objectives and shared vision of the PI. The outcome of PI Planning is a list of PI objectives and a program board. After the PI planning, agile teams start their Development Iterations, each lasting 2-4 weeks. Each iteration starts with Iteration Planning where the team plan and agrees on what to develop. Following by an Iteration Execution where the team implements

the increment. During each iteration, the Backlog Refinement event is used 1-2 times to refine backlog items. Iterations end with an Iteration Review event where the team reviews iteration results and adjusts the team backlog. Iteration Retrospective is the final phase of iteration, and it is used to identify ways to improve the team practices. After each development iteration, an ART-wide System Demo is held, where each agile team presents their results to ART stakeholders.

During agile team development iterations, a couple of ART events are held that keep the train on the right track. These events include a Scrum of Scrums (SoS) event with Scrum Masters of each agile team and a PO Sync event with Product Owners from each agile team and a Product manager. SoS and PO Sync can also be combined into a single event called an ART Sync. These events are typically held weekly and are used to provide visibility into how the ART is progressing toward PI objectives. Preparing for the next PI planning event is a continuous process during each Program Increment and it is used to make sure that the backlog and content are ready for the next PI planning event. Each Program Increment ends with an Inspect and Adapt event where the solution state is demonstrated and evaluated by the ART. The result of this event is a set of improved backlog items that go into the Program Backlog, ready for the next PI Planning event (Scaled Agile Framework 2022).

The last iteration in each Program Increment is called an Innovation and Planning (IP) iteration. This iteration is used to prepare for the next PI Planning event and to dedicate time for innovation and continuous education. It can also be used as a buffer to finalize tasks that are needed to meet planned PI objectives from the current PI (Scaled Agile Framework 2022). An example schedule for ART and Team level events of SAFe Program Increment is illustrated in Figure 2.

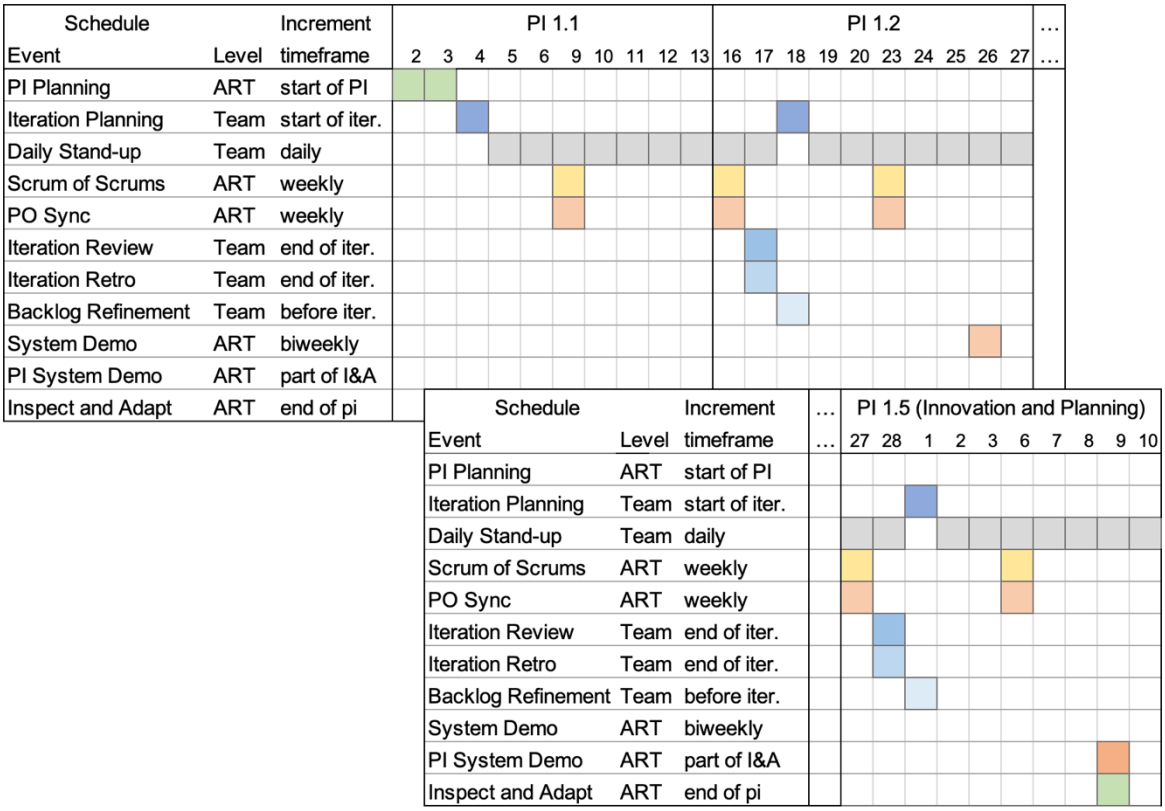


Figure 2. Example schedule for Program Increment in Scaled Agile Framework

The latest 5.1 version of SAFe includes four (4) different configurations that can be utilized depending on the size of the business. Essential SAFe is the basic configuration of SAFe that includes minimum elements that ensure the success of development and agile product delivery at the program level (Figure 3). Large Solution SAFe is built on top of essential, excluding portfolio management and it is meant for large and complex solutions (Scaled Agile Framework 2022). Portfolio SAFe adds an additional level that is used to enable portfolio management and support for multiple simultaneous solution development and management. Full SAFe includes all configurations and is meant for massive projects with multiple teams and hundreds of people.

Levels focus / Configurations	Essential SAFe	Large Solution	Portfolio	Full SAFe
Team *Team and technical agility*				
Program *Agile product delivery*				
Large solutions *Enterprise solution delivery*				
Portfolio *Lean portfolio management & organizational agility*				

Figure 3. Adoption configurations and levels of SAFe (Digite 2022)

SAFe has introduced critical roles in addition to the usual agile team of Agile Scrum methodology. These additional roles ensure the success of an Agile Release Train. Release Train Engineer (RTE) is a servant leader of ART whose main responsibility is to facilitate ART events and to help agile teams in delivering value. RTE is like a Scrum Master, but for the whole ART. The Product Manager is responsible for defining and supporting the building of a product. The System Architect is responsible for defining and communicating the architecture of the system and development practises set in place for the organisation. Business owners are a group of key stakeholders who are the main ones responsible for the business outcomes of ART (Scaled Agile Framework 2022).

Large Solution SAFe configuration includes additional roles, that ensure the success of larger projects. Solution Train Engineer that facilitates and guides multiple ARTs, called a Solution Train. The Solution Manager is responsible for defining large-scale business solutions. The Solution Architect is responsible for defining a shared technical and architectural vision across the Solution Train. Portfolio SAFe configuration includes roles of Epic Owners, who coordinate portfolio epics, and Enterprise Architect, who is responsible for technology strategy and roadmap (Scaled Agile Framework 2022).

2.2 Serious games

Serious games have become an established field of study. It has a rich and interdisciplinary history, rooted in fundamental debates that explore concepts of play. There are historical precedents for the use of games for non-entertainment purposes, where play has a major role in educational terms. Plato, for example, believed that certain behaviours exhibited in play

would reinforce similar behaviours as an adult. Since the 19th century it has been assumed that children's play and games are a necessity for human development (Wilkinson 2016). Many academic publications quote “Play is the work of children” and suggests attributing the phrase to Jean Piaget (1896-1980), a Swiss psychologist known for his work on child development (Wilkinson 2016, Pellegrini and Bjorklund 2004, Boucher and Downing and Shemilt 2014, Resnick 2007). Some sources (Carver 1986) suggest attributing the phrase to Friedrich Froebel (1782-1852), a German educator who invented kindergarten (Froebel 2017). Still, the phrase rooted in Piaget’s philosophy is the foundation for modern Serious Games development.

Serious Games leverage the success of entertainment value of the gaming industry and digital games. In addition to this, serious games provide serious value through the educational element. In contrast to Piaget’s work on child development and play, the assumption is that Serious Games is to be used like a toy. Using digital games as toys means that the activity itself is motivating because it is fun (Ritterfeld et al. 2009).

The next subchapters present different types of serious games used for educational purposes in the software engineering industry. There are many games designed to support learning, but the games presented in the next subchapters reflect different areas of the software industry or techniques that software developers could use to develop better quality software more efficiently.

2.2.1 Hard Choices game

Hard Choices game is a board game designed by Brown et al. (2011). It is a game that teaches concepts of technical debt. The game is playable by 2 – 4 competing people and it takes 15-20 minutes to play a single round, which can be repeated several times. The game consists of a board game (Figure 4), a regular six-sided die, markers, tool cards and bridge cards. The player who gets the most points in the game wins the game.

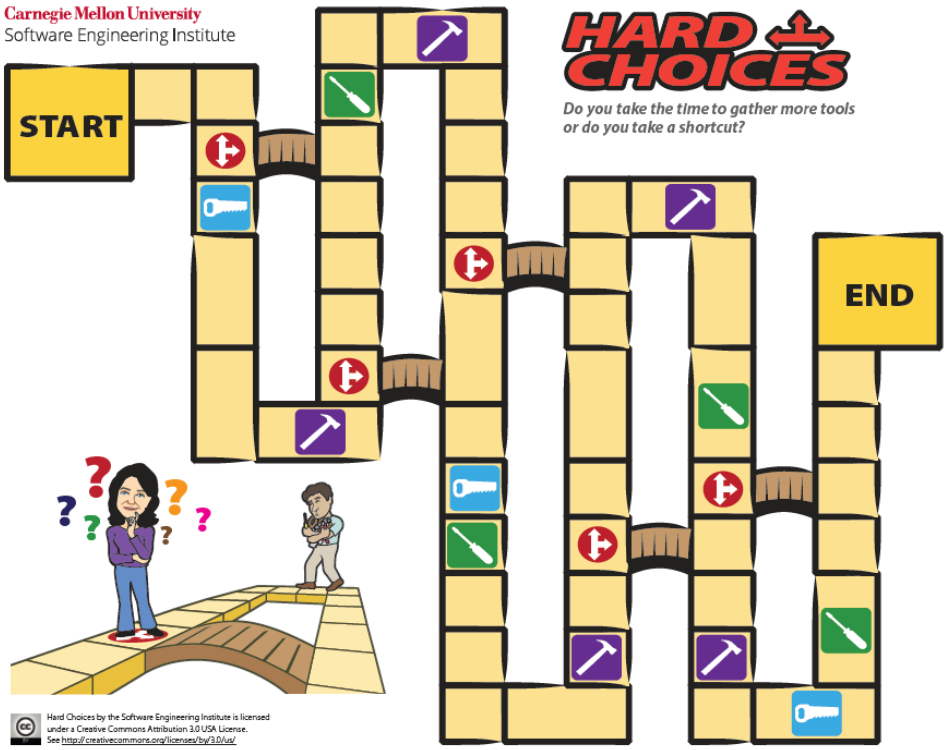


Figure 4. Hard Choices game explained (Brown et al. 2011).

Players may choose to move in any direction within the roll of the dice. Players gain points by collecting tool cards by landing in the corresponding tile and finishing the game before the other players. The first player who finishes the round first gets 7 points, the second 3 points, and the third 1 point. Every tool card is worth 1 point. Players can choose to take shortcuts in bridge sections of the game, when taken the player gains a penalty in form of a bridge card that deducts dice value for the rest of the game. This makes players re-evaluate their strategies at every roll and decide whether to take shortcuts that will accumulate over time like technical debt in software development. This game is a concrete example that is designed with serious games in mind that are directly related to the software engineering concept of technical debt.

2.2.2 Scrum Lego simulation game

For teaching Scrum in practice, Paasivaara et al. (2014) introduced an onsite simulation that utilized Lego building blocks. The simulation was run in two instances of the Project Management Course at Aalto University. In each instance, the students were divided into Scrum Teams of 7-8 people. Scrum Teams were also accompanied by one Scrum Master

and one Product Owner. Scrum Teams were provided with post-it notes, pens and a box of LEGO building blocks. The aim for each Scrum Team was to build a new product for the market, an Antarctic exploration set that included different constructables: a luxury cabin, sauna, helicopter with a landing pad, snow scooter, snow plot and a fence that surrounded the premise. A completed set of Agile Parrot variation of the same Lego simulation game can be seen in Figure 5.

Figure 5. Scrum Lego simulation game (Agile Parrot 2020).

The game was designed to teach participants the Scrum process and roles in practice, requirement management and the importance of customer collaboration, estimating work, collaborating as a team, and visualizing work and progress. The overall reception from participants of the game was highly positive. Feedback from the participants demonstrated that they learned more than expected by the learning goals set at the beginning of the simulation.

2.2.3 Respond to Change or Die: An Educational Scrum Simulation for Distributed Teams

The most recent study conducted by Christensen and Paasivaara (2022) was a digital playful simulation called the DST Scrum simulation. The goal of this online simulation was to teach Scrum in practice to university students. The simulation was based on a multiplayer game "Don't Starve Together" (DST). The idea for the game was born when the Covid-19 lockdown prevented the physical interaction of participants of the original Scrum Lego

simulation. The goal of the simulation was to fulfil the product goal (Figure 6), and to build a base for the product owner.

Figure 6. DST Scrum simulation product goal (Christensen and Paasivaara 2022).

The product goal was described through five epics: food, light, a place to sleep, companionship and safety. These epics were linked to labels that were shown with user stories or tasks in the Agile board created in Trello. Each of these user stories was described in the card of the Agile board with a template “As the PO I would like <item or animal>, so <receive benefit>” (Christensen and Paasivaara 2022).

Results of the study indicated that participants learned to communicate with other team members and the PO, estimate tasks, Scrum events and how to use Scrum in practice. The complexity of DST as a play space was seen as one of the challenges reported by participants. This was especially evident for participants who didn’t have enough video game experience. The overall results indicated that participants felt that the simulation was effective to learn Scrum in practice.

3 Research methods

This chapter describes the process by which the research was carried out.

3.1 Research questions

The purpose of this study is to find out how serious games can be used for learning in the context of scaled agile transformation, what participants learn and how they experience serious games as a learning method. Research questions are therefore:

3.1.1 RQ1: How to create a serious game that allows users to learn SAFe?

3.1.2 RQ2: How do the members of the SAFe organisation experience serious games as a learning method?

3.1.3 RQ3: What SAFe-related features do participants learn from serious games simulation?

3.2 Research process

The research was executed in three (3) phases using mixed methods of field research and design science.

The first phase was used to do a literature review on existing serious games in the context of software engineering and especially in scaled agile. Preliminary field research based on a survey questionnaire was performed on the case organisation. The purpose of the survey was to get a preliminary understanding of the organisational knowledge of agile practices. This knowledge was used to get more focus on the research problem and to pinpoint research on the areas of agile practices that were lacking the most knowledge within the participants in the organisation.

The second phase was used to design and build an artefact (a serious game) in the context of scaling agile. The artefact design was based on existing literature and on the information gathered from the first phase. The artefact was initially planned to be either a digital one (video game) or a physical one (board game). The plan was to use existing serious game from the literature and adapt it to the current challenge. This was later changed so that the initial concept from existing serious game research was adapted to another game.

The third and final phase was used to experiment with the artefact in a scaled agile context and to gather results using a simulation survey. The results of the experiment were analysed and reported to the research community to conduct further research on the topic.

3.2.1 Preliminary field research

The main purpose of preliminary field research was to determine the current understanding of Agile methodologies among the people in the SAFe organisation into which the case company was being transformed. The field research was implemented by a survey questionnaire, and it was conducted using a Webropol survey tool. The survey questionnaire was distributed by web link via email among the people in the SAFe organisation. The questions used in the survey were based on the results of a previous study "Towards a Standardized Questionnaire for Measuring Agility at Team Level" (Looks et al. 2021).

The questions from the earlier research were modified to better target the situation in the case organisation. The original questions were decided to be modified because some of the questions were difficult to understand. In addition, references to 'project', which was used to refer to the development process in the original study, were removed from the questions. The survey questionnaire was also translated into the Finnish language because most of the people in the case organisation were native Finnish speakers. Participants were given the opportunity to give their responses in English or in Finnish.

Survey questions Q2-Q7 were based on a 7-point Likert scale. On the scale, participants were asked to choose between 1-7, verbally presented with values ‘totally agree’, ‘agree’, ‘rather agree’, ‘neutral’, ‘rather disagree’, ‘disagree’, and ‘totally disagree’. Survey question Q8, “How important do you see agile development principles?”, was also based on a 7-point Likert scale but with different verbal presentation values of ‘particularly important’, ‘important’, ‘rather important’, ’neutral’, ‘rather unimportant’, ‘unimportant’, ‘particularly unimportant’.

The survey questionnaire was complemented by two additional questions that were not originally part of the standardized questionnaire. First question Q1, “What is your primary role in an Agile organisation?”. This question was used to gain more insight into whether the problem in the understanding of agile practices was within the agile team members

(Developers) or within some other role (Scrum Master, Product Owner) in the SAFe organisation. The last question, “Feel free to share your own views on team agility or thoughts from the survey (optional)”. This question was created for the participants to be able to say openly about Agility in general and to give feedback in their own words. For a full list of the survey questionnaire, see Tables 2-3 (Appendix 1).

There were 40 recipients in the email that contained a web link to the survey. Survey participants were given 7 days to respond and one additional email as a reminder was sent to the participants 2 days before the deadline. 16 out of 40 participants submitted responses to the survey.

3.2.2 Design science for building an artifact

The goal of this phase was to build an artefact that would provide a simulation space to learn the Scaled Agile Framework. The full process of designing and implementing an artefact for serious games simulation is presented in chapter 4.

3.2.3 Simulation and a survey conducted to gather results

The third phase was used to experiment with the artefact designed in the previous phase in a scaled agile context and to gather results from the experiment using a survey.

One of the key purposes of the simulation was to create an experience for the participants that would allow them to learn the Scaled Agile Framework in practice. It would normally take approx. 8-12 weeks to run an actual Program Increment in SAFe, to implement this kind of simulation just for learning purposes could be seen as a waste of important development resources. By utilizing a simulation that was specially designed to offer a similar experience of running a Program Increment in practice, the learning period could be expedited to 3 weeks, with effective use of the time of 8-12 hours/development resource. To accomplish this, the actual implementation part of real-world software development was substituted with activities in a simulated environment, in this case in an interactive multiplayer game.

Results of the simulation were collected using a survey questionnaire, that was conducted using a Webropol survey tool. The survey questionnaire was distributed by web link via

email among the simulation participants. The questions used in the survey were based on a previous study “Respond to change or die an educational scrum simulation for distributed teams” by Christensen and Paasivaara (2022).

Some of the questions from the earlier study were modified and dropped to better target the Scaled Agile Framework. ‘Product backlog’ and ‘Sprint backlog’ were renamed to ‘Program backlog’ and ‘Iteration backlog’. ‘Scrum team’ was changed to ‘Agile team’. ‘Conducting Sprint Planning / Review and Demos / Retrospectives” were renamed to ‘PI Planning / System Demo / PI Retrospective’. A full list of questions used in the simulation survey can be seen in Table 6 (Appendix 3).

Survey question Q1 options were based on a 4-point scale. On the scale, participants were asked to choose between 1-4, verbally presented with values ‘I didn't learn anything new about this topic in the simulation’, ‘I heard about this in the simulation’, ‘Due to the simulation I can explain what the topic is’, ‘Due to the simulation I learned how to use/apply this topic in practice’.

Survey question 2 options were based on a 5-point Likert scale. On the scale, participants were asked to choose between 1-7, verbally presented with values ‘strongly agree’, ‘agree’, ‘neutral’, ‘disagree’, and ‘strongly disagree’.

The results of the simulation survey and this study can be found in chapter 5. This information can be used to conduct further research on the topic.

4 Simulation design

This chapter describes the process of designing and running the first iteration of the simulation. The first chapter presents my own thoughts and ideas that I had during the process. The second chapter presents simulation goals and challenges. The third chapter illustrates the connection between serious games and software engineering. The fourth and final chapter introduces the first iteration of serious games simulation.

4.1 The process of designing a simulation for learning

The initial idea for the simulation was to create a tabletop board game that would be used to simulate Scaled Agile Framework and Scrum principles in practice. The idea was to present a similar but alternative solution to the Lego building simulation. The aim was to provide a platform for the participants where simple but concrete tasks could be planned and carried out. A hand-drawn paper sketch prototype was created that was based on chance and luck (Figure 7). Features were cards that illustrated a grid with numbers; players would roll a die that would simulate software implementation. Soon after a few rounds, I realised that the simulation isn't the best when it is based on chance, and therefore the prototype was dropped.

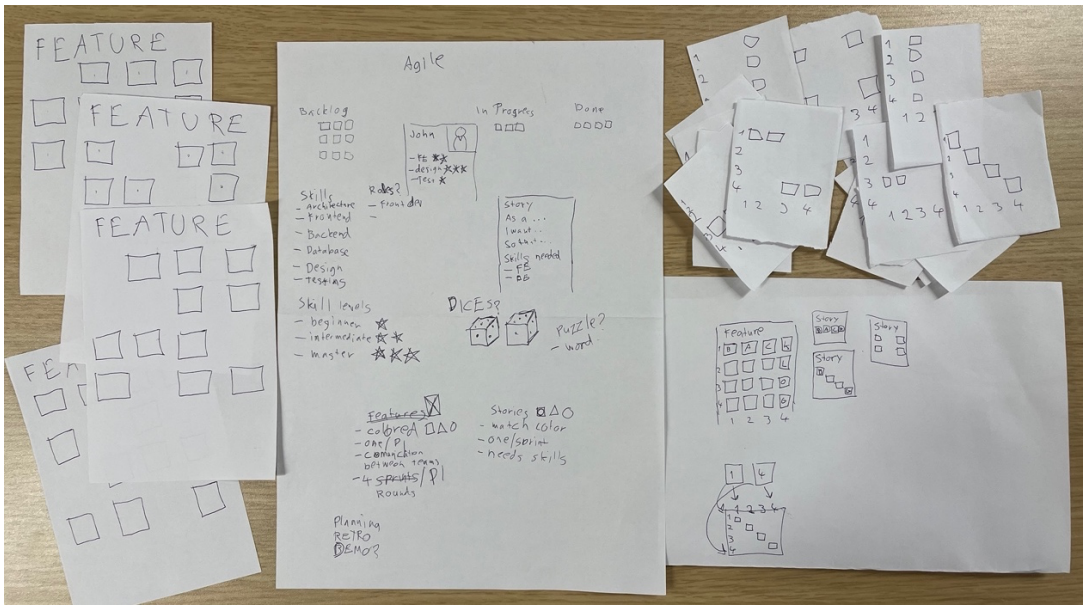


Figure 7. Hand drawn paper sketch prototype.

I started to think about my issue from a wider perspective and looked for the most recent papers published on serious games and simulations. My examiner, Maria Paasivaara had previously examined another similar thesis "DST Scrum simulation" that was based on a multiplayer game "Don't Starve Together". Previous research that was carried out had promising results. I started to gather information about the game itself and bought myself a pair of copies from Steam. I started to play the game and tried to figure out whether it could be used to carry out my research. The first couple of hours of the game was very confusing, it was hard to figure out what to do in the game. After a while, I got a hang of the game and managed to gather resources and build a decent outpost. I started to think about the

simulation that I would be presenting to my fellow participants. I studied how to set up a dedicated server and how I could act using a “God mode” from the console. It was clear that I could carry out a Scrum simulation based on this game. I started to think about Scaled Agile Framework with this concept but couldn’t get my head around how different teams could utilize this on a larger scale. I also didn’t like the fact that the game was somewhat too hard for a beginner.

The idea behind DST Scrum simulation was good and I wanted to use a similar approach with a different multiplayer game. I started to gather information about other games that could be used to carry out a simulation. I thought about games like Minecraft, Stationeers and Satisfactory. Games that are open and give players the freedom to build and use their own minds as they struggle to put their ideas into the game world. Christensen and Paasivaara (2022) also considered using Minecraft (Mojang 2009-2022) in their simulation because of its popularity and Lego-like nature, but it was dropped because there was not enough experience with the game. This was also one of the reasons in my case. I also considered Stationeers (Rocketwerkz 2017-2022), which is a game where players try to survive in harsh conditions (e.g., Mars) by building a station or base using resources that can be found on the planet's surface. This game was dropped because of its complexity, there wouldn’t be enough time for the participants to get familiar with game mechanics. Satisfactory (Coffee Stain Studios 2016-2022) is a first-person open-world factory-building game where players extract resources from the environment and process them into materials, which are then reprocessed into more advanced parts. I had personally played Satisfactory for over 500 hours, so I had a good understanding of the game mechanics and open-world concepts that the game contained. With my previous knowledge of the game, I was confident enough to start designing the actual simulation.

4.2 Simulation goals and challenges

As stated in chapter 3.2.3, the main goal of the simulation was to create a simulated experience for the participants so that they could learn Scaled Agile Framework in practice. It was also stated that in practice, the training part of the simulation needed to be more efficient than running a traditional 8-12 weeks of Program Increment in SAFe. It was decided that the learning period of the simulation would be 3 weeks, with effective use of the time

of 8-12 hours/development resource. To accomplish this, some parts of the Scaled Agile Framework needed to be put aside or shortened. The main goal for the participants was to get into SAFe and see the process in practice.

One of the key factors of a simulation based on serious games is to enable a favorable environment for the participants to learn. A favourable learning environment can be achieved if the participants are willing and open to learning new things and have a sufficient understanding of the game mechanics used in the simulation. However, the game of the simulation dictates the mechanics built into it, and this cannot be taught except by playing the game itself. Therefore, this means that participants must have some prior knowledge of the game used in the simulation. This is one of the challenges of running a serious game-assisted simulation, the lack of previous experience with the game itself.

4.3 Serious games and software development

To connect the serious games method of learning to software development, I started to think about the elements in the game that could be used to substitute concepts of software development. The basic idea for using serious games for learning in software engineering is to substitute the development part of software development with actions carried out in play space. This means that implementation (coding), testing (verification) and presentation (demo) parts are substituted with play space elements. Design and interaction between participants would remain the same as they would be with an actual software project.

I wanted to create a simulation where participants could learn basic concepts of the Scaled Agile Framework and some additional aspects of software development that have no direct relation to the framework itself. The idea was to create a simulation space where participants could build factories that have specific requirements, as specifications would normally be defined in software development. I created features that present SAFe stakeholder needs. These needs included different production pipelines (Figure 8) which were related to different resources: Iron, Copper, Concrete, Power source, Storage and an advanced Power source that would be a self-sustaining source of electricity. Features were designed to include dependencies (Figure 14) between teams implementing these features, e.g., Iron, Copper and Concrete production features had dependencies with the power source and storage solution.

Production does not run without electricity and factories needed to have a place to store produced products.

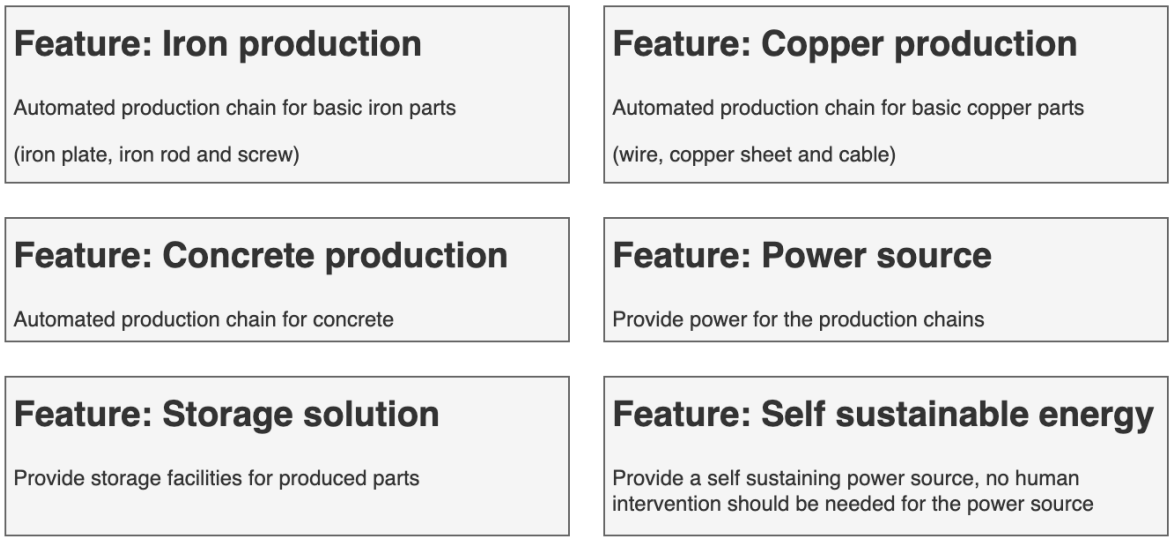


Figure 8. Features designed for the simulation

An additional aspect of non-functional requirements (NFR) from software development was included in the simulation. Included NFRs were related to Security, Safety, Maintainability and Extendibility (Figure 9). This restricted participants to think about and design their factories in more detail. Factories needed to have foundations, walls, roofs and at least one door to conform Security aspect of NFRs. Factory corridors needed to have enough room for an inspection, and they needed to be designed in a way that they could later be extended or upgraded. When factories contained exterior or roofs where players could walk, they needed to have railings to conform Safety aspect of NFR's.

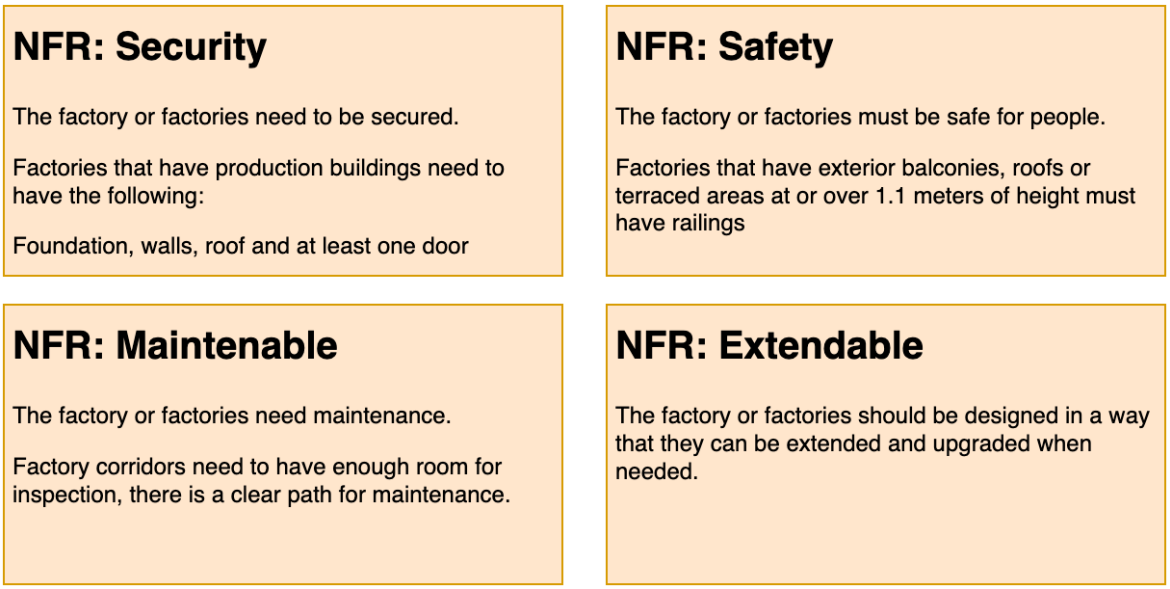


Figure 9. Non-functional requirements designed for the simulation

Each of these main pipeline features contained one or more story-level items that presented specific resource needs for the foreman (Figure 10). E.g., the Iron production feature included production factories for Iron Plates, Iron Rods and Screw resources.

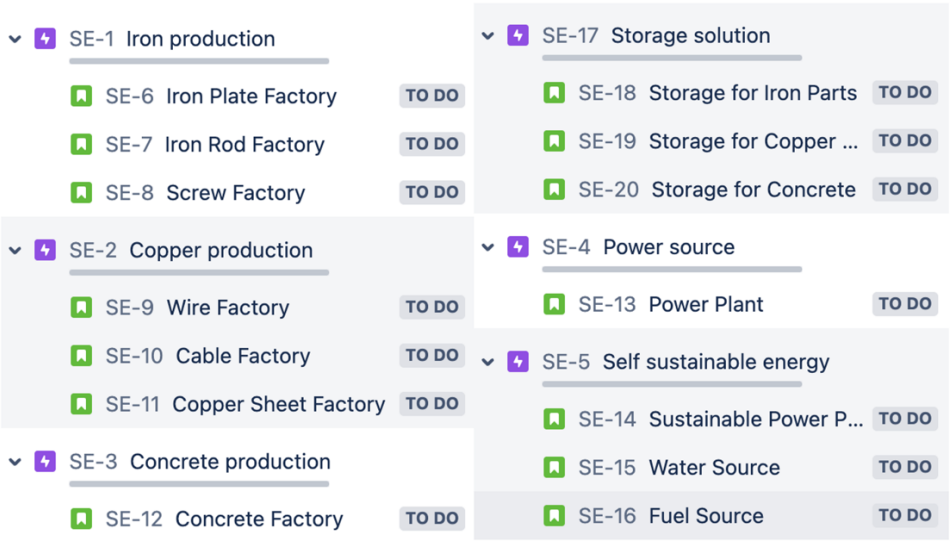


Figure 10. Features and their stories written out

In general, this meant that each story described a resource need from the foreman. E.g., the Iron Plate resource need was described in a story with a description “As a Foreman, I want an Iron Plate Factory that produces resources 100 units/minute so that our daily target quota will be met.” (Figure 11). These stories also included a list of items (Definition of Done) that needed to be done for the story to be considered done.

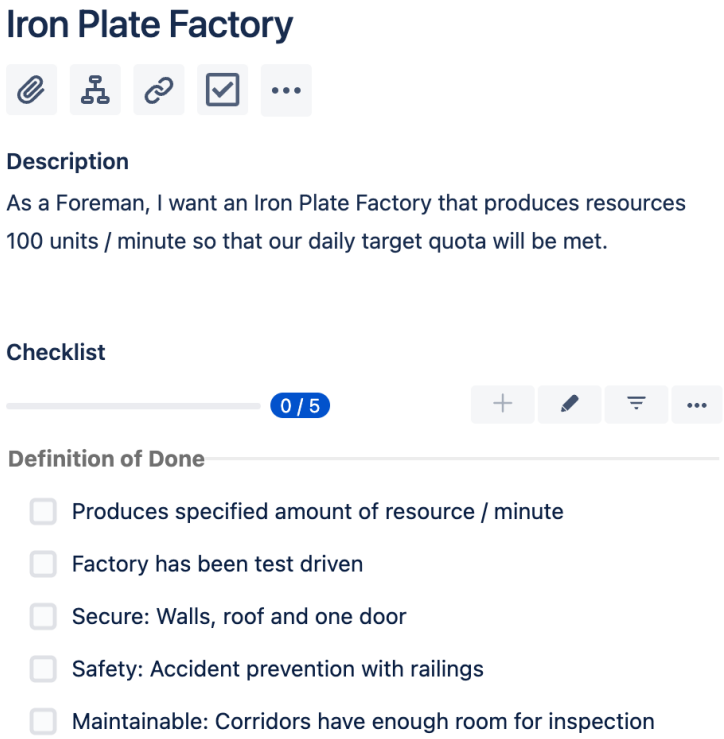


Figure 11. Template for stories with Definition of Done

As stated in the previous chapter, one major issue of running a simulation with serious games is the previous game knowledge of participants in the simulation, in this case, the knowledge of a Satisfactory game. This is quite problematic because it can take several hours to become familiar with the game mechanics. To overcome or alleviate this challenge, I didn't want the simulation to start from scratch. I wanted to create a more approachable start for the simulation for all participants. Therefore, I prebuilt the simulation game world with my 13-year-old son to a more favourable point where everything was not required to be crafted by hand. We crafted some resources for the simulation beforehand, so that these resources could be used to start building the necessary factories depicted by SAFe features and stories.

4.4 First iteration of simulation

It was planned that the first iteration would be used to learn the Scaled Agile Framework in practice. Initially, it meant the whole simulation would be carried out synchronously in one or two sessions that would last 3-4 hours. It was soon realized that in practice this would be difficult to arrange given the current workload of the development team. The gameplay part of the simulation was decided to be carried out asynchronously. I still felt that the simulation needed some synchronisation, especially in the planning part. This resulted in the simulation being run in mixed methods where the teams would meet for planning but would otherwise work in smaller teams. In practice, this meant that the simulation contained synchronized events for introduction, planning and retrospective, and asynchronized events that were used for gameplay and demos, just like in the real world.

4.4.1 Preliminary session

People in the organisation were invited to the simulation via email and through Teams chats. The preliminary session for the simulation was held with volunteer participants. There was a total of 8 volunteers participating in the preliminary session. The session started with an introduction to the simulation using serious games, in this case, a simulation which is run with a Satisfactory game, factory management and planning game. After the introduction, I presented the first steps needed to do before the actual simulation can begin (Figure 12). It was planned that the Satisfactory game would be purchased via Steam, so a Steam account was needed for all participants. While participating in the preliminary session, participants created a Steam account or indicated that they would be using their existing Steam account. Every participant sent me their Steam ID so I could send them the game using a Steam gift.

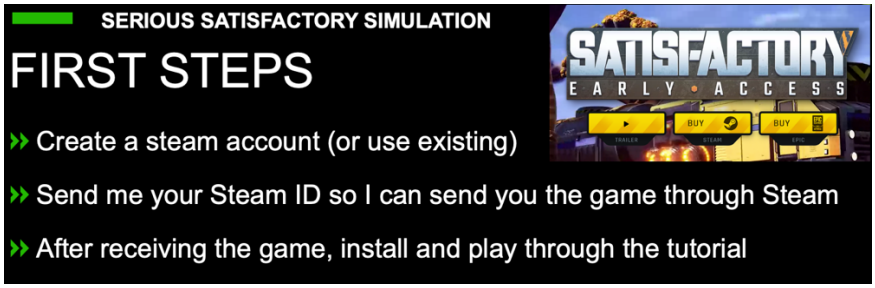


Figure 12. First steps presented at preliminary session

After receiving the game, I instructed the participants to install Steam and the Satisfactory game on their PC and play the game tutorial through. A satisfactory tutorial consists of a "TIER 0: Onboarding", playing the game through the first steps that included HUB upgrades 1-5. Basically, participants gathered resources and build small-size factories to get the feel of the game before the simulation. The preliminary session was concluded by presenting the next steps for the simulation: an introduction and practicalities session that would present the actual simulation part and practicalities. After the session, one participant did not participate in the simulation. The simulation was therefore conducted with participants of 7 volunteers.

4.4.2 Simulation introduction

Next week after the preliminary session, when participants had spent some time with tutorials, a simulation introduction session was held. In this session, I presented how serious games are utilized to perform simulation and how the implementation part of software development is replaced by building tasks in a gaming environment. I presented the goals of the simulation that included topics of learning SAFe in practice and how to utilize Spike type of Story while providing an alternative learning experience for participants. Simulation contained synchronous events for intro, planning and retro, and asynchronous events for gameplay and iteration demos. Simulation progress was tracked using Jira Cloud where participants created their accounts and joined my simulation instance. I presented the simulation schedule (Figure 13) which indicated the timing for each event of the simulation.

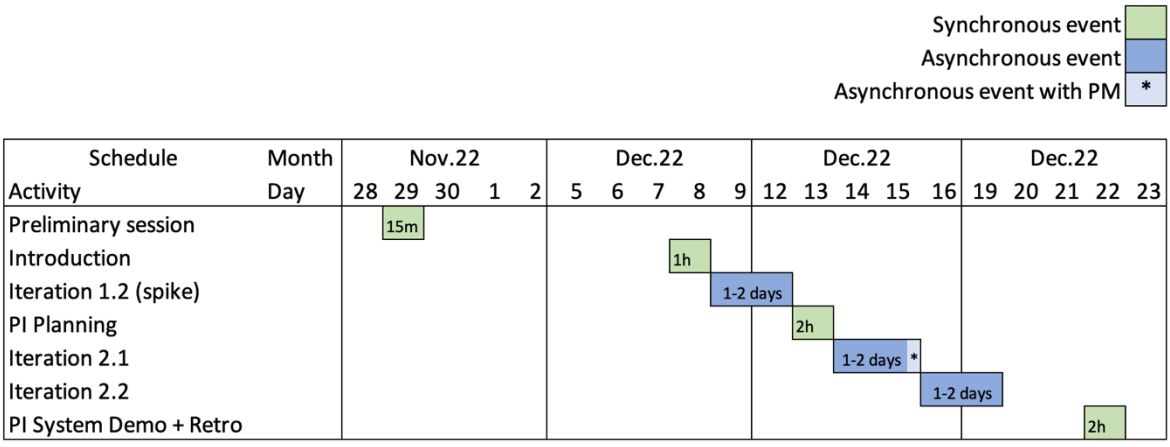


Figure 13. Simulation schedule

Instructions for team formation were provided for participants. The idea was to create teams of 2-3 people, participants decided to form 2 teams of 3-4 people. The simulation instance was already set to run on a dedicated server, meaning that it was going to be always on for the duration of the simulation timeframe. Details on how participants could connect to a game instance were provided during the practicalities part of this session. The session ended with notes on the next steps, “start gathering enough information so that next week PI Planning would be effective as possible”.

4.4.3 PI Planning

The program Increment Planning event started with an introduction to the business context and product vision. I presented the story behind the Satisfactory game. I was acting in the roles of Program Manager, System Architect, Product Owner and Agile Coach for both teams.

“FICSIT Inc. is building a top-secret project in space called Project Assembly, which requires resources to be produced at the surface of the planet. Part factories built by participants play a key role in the success of the project.”

The event continued with Architecture vision and development practicalities that connected another software engineering aspect to the simulation, Microservices architecture. This meant that Part factories needed to be small as possible and independent, materials are the input for the factory and the output is the produced parts. The Architecture vision also included non-functional requirements described in section 4.3 (Figure 9).

After the business context and architecture vision were presented, teams started their work in team breakouts. Both teams were divided into different rooms where they could start deciding and designing what features to implement. Teams planned their work and presented their scheduled plan (Figure 14) which also indicated dependencies between teams.

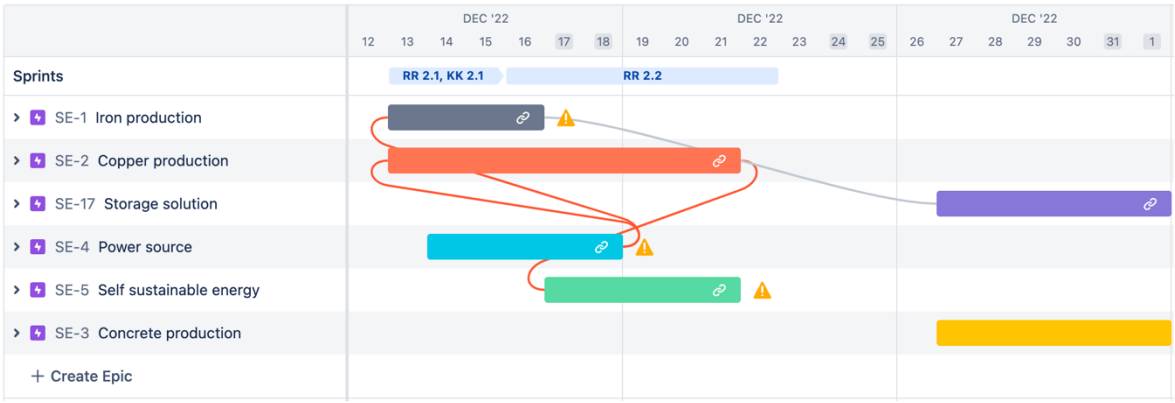


Figure 14. Dependencies between features

4.4.4 Iteration review

After the PI Planning, teams had time to work on their implementation in the simulation for 2 days. An asynchronous event for both teams was held that was meant to carry out an iteration review. This was used to provide feedback, not just about the implementation but also for the technical difficulties related to game mechanics that some participants were experiencing. This event was held only once for time consumption reasons for both teams. As roles described in the previous chapter, I gave out feedback as Product Owner to both teams during these sessions.

Figure 15. Iteration review for one of the teams

4.4.5 Inspect and Adapt

Teams had one week left after the iteration review session to complete the work that they had planned. During the System Demo event, I was acting in the role of Product Owner and presented completed work for both teams. During this event, we went through each factory that participants had built and how the factory layout was created. Both teams managed to build their assigned features and work and therefore each participant was granted a virtual cup of coffee (Figure 16) that is only bought in-game using tickets acquired in-game.

Figure 16. System Demo with a cup of coffee

Inspect and Adapt session was held at the same session after System Demo. In this session, I created a retrospective board for all participants to give out feedback on what went well, what caused problems and what can be done differently. Participants responded that PI objectives were clear, and the PI planning event was successful. Necessary support was available for variating problems. They also wrote that the game choice was successful. Participants indicated that there were too few teams for scaled agile simulation and that there just wasn't enough time to participate in the simulation that they would have hoped for. Participants indicated that next time familiarization with work tools and information sharing could be improved and there should be a realistic (Ironman mode) where players are not allowed to build stairs that reach the skies.

5 Results

This chapter presents the results of the research. The first section presents the results of the survey of preliminary field research. The second chapter briefly goes over the results from initial prototype testing. The third and final chapter presents serious games simulation survey results.

5.1 Preliminary field research

The survey got 16 submitted responses out of 40 participants of the organisation. 6 submitted responses were given based on the first survey email, and 10 more submitted responses were given after the reminder email. Question Q1 results (Figure 17) indicated that 11 of the responses were given by Agile team members, 3 by Scrum masters, 0 by product owners, and 2 by having another role in the organisation. A full list of responses to the survey questionnaire can be seen in Appendix 2.

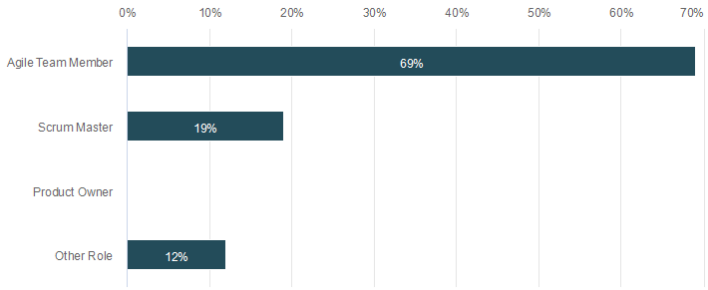


Figure 17. Preliminary survey question Q1 responses

Question Q2. How does your team communicate: Participants responded that communication with the customer or his/her representative is lacking the most compared to other communication-related questions. Few participants felt that every team member isn't aware of the tasks of other team members. There were few responses that indicated that members aren't provided with enough appreciation for their work. Team meetings and communication is seen as the most agile within the team.

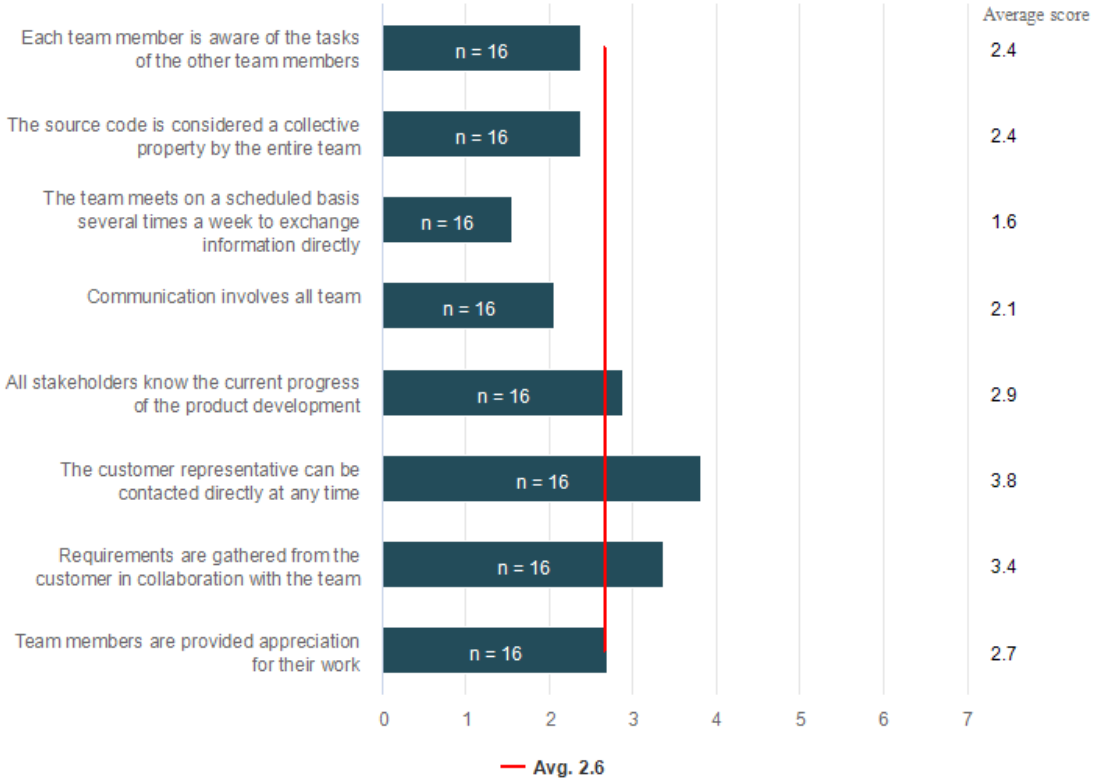


Figure 18. Preliminary survey question Q2 responses

Question Q3. What is the attitude for change in your team: Participants responded that iteration does not always end with the delivery of a working product to the customer. They also feel that the customer does not regularly inspect the business outcome of the delivered product. There was a strong agreement that the customer requirements can be adapted with an alternative solution.

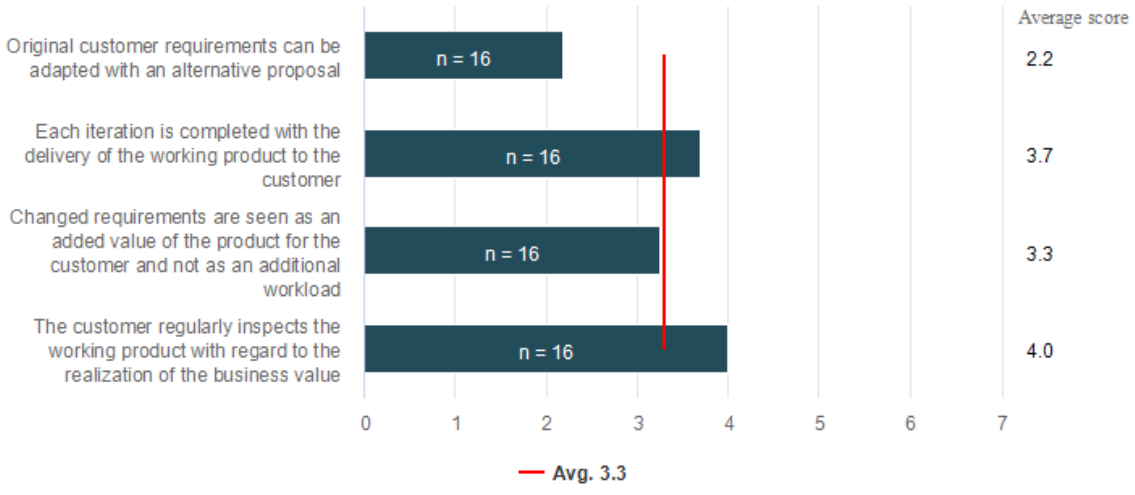


Figure 19. Preliminary survey question Q3 responses

Question Q4. How does your team experience iterative development: Some of the participants feel that development cannot be started without fully defining the requirements and that the detailed plan for development isn't made just before the next iteration.

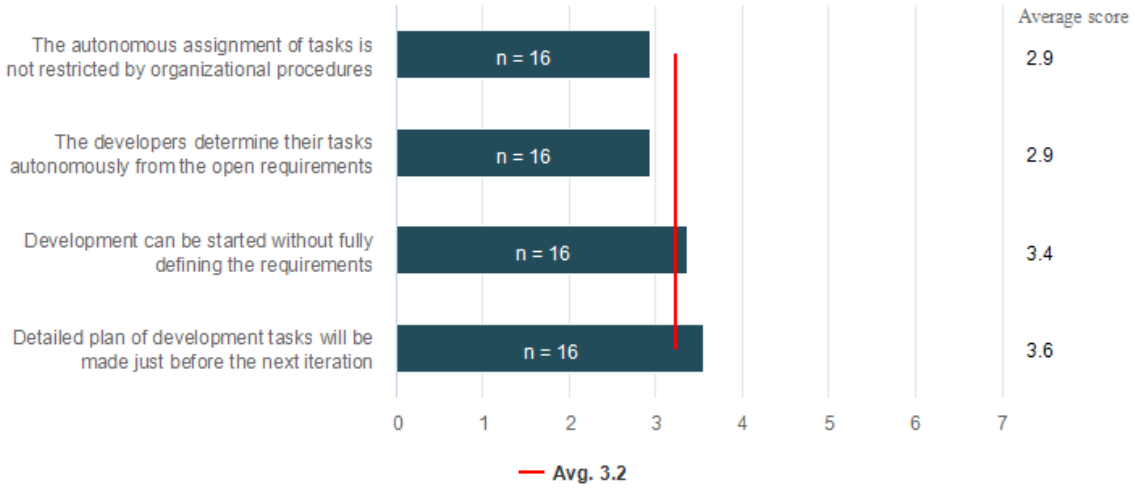


Figure 20. Preliminary survey question Q4 responses

Question Q5. How extensively is your team self-organized: Most of the participants see that they are self-organised, but few participants feel that some decisions regarding the execution of their work cannot be made within the team level.

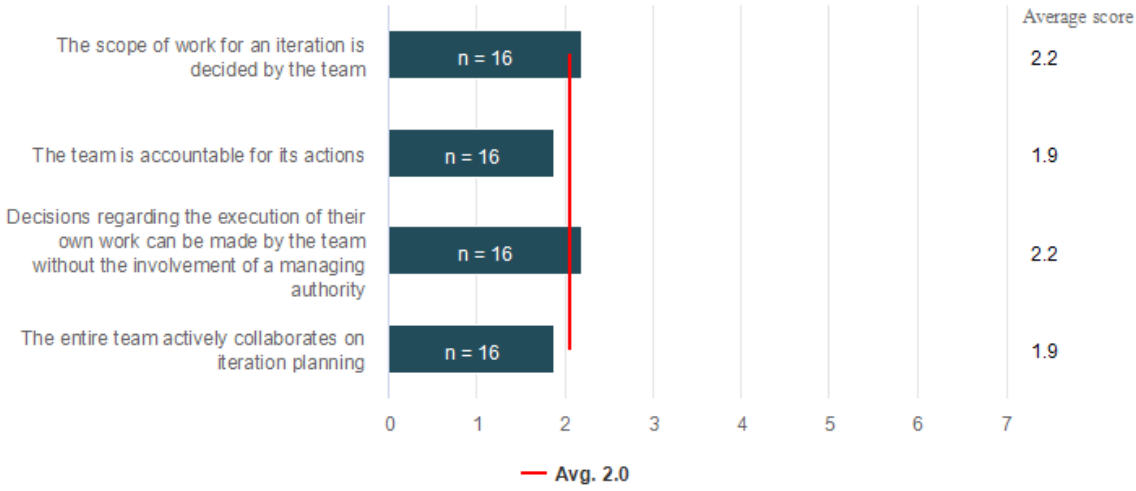


Figure 21. Preliminary survey question Q5 responses

Question Q6. How product driven the team is: Participants do not see that customer representative is involved with decisions related to features. Some of the participants see that subject matter experts are not involved in requirement identification as they should be. Some

participants feel that overtime is sometimes used to carry out the work in iterations. Documentation and its value are dividing the responses on both ends.

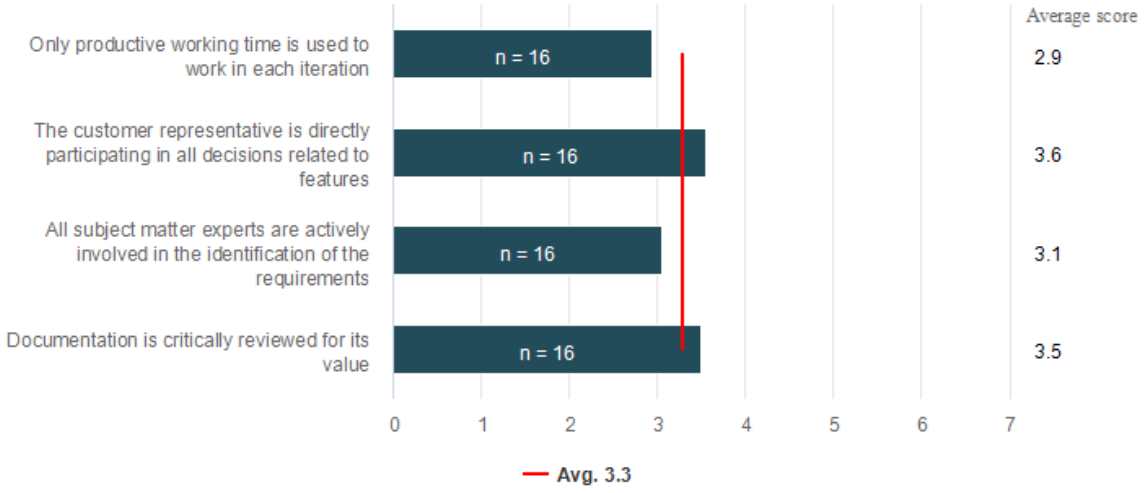


Figure 22. Preliminary survey question Q6 responses

Question Q7. Does your team continuously improve the development process: Participants responded that retrospectives are regularly used to improve the development process. Some participants see that everybody isn't actively participating in process improvement. Some see that the retrospective insights aren't turned into concrete improvement measures. There was a strong feeling from some of the participants that improvements cannot be explored experimentally during the process.

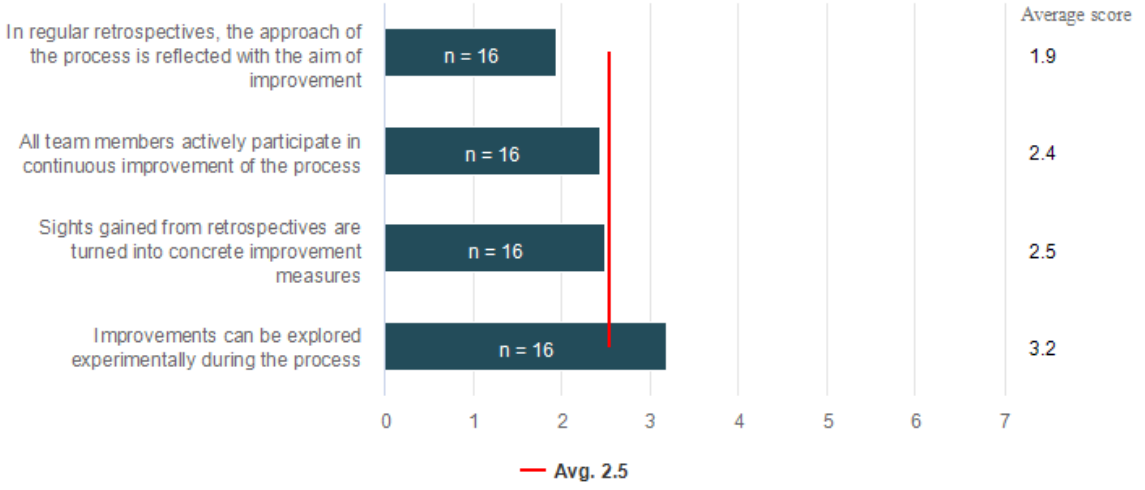


Figure 23. Preliminary survey question Q7 responses

Question Q8. How important do you see agile development principles: Participants mainly responded that they see agile development principles as important. There were few responses that were neutral about the communication frequency, that the product should be developed in several iterations, and that the agile team should operate autonomously as a self-organising team.

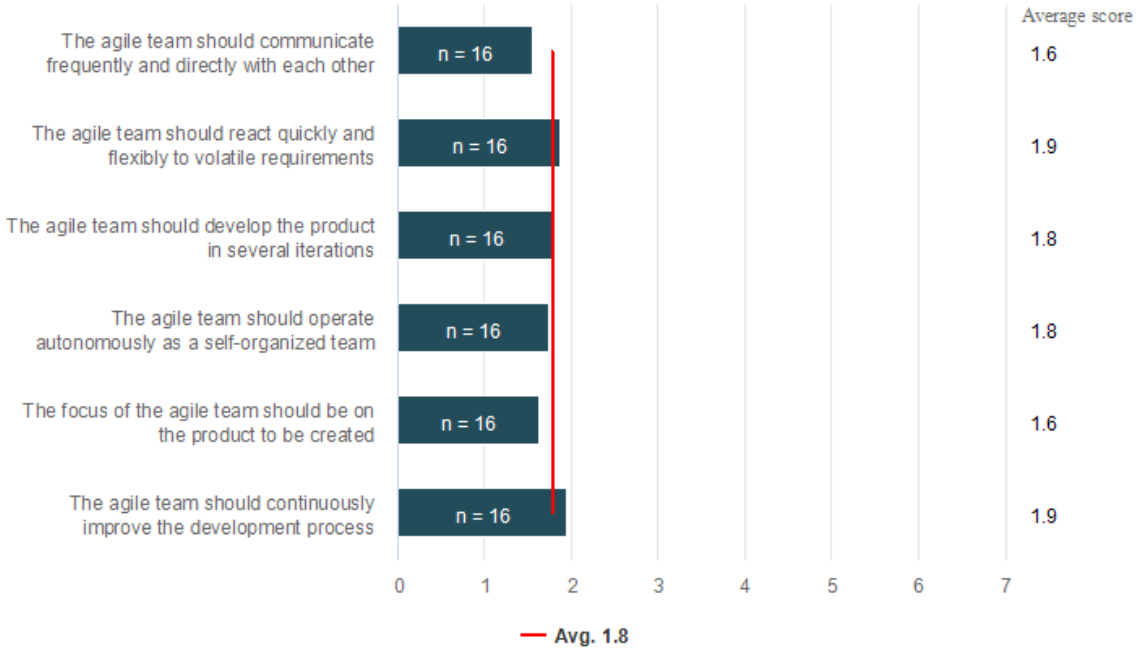


Figure 24. Preliminary survey question Q8 responses

Question Q9. Feel free to share your own views on team agility or thoughts from the survey (optional): There were 3 submitted responses to the last question, which was an open ended question (Table 4). Participants see that there is still a lot to learn, and some agile ceremonies are questioned within the team. One response indicated that there isn't enough feedback from the customer. One participant feels that they are not given enough time to work on the agile development project.

Table 4. Responses to preliminary field research survey question Q9

Responses in English (translated)
The principles and benefits of agile development are fairly well digested within the team, but the change from the old takes time and the need for certain ceremonies are questioned.

The team is adaptable and receptive to feedback. There is still a lot to learn about the methodology through trial and error. Customer feedback is lacking, and this is where there is room for improvement between consultants and developers.
The main problem with agile development now is that team members are involved in many other projects at the same time. These other projects have been prioritised over the agile development project, so this project is being done when there is not enough time for other projects. The agile project is moving forward mainly because a few contributors are enthusiastic about it and are willing to work long hours and dedicate time to it, even on weekends.

Based on the submitted responses in general, customer interaction and collaboration was seen as the most critical part to improve in terms of agility. It should be made clear that iterations end with the delivery of a working product to a customer. The customer or representative should inspect the business value gained from the delivery. It should be made aware that the development work could be started without comprehensive and detailed requirement documentation and that the development should be iterative, creating value for the customer in as small pieces as possible. Also based on the responses, it should also be made clear to the team that they have the power to make the executive decisions on the features described by the customer representative. Documentation and its value should be made clear to the team. Retrospective insights should result in concrete steps that would improve the development process in the future. It should be made clear to the team to experimentally explore new things and that making errors during the process is allowed, it is all about learning new things and improving the development process. The idea should be made clear that the team is able to adjust the process and communication preferences on their own as they see fit. For the issue of resource management that is out of the team level, the business owners should be made clear that team members should be dedicated to working on the tasks defined in the agile organisation. Working overtime to allow agile work to be carried out is not an option.

5.2 Prototype testing

The first run on the prototype was carried out by myself and my 13-year-old son. The purpose was to gather information if the features that could be completed in the designed timeframe. After I had set up initial features and stories for the prototype, we run the

simulation as our participants would run it. We picked one of the features for each of us and started to build stories related to those features. Initial idea was to run the simulation in fully synchronized mode, where all participants would be running the simulation at the same time. I had set the time for completing the first PI iteration to 30 minutes. Once the simulation was completed with the designed prototype, it became clear that it was too difficult to build all the necessary requirements within the given timeframe. It was decided to include more time to complete each iteration in the simulation. Also, some of the non-functional requirements were missing from the prototype simulation that was added afterwards because of the testing carried out with the prototype.

5.3 Simulation survey results

Survey got 6 submitted responses out of 7 simulation participants. The simulation survey contained 2 close-ended and 1 open-ended question. Question Q1 contained 15 learning topics on how participants felt that they learned from the simulation. Question Q2 contained 17 evaluation topics of the simulation. A full list of responses to the survey questionnaire can be seen in Appendix 4.

Table 6. Responses to simulation survey question Q1

	I didn't learn anything new about this topic in the simulation	I heard about this in the simulation	Due to the simulation I can explain what the topic is	Due to the simulation I learned how to use/apply this topic in	Average	Median
The program backlog	33,4%	33,3%	33,3%	,0%	2,0	2,0
The iteration backlog	16,6%	50,0%	16,7%	16,7%	2,3	2,0
Iterative development	,0%	50,0%	16,7%	33,3%	2,8	2,5
Importance of feedback	16,6%	16,7%	16,7%	50,0%	3,0	3,5
Product vision	16,7%	33,3%	50,0%	,0%	2,3	2,5
Spike (Story)	16,7%	33,3%	33,3%	16,7%	2,5	2,5
Non-functional requirements	,0%	33,3%	50,0%	16,7%	2,8	3,0
Definition of Done	33,3%	33,3%	16,7%	16,7%	2,2	2,0
Monitoring project progress	16,7%	33,3%	50,0%	,0%	2,3	2,5
Selecting/assigning work	16,7%	33,3%	33,3%	16,7%	2,5	2,5

Working in Agile team	16,7%	16,7%	33,3%	33,3%	2,8	3,0
Collaborating with other Agile teams	,0%	16,7%	83,3%	,0%	2,8	3,0
Conducting PI Planning	33,4%	33,3%	33,3%	,0%	2,0	2,0
Conducting System Demo	33,3%	16,7%	33,3%	16,7%	2,3	2,5
Conducting Adapt and Inspect (PI Retrospective)	16,7%	33,3%	50,0%	,0%	2,3	2,5

Q1 Topic 1. **The program backlog**: Participants responded with mixed answers ranging from not learning anything about the program backlog to understanding it enough to be able to explain what the program backlog is.

Q1 Topic 2. **The iteration backlog**: Participants indicated somewhat mixed results with most of the participants answering that they heard about iteration backlog in the simulation.

Q1 Topic 3. **Iterative development**: Participants indicated that they heard about it in the simulation and half of the responses indicated that participants could apply iterative development now due to the simulation.

Q1 Topic 4. **Importance of feedback**: Participants responded with mixed responses from not learning anything to understanding and applying the importance of feedback in practice. Half of the responses indicated that they can now apply this topic in practice due to the simulation.

Q1 Topic 5. **Product vision**: Participants indicated that they heard it in the simulation and due to the simulation, they can explain what the product vision is and what it is used for.

Q1 Topic 6. **Spike (Story)**: Participant responses were in the middle range, two of the responses indicated that they heard about the topic in the simulation and two indicated that they can explain what a spike type of story is.

Q1 Topic 7. **Non-functional requirements**: Participants responded that they heard it in the simulation and most of the participants can now explain what non-functional requirements are due to the simulation.

Q1 Topic 8. **Definition of Done**: Participants responded with mixed answers mainly indicating that they didn't learn anything new, or they heard about the definition of done in the simulation.

Q1 Topic 9. **Monitoring project progress**: Participants responded in the middle range; half of the responses indicated that they can now explain how the project progress is monitored due to the simulation.

Q1 Topic 10. **Working in Agile team**: Participants indicated mainly that due to the simulation, they can explain the basic principles of working in an agile team or that they can now apply this topic in practice due to the simulation.

Q1 Topic 11. **Collaboration with other Agile teams**: Participants responded to this question more consistently than any other question. 5/6 participants responded that they feel that due to the simulation, they can now explain what collaboration with other Agile teams mean

Q1 Topic 12. **Conducting PI Planning**: Participants responded with mixed answers from not learning anything new to explaining how to conduct PI Planning.

Q1 Topic 13. **Conducting System Demo**: Participants responded with mixed answers, mostly indicating either not learning anything new or now being able to explain how to conduct a System Demo event.

Q1 Topic 14. **Conducting Adapt and Inspect (PI Retrospective)**: Participant responses indicated that they heard about Adapt and Inspect during the simulation and half of the responses indicated that they can now explain how an Adapt and Inspect event can be conducted.

Answers to the simulation evaluation question Q2 topics indicated that participants mostly agreed that they felt confident that they were learning during the simulation, the simulation was an effective way to learn, the simulation content was connected to the knowledge they already had and that the variations in the simulation helped them to keep their attention to the simulation. Participants felt that the simulation progressed at an adequate pace, and it did not become monotonous. They also agreed that the way the simulation works suited their way of learning. Participants also indicated liking the simulation design and that it was easy to understand how the simulation was related to software development. Participants indicated that they would recommend the simulation to other people.

Table 7. Responses to simulation survey question Q2

	strongly agree	agree	neutral	disagree	strongly disagree	Average	Median
Passing through the simulation, I felt confident that I was learning.	33,3%	50,0%	16,7%	,0%	,0%	1,8	2,0
The simulation content was connected to other knowledge I already had.	50,0%	16,6%	16,7%	16,7%	,0%	2,0	1,5
The way simulation works suited my way of learning.	33,3%	50,0%	,0%	16,7%	,0%	2,0	2,0
The simulation content was relevant to my interests.	16,7%	16,7%	33,3%	33,3%	,0%	2,8	3,0
The variations in the simulation helped me to keep my attention to simulation.	16,6%	50,0%	16,7%	16,7%	,0%	2,3	2,0
The simulation was easy to get into, game tutorial was all I needed.	,0%	16,7%	33,3%	16,7%	33,3%	3,7	3,5
The simulation design was attractive.	66,6%	16,7%	16,7%	,0%	,0%	1,5	1,0
It was easy to understand how the simulation relates to software development.	50,0%	33,3%	16,7%	,0%	,0%	1,7	1,5
The simulation was an effective way to learn.	33,3%	50,0%	,0%	16,7%	,0%	2,0	2,0
I would like to play this simulation again.	33,3%	33,3%	16,7%	16,7%	,0%	2,2	2,0
I would recommend this simulation to other people.	66,7%	33,3%	,0%	,0%	,0%	1,3	1,0
I would have liked to play the simulation for a longer time.	50,0%	16,6%	16,7%	16,7%	,0%	2,0	1,5
The simulation progressed at an adequate pace and did not become monotonous.	16,7%	50,0%	33,3%	,0%	,0%	2,2	2,0
I was confused during the simulation.	16,7%	33,3%	,0%	33,3%	16,7%	3,0	3,0
I was frustrated during the simulation.	16,7%	16,7%	,0%	33,3%	33,3%	3,5	4,0
This simulation was properly challenging, it was not too easy nor difficult.	16,7%	16,7%	33,3%	33,3%	,0%	2,8	3,0
I active participated in the reflection event after the simulation.	33,3%	,0%	16,7%	16,7%	33,3%	3,2	3,5

Mixed answers from the participants were received for a few of the questions. Some felt that they were confused and frustrated during the simulation, and some felt the opposite. The simulation was seen as properly challenging by a few participants but most of the answers indicated that it was too difficult. Some of the participants actively participated in the reflection event. Most of the participants would have liked to play the simulation for a longer time, but only some of the participants would have liked to play the simulation again.

Participants mostly disagreed or strongly disagreed that simulation content was relevant to their interests. They also indicated that the simulation was not easy to get into, and the tutorial didn't cover all the basics that they needed to learn before the simulation.

The survey questionnaire included 1 open-ended question Q3, to which every survey participant responded (Table 6). There were indications from participant responses that the simulation felt too rushed, which should be refined for future simulations. Also, it came clear that more tutorials or onboarding is needed for the simulation to be successful. Still, half of the participants responded that they felt that the simulation was a fun and interesting way of learning agile methodology.

Table 8. Responses to simulation survey question Q3

Responses in English
This simulation was quite fun way to learn agile concepts. I can easily see the use in other educations as well.
Happy to learn anything new
Through the game you will learn the SAFe model and terminology in software development
I would have liked to have more time to prepare and to better fit the events and game playing time in my schedule. Now everything felt rushed and it felt like there was no time to think, only do. Which often is the reality but is not an effective way of learning for me. Like in real life, a lot of my game-play time was spent running around like a headless chicken, jumping off cliffs injuring myself while trying to find everyone else :D
I didn't really have any experience with games before this, so even the tutorial was challenging and time-consuming. "What LMB?" That's the kind of acronym that SAFe and the whole IT world is full of.

There were many new things for me, the game, Discord... so I almost panicked. I also didn't test my laptop before the sessions, so some of the fun was missed. I think this simulation would work well for people who are more familiar with games. It's easier to learn new things when you have something familiar to work with. The SAFe terms become very concrete with this and it was easy to see how many aspects of the simulation was related to software development, such as not knowing anything about the subject to begin with. A fun way to learn and this could even mitigate resistance to change.
Overall, I felt that this way of learning about agile methodology was interesting and simulation was a good way to link the theory with action. I liked it and would prefer this over a classroom lecture for sure. This was obviously a very short test run, so there was no time to concentrate to all aspects thoroughly, but given a bit more time, I believe this would be very efficient way of learning.

6 Discussion

This chapter presents discussion of this research results. The first section presents research questions and their relation to previous research. The second section explains the limitations of this study. This chapter ends with future research considerations.

6.1 Research questions

RQ1: How to create a serious game that allows users to learn SAFe?

In this research, I designed and evaluated a serious games simulation that was built on top of a PC game, Satisfactory. A simulation play space was created for participants to plan and carry out simple but concrete tasks. In the simulation, participants were able to learn various aspects of the Scaled Agile Framework. The simulation that I presented was in line with already previous studies of “Respond to Change or Die” by Christensen and Paasivaara (2022) and “Scrum Lego simulation game” by Paasivaara et al. (2014). These previous studies were meant for learning Scrum in practice, this study extended the learning to the Scaled Agile Framework.

RQ2: How do the members of the SAFe organisation experience serious games as a learning method?

Participant reaction to the simulation was mainly positive. Some members of the SAFe organisation saw serious games as something new, inspiring and an alternative way for learning when properly designed and conducted. The main issue that arises from utilizing serious games is the lack of previous knowledge of the game mechanics as mentioned by Christensen and Paasivaara (2022) as well.

RQ3: What SAFe-related features do participants learn from serious games simulation?

This study indicated that through the simulation, participants learned topics of 1) collaboration with other agile teams, 2) monitoring project progress, 3) non-functional requirements and 4) the importance of feedback in iterative development. These findings correlate with a previous study by Christensen and Paasivaara (2022).

6.2 Limitations

This work was limited to the case organisation and the simulation was held with 7 members of the organisation, which in this study limits the learning experience of the simulation in the context of large-scale agile and the responses received for the final simulation survey questionnaire.

One of the major limitations of this study was the resource utilization of simulation participants. This was also evident from the survey responses given by the simulation participants. This limitation was also known before the simulation and was mitigated through simulation state building where participants didn't start the simulation from scratch but from a more approachable state.

For these reasons, the results of this research are limited to a small number of samples.

6.3 Future research

While the serious games simulation presented in this research did cover some learning topics relatively well, it didn't cover other topics enough to be considered a ready-to-be-used

solution as its current form. Therefore, I would recommend further research to be conducted based on the findings of this research.

7 Conclusion

This chapter concludes this research. In this study, I presented an alternative solution for supporting participants' learning and adopting the Scaled Agile Framework. While this research indicates that there were problems related to the timetable of participants and not necessarily providing new information for seasoned ones, this research shows that serious games can be utilized to support the learning of software development processes like the Scaled Agile Framework through playful simulation.

This study indicated that through the simulation, participants learned topics like collaboration with other agile teams and the importance of feedback in iterative development among other topics. The results of this study indicate that serious games can be a valuable tool for learning Scaled Agile Framework and other related software development activities when set up and conducted properly.

One, not that obvious beneficial aspect of the simulation, is the relationship and trust-building element between participants. The simulation was run with participants during off-hours, which disconnected them and their colleagues from a standard working environment. Running in a game with your team members and building things, or fighting venomous spiders together really brings out the real you. This kind of interaction and play spaces bond team members together in a way that wouldn't normally happen in a working environment.

References

Sidky, A.S., 2007. A structured approach to adopting agile practices: The agile adoption framework (Doctoral dissertation, Virginia Tech).

Putta, A., Uludağ, Ö., Hong, S.L., Paasivaara, M. and Lassenius, C., 2021, October. Why Do Organizations Adopt Agile Scaling Frameworks? A Survey of Practitioners. In Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) (pp. 1-12).

Dikert, K., Paasivaara, M. and Lassenius, C., 2016. Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software, 119, pp.87-108.

Kuss, D.J. and Griffiths, M.D., 2012. Internet gaming addiction: A systematic review of empirical research. International journal of mental health and addiction, 10(2), pp.278-296.

Connolly, T.M., Boyle, E.A., MacArthur, E., Hainey, T. and Boyle, J.M., 2012. A systematic literature review of empirical evidence on computer games and serious games. Computers & education, 59(2), pp.661-686.

CGI (2023) Company Overview [online]. Available at: <https://www.cgi.com/en/overview> [Accessed: January 8, 2023].

CGI Profio360 (2023) CGI.com [online]. Available at: <https://www.cgi.com/en/solutions/cgi-profio360> [Accessed: January 8, 2023].

Royce, W. W. (1987, March). Managing the development of large software systems: concepts and techniques. In Proceedings of the 9th international conference on Software Engineering (pp. 328-338)

Šmite, D., Moe, N., Šāblis, A. and Wohlin, C., 2017. Software teams and their knowledge networks in large-scale software development. Information and Software Technology, 86, pp.71-86.

Looks, H., Fangmann, J., Thomaschewski, J., Escalona, M. and Schön, E., 2021. Towards a Standardized Questionnaire for Measuring Agility at Team Level. Lecture Notes in Business Information Processing, pp.71-85.

Rodríguez, P., Mäntylä, M., Oivo, M., Lwakatare, L. E., Seppänen, P., & Kuvaja, P. (2018). Advances in Using Agile and Lean Processes for Software Development. Advances in Computers.

Fowler, M. and Highsmith, J., 2001. The agile manifesto. Software development, 9(8), pp.28-35.

Agile Alliance. 2022. What is Scrum? [online]. Available at: <https://www.agilealliance.org/glossary/scrum/> [Accessed 16 October 2022].

Scaled Agile Framework (2022). SAFe 5.1 Framework [online]. Available at: <https://www.scaledagileframework.com/> [Accessed 10 December 2022].

Kittlaus, H., & Fricker, S. (2017). Software Product Management. https://doi.org/10.1007/978-3-642-55140-6

Ebert, C., & Paasivaara, M. (2017). Scaling Agile. IEEE Software, 34(6), 98-103. https://doi.org/10.1109/ms.2017.4121226

Paasivaara, M., Heikkilä, V., Lassenius, C., & Toivola, T. (2014). Teaching students scrum using LEGO blocks. Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014. doi:10.1145/2591062.2591169

Digite (2022). 6 Scaled Agile Frameworks - Which One Is Right For You? [online]. Available at: <https://www.digite.com/blog/scaled-agile-frameworks/> [Accessed 10 December 2022].

Wilkinson, P. (2016). A Brief History of Serious Games. In: Dörner, R., Göbel, S., Kickmeier-Rust, M., Masuch, M., Zweig, K. (eds) Entertainment Computing and Serious Games. Lecture Notes in Computer Science(), vol 9970. Springer, Cham. https://doi.org/10.1007/978-3-319-46152-6_2

Emslie, A. and Mesle, C.R., 2009. Play: The use of play in early childhood education. Gyanodaya: The Journal of Progressive Education, 2(2), pp.1-26.

Pellegrini, A.D. and Bjorklund, D.F., 2004. The ontogeny and phylogeny of children's object and fantasy play. Human Nature, 15(1), pp.23-43.

Boucher, S., Downing, J. and Shemilt, R., 2014. The role of play in children's palliative care. Children, 1(3), pp.302-317.

Resnick, M., 2007, June. All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten. In Proceedings of the 6th ACM SIGCHI conference on Creativity & cognition (pp. 1-6).

Carver, N.K., 1986. Reading readiness: Aspects overlooked in structured readiness programs and workbooks. Childhood education, 62(4), pp.256-259.

Froebel, F. (2017). Revival: Autobiography of Friedrich Froebel (1915) (12th ed.). Routledge. https://doi.org/10.4324/9781315122489

Ritterfeld, U., Cody, M.J. and Vorderer, P. (2009) Serious games: Mechanisms and effects. New York: Routledge.

Agile Parrot (2020). LEGO Scrum - cheat sheet, step by step [online]. Available at: <https://agileparrot.com/agile-workshops/lego-scrum/> [Accessed 10 December 2022].

Christensen, E.L. and Paasivaara, M. (2022) "Respond to change or die," Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Software Engineering Education and Training [Preprint]. Available at: https://doi.org/10.1145/3510456.3514145.

Brown, N., Nord, R., Ozkaya, I., Kruchten, P., & Lim, E. (2011). Hard choice: A game for balancing strategy for agility. 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T). doi:10.1109/cseet.2011.5876149

Mojang (2009-2022) Minecraft [Video game]. Available at: <https://www.minecraft.net/> [Accessed 9 December 2022].

Rocketwerkz (2017-2022) Stationeers [Video game]. Available at: <https://store.steampowered.com/app/544550/Stationeers/> [Accessed 9 December 2022].

Coffee Stain Studios (2016-2022) Satisfactory [Video game]. Available at: <https://www.satisfactorygame.com/> [Accessed 9 December 2022].

Appendix 1. Preliminary field research survey questions

Table 2. Preliminary field research survey questionnaire in English

Q1. What is your primary role in Agile organisation?	
Role	Agile Team Member
Role	Scrum Master
Role	Product Owner
Role	Other Role
Q2. How does your team communicate?	
Communicative	Each team member is aware of the tasks of the other team members
Communicative	The source code is considered a collective property by the entire team
Communicative	The team meets on a scheduled basis several times a week to exchange information directly
Communicative	Communication involves all team members
Communicative	All stakeholders know the current progress of the product development
Communicative	The customer representative can be contacted directly at any time
Communicative	Requirements are gathered from the customer in collaboration with the team
Communicative	Team members are provided appreciation for their work
Q3. What is the attitude for change in your team?	
Change-affine	Original customer requirements can be adapted with an alternative proposal
Change-affine	Each iteration is completed with the delivery of the working product to the customer
Change-affine	Changed requirements are seen as an added value of the product for the customer and not as an additional workload
Change-affine	The customer regularly inspects the working product with regard to the realization of the business value
Q4. How does your team experience iterative development?	

Iterative	The autonomous assignment of tasks is not restricted by organisational procedures
Iterative	The developers determine their tasks autonomously from the open requirements
Iterative	Development can be started without fully defining the requirements
Iterative	Detailed plan of development tasks will be made just before the next iteration
Q5. How extensively is your team self-organized?	
Self-organized	The scope of work for an iteration is decided by the team
Self-organized	The team is accountable for its actions
Self-organized	Decisions regarding the execution of their own work can be made by the team without the involvement of a managing authority
Self-organized	The entire team actively collaborates on iteration planning
Q6. How product driven the team is?	
Product-driven	Only productive working time is used to work in each iteration
Product-driven	The customer representative is directly participating in all decisions related to features
Product-driven	All subject matter experts are actively involved in the identification of the requirements
Product-driven	Documentation is critically reviewed for its value
Q7. Does your team continuously improve the development process?	
Improvement-oriented	In regular retrospectives, the approach of the process is reflected with the aim of improvement
Improvement-oriented	All team members actively participate in continuous improvement of the process
Improvement-oriented	Sights gained from retrospectives are turned into concrete improvement measures
Improvement-oriented	Improvements can be explored experimentally during the process
Q8. How important do you see agile development principles?	

Weighting Communicative	The agile team should communicate frequently and directly with each other
Weighting Change-affine	The agile team should react quickly and flexibly to volatile requirements
Weighting Iterative	The agile team should develop the product in several iterations
Weighting Self-organized	The agile team should operate autonomously as a self-organized team
Weighting Product-driven	The focus of the agile team should be on the product to be created
Weighting Improvement-oriented	The agile team should continuously improve the development process
Q9. Feel free to share your own views on team agility or thoughts from the survey (optional)	

Table 3. Preliminary field research survey questionnaire in Finnish

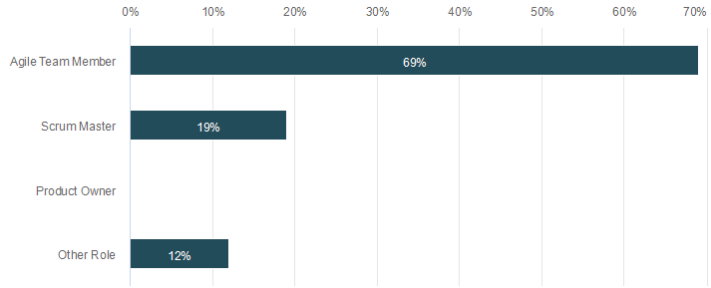
Q1. Mikä on pääasiallinen roolisi Agile organisaatiossa?	
Role	Agile Team Member
Role	Scrum Master
Role	Product Owner
Role	Muu Rooli
Q2. Miten näet tiimin kyvyn kommunikoida?	
Communicative	Jokainen tiimin jäsen on tietoinen muiden tiimin jäsenten tehtävistä
Communicative	Lähdekoodia pidetään koko tiimin yhteisenä omaisuutena
Communicative	Tiimi kokoontuu sovitusti useita kertoja viikossa vaihtaakseen tietoja keskenään
Communicative	Viestintään sisällytetään kaikki tiimin jäsenet
Communicative	Kaikki sidosryhmät tietävät tuotekehityksen tämänhetkisen edistymisen
Communicative	Asiakkaan edustajaan voidaan ottaa yhteyttä milloin tahansa
Communicative	Vaatimukset kerätään asiakkaalta yhteistyössä tiimin kanssa

Communicative	Tiimin jäsenille annetaan arvostusta heidän työstään
Q3. Kuinka muutoksiin suhtaudutaan tiimissänne?	
Change-affine	Alkuperäisiä asiakasvaatimuksia voidaan mukauttaa vaihtoehtoisilla ehdotuksilla
Change-affine	Jokainen iteraatio päättyy toimivan tuotteen toimittamiseen asiakkaalle
Change-affine	Muuttuneet vaatimukset nähdään asiakkaan kannalta tuotteen lisäarvona eikä ylimääräisenä työmääränä
Change-affine	Asiakkaan edustaja tarkastaa säännöllisesti tuotteen toimivuutta liiketoiminta-arvon toteutumisen kannalta
Q4. Miten tiimissä koetaan iteratiivinen kehitys?	
Iterative	Töiden tehtäväksiannot voidaan toteuttaa organisaatiosta riippumattomasti
Iterative	Kehittäjillä on vapaus määrittellä toteutus avoimista vaatimuksista
Iterative	Työt voidaan aloittaa ilman täydellistä vaatimusmäärittelyä
Iterative	Toteutettavat työt määritellään tarkalla tasolla vasta juuri ennen seuraavaa iteraatiota
Q5. Kuinka laajasti tiimi on itseorganisoitunut?	
Self-organized	Tiimi päättää iteraation työn laajuuden itse
Self-organized	Tiimi on vastuussa toiminnastaan
Self-organized	Tiimi voi tehdä oman työnsä toteuttamista koskevat päätökset ilman johtavan tahon osallistumista toimintaan
Self-organized	Koko tiimi osallistuu toteutettavan iteraation suunnitteluun
Q6. Kuinka tuotekeskeinen tiimi on?	
Product-driven	Iteraatioiden toteutukseen käytetään vain tuottavaa työaikaa
Product-driven	Asiakkaan edustaja osallistuu ominaisuuksiin liittyvissä päätöksissä
Product-driven	Kaikkia aihetta koskevia asiantuntijoita käytetään tunnistamaan vaatimuksia
Product-driven	Dokumentaation tarpeellisuutta arvioidaan kriittisesti
Q7. Parannetaanko tiimissänne jatkuvasti toimintaa?	

Improvement-oriented	Säännöllisissä retrospektiiveissä tutkitaan prosessia ja miten sitä voitaisiin parantaa
Improvement-oriented	Kaikki tiimin jäsenet osallistuvat aktiivisesti prosessin jatkuvaan parantamiseen
Improvement-oriented	Retrospektiiveistä saadut näkemykset muutetaan konkreettisiksi parannustoimenpiteiksi
Improvement-oriented	Parannuksia voidaan tutkia kokeellisesti prosessin aikana
Q8. Kuinka tärkeänä näet ketterän tiimin periaatteet?	
Weighting Communicative	Ketterän tiimin tulisi kommunikoida usein ja suoraan toistensa kanssa
Weighting Change-affine	Ketterän tiimin tulisi reagoida nopeasti ja joustavasti muuttuviin vaatimuksiin
Weighting Iterative	Ketterän tiimin tulisi kehittää tuotetta useissa iteraatioissa
Weighting Self-organized	Ketterän tiimin tulisi toimia itsenäisesti itseorganisoituneena tiiminä
Weighting Product-driven	Ketterän tiimin fokus tulisi olla tuoteen kehityksessä
Weighting Improvement-oriented	Ketterän tiimin tulisi jatkuvasti parantaa kehitysprosessia
Q9. Kerro vapaasti oma näkemyksesi tiimin ketteryydestä tai kyselyssä ilmenneistä mietteistä (vapaaehtoinen)	

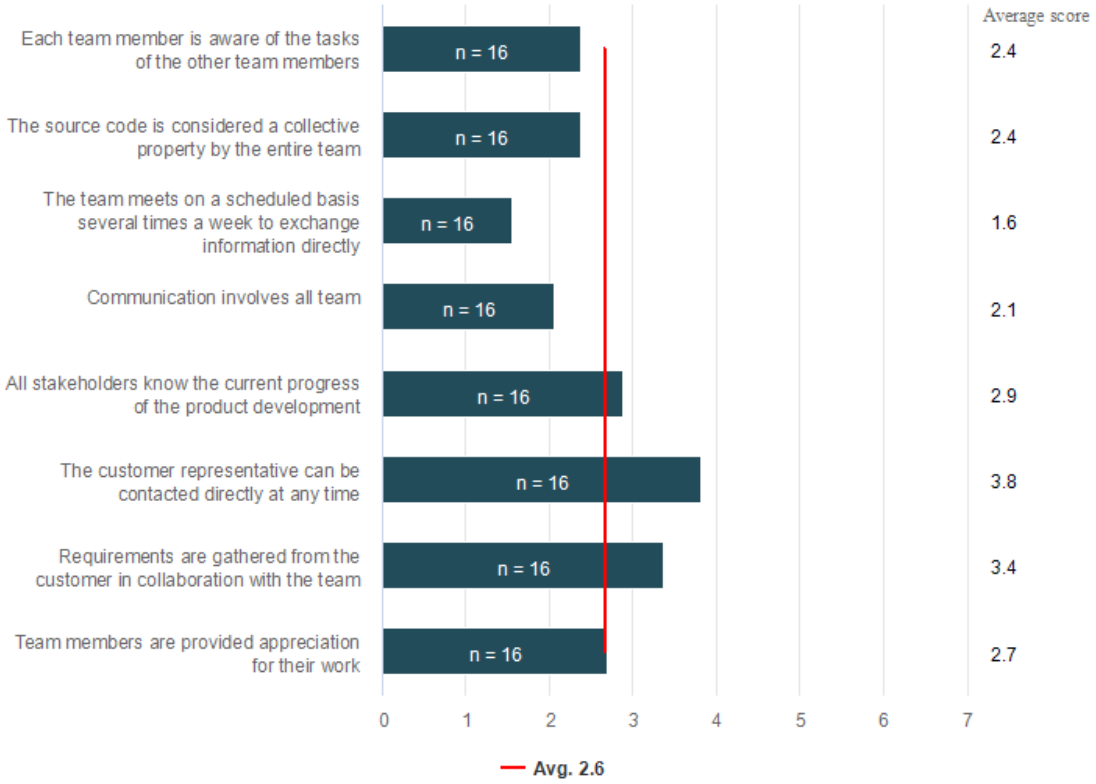
Appendix 2. Preliminary field research survey results

Q1. What is your primary role in Agile organisation?



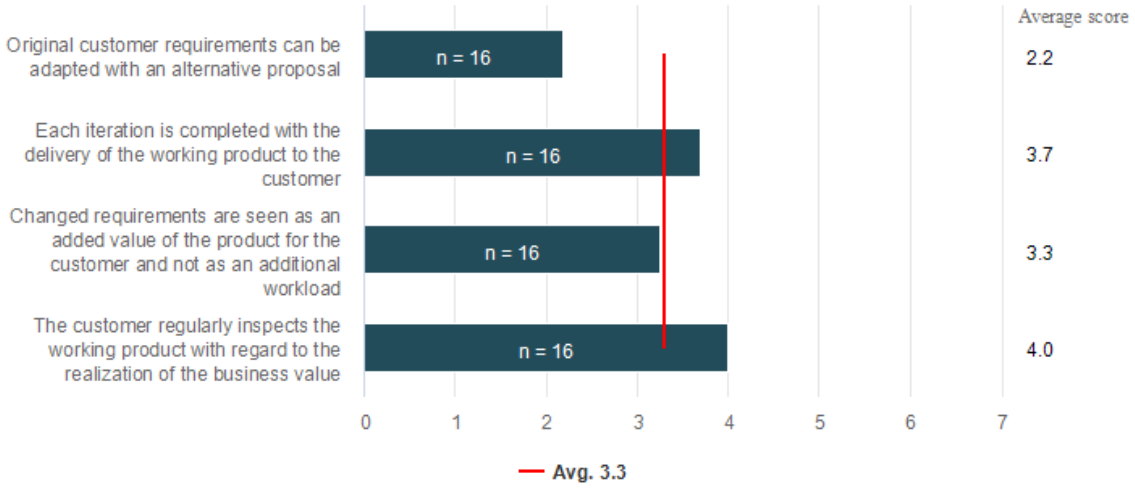
	n	Percent
Agile Team Member	11	68.8%
Scrum Master	3	18.7%
Product Owner	0	0.0%
Other Role	2	12.5%

Q2. How does your team communicate?



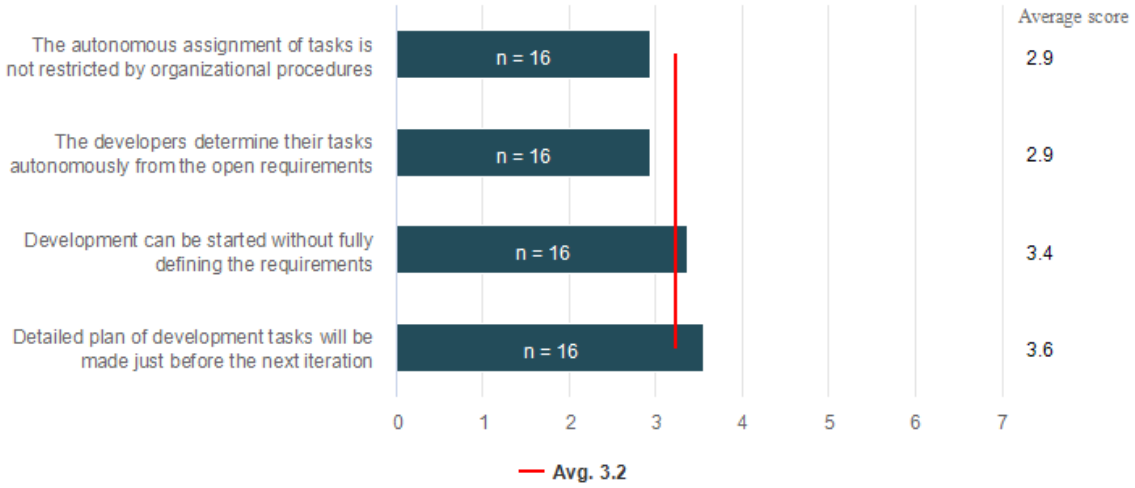
	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
Each team member is aware of the tasks of the other team members	18.7%	43.8%	25.0%	6.2%	6.3%	0.0%	0.0%	2.4	2.0
The source code is considered a collective property by the entire team	18.7%	43.8%	18.7%	18.8%	0.0%	0.0%	0.0%	2.4	2.0
The team meets on a scheduled basis several times a week to exchange information directly	50.0%	43.8%	6.2%	0.0%	0.0%	0.0%	0.0%	1.6	1.5
Communication involves all team members	37.5%	31.2%	18.8%	12.5%	0.0%	0.0%	0.0%	2.1	2.0
All stakeholders know the current progress of the product development	12.5%	25.0%	37.5%	18.7%	0.0%	6.3%	0.0%	2.9	3.0
The customer representative can be contacted directly at any time	0.0%	25.0%	18.7%	37.5%	0.0%	6.3%	12.5%	3.8	4.0
Requirements are gathered from the customer in collaboration with the team	12.5%	25.0%	18.7%	18.7%	12.5%	6.3%	6.3%	3.4	3.0
Team members are provided appreciation for their work	12.5%	37.5%	25.0%	18.7%	6.3%	0.0%	0.0%	2.7	2.5

Q3. What is the attitude for change in your team?



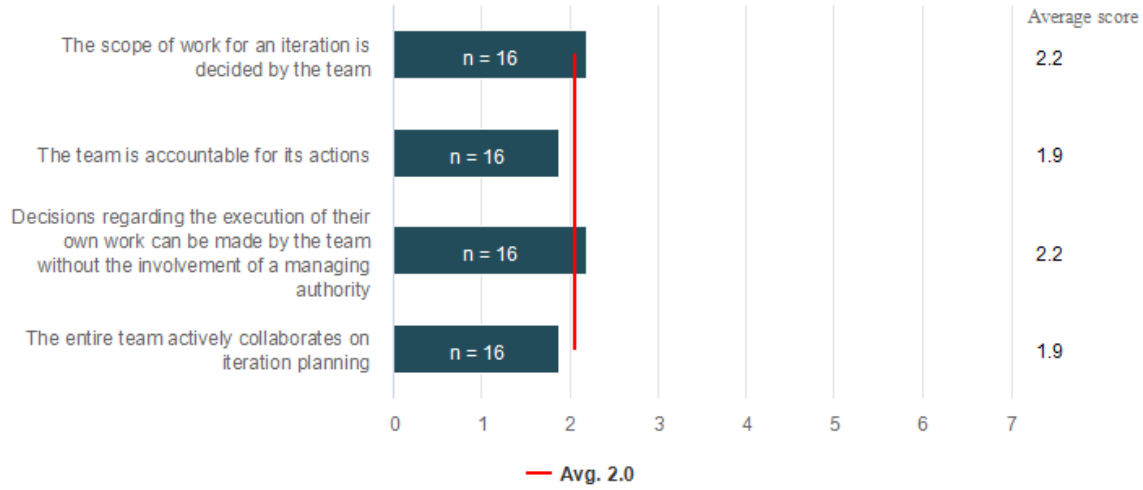
	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
Original customer requirements can be adapted with an alternative proposal	12.5%	68.8%	6.2%	12.5%	0.0%	0.0%	0.0%	2.2	2.0
Each iteration is completed with the delivery of the working product to the customer	0.0%	25.0%	25.0%	25.0%	12.5%	6.2%	6.3%	3.7	3.5
Changed requirements are seen as an added value of the product for the customer and not as an additional workload	6.3%	25.0%	25.0%	25.0%	18.7%	0.0%	0.0%	3.3	3.0
The customer regularly inspects the working product with regard to the realization of the business value	0.0%	31.2%	18.7%	12.5%	6.3%	18.8%	12.5%	4.0	3.5

Q4. How does your team experience iterative development?



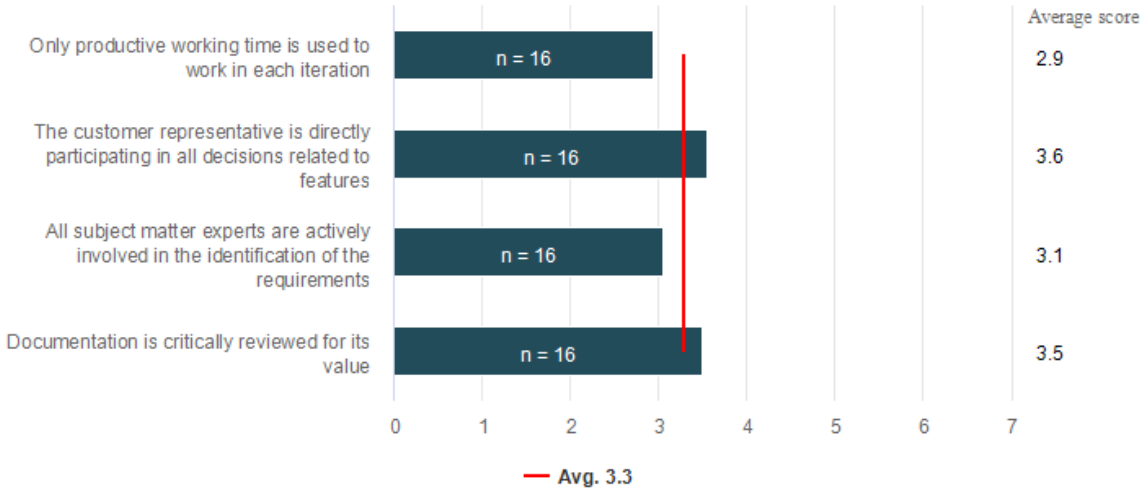
	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
The autonomous assignment of tasks is not restricted by organisational procedures	6.3%	31.2%	37.5%	18.7%	0.0%	6.3%	0.0%	2.9	3.0
The developers determine their tasks autonomously from the open requirements	6.3%	31.2%	31.2%	25.0%	6.3%	0.0%	0.0%	2.9	3.0
Development can be started without fully defining the requirements	18.7%	18.7%	18.7%	18.8%	6.3%	12.5%	6.3%	3.4	3.0
Detailed plan of development tasks will be made just before the next iteration	0.0%	25.0%	37.5%	12.5%	6.3%	18.7%	0.0%	3.6	3.0

Q5. How extensively is your team self-organized?



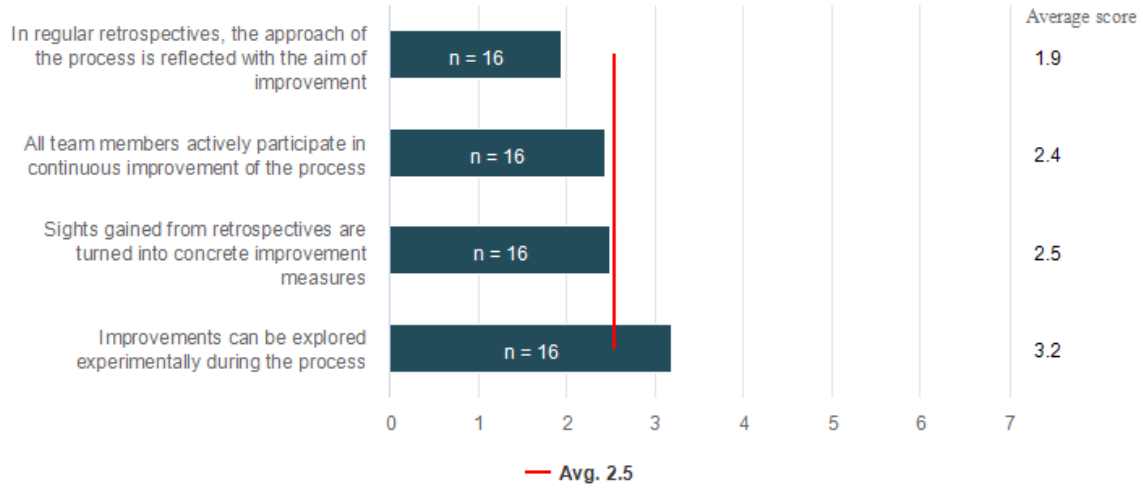
	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
The scope of work for an iteration is decided by the team	18.7%	50.0%	25.0%	6.3%	0.0%	0.0%	0.0%	2.2	2.0
The team is accountable for its actions	31.2%	56.3%	6.2%	6.3%	0.0%	0.0%	0.0%	1.9	2.0
Decisions regarding the execution of their own work can be made by the team without the involvement of a managing authority	25.0%	50.0%	12.5%	6.2%	6.3%	0.0%	0.0%	2.2	2.0
The entire team actively collaborates on iteration planning	37.5%	43.8%	12.5%	6.2%	0.0%	0.0%	0.0%	1.9	2.0

Q6. How product driven the team is?



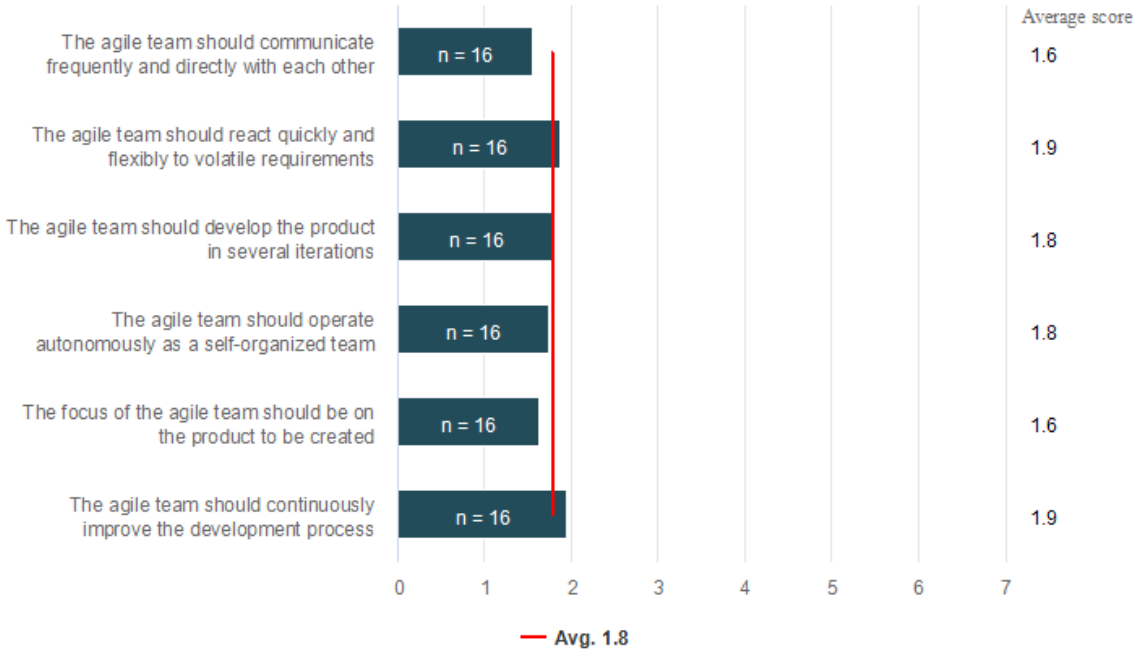
	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
Only productive working time is used to work in each iteration	12.5%	37.5%	12.5%	25.0%	6.2%	6.3%	0.0%	2.9	2.5
The customer representative is directly participating in all decisions related to features	12.5%	6.3%	37.5%	18.7%	6.3%	18.7%	0.0%	3.6	3.0
All subject matter experts are actively involved in the identification of the requirements	12.5%	25.0%	31.2%	12.5%	12.5%	6.3%	0.0%	3.1	3.0
Documentation is critically reviewed for its value	6.2%	31.2%	12.5%	25.0%	12.5%	6.3%	6.3%	3.5	3.5

Q7. Does your team continuously improve the development process?



	totally agree	agree	rather agree	neutral	rather disagree	disagree	strongly disagree	Average	Median
In regular retrospectives, the approach of the process is reflected with the aim of improvement	31.2%	50.0%	12.5%	6.3%	0.0%	0.0%	0.0%	1.9	2.0
All team members actively participate in continuous improvement of the process	37.5%	25.0%	12.5%	12.5%	6.2%	6.3%	0.0%	2.4	2.0
Sights gained from retrospectives are turned into concrete improvement measures	31.2%	31.2%	12.5%	12.5%	6.3%	6.3%	0.0%	2.5	2.0
Improvements can be explored experimentally during the process	18.7%	18.7%	12.5%	37.5%	6.3%	0.0%	6.3%	3.2	3.5

Q8. How important do you see agile development principles?



	particularly important	important	rather important	neutral	rather unimportant	unimportant	particularly unimportant	Average	Median
The agile team should communicate frequently and directly with each other	56.3%	37.5%	0.0%	6.2%	0.0%	0.0%	0.0%	1.6	1.0
The agile team should react quickly and flexibly to volatile requirements	31.2%	50.0%	18.8%	0.0%	0.0%	0.0%	0.0%	1.9	2.0
The agile team should develop the product in several iterations	31.2%	62.5%	0.0%	6.3%	0.0%	0.0%	0.0%	1.8	2.0
The agile team should operate autonomously as a self-organized team	43.8%	43.8%	6.2%	6.2%	0.0%	0.0%	0.0%	1.8	2.0
The focus of the agile team should be on the product to be created	43.8%	50.0%	6.2%	0.0%	0.0%	0.0%	0.0%	1.6	2.0
The agile team should continuously improve the development process	37.5%	31.2%	31.3%	0.0%	0.0%	0.0%	0.0%	1.9	2.0

Q9. Feel free to share your own views on team agility or thoughts from the survey (optional)

Responses in Finnish
Tiimissä on aika hyvin sisäistetty ketterän kehityksen periaatteet ja hyödyt, mutta muutos vanhasta ottaa aikaa ja tiettyjen seremonioiden tarpeellisuutta kyseenalaistetaan.
Tiimi on muokkautuva ja ottaa palautetta vastaan. Menetelmissä on vielä paljon opittavaa yrityksen ja erehdyksen kautta. Asiakkaan palaute on puutteellista ja tässä on konsulttien ja kehittäjien välissä parannettavaa.
Suurin ongelma ketterässä kehittämisessä on nyt se, että tiimin jäsenet ovat mukana monissa muissakin projekteissa yhtäaikaa. Nämä muut työt on priorisoitu tärkeämmäksi kuin ketterä kehityshanke, jolloin tätä projektia tehdään silloin kun muista projekteista aikaa jää. Ketterä projekti etenee lähinnä siksi, että muutama tekijä on innostunut asiasta ja suostuu tekemään pitkää päivää ja käyttää asiaan aikaa myös viikonloppuisin.

Responses in English (translated)
The principles and benefits of agile development are fairly well digested within the team, but the change from the old takes time and the need for certain ceremonies are questioned.
The team is adaptable and receptive to feedback. There is still a lot to learn about the methodology through trial and error. Customer feedback is lacking and this is where there is room for improvement between consultants and developers.
The main problem with agile development now is that team members are involved in many other projects at the same time. These other projects have been prioritised over the agile development project, so this project is being done when there is not enough time for other projects. The agile project is moving forward mainly because a few contributors are enthusiastic about it and are willing to work long hours and dedicate time to it, even on weekends.

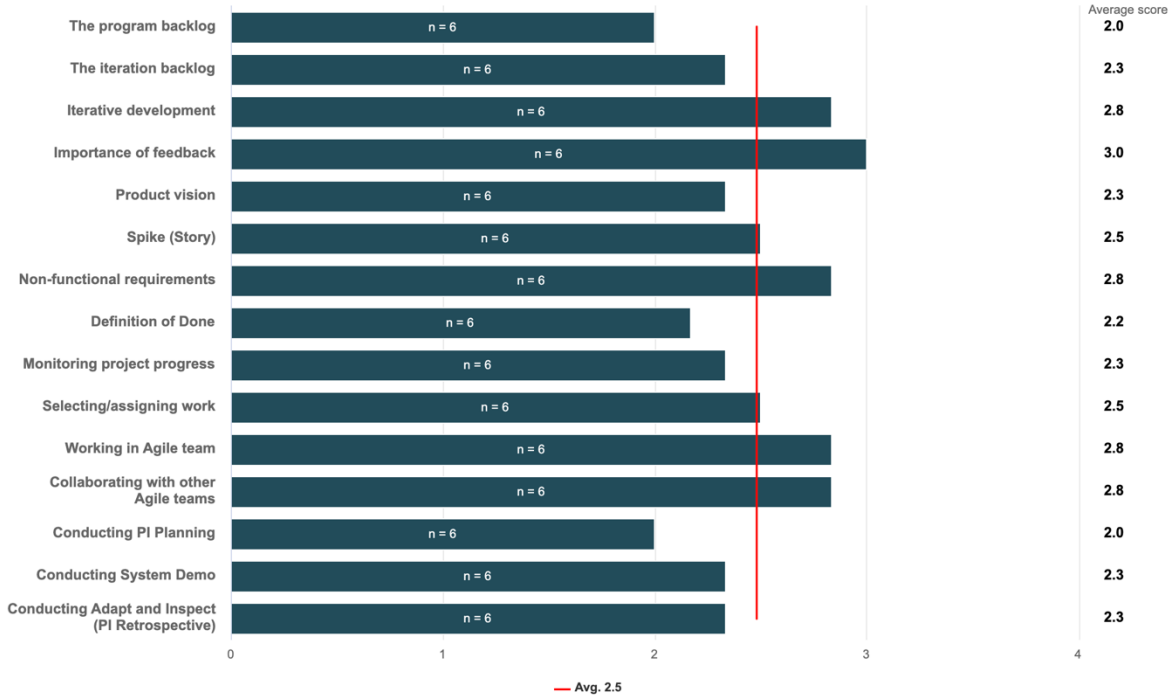
Appendix 3. Seriously Agile at Scale simulation survey questions

Table 5. Seriously Agile at Scale simulation survey questionnaire

Q1. Learning topics	
The program backlog	Definition of Done
The iteration backlog	Monitoring project progress
Iterative development	Selecting/assigning work
Importance of feedback	Working in Agile team
Product vision	Collaborating with other Agile teams
Spike (Story)	Conducting PI Planning
Non-functional requirements	Conducting System Demo
Conducting Adapt and Inspect (PI Retrospective)	
Q2. Simulation evaluation	
Passing through the simulation, I felt confident that I was learning.	
The simulation content was connected to other knowledge I already had.	
The way simulation works suited my way of learning.	
The simulation content was relevant to my interests.	
The variations in the simulation helped me to keep my attention to simulation.	
The simulation was easy to get into, game tutorial was all I needed.	
The simulation design was attractive.	
It was easy to understand how the simulation relates to software development.	
The simulation was an effective way to learn.	
I would like to play this simulation again.	
I would recommend this simulation to other people.	
I would have liked to play the simulation for a longer time.	
The simulation progressed at an adequate pace and did not become monotonous.	
I was confused during the simulation.	
I was frustrated during the simulation.	
This simulation was properly challenging, it was not too easy nor difficult.	
I active participated in the reflection event after the simulation.	
Q3. General feedback and comments	

Appendix 4. Seriously Agile at Scale simulation survey results

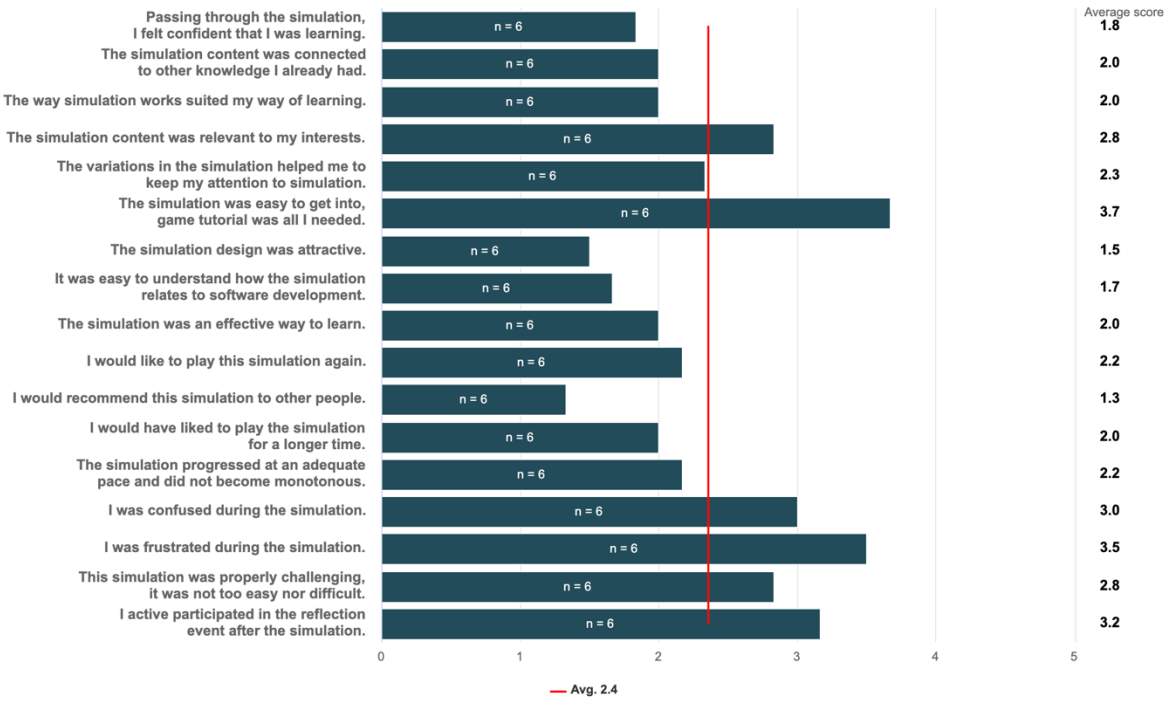
Q1. Learning topics



	I didn't learn anything new about this topic in the simulation	I heard about this in the simulation	Due to the simulation I can explain what the topic is	Due to the simulation I learned how to use/apply this topic in practice	Average	Median
The program backlog	33,4%	33,3%	33,3%	,0%	2,0	2,0
The iteration backlog	16,6%	50,0%	16,7%	16,7%	2,3	2,0
Iterative development	,0%	50,0%	16,7%	33,3%	2,8	2,5
Importance of feedback	16,6%	16,7%	16,7%	50,0%	3,0	3,5
Product vision	16,7%	33,3%	50,0%	,0%	2,3	2,5
Spike (Story)	16,7%	33,3%	33,3%	16,7%	2,5	2,5
Non-functional requirements	,0%	33,3%	50,0%	16,7%	2,8	3,0
Definition of Done	33,3%	33,3%	16,7%	16,7%	2,2	2,0
Monitoring project progress	16,7%	33,3%	50,0%	,0%	2,3	2,5
Selecting/assigning work	16,7%	33,3%	33,3%	16,7%	2,5	2,5
Working in Agile team	16,7%	16,7%	33,3%	33,3%	2,8	3,0

Collaborating with other Agile teams	,0%	16,7%	83,3%	,0%	2,8	3,0
Conducting PI Planning	33,4%	33,3%	33,3%	,0%	2,0	2,0
Conducting System Demo	33,3%	16,7%	33,3%	16,7%	2,3	2,5
Conducting Adapt and Inspect (PI Retrospective)	16,7%	33,3%	50,0%	,0%	2,3	2,5

Q2. Simulation evaluation



	strongly agree	agree	neutral	disagree	strongly disagree	Average	Median
Passing through the simulation, I felt confident that I was learning.	33,3%	50,0%	16,7%	,0%	,0%	1,8	2,0
The simulation content was connected to other knowledge I already had.	50,0%	16,6%	16,7%	16,7%	,0%	2,0	1,5
The way simulation works suited my way of learning.	33,3%	50,0%	,0%	16,7%	,0%	2,0	2,0
The simulation content was relevant to my interests.	16,7%	16,7%	33,3%	33,3%	,0%	2,8	3,0
The variations in the simulation helped me to keep my attention to simulation.	16,6%	50,0%	16,7%	16,7%	,0%	2,3	2,0
The simulation was easy to get into, game tutorial was all I needed.	,0%	16,7%	33,3%	16,7%	33,3%	3,7	3,5

The simulation design was attractive.	66,6%	16,7%	16,7%	,0%	,0%	1,5	1,0
It was easy to understand how the simulation relates to software development.	50,0%	33,3%	16,7%	,0%	,0%	1,7	1,5
The simulation was an effective way to learn.	33,3%	50,0%	,0%	16,7%	,0%	2,0	2,0
I would like to play this simulation again.	33,3%	33,3%	16,7%	16,7%	,0%	2,2	2,0
I would recommend this simulation to other people.	66,7%	33,3%	,0%	,0%	,0%	1,3	1,0
I would have liked to play the simulation for a longer time.	50,0%	16,6%	16,7%	16,7%	,0%	2,0	1,5
The simulation progressed at an adequate pace and did not become monotonous.	16,7%	50,0%	33,3%	,0%	,0%	2,2	2,0
I was confused during the simulation.	16,7%	33,3%	,0%	33,3%	16,7%	3,0	3,0
I was frustrated during the simulation.	16,7%	16,7%	,0%	33,3%	33,3%	3,5	4,0
This simulation was properly challenging, it was not too easy nor difficult.	16,7%	16,7%	33,3%	33,3%	,0%	2,8	3,0
I active participated in the reflection event after the simulation.	33,3%	,0%	16,7%	16,7%	33,3%	3,2	3,5

Q3. General feedback and comments

Responses in English
This simulation was quite fun way to learn agile concepts. I can easily see the use in other educations as well.
Happy to learn anything new
Through the game you will learn the SAFe model and terminology in software development
I would have liked to have more time to prepare and to better fit the events and game playing time in my schedule. Now everything felt rushed and it felt like there was no time to think, only do. Which often is the reality but is not an effective way of learning for me. Like in real life, a lot of my game-play time was spent running around like a headless chicken, jumping off cliffs injuring myself while trying to find everyone else :D
I didn't really have any experience with games before this, so even the tutorial was challenging and time-consuming. "What LMB?" That's the kind of acronym that SAFe and the whole IT world is full of. There were many new things for me, the game, Discord... so I almost panicked. I also didn't test my laptop before the sessions, so some of the fun was missed. I think this simulation would work well for people who are more familiar with games. It's easier to learn new things when you have something familiar to work with. The SAFe terms become very concrete with this and it was easy to see how many aspects of the simulation was related to software development, such as not knowing anything about the subject to begin with. A fun way to learn and this could even mitigate resistance to change.
Overall, I felt that this way of learning about agile methodology was interesting and simulation was a good way to link the theory with action. I liked it and would prefer this over a classroom lecture for sure. This was obviously a very short test run, so there was no time to concentrate to all aspects thoroughly, but given a bit more time, I believe this would be very efficient way of learning.

Appendix 5. Serious Satisfactory Handout


LUT
University
LAND
OF THE
CURIOUS

PRELIMINARY SESSION 30.11.2022

SERIOUS SATISFACTORY HANDOUT

Scaled Agile Simulation Using Serious Games

SCALED AGILE SIMULATION USING SERIOUS GAMES

SERIOUS SATISFACTORY

- **Satisfactory** is a game of factory management and planet exploitation
- Game focuses on constructing buildings and linking them together with conveyor belts. It is possible to create a factory that can handle entire item construction process.
- Agile teams plans and executes a Program Increment in a simulation where Agile teams build an automated factory.

SE-1 Iron production	
SE-6 Iron Plate Factory	TO DO
SE-7 Iron Rod Factory	TO DO
SE-8 Screw Factory	TO DO
SE-2 Copper production	
SE-9 Wire Factory	TO DO
SE-10 Cable Factory	TO DO
SE-11 Copper Sheet Factory	TO DO
SE-3 Concrete production	
SE-12 Concrete Factory	TO DO
SE-17 Storage solution	
SE-18 Storage for Iron Parts	TO DO
SE-19 Storage for Copper ...	TO DO
SE-20 Storage for Concrete	TO DO
SE-4 Power source	
SE-13 Power Plant	TO DO
SE-5 Self sustainable energy	
SE-14 Sustainable Power P...	TO DO
SE-15 Water Source	TO DO
SE-16 Fuel Source	TO DO

SERIOUS SATISFACTORY SIMULATION

FIRST STEPS

- Create a steam account (or use existing)
- Send me your Steam ID so I can send you the game through Steam
- After receiving the game, install and play through the tutorial

SERIOUS SATISFACTORY SIMULATION

SATISFACTORY TUTORIAL

- To get feel of the game before the simulation
- Play through TIER 0: Onboarding
  - HUB Upgrades 1-5
- When in doubt you can watch tutorial
  https://www.youtube.com/watch?v=Fg0r5hcYvHU

SERIOUS SATISFACTORY SIMULATION

NEXT STEPS

- We are running the simulation on dedicated server (always on)
- Simulation contains
  - Synchronized events (intro, planning and retro)
  - Asynchronized events (gameplay and demos)
- Simulation starts next week

SE-1 Iron production	
SE-6 Iron Plate Factory	TO DO
SE-7 Iron Rod Factory	TO DO
SE-8 Screw Factory	TO DO
SE-2 Copper production	
SE-9 Wire Factory	TO DO
SE-10 Cable Factory	TO DO
SE-11 Copper Sheet Factory	TO DO
SE-3 Concrete production	
SE-12 Concrete Factory	TO DO
SE-17 Storage solution	
SE-18 Storage for Iron Parts	TO DO
SE-19 Storage for Copper ...	TO DO
SE-20 Storage for Concrete	TO DO
SE-4 Power source	
SE-13 Power Plant	TO DO
SE-5 Self sustainable energy	
SE-14 Sustainable Power P...	TO DO
SE-15 Water Source	TO DO
SE-16 Fuel Source	TO DO

INTRODUCTION 8.12.2022

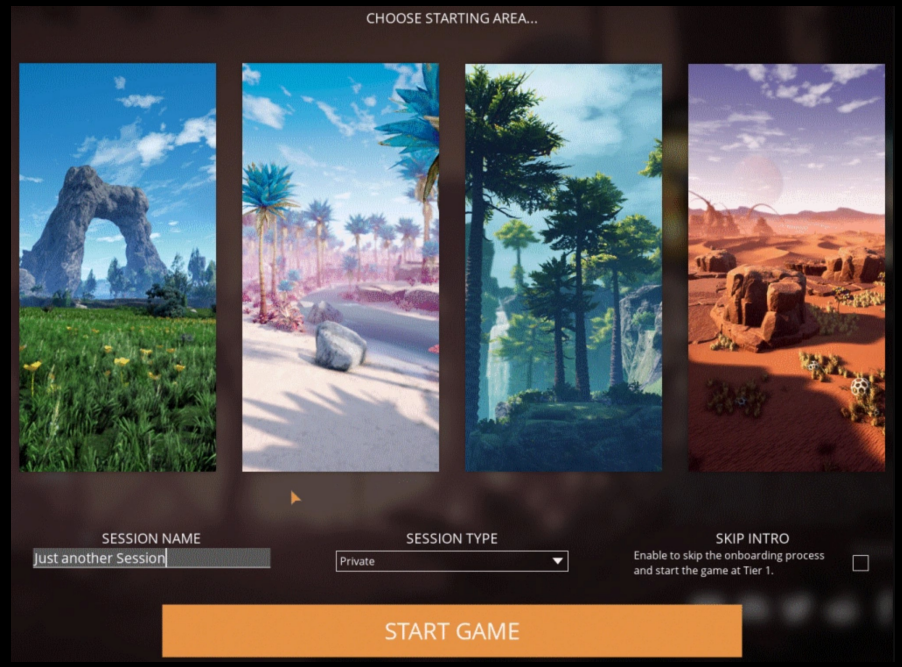
INTRODUCTION AND PRACTICALITIES

Serious Satisfactory Simulation

SERIOUS SATISFACTORY SIMULATION

INTRODUCTION

- We are utilizing Serious Games to perform a simulation
  - Simulation does not contain actual implementation part of software development
  - Implementation is replaced by building tasks in open world gaming environment
- Goal of the simulation
  - Learn Scaled Agile Framework in practice
  - Learn how to utilize Spike type of Story
  - Provide alternative learning experience



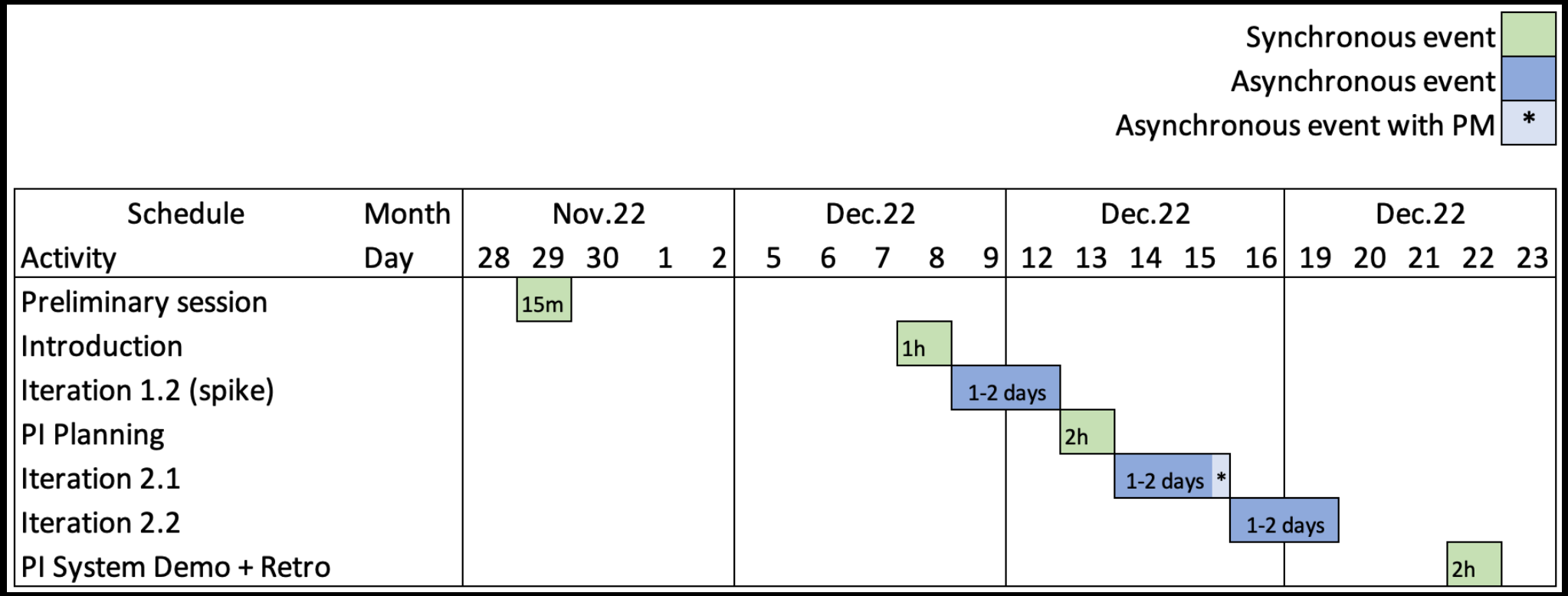
SERIOUS SATISFACTORY SIMULATION

PRACTICALITIES

- We will be executing a Program Increment (PI) in shorter time frame
  - 8-12 weeks -> 3 weeks (8-12 active hours)
- Simulation contains
  - Synchronous events (intro, planning and retro)
  - Asynchronous events (gameplay and demos)
- Simulation is run on dedicated server (always on)
- Simulation progress is tracked using Jira Cloud
  - Agile Teams update issues and tracks progress
- Simulation is executed with 2-3 teams (team formation)

SERIOUS SATISFACTORY SIMULATION

SIMULATION SCHEDULE



SERIOUS SATISFACTORY SIMULATION

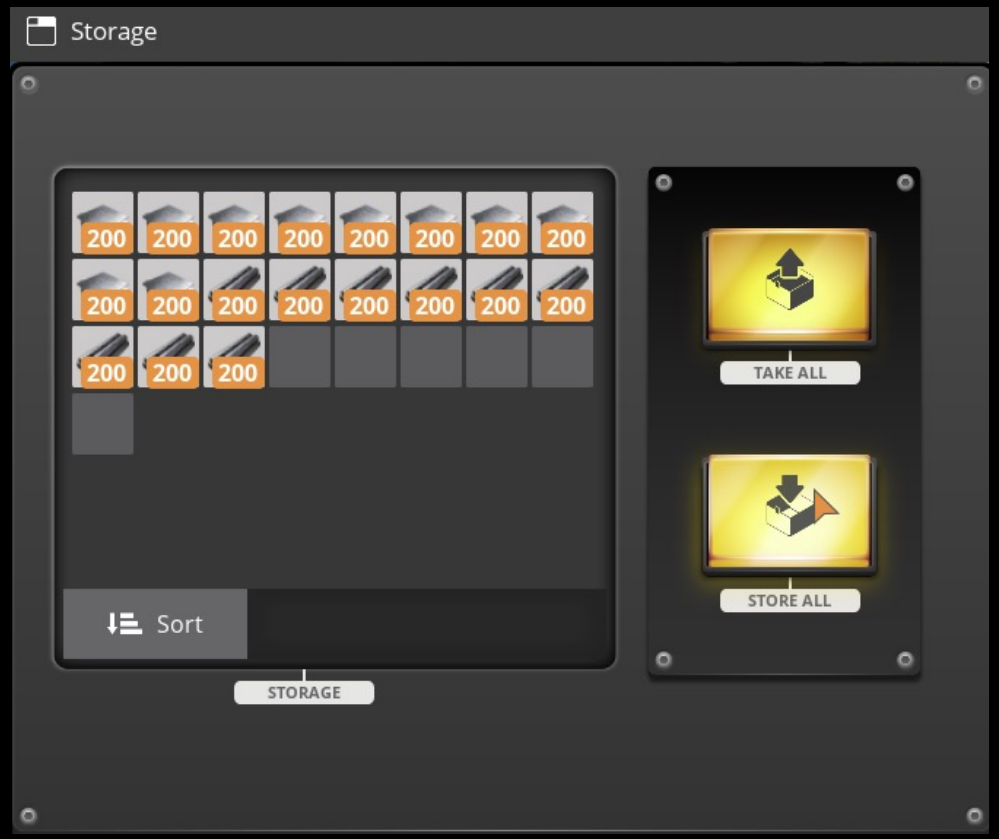
TEAM FORMATION

- Provide your view on timetable
  - When you plan to be executing the iteration (gaming part)
  - After working hours, weekends, etc.
- Create teams of 2-3 people
  - Let's try to form teams with matching timetables
  - Choose your own name for the team
- Idea is to work together as a team
  - Communicate with your team
  - Communicate with other team(s), dependencies!

SERIOUS SATISFACTORY SIMULATION

SIMULATION PRACTICALITIES

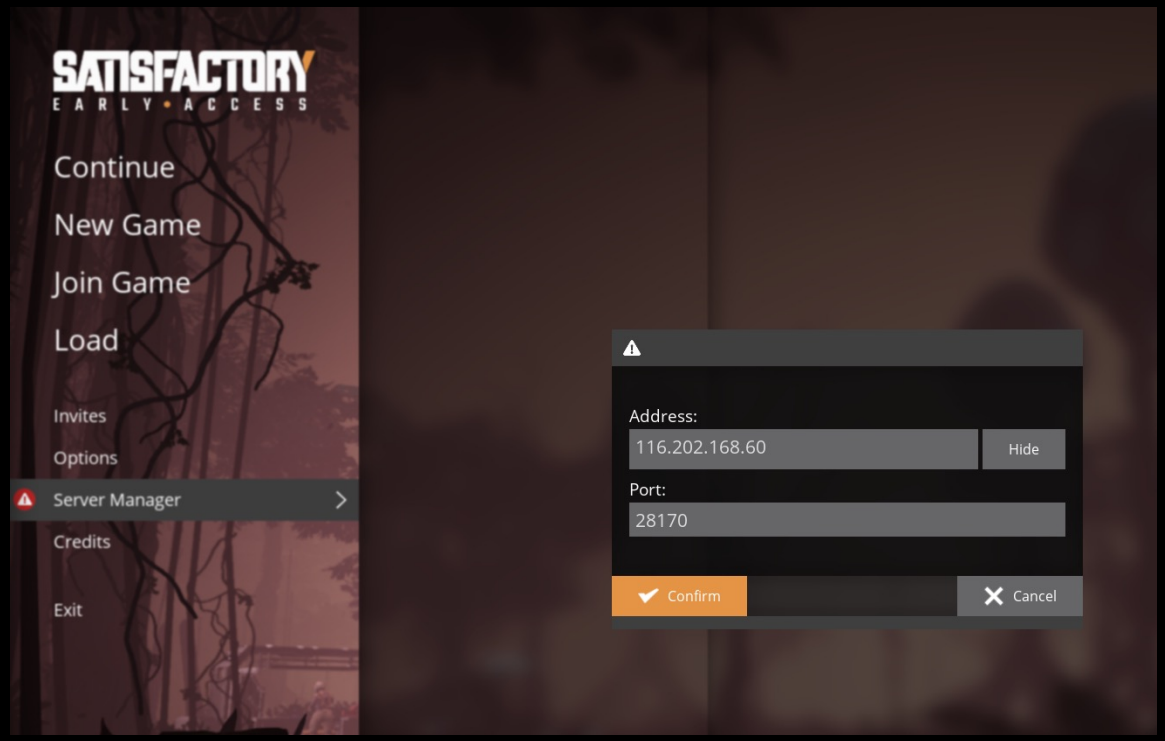
- Resources can be found in personal storage boxes next to the HUB
- These resources can be used to carry out PI Features
- BEWARE: there is only limited amount of resources
  - You need to create more for future Features



SERIOUS SATISFACTORY SIMULATION

DEDICATED SERVER

IP Address: **116.202.168.60**
Game Server Port: **28170**

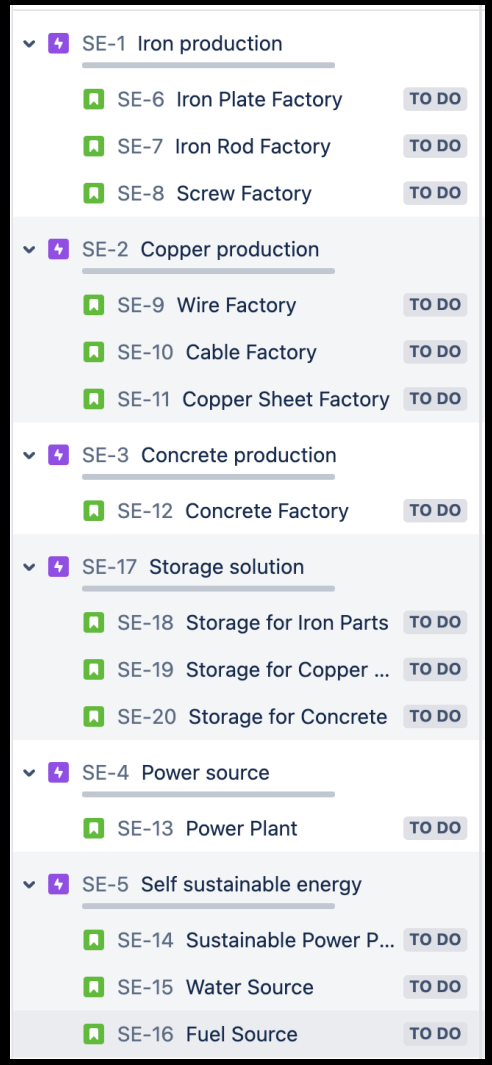


- **How to Connect to the Server**
  - **Step One:** Open Satisfactory.
  - **Step Two:** Click **Server Manager** in the main menu.
  - **Step Three:** Click **+ Add Server**.
  - **Step Four:** Enter server IP Address and Server Port.
  - **Step Five:** After adding the server, go to the **Status** tab and press the **Join Game** button in the bottom right.

SERIOUS SATISFACTORY SIMULATION

START FROM ITERATION 1.2

- Gather enough information so that PI Planning is effective as possible
- Feel free to create any additional features or stories in Jira
- What information do you need to plan a Program Increment? (spike)
- Definition of Done (spike)
  - Information is gathered / documented so you can answer to questions:
    - What is needed to build part factories described at Story level for each Feature?
    - How to comply non-functional requirements?



PI PLANNING 13.12.2022

PI PLANNING

Serious Satisfactory Simulation

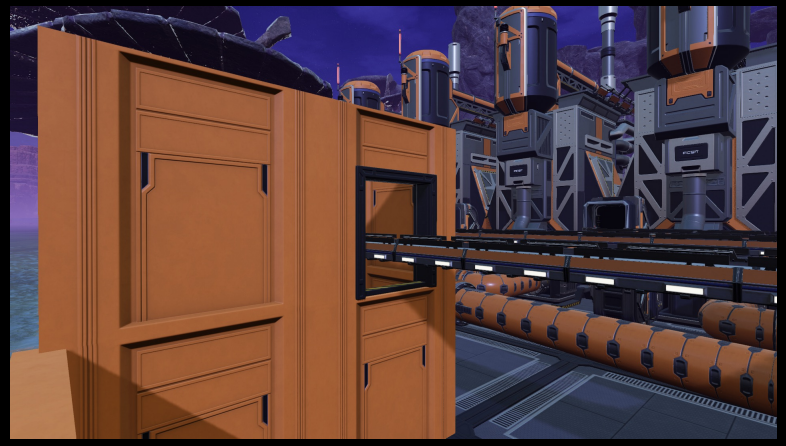
SERIOUS SATISFACTORY SIMULATION

BUSINESS CONTEXT AND VISION

- FICSIT Inc. has a strong focus in R&D, Engineering, Pioneering
- The company is building a top-secret project in space called Project Assembly
  - Due to classified nature of the project, additional details won't be disclosed related to the project or its purpose
- Project Assembly requires resources which will be transferred using a Space Elevator
- Part Factories located at the surface of the planet will play a key role
  - Provides needed resources for the project
  - Self-sustainable (fully automated)

SERIOUS SATISFACTORY SIMULATION

ARCHITECTURE VISION AND DEVELOPMENT PRACTICES

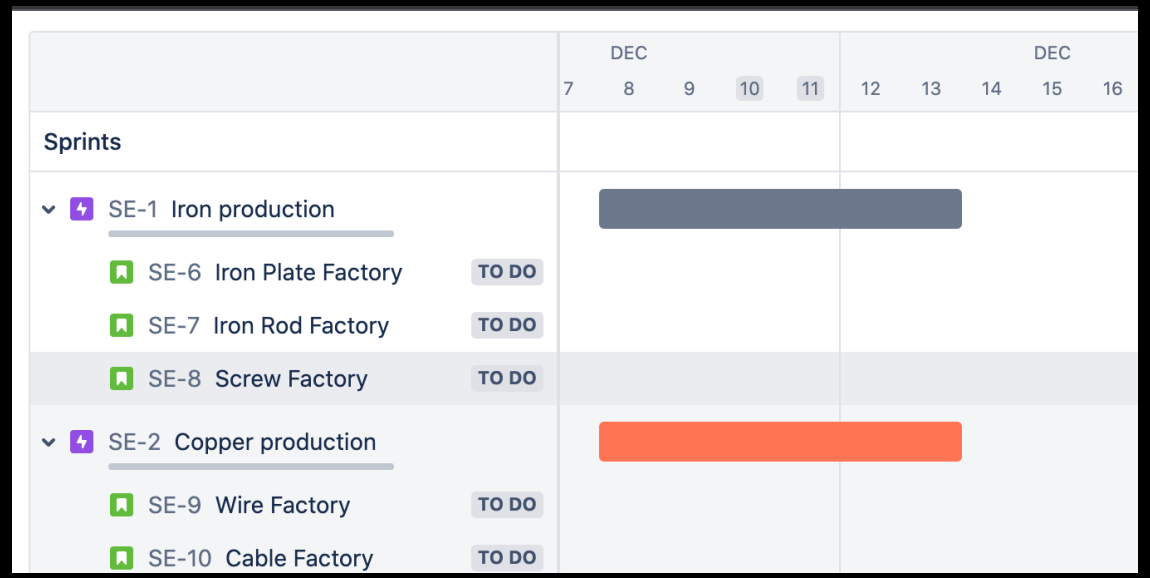


- Part Factories should be small as possible and independent (Microservices architecture)
  - Input of materials, output(s) of produced parts
  - Electricity is an exception, it can be shared between factories
- Factories must follow non-functional requirements
  - Security: Foundation, walls, roof and at least one door
  - Safety: Balconies, roofs, or terraced areas are at or over 1.1 meters of height must have railings
  - Maintainable: Corridors need to have enough room for inspection, clear path for maintenance
  - Extendable: Should be designed in a way that they can be extended if needed

SERIOUS SATISFACTORY SIMULATION

TEAM BREAKOUTS

- Decide on features that your team would implement
- Plan your work on Jira Cloud
  - Scheduling
  - Dependencies
  - Risks
- After breakouts present your initial plan



PI PLANNING 13.12.2022

INSPECT AND ADAPT

Serious Satisfactory Simulation

SERIOUS SATISFACTORY SIMULATION

SYSTEM DEMO

- https://serious-satisfactory.atlassian.net/wiki/spaces/SE/blog/2022/12/22/5800009/Factories

SERIOUS SATISFACTORY SIMULATION

RETROSPECTIVE

- https://app.funretrospectives.com/agendas/-NJt04KTdTZK3FlcjC6m

SERIOUS SATISFACTORY SIMULATION

SIMULATION FEEDBACK

- https://link.webropolsurveys.com/S/DD1B382EA8F01CEC

REFERENCES

- https://doi.org/10.1145/3510456.3514145


LUT
University