# An Onboarding Model for Integrating Newcomers into Agile Project Teams

Peggy Gregory[1], Diane E. Strode[2], Helen Sharp[3], Leonor Barroca[3]

[1] School of Psychology & Computer Science, University of Central Lancashire, Preston, UK
ajgregory@uclan.ac.uk (corresponding author)

[2] School of Information Technology, Whitireia Polytechnic, Wellington, New Zealand
diane.strode@whitireia.ac.nz

[3] School of Computing and Communications, The Open University, Milton Keynes, UK
helen.sharp@open.ac.uk; leonor.barroca@open.ac.uk

## Abstract

**Context:** A stable team is deemed optimal for agile software development project success; however, all teams change membership over time. Newcomers joining an agile project team must rapidly assimilate into the organisational and project environment. They must do this while learning how to contribute effectively and without seriously interrupting project progress. **Objective:** This paper addresses how newcomers integrate into an established agile project team and how agile practices assist with onboarding. **Method:** A single, qualitative case study approach was used, investigating a co-located agile project team in a large IT department who regularly onboard inexperienced newcomers. Analysis was abductive, consisting of inductive coding and theming using categories from an existing onboarding theory. **Results:** We describe the team's onboarding practices and adjustments and present an agile onboarding model that encompasses onboarding activities, individual adjustments, and workplace adjustments. **Conclusions:** A mixture of general and specific agile onboarding practices contribute to successful onboarding in an agile team. We provide practical guidelines to improve onboarding practice in agile teams. Our major new contribution is an extended model of onboarding for agile teams.

**Keywords**: Agile Onboarding adjustments; Onboarding activities; Scrum; Self-organizing team onboarding; Sustaining agile; Team onboarding; Newcomers

## 1 Introduction

Software development is a knowledge-intensive activity that relies on people with advanced technical knowledge, skills, experience, and domain knowledge. One commonly accepted way to organise software development is to adopt the process, practices, and mindset of agile software development. Agile software development approaches are currently used widely in co-located, distributed, and large-scale systems development projects (Stavru, 2014; Version One, 2019), and more recently in fully virtual projects due to the Covid-19 pandemic (da Camara *et al.,* 2020; Handke *et al.,* 2020). Within these environments agile development optimally occurs in self-organising teams that are autonomous, cross-functional, and self-improving (Hoda and Murugesan, 2016). Newcomers to any software development environment face challenges in becoming fully integrated and productive team members. The challenges involve acquiring organisational knowledge, project knowledge, product and domain knowledge, and knowledge of the technical environment. There are, in addition, particular challenges for joining an agile software development environment. These include understanding and becoming proficient in the agile approach used by the team, undergoing socialization into an intensive, self-organising team environment (Buchan, MacDonell and Yang,2019), and, for those completely new to agile, learning about and adopting an agile mindset and agile behaviours and being more holistically involved in

different aspects of a project. The subject of this study is an investigation of practitioner experiences of onboarding into an agile team.

Onboarding is the term used to describe the process through which new employees join and integrate into an organisation. There is extensive literature on onboarding in organisations extending back to the 1970s (Bauer and Erdogan,2011; Van Maanen and Schein, 1977), and significant research into onboarding in Open Source Software Development projects (Fagerholm *et al.,* 2014; Steinmacher and Gerosa,2014). Most of the literature on onboarding in software development organisations is agnostic about the development approaches used, namely whether agile or not (Dagenais *et al.,* 2010; Sharma and Stol, 2020). There is some research into onboarding in agile software development project teams: Buchan, MacDonell and Yang (2019) investigate effective onboarding techniques for agile teams; Britto, Cruzes, Smite, and Sablis (2018) look at onboarding in globally distributed legacy projects but without highlighting the specificities of agile; Dagenais, Ossher, Bellamy, Robillard, & de Vries (2010) identify orientation aids that help the integration of newcomers into agile projects; and Barroca, Gregory, Kuusinen, Sharp & AlQaisi (2018) indicate that effectively integrating newcomers is a factor in sustaining agile in the long-term.

We expected to find that onboarding into agile software project teams would exhibit many similarities to general organisational onboarding, but that there would also be differences because of the need for newcomers to understand the agile practices of this team, to adopt an agile mindset, and to effectively integrate into projects where self-organising teamwork is the norm. Therefore, we aimed to explore the onboarding experiences of newcomers and their more experienced colleagues in a specific context, looking at how newcomers are integrated and how they learn the unique agile approach of the team. The two research questions for this study are: RQ1: *How do newcomers integrate into an ongoing agile project team?* And RQ2: *How do agile practices particularly assist with onboarding into an ongoing agile project team?* To address these, we undertook a single qualitative case study in a co-located agile project team, based in the IT department of a UK university (not the home institution of any of the authors), who regularly onboard inexperienced newcomers. We found a mixture of general and agile onboarding practices contributed to the onboarding process and that specific agile practices were adapted to assist with onboarding. We also found adjustments that had to be made by newcomers and the workplace to support integration, some of which were particularly related to an agile context.

This paper extends a previous publication (Gregory *et al.,* 2020) by adding more detail about the study, specifically the literature, research method, and participants. It also contains a more detailed presentation and discussion of the findings which leads to a new model of onboarding not previously presented. This new model is an adaptation and extension of Bauer's onboarding model (2010) and includes new concepts for the agile context; this is a major new contribution.

This paper is organised as follows. In section 2 we review pertinent literature on onboarding and describe Bauer's onboarding model (2010) which was used after our initial analysis to frame our findings. In section 3 we present our research method, and in section 4 we describe the agile project team we studied. In section 5 we introduce our model of onboarding in agile teams and explore it in relation to our findings. In section 6 we discuss our answers to the research questions, relate our findings to that of other studies, summarise our contributions, and discuss limitations of the work. In section 7 we draw conclusions and indicate future work.

## 2 Background

"*Organizational socialization, or onboarding, is a process through which new employees move from being organizational outsiders to becoming organizational insiders. Onboarding refers to the process that helps new employees learn the knowledge, skills, and behaviors they need to succeed in their new organizations.*" (Bauer and Erdogan,2011 p.51)

In onboarding, a central idea is that of the *newcomer*, who is a new staff member joining an organisation. Newcomers are also defined as people moving within the organisation, for example from one department to another or from one team to another. Other concepts are that of the *outsider* and *insider*. An outsider is a stranger to the organisation or team whereas an insider is an established staff member. People moving from one team to another are organisational insiders, but team outsiders. Becoming an insider is a process that happens over time, and onboarding is the initial part of that process. "*To cross inclusionary boundaries means that [an outsider] becomes an insider with all the rights and privileges that go with such a position.*" (Van Maanen and Schein, 1977 p.21)

Onboarding literature emerged in the field of organisation studies in the 1970s when Van Maanen and Schein (1977) defined the concepts of organisational socialisation, newcomers, insiders, and outsiders. Their idea was that organisations have functional, hierarchical, and inclusionary boundaries that newcomers cross as they are socialised from being outsiders to being insiders. They identified six inter-related dimensions of organisational socialisation: collective vs individual; formal vs informal; sequential vs random steps; fixed vs variable; serial vs disjunctive; and investiture versus divestiture. (Van Maanen and Schein, 1977 p.37).

Onboarding in commercial software development organisations was studied by Sharma and Stol (2020). After a review of empirical studies, these authors noted that most onboarding studies focus on newcomers in open source software communities. They only found nine empirical studies of onboarding in commercial software development. Only two of these studies mentioned agile environments: Dagenais et al. (2010) and Britto et al. (2018). Sharma and Stol developed and tested a theoretical model of the relationship between onboarding activities (orientation, training, and support), onboarding success, organisational fit (job satisfaction and workplace relationship quality) and turnover intention. One key result was that orientation and support are strongly related to onboarding success.

Dagenais, Ossher, Bellamy, Robillard, and de Vries (2010) identified three integrating factors for newcomer developers to software projects: early experimentation, internalizing the various structures and cultures, and frequent progress validation. In their grounded theory study, they interviewed 18 developers who had recently joined an ongoing project. Most interviewees were experienced developers, and all were working in agile teams, but agility was not a particular focus of the study. The only agile practice highlighted as being helpful and effective for orientation was the daily Scrum meeting.

Britto, Cruzes, Smite and Sablis (2018) study onboarding in three globally distributed legacy software development projects, using the onboarding model of Bauer (2010) to structure their research. One key finding was that onboarding remote developers to an ongoing agile project was the greatest challenge. This was because the agile approach involved minimal documentation meaning that new developers had to engage in continuous dialogue with mentors to understand the project. Britto, Smite, Damm, and Börstler (2020) advanced knowledge of onboarding into large-scale globally distributed software projects by identifying, based on a single case study, that in this context effective performance depends on distance from mentors, formal training approach, fit with socio-cultural background, task allocation,

and team stability. Both these papers investigated agile teams, but the focus was on onboarding in globally distributed projects rather than agile teams.

Yates, Power, and Buckley (2020) investigate how newcomer software developers understand the code base of an existing system by exploring the different types of information passed from insider experts to newcomers during onboarding. They report that those coaching newcomers can provide four views of a program (temporal, structural, algorithmic, and rationale) and that each view is valuable for onboarding into software development projects. They don't specify whether participants in their study used agile or non-agile development approaches.

Onboarding in co-located agile project teams is directly addressed by Buchan, MacDonell, and Yang (2019), who investigated software practitioners' perceptions of the efficacy and value of onboarding practices used in agile contexts. From an initial systematic literature survey of software development literature, they identified 11 onboarding goals organised into 5 themes, that they determined were relevant for onboarding onto agile software teams (see Table 1), although only two goals specifically mention agile (in the Development Process theme). They conducted an empirical interview study in New Zealand with 11 participants from 8 organisations, of whom all 11 were new to the product being developed, 10 were new to working in an agile team, 7 were new to the organisation and 5 were recent graduates. The study identifies and describes 24 effective onboarding techniques linked to the 11 onboarding goals. The authors acknowledged their list of techniques is unlikely to be exhaustive given the study's limitations. Although the techniques listed are a mix of traditional approaches, such as mentoring and formal training, and agile techniques, such as pair programming and stand-up meetings, the authors do not discuss or reflect on why specific agile practices are useful for onboarding.

*Table 1: Software Development Onboarding Goals (adapted from Buchan, MacDonell and Yang, 2019 p.3)*

| **Theme** | **Onboarding Goals** |
|---|---|
| Cultural Context | Understand and fit in with company culture |
| | Understand and fit in with team norms |
| Job Responsibility | Understand and meet others' expectations of one's own role's responsibilities. |
| | Understand the responsibilities, expertise, and authority of other team members |
| | Understand what work to do and when |
| Standard of Work | Understand how to code and test to the team's expectations |
| | Understand the team's standards of team quality |
| Development Process | Understand and adopt the Agile mindset |
| | Know how to use Agile artefacts and techniques used by the rest of the team |
| Project Knowledge | Understand the short, medium, and long-term work structures, aims and implications |
| | Understand the product/project domain knowledge and terminology |

In summary, this review of extant literature on onboarding in software development projects indicates that, apart from Buchan et al.'s (2019) study, knowledge specific to onboarding into agile software development projects is limited. Our work distinguishes itself from that of Buchan et al.(2019) in several ways. We look at onboarding in situ through the case study of an agile team, exploring the views of all team members including newcomers and established members and learning about the history and development of onboarding in the team over time. This provides a more in-depth view of the phenomenon than is possible from an interview study of participants from different organisations. We specifically explore how agile practices are used and adapted by the team to become effective onboarding techniques. Finally, we map our findings to a theoretical frame from the onboarding literature and develop a model of onboarding in the agile context.

## 2.1 Bauer's Onboarding Model

Bauer's model for successful onboarding (Bauer and Erdogan,2011; Bauer, 2010) provides a set of guidelines for onboarding people into organisations. Bauer's model is generic to all onboarding environments and situations. The six *onboarding functions* in Bauer's model (2010) are described in Table 2. Bauer's model also introduces the concept of *new employee adjustments* that are identified as levers that help newcomers during onboarding. These adjustments (see Table 3) are also described as 'levers' that organisations can deliberately target during onboarding programs to ensure that new employees get an optimal experience and are more likely to onboard successfully.

*Table 2: Bauer's Onboarding Functions (2010)*

| Onboarding Function | Description |
|---|---|
| ***Recruiting process*** | The process that provides information to newcomers and helps them form realistic expectations of the organisation and their role. The recruiting process can be separate from the onboarding process but has been shown to be more effective if integrated into onboarding. |
| ***Orientation*** | The process of helping newcomers to understand the important aspects of their jobs and of the organisation including the organisation's culture, values, goals, history, and power structure. Orientation includes formal face-to-face, written guidelines, and online programmes for providing key information to newcomers. Orientation includes socialization, which involves making newcomers feel welcome by introducing them to co-workers and other people in the organisation. |
| ***Support tools and processes*** | Support tools include a written onboarding plan for newcomers that includes timelines, goals, responsibilities, support systems, and how to access assistance. Attending regular meetings with a variety of stakeholders within the organisation is a mechanism for support of newcomers. Online support tools are another mechanism for onboarding but have been shown to be somewhat less effective than regular face-to-face orientation sessions. |
| ***Coaching and support*** | Coaching, mentoring, and having role models are mechanisms for helping newcomers learn about the organisation and their role, and to navigate the social and political aspects of the organisation. Coaching and mentoring can be external or internally sourced. Using mentors is shown to improve newcomer knowledge of the organisation. |
| ***Training*** | Training includes learning hard, soft, and onboarding skills. Training can be informal (learning-on-the-job) or formal (mandatory scheduled courses). |
| ***Feedback tools*** | Feedback and guidance provide newcomers with information on progress, strengths, and weaknesses. Feedback can be formal (e.g. performance appraisals) and informal (e.g. the newcomer is proactive in asking questions about the expectations and evaluations of co-workers and supervisors). |

*Table 3: Bauer's New Employee Adjustments (2010)*

| Adjustments | Description |
|---|---|
| ***Self-efficacy*** | Self-confidence in performing the job role. More confident new employees will be more motivated and successful than less confident counterparts. |
| ***Role clarity*** | How well a newcomer understands his or her role and the expectations of the role. |
| ***Social integration*** | Integrating socially into the organisation involves forming effective working relationships with co-workers. |
| ***Knowledge of the [organisational] culture*** | Understanding the organisation's values, goals, politics, and unique language. |

We used Bauer's model (2010) to frame the third phase of our analysis because it is empirically based, highly cited in many fields, and hasn't been specifically applied to agile software development project team onboarding. The method is described in detail below.

## 3 Method

An organisation approached our research group and asked for assistance in understanding how team members shift from an individual view of working to a team-oriented view when they joined an agile team for the first time. A literature search found few papers investigating how new members are helped when joining an agile team, although there is much exploration in the agile adoption and transformation literature about how entire teams or organisations transition from traditional project management towards agility (Gandomani and Nafchi, 2015). To fill this gap a single case study method was selected, and through discussion with our key contact we decided to investigate the phenomenon of onboarding into an agile team so we could explore the complex and interrelated factors that affect the experience to a depth not possible with other research designs. The unit of analysis was the co-located agile software development team (Yin, 2018). The University of Central Lancashire gave ethical approval for the research.

We took the perspective of the whole team, including newcomers and insiders. 'Newcomers' for this study are new hires and those who had worked up to a year in the project team; 'insiders' are established team members who had worked for a year in the project team or more and included the team lead who had been in the team for the longest time. Because the team had diverse employment patterns (see the team description in Section 4) there were several 'new' aspects of the job that newcomers could experience when they started with the team. These included being new to working in: this organisation, this team, an agile team, a software development team, and a professional job. Some employees were new to *all* these aspects when they joined the team, whereas others were only new to some of these aspects. We collected data from the whole team because the team worked as a unit and everyone helped with onboarding in some way. Newcomers were able to provide current insights into what it was like being new to the team. Insiders were able to provide a wider range of insights because they could still remember their experiences of being new but also had reflected on them and had observed others going through the onboarding process, so they were able to provide a different perspective.

Data was collected using interviews, observations, and informal discussions. All staff in the agile team were invited to take part in the study, provided with an information sheet about the research, and asked to consent. All the team consented to being observed and more than half to being interviewed. Table 4 shows the profiles of interviewees. Details about the history, context, and development of the team were collected during several online and face-to-face informal meetings between the lead researcher and the team lead before and after the main study period.

Nine interviews took place between October and December 2018. Two researchers carried out the interviews. The interviews were semi-structured and followed an interview schedule (see appendix). However, the interviewers strived to remain open to new ideas and probed for additional information when necessary or relevant to the topic. Interviews were on average 36 minutes long with a range between 22 and 53 minutes. The interviews were audio recorded with participants' consent and subsequently transcribed and input into NVivo for analysis. They were the primary source of data for the study.

Three observations of the team's working practices took place during October 2018. The lead researcher carried out the observations which included a two-hour Sprint Planning Meeting, three hours of a normal working morning in the team space, and a two-hour Sprint

Review and Retrospective Meeting. The purpose of the observations was to extend our understanding of the context by getting to know the team members, watching how they worked, and examining aspects of the team culture. Hand-written field notes were used to record observations, without a template, during and immediately after the session. They included who was present, a diagram of the room layout and seating plan, description of the activities observed, a summary of the main points of conversation, and the researcher's reflections. Field notes were typed up and input into NVivo. These were not coded in the same detail as the interviews but underwent light-touch coding. This data was used to corroborate findings from the interviews and to understand aspects of the case study in more detail, for instance how the team worked together during sprint planning.

We used a systematic qualitative coding approach to analyse our interview data (Saldaña, 2015). This is an approach for identifying patterns and developing theory from a body of data. Using an abductive process as suggested by Timmermans and Tavory (2012), we moved iteratively between generating our own codes and using categories from Bauer (2010) to shape our analysis. Generating our own codes allowed us to freely explore all aspects of our data without limitations, while the use of theory helped us to pull together insights into a coherent framework and link our findings to those of others. The abductive process required four analytical stages:

1. During the first stage, all the interview and observational transcripts were read to gain an overview of the team's history, work practices, and culture. Descriptive information about the interviewees, such as previous experience of agile methods and length of time in the team, was also tabulated (see Table 4).
2. During the second stage, interview data were inductively analysed by the first author without using existing theory to guide the process. This involved using two cycles of coding as described by Saldaña (2015 p.68). In the first cycle the first author closely examined the data and tagged relevant sections with initial codes. In the second cycle the codes were grouped into four organising patterns or themes. This stage was inductive, exploratory, and laid the ground for the subsequent analysis. However, the themes from this stage were not used again as they related to the whole approach the team used, not just onboarding.
3. During the third stage, we returned to the literature and identified Bauer's model of onboarding (2010) as a relevant theory for exploring onboarding. We used this as a frame for our analysis because it provided a useful set of meta-level categories. Saldaña calls this approach 'elaborative coding' in which a previous study's categories are used to aid the analysis of the current study (2015 p.255). During this stage the second author undertook another cycle of low-level coding and then applied Bauer's six Onboarding Functions as organising patterns. The outcome was 27 codes grouped into the six sub-categories from Bauer's model (Table 5).
4. During the fourth stage, we returned to Bauer's model (2010) and looked at adjustments. The first author undertook another cycle of low-level coding and then applied Bauer's adjustments as organising patterns. Bauer's four adjustments did not account for all the adaptations we observed in the data therefore some new subcategories were identified. The outcome was 10 codes grouped into 7 sub-categories and 2 adjustment categories (Table 6).

All these results were discussed in depth first between the first and second authors, then with the whole research team and the research participants.

## 4 The History and Nature of the Agile Team

The agile software development team was a unit based in a UK university within the IT services section (ISS). Over six years, the unit increased from two members at inception to 15 at the time of the study although the increase was not linear, as the team also grew and shrank during the course of each year depending on how much work they had to do. During the case study, the unit acted as a single team and followed a whole-team approach. The unit had a twofold remit, first to develop mobile applications for the university, and second to investigate new ideas and technology for future innovation. As UK universities run as large businesses, this unit was a professional IT service developing business applications to enhance the functioning of the university. They mainly worked on new projects but also had to maintain previously deployed apps and systems. At the time of the study the team were two months into developing a new product. Most of the team were working on the new product but some were doing maintenance tasks on existing products.

Team membership and size changed over time depending on workload, consisting of full-time and part-time staff. At the time of this study, the team comprised 6 full-time staff, 6 part-time staff and 3 student apprentices. The team lead was full-time and had a dual role as Scrum Master and line manager. Of the part-time staff, three had full-time roles elsewhere within the university and were seconded part-time to this team. The other three part-time staff were students on undergraduate or postgraduate courses at the university who were working on the project team, typically for two or three mornings or afternoons per week. In contrast to the university students, student apprentices worked full-time with the team for most weeks but attended block courses, usually for one week per month, at their teaching institute which was outside the university. The student apprentices started straight from school or college without prior work experience. Most of the team were in their 20s with little or no previous work experience except the Scrum Master, Product Owner and Conversation Specialist who were in the 35-55 age group and had a wide range of previous work experience. Some of the full-time staff had started as part-time students and gained full-time permanent posts as new graduates. There was regular staff turnover as part-time students and apprentices usually left after graduating, and full-time staff were often attracted by jobs outside Higher Education.

The team were co-located in an open-plan office space with an adjacent meeting room. The developers used a hot-desk system and often changed the configuration of their desks to suit themselves. The team used a Scrum approach, running two-week sprints, with the last Friday used as a non-Sprint day to complete other work. They used the following agile practices: daily stand-ups, sprint planning, sprint refinement, sprint review, retrospective meetings, product demos and had a Scrum wallboard (Schwaber and Sutherland, 2020). The team lead held weekly one-to-one meetings with staff if they wanted them. The general feeling among the team was stated by a staff member who had been with the team for a year, *"Personally, I love it. It's very relaxed. It's quite dynamic, the way we do things. It's just a nice workplace" [SD1].*

Details of the nine members of the team who were interviewed are shown in Table 4. The team lead [TL] and assistant project manager [PM] were both established insiders and had been in the team longer than anyone else. The three full-time, permanent software developers [SD1], [SD2], and [SD3] had all been in the team for at least a year and were therefore 'insiders' in this study. The four 'newcomers' we interviewed [NC1], [NC2], [NC3] and [NC4] were also part-time and temporary in the team. This was the most common type of newcomer in the team as there was more turnover of part-time temporary staff than of full-time staff.

*Table 4: Profile of Interviewees – TL = Team Lead, PM = Project Manager, SD = Software Developer, NC = Newcomer*

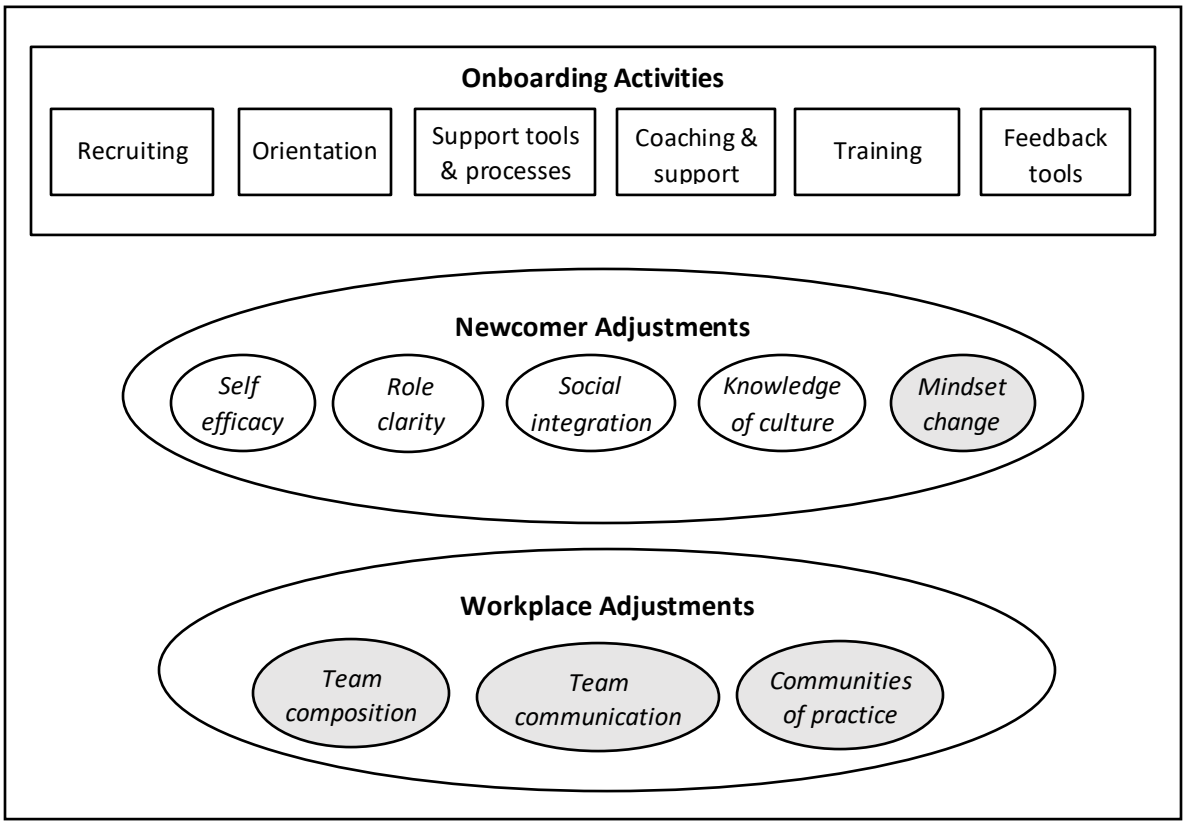
| | Role and code name | Work Mode | Duration in role | Experience |
|---|---|---|---|---|
| 1 | Team lead, Scrum master [TL] | FT Perm | 10 years | Degree, Certified Scrum Master<br>Prior development and agile experience in other organisations |
| 2 | Assistant project manager [PM] | FT Perm | 6 years | Degree<br>4.5 years PT in this team while a student then FT<br>No prior development, agile, or work experience before joining this team |
| 3 | Software developer 1 [SD1] | FT Perm | 5 years | Degree<br>4 years PT in this team while a student then FT<br>No prior development, agile, or work experience before joining this team |
| 4 | Software developer 2 [SD2] | FT Perm | 1 year | Degree<br>3.5 years prior development experience<br>Prior development and agile experience in another workplace |
| 5 | Software developer 3 [SD3] | FT Perm | 1 year | Degree<br>2 years PT in this team while a student, then agile development experience in another workplace, before returning to a full-time role in this team |
| 6 | Apprentice developer [NC1] | FT Temp | 8 months | Studying<br>No prior development, agile, or work experience before joining this team |
| 7 | Software developer [NC2] | PT Temp | 4 months | Studying<br>Minimal prior development experience<br>No prior agile or work experience |
| 8 | Software developer [NC3] | PT Temp | 3 months | Degree<br>Prior agile and development experience<br>3 years FT permanent developer at the university in another section, seconded to this team PT temp |
| 9 | Conversation specialist [NC4] | PT Temp | 3 months | Degree<br>Many years of work experience as a teacher<br>No prior development or agile experience. New to working at the university |

## 5 An Onboarding Model for Agile Teams

Following the abductive analysis approach described in section 3, we developed an onboarding model for agile teams. This new model is an adaptation and extension of Bauer's onboarding model (Bauer, 2010). Figure 1 illustrates the onboarding model for agile teams, which is fully explained in the detailed sections that follow.

The three categories of the model shown in Figure 1 are presented along with their sub-categories and codes in sections 5.1 Onboarding Activities, 5.2.1 Newcomer Adjustments, and 5.2.2 Workplace Adjustments. Data sources for the descriptions in this section are interviews, observations, and informal discussions with team members. Verbatim quotations come from interviews and are followed by the role code name of the interviewee (indicated in Table 4). As recommended by Saldaña (2015 pp. 71-74) we do not quantify how many participants mentioned each code, because this is a qualitative case study, In the following sections we use relevant quotes to illustrate the codes presented and ensure that overall we use at least one quotation from each interviewee.

*Figure 1: An Onboarding Model for Agile Teams (adapted and extended from Bauer, 2010). Shaded shapes indicate new sub-categories identified in this study*



## 5.1 Onboarding Activities

The team's onboarding activities are described in this section. These are organised into six sub-categories and 27 codes (see Table 5). Together they describe the factors that play a role in onboarding newcomers to the team.

*Table 5: The Onboarding Activities Category, with sub-categories and codes*

| *Category* | *Sub-Category* | *Code* |
|---|---|---|
| **Onboarding Activities** | ***Recruiting*** | Long-term recruitment strategy |
| | | Onboarding during recruitment |
| | ***Orientation*** | New staff pack |
| | | Working with the client pack |
| | | Agile method pack |
| | | Socialising |
| | ***Support tools & processes*** | Information radiator |
| | | Communication tools |
| | ***Coaching & support*** | Mentoring |
| | | Role modelling |
| | | Pair programming |
| | | Ceremonies |
| | | Daily Stand-up meetings |
| | | Co-location |
| | | Encouraging teamwork |
| | | Encouraging learning |
| | | Empathy |
| | ***Training*** | Immersion |
| | | Self-study |
| | ***Feedback tools*** | One-to-ones |
| | | Immediate feedback |
| | | Code reviews |
| | | Testing |
| | | Retrospectives |
| | | Sprint Reviews |
| | | Sprint Refinements |
| | | Small tasks |

### 5.1.1 Recruiting

The recruitment process was formal and standardized for all staff who join the organisation. The process differs for full-time permanent and part-time or fixed term (university students, student apprentices, and some others) newcomers. The team lead explained that experience and technical knowledge are expected of full-time permanent applicants, and technical knowledge is expected of university students, but neither are expected of apprentice student applicants. Once hired, full-time members get an institutional induction. All newcomers get a personal welcome from the team lead and are assigned a mentor.

*Long-term recruitment strategy*: The unit had a long-term recruitment strategy (Bauer, 2010) that involved hiring students and apprentices who would work within the team as part-time employees whilst completing their studies. In some cases, these temporary student employees would finish their degree and then become full-time permanent staff members. This approach provided permanent staff who required minimal onboarding because they had a pre-existing good team fit, and understood the organisation, the unit's goals, products, technologies, stakeholders, and the team's agile approach.

*Onboarding during recruitment*: During recruitment, newcomers' knowledge gaps were identified (Bauer, 2010). *"One of the things I do is in the interviews when we take people on, I try to understand what their understanding of agile is, to see how much of a gap there is to see how much of a gap there is to get them in. " [TL].* The team did not expect newcomers to be familiar with agile approaches before they joined the team. With an understanding of a newcomer's knowledge gap, the team lead could ensure the newcomer received appropriate resources and assistance to help them learn the agile approach.

### 5.1.2 Orientation

The team lead and the project manager were responsible for formal orientation (Klein and Polin, 2012). They had developed various information 'packs' over a number of years and shared these with new recruits as soon as they had accepted a post. On their first day newcomers would first spend time with the team lead or project manager and then be introduced to the team and start working alongside another team member.

*"New staff" pack*: This document described things that new employees need to know, was given to all newcomers, and provided an easy reference for them in the first weeks.

*"How our team works with the client" pack*: The document explained how the team writes user stories (short, simple descriptions of software features), what communication the team expects from the client, and how the team tests and signs-off products. It was sent to clients but also helped newcomers learn how the team worked with clients.

*Agile method pack*: *"New team members, I now send them a guide, the principles behind it. A Scrum Guide. I talk about the fact that this is what they do" [TL].* The Scrum guide is a standard text on the agile method Scrum that is widely used as a reference (Schwaber and Sutherland, 2020). This helped newcomers learn about agile and about the team's agile approach.

*Socialising*: The project team made efforts to socialise with, and get to know one another, because they found this helped newcomers and insiders to develop trust and be more confident in interacting and communicating with one another. The team lead introduced techniques within the workplace, *"for example, practices that we've encouraged in our full-time team meeting, we'll say 'what can you present to the team that you think is valuable or about yourself?', you know breaking down those barriers, it could be about anything. So [a team member] recently did one about e-capture and [another team member] did one about his passion for Rubik's Cubes"* [TL]. They also organised social events outside the workplace, such as playing games at lunchtime or going to the pub. For newcomers this was the start of settling into the team and building a trust relationship with other team members, which is an important part of onboarding for newcomers into any workplace (Klein and Polin, 2012) It was also a continuous practice that helped to maintain trust for the whole team. Trust is a crucial team quality in agile teams that is specified in founding agile documents (e.g. the Agile manifesto (Fowler and Highsmith, 2001)) and mutual trust is also critical for high functioning teams in general (Salas, Sims and Burke, 2005).

### 5.1.3 Support Tools and Processes

The team used various online communication tools that newcomers had to learn. There was also a physical board in the room that was usually propped up on the floor. None of the tools contained comprehensive information and they needed to be supplemented by conversations as the status of a task or a problem could change rapidly, and those changes were not necessarily documented.

*Information radiator*: An information radiator is a physical object, such as a wallboard, that provides a visual summary of the team's work and is kept plainly in sight. They are commonly used in agile teams (Cockburn, 2006) to coordinate work, and serve many purposes. The team's wallboard displayed information about user-stories under construction, specific tasks to be addressed, task allocation, task progress, project issues, and work that was considered complete (Sharp, Robinson and Petre, 2009). The wallboard had physical and virtual versions, although they preferred the physical board. The established members thought it was useful for newcomers. *"Sometimes the team don't necessarily engage quite as much with a digital thing as with a physical thing, it seems to be a bit more natural…I think it helps [the newcomers] as well because it's a more instantaneous way to look and see where things are."[SD1].* The wallboard was viewed by one newcomer (an architect) as useful for developers but not for him, *"it's all development tasks that are on the board… But then, my work is stuff that just supports all of that, and sometimes it's like, I want to write a story that is… 'As an architect, I want'" [NC3].* Therefore, while this agile practice was useful for most newcomers because it gave them easy access to a project overview, it was not useful for all newcomers.

*Communication tools*: The team used communication tools including Microsoft Teams, Slack, TFS, and email. Communication tools are frequently used in co-located agile teams (Calefato *et al.,* 2020). These tools also helped newcomers as they could spend time reading through them to learn about the status of the project. However, there was often a lot of detail missing from them because the team were co-located and often used rapid verbal communication to share knowledge rather than tools (Sharp and Robinson,2010). New part-timers said they couldn't entirely rely on looking at Teams to catch-up with what had happened while they were away, so they had to spend time talking to their team-mates on their first visit during the week.

### 5.1.4 Coaching and Support

Many facets of the agile approach facilitated support for newcomers, such as pair programming, agile ceremonies, co-location, and the whole team approach. Newcomers started contributing to production code on their first day. Since there was so much to learn, they were supported through pair programming while doing productive work. The agile ceremonies (explained below in the ceremonies section) were used as opportunities for learning about the agile way of working and about the current project. When new staff started, extra time was allocated to explain the ceremony and to share project knowledge. Co-location made it easy to ask for support and team members developed methods to signal their availability, such as putting on headphones when they didn't want to be disturbed. More widely-used support practices for new employees such as mentoring and role modelling were also used (Bauer, 2010). The team lead encouraged a supportive and learning-oriented atmosphere and team members showed empathy for newcomers as they could remember their own experiences of starting out. At the time of our case study, the team had several experienced full-time members who could take on mentoring and coaching roles. In previous years, the team lead was the only one with enough experience to mentor new staff.

*Mentoring*: Mentoring was viewed as an important part of the onboarding experience for most newcomers. The team lead was frequently referred to as a good mentor, but he also recognised the unacknowledged mentoring role that established team members undertook as they communicated informally with newcomers in the team workspace: *"from my perspective the mentoring aspect of things, it helps both with the integration into the unit, the integration with the technology stack and the integration into the agile way, and it's kind of almost subliminal. The messages come across from the team members rather than from me, which, I hope, [the newcomer] would learn better because of that" [TL].*

*Role modelling*: The more experienced team members noted that role modelling desired behaviours was beneficial for newcomers. This was particularly important for getting across the importance of honesty and transparency, *"I try and get rid of the stigma … and set an example, and the rest of the team will realise that it's fine to say 'I don't know how to do that. I don't know what this is or that is, or I need help with this'" [SD3].* Another type of role modelling was achieved by established staff showing an eagerness to continuously self-improve by learning new technologies: *"I do a lot of learning outside of work at the moment, especially with all the new stuff that we're doing" [PM].*

*Pair programming*: Pair programming is a technique in which two programmers work side by side on one task at one workstation. The team lead recognised that pair programming was useful to support newcomers to learn the code base, the team's coding processes, and specific facets of their approach, such as testing conventions. It was particularly useful in the first few weeks of onboarding when it was used as the primary mechanism for exchanging knowledge and mentoring. *"When I turned up first day we started working together and they were like showing me something … I was paired with my colleague and then they put me with some other colleagues. They tried different things … [it lasted] maybe a month"* [NC2]. It was also a way of helping newcomers to acclimatise to working intensively with other team members and engage in regular technical communication that happens in the agile development environment. *"When they first come in, I pair them up with a full-time member … the same full-time member for about 2 to 3 weeks until we then release them to work on their own on a particular area." [TL].* This practice required the team to contain sufficient established members to take on the role. It was therefore a practice that became easier as the team grew.

*Ceremonies*: Scrum has four 'ceremonies': the daily stand-up, the sprint planning meeting, the sprint review meeting, and the sprint retrospective meeting (Schwaber and Sutherland, 2020). As part of the immersion approach these were explained to newcomers the first time they attended. For example, just before the stand-up meeting, a newcomer would have the process explained so they knew what they were expected to do, *"they were very good at explaining everything they did, explaining why they had stand-ups in the morning, and explain the meetings, you know, before and [at] the end of the sprints. They explained that before they happened" [NC4].* During the ceremonies experienced staff members would model knowledge, skills, and behaviours that newcomers needed to learn. When a ceremony took place soon after a newcomer started, the team would take more time to provide extra explanations about the project and its challenges than they normally would. In this way everyone in the team adapted to account for the experience of attendees. The ceremonies exemplify the whole-team ethos of agile approaches. Newcomers were exposed to, and learned about, all aspects of the project by taking part in them.

*Daily stand-up meetings*: Daily stand-ups are one of the Scrum ceremonies. They are short, daily, time-boxed status meetings that the whole team participates in. They are widely used in agile teams and research has found that junior members often find them particularly useful (Stray, Viktoria, Moe and Bergersen, 2017). In this team, stand-ups were viewed as an essential communication vehicle that were found to help newcomers. *"A lot of the communication comes at the stand-up in the morning ... We also have another, sometimes, in the afternoon if someone's come in just to get them on board. So, we might have two stand-ups" [SD2].* One developer, with only one year of experience on the team, also thought stand-ups were useful for getting help, *"If you're stuck on something, don't know how to do something or you're just lost, then it's a good place to air that and usually, somebody will, oh I'll help you with that." [SD3].* However, one of the apprentice newcomers

noted that she did not yet understand all the technical terminology being used, *"If I understood their language, then I would probably understand more" [NC3*].

*Co-location*: The agile practice of co-locating the whole team in one room allows them to talk to each other informally and to overhear others' conversations. It allows for close co-operation between team members and is regarded as contributing positively to team performance where it is possible (Dingsøyr and Lindsjørn, 2013). It was an important mechanism that helped our case study team to rapidly onboard newcomers by making it easier for them to learn and ask for help, *"…look at this code, or something like that but also, just asking the person you're sat next to… If you don't know something, there's a good chance the person next to you does" [SD3]*. Being able to overhear conversations also helped newcomers, *"It does [give a] general sense of what other people are doing, even if it's just overhearing them … talking between themselves" [NC1].*

*Encouraging teamwork*: Working as a whole team, and building trust are essential to agile approaches (Fowler and Highsmith, 2001). The active encouragement of teamwork and regular whole-team activities helped newcomers to settle into the team. The established team members encouraged knowledge sharing and supportive behaviours among the team in general, and particularly towards newcomers. Newcomers reported that this made them feel able to ask questions, *"everyone is very friendly, and ask if you want anything and yeah, you are encouraged to talk to people." [NC3].* The level of trust between newcomers and established staff was perceived as good, *"There's a lot of trust… especially with the student developers as well, there's a lot of trust for them to do work, … once they're part of the team, and they fit and work as part of the team, we trust them to do work … Everyone is very helpful, very friendly ... it feels very inclusive, very inclusive, it's not sort-of developers and non-developers" [NC3].*

*Encouraging learning*: Learning is an important part of onboarding (Bauer, 2010), and continuous learning is an important part of agile working (Fowler and Highsmith, 2001). Regular reflection is embedded into agile processes to help teams maintain and improve their software development processes (Babb, Hoda and Nørbjerg, 2014). Newcomers were encouraged to learn new things early on. This newcomer reports how she appreciated being encouraged to be adventurous, *"[The TL] is very good at encouraging you to take on more challenging things. … He'll suggest, why doesn't [NC4] do that, why don't you do that [NC4]? Initially, I'll go ohhh (shouting in confusion and panic!) and then… But in a good way, it is good to push your staff, isn't it? It is good to learn new things and yeah. Yeah, it is good. Scary but good. Good scary" [NC4].*

*Empathy*: Because some established team members had previously been student members themselves, they could recall their own experiences and they said this helped them to understand newcomers' issues. As the team usually contained several newcomers it was important that more experienced staff were sympathetic to the newcomer experience (Pavlina, 2020). *"I'd like to think anyway, that we treat the students, especially with my background as a student developer, that we're all treated as equals. We don't really have the junior developer syndrome that some teams suffer from where they're handed lesser tasks … Sometimes if a part-time student is only in for 3 hours or something, then there might be a situation where we might suggest things for them, just to maximise that time that they have. But it's more for their benefit because I know how frustrating it is to get into a piece of work and then have to down tools and go to lectures" [SD1].*

### 5.1.5 Training

The approach taken to training was to get newcomers to work closely with more experienced team members while being immersed in the working environment. This was very effective for

bringing them up to speed with technical and requirements aspects of the work. There was no formal training in agile methods, and few opportunities to learn from others outside the team as there was only one other agile team in the IT section. Many newcomers undertook self-study to learn unfamiliar technologies and agile practices.

*Immersion*: (or experiential learning). Newcomers started working on their first day and much of their learning and socialisation was accomplished by being a productive member of the team. This is very much in line with the agile approach, in which teams work to deliver value as soon as they can (Fowler and Highsmith, 2001). The team lead explained: *"Generally we try to let them get their hands into a piece of work, learn literally on the job, so we give them a sort of induction into what their sort of expectations are in the team, what they can do to get support and all that kind of stuff and just let them loose and fit right in" [PM].* A newcomer's perception reinforced this*, "I was very much thrown in at the deep end, 'Here are some meetings. Yeah, let's go ahead with it,' and very much learning on a day-to-day basis with the team how they do it. It's really largely practice or very practical, with some explanations when necessary… before we went into the meeting and we were voting with our animal cards[1]* and things, that was explained to me before we went in, we do this, so… I got in there and wasn't surprised by what happened" [NC3]*.

*Self-study*: Newcomers who were not experienced with agile methods were asked to read about it before starting and were given links to online resources (Klein and Polin, 2012). *"In the interviews, we tend to ask them if they have any experience of agile, and if they say no, we say, 'That's fine, but we recommend you look into it'" [PM].* The team lead expected newcomers to self-study and would request them to do so, *"when we took him on, we said 'you need to do some learning outside of work if you want to continue with the team'" [TL].* For some newcomers, the self-study was self-motivated, *"I did a lot of background work …I did lots of reading [about Alexa] on the internet… A couple of courses on Udemy …At home, I am doing Python and Excel, I am doing a course on Excel. And … I have just signed up for, … user stories" [NC4].* It is perhaps not surprising that self-study is encouraged in a university setting, but self-improvement is also present in successful agile teams (Hoda and Murugesan, 2016). This might be contentious in other settings, but it was a technique noted by Buchan *et al.* (2019)

### 5.1.6 Feedback Tools

The immersive approach to embedding newcomers meant that they received feedback about their work early and often. Newcomers were given small tasks and worked in a pair, so they received regular informal feedback from the team members they were working with and received testing feedback quickly because they were developing small features. More formal feedback was received during sprint reviews, code reviews and retrospectives. These were particularly useful for newcomers but were also normal working practice.

*One-to-Ones*: All full-time newcomers had regular, often weekly, one-to-one meetings with the team lead (Klein and Polin, 2012). These started soon after they joined the team and continued even for more experienced staff. It gave them a chance to receive guidance about technical issues and reflect on their work practice.

*Immediate feedback*: Newcomers would get timely feedback about their work because the team were working in two-week sprints and new features were tested and integrated as soon as they were ready. Feedback is important for any newcomer (Bauer, 2010), but the agile practice of working incrementally and iteratively (Fowler and Highsmith, 2001) helped newcomers because they got regular feedback while they were working on a task. One

[1] At the time of this interview animal cards were being used to size stories during planning meetings

newcomer explained after her work was tested, *"people do point things out, but in an ok way… but it is always nicely done". [NC3].* Over the course of a two-week sprint a newcomer may receive feedback from a wide range of roles including team members, the Team Lead, the Product Owner, project stakeholders, and end users.

*Code reviews*: Informal code reviews, in which a group of developers read and talk through a section of code, were used for providing feedback and to aid learning for everyone including newcomers. These helped newcomers to learn by listening to the whole group discussing detailed points about coding practice and problem solving. *"We do a group code review each week to see what we've been going over, to learn off each other. That meeting is primarily just for the programmers and the apprentices" [SD2].* A newcomer apprentice, who had not yet presented at a code review, explained how useful they were as a way of getting familiar with the team's code, *"At the moment I don't quite understand everything. But it is useful because it can be quite scary to have a look at the [code], it makes it a bit more familiar" [NC1].*

*Testing*: Unit tests were viewed as an early feedback mechanism for newcomers. Test-driven development, which is a recommended practice for agile development (Beck, 2000), was also used alongside pair programming to assist newcomers, *"We do try pair programming, especially with the students… so, when [NC2] started, we actually added some test-driven development with him to introduce him to what we're working on, how we work" [SD2].* A newcomer explained, *"which is really good … that extra bit of testing is, and then I can see whether it does what I hoped it will do and if it works" [NC4].*

*Retrospectives*: Sprint retrospectives are one of the Scrum ceremonies, during which the agile team reflects on and improves their work processes. They run at the end of each sprint. This involves identifying and discussing obstacles, feelings, previous actions, background reasons, and plans (Andriyani, Hoda and Amor, 2017). Retrospectives helped newcomers in the team to learn about reflection, continuous improvement, and key agile values. The team lead sometimes discussed aspects of agility in retrospectives. Established team members viewed them as a valuable tool for the whole team, including newcomers. *"We do it [give feedback] at the retrospectives or we give feedback on how we did, what we liked, what we'd improve. So that's more feedback as a team" [SD2].*

*Sprint reviews*: Sprint reviews are one of the Scrum ceremonies, during which the latest version of the product is reviewed by the customer and the agile team (Schwaber and Sutherland, 2020). These reviews helped newcomers adapt to receiving customer feedback and critique which are both essential elements of agile working. Feedback at the sprint review was concerned with technical matters, *"… we do a review session where we demo the build. Hopefully, it works, and we can celebrate, or there will be some critique about the way it's been implemented or the design choices, or that kind of stuff" [SD1].*

*Sprint refinements*: Sprint refinements are part of Sprint planning, which is one of the Scrum ceremonies during which user stories are discussed, refined, estimated, and selected for inclusion into the next sprint (Schwaber and Sutherland, 2020). These helped newcomers to understand requirements for the whole product, learn about estimation, and make team decisions. During the sprint refinement session that was observed the team had a discussion, geared towards newcomers, about how to make more realistic estimates of how much work they could take on in a sprint. At the time of the study the team were trying to get better at planning. They had separated out the sprint refinement meetings from the sprint planning meetings so they could spend more time going through user stories in detail before having a sprint planning session. *"We have Sprint refinements before we do a planning, where we go through each of the work items and ask a lot of questions" [SD3].*

*Small tasks*: Agile teams often break down large user stories into smaller tasks, so that developers can code and test them in a few days. This is an important practice for ensuring that estimation and sprint planning are as accurate as possible (Cohn, 2004). While it is generally considered good practice to start with small tasks (Davis and Kleiner, 2001), it was useful for the team to be able to give newcomers smaller, simpler tasks at first. Newcomer part-time university students were deliberately given tasks that could be completed within a working session, rather than having to stop halfway through and pick it up again several days later when they next came in. *"We'll give smaller tasks to the students because there's just not enough time ... if we've got a small user story, say, getting the next timetable event from an API, that's something that we could see a student doing" [SD2].* Minor bug fixes were often an entry point for newcomers, *"I'll have like a list of bugs that need fixing because generally, we don't want to pull the full-timers out of sprint." [PM].*

## 5.2 Adjustments

Bauer (2010) describes 'adjustments' as levers related to job roles and the social environment that help newcomers to onboard successfully, and lists four adjustments: self-efficacy, role clarity, social integration and knowledge of culture. We found evidence in this case study of Bauer's four adjustments but also identified some additional adjustments, several of which were highlighted because they were challenges for the team. We have therefore extended Bauer's concept of adjustments by adding four new adjustments and distinguishing between two categories of adjustment. First, we present *newcomer adjustments*, which include Bauer's original four adjustments to which we have added a fifth, *mindset change*. Newcomer adjustments relate to the individual who is adapting to a new environment, such as their belief in their ability to succeed, or how well they integrate into the team. Second, we present *workplace adjustments*, which is a completely new category containing three new adjustments: *team composition*, *team communication* and *communities of practice*. Workplace adjustments relate to the way the workplace is set up to cope with integrating newcomers, such as how teams are configured or whether there is an appropriate community of practice to support team members. The adjustment categories, sub-categories and codes are summarised in Table 6.

*Table 6: The Newcomer Adjustments and Workplace Adjustments categories, with sub-categories and codes*

| *Category* | *Sub-category* | *Code* |
|---|---|---|
| **Newcomer Adjustments** | ***Self-efficacy~*** | Empowerment |
| | ***Role clarity~*** | Reimagining |
| | ***Social integration~*** | Joining a team |
| | ***Knowledge of culture~*** | Knowledge of agility |
| | ***Mindset change**** | Tackling problems |
| | | Becoming agile |
| **Workplace Adjustments*** | ***Team composition**** | Adjusting the team |
| | | Mentor availability |
| | ***Team communication**** | Accommodating part-timers |
| | ***Communities of practice**** | Agile community of practice |

*Key: ~ = Bauer's adjustments (2010) * = New adjustments identified in this study*

### 5.2.1 Newcomer Adjustments

In this section we describe how the five newcomer adjustments were relevant for newcomers in the case study.

#### 5.2.1.1 Self-efficacy

Self-efficacy is defined as an individual's belief in their ability to succeed in a situation or accomplish a task (Bandura, 1982). Self-efficacy is related to empowerment because belief

in one's ability to achieve a task is a precursor to being able to work independently to achieve that task. Work in agile teams is predicated on the assumption that team members are empowered and can self-organise to accomplish the tasks they take on, either by working on their own or working together with other team members (Fowler and Highsmith, 2001).

*Empowerment*: had occasionally been an issue within the team, although it improved over time as the number of full-time experienced members in the team increased. The team lead identified a difficulty with onboarding younger newcomers who had never worked in a self-organising, empowered team. He thought they needed to be helped, *"when they're just out of university and they've come from an academic background that doesn't teach team work very well, doesn't teach about empowerment … sometimes in conversations, they may turn to me in terms of a position of authority and I'm like, no you go and do that, so I've tried to set up things where they have their own meetings and they run their own meetings so I may well initiate something and step out and say well there you go, you don't need to talk to me anymore, just sort it out yourselves." [TL].* However, at times of pressure, a command-and-control approach did emerge, "*and then I'll pull someone out of sprint and go, 'This needs fixing,' or I'll say, 'This will be fixed at the end of the sprint, depending on how urgent it is'" [PM].*

#### 5.2.1.2 Role clarity

It was important for newcomers to understand their role within the team. Most developers had specialist skills such as front end, back end, or systems development skills, and would often choose tasks that played to those strengths. However, members of an agile team must be prepared to take on a wide variety of tasks and cannot only work to their specialist strengths. Role clarity was developed over time in relation to other members of the team, and with the help of the team lead who supported newcomers individually to find their place. Reimagining was one technique used by the team lead that helped staff to think themselves into a new role.

*Reimagining*: The team lead used one-to-one reflection meetings to discuss ways of working with newcomers to help them reimagine themselves in their new role. This helped them make the transition, *"when I've taken students on and they've transitioned to being full-time members of staff, I've tried to coach them to say you need to reimagine yourself in the new role. So [newcomer]…, she was an administrator but now she's a, well technically her title is [new role], but that's actually different to what she does and she's had to reimagine herself in those new roles because she's no longer doing the roles that she was doing earlier on" [TL].*

#### 5.2.1.3 Social integration

Social integration helps newcomers become part of an agile team because team members work so closely together. At the time of the case study the team were physically co-located and had created a team atmosphere that could accommodate periods of concentrated work, periods of problem solving and discussion, and periods of relaxation and fun. Newcomers highlighted the friendliness of the team.

*Joining a Team*: Newcomers reported finding it easy to integrate into the team as there was a friendly atmosphere at work; *"They are very, very friendly, it's not like any other office I've worked in. But I never worked in tech before, so I don't know" [NC4]*. Newcomers reported feeling comfortable to ask team insiders for help and support very quickly … *"Everyone is really friendly and helpful, because I am starting to do the development side, everyone is ... even though I am doing very elementary stuff ... they are very helpful and always willing to give me their time" [NC1].* As a team they often played games during their breaks and did

other social activities together *"We make sure we have nights out, like meals, nights out, things like that around Christmas, around the start of each term, when we get new staff, that kind of thing. We went out for drinks last week" [PM].*

#### 5.2.1.4 Knowledge of culture

Team culture is the shared context and belief system that operates at several levels to influence how things are done. The team had a strong and unique identity within the organisation as they had adopted the Scrum method, which was not used by other teams. They did a different type of work from the other IT Services teams and had an unusual make-up combining full-time, part-time and student employees. The co-location of the team and regular agile ceremonies helped newcomers to get to know the team and adopt their work culture. The team members reported that the best way to help a new team member to learn their culture was through experiential learning within the team.

*Knowledge of agility*: The team lead thought the main onboarding issue was integrating relatively young and inexperienced part-time newcomers with little knowledge of how agile works in a professional environment into a team that had a strong agile culture. He believed that immersion was the best way to achieve that quickly. *"It's just them being immersed in it, and for part-time that's hard, because up to 15 hours a week, whilst doing other learning, and whilst you're young and having a social life, and everything else. Finding space in their brain for this is hard, and it's being able to get over the principles and culture, which is what I want to focus on" [TL].* The team lead had high expectations of newcomers and struggled to convey agile principles. *"And we now have a very high bar of workforce that are … highly motivated and through that, there's an expectation that you have to fit into that kind of ethos as well, and that becomes a barrier for recruiting new students because the bar is so high" [TL].*

#### 5.2.1.5 Mindset change

Mindset change is important for agile teams because research into the agile mindset has found it is a significant concept that is an essential part of agility (Miler and Gaida, 2019; Mordi and Schoop,2020). A mindset refers to someone's attitudes or ways of thinking and therefore differs from knowledge of culture. Mordi and Schoop (2020) use literature and primary data to define the agile mindset as "*a mindset based on the values and principles of the Agile Manifesto, whose main characteristics are trust, responsibility and ownership, continuous improvement, a willingness to learn, openness and a willingness to continually adapt and grow"*. The agile mindset comprises a wide range of features that Miler and Gaida (2019), in their survey of agile professionals and literature, group into four categories: support for business goals, relationships within the team, individual features (i.e. openness to change, continuous learning), and organisation of work. In their study the five most highly ranked features were: searching for problem solutions, being motivated, helping each other, mutual listening and focussing on achieving a common goal. The mindset changes that the team lead mentioned as being particularly challenging for newcomers were taking responsibility, working incrementally, being experimental, and reflecting on how to improve. While we do not suggest that newcomers need to develop an agile mindset during onboarding, we identify from this study that they need to start that journey

*Tackling problems*: For many newcomers to this team, this was their first professional job experience in the IT sector. Whilst this meant they were keen to do well and enthusiastic about tackling new technical challenges, it also meant they didn't have previous experience of working on business-critical software in a professional environment or of dealing with managers, stakeholders or administrators within the workplace … *"it's about a perceived mental barrier and how to approach the work. Because they're so new they also don't*

*understand how to tackle problems. It's a case of, well just start, just get started it doesn't matter if you throw it all away" [TL]*. This learning had to start during their first few months in the job.

*Becoming agile*: Project team members, especially newcomers, found it hard to maintain agile processes without prompts from the team lead or the project manager, *"if me or [the TL] aren't in the office, stand-ups don't happen, and so we're really trying to encourage, 'This is your meeting, this is for you to help each other'" [PM].* Some newcomers found it hard to adapt to working incrementally to deliver a vertical slice of functionality in a sprint. Sometimes this was linked to expectations based on previous work or educational experiences. Experiences of success, such as completing sprints on time and understanding the benefits of retrospectives, helped team members towards a mindset change.

### 5.2.2 Workplace Adjustments

Workplace adjustments describe ways in which the organisation and insiders need to adjust to support the onboarding of newcomers into agile teams. These are different from individual adjustments in that they are not adaptations the newcomer makes but are changes the workplace needs to make so it is better set up to cope with integrating newcomers. This is a completely new category that extends Bauer's theory (see Figure 1 and Table 6) containing three sub-categories: *team composition*, *team communication* and *communities of practice*.

#### 5.2.2.1 Team composition

Agile teams are designed to be close-knit, well-balanced units (Whitworth and Biddle, 2007). When a newcomer arrives, the team is changed because its composition is no longer the same. As a result, the whole team needs to adjust to accommodate the new team configuration. In this case study, the team had a regular throughput of newcomers who often worked for a couple of years and then left. In their early history the team found it difficult to develop a strong ethos because of high staff turnover. Over time the team composition changed, so that at any one time they had a stable core of (usually full-time) insiders and a smaller number of newcomers. This change enabled established members to develop their skills so they could take on mentoring roles.

*Adjusting the team*: Over time the team evolved to consist of more established members and fewer newcomers. This balance improved their ability to continuously improve *"We've been through a lot of iterations of how we approach our work, and I think we're hitting a sweet spot of getting things done, with having more full-time members" [SD1].*

*Mentor availability*: Newcomers needed guiding through their first few days in the team. This role was best filled by the team lead or project manager, but they were not always available and didn't have control of when other meetings or events would take them away from the team. Additionally, there was never an overlap when a new team member could start before their predecessor left. *"How do you embed that knowledge into people when they onboard? And it's not just part-time members, it's full-time members, when they first come into the team. The problem with the University recruitment process is that someone leaves before you can fill their shoes to a certain extent." [TL].* This made a hand-over period impossible. It was therefore essential that experienced team members were available to mentor newcomers because the team lead was often away.

#### 5.2.2.2 Team communication

Agile teams aim to be well-integrated and have effective ways of communicating within the team (Sharp and Robinson,2010). When a newcomer joins an agile team, experienced team members may temporarily need to change the way they communicate to explain to the newcomer aspects of the work that are normally taken for granted.

*Accommodating part-timers*: A recurrent theme among the team was connecting with, and sharing knowledge with, the part-time newcomers. Both established members and part-time newcomers saw this as an issue, *"…for part-time members or [those] who can't attend, and it's probably trying to find ways of bridging the gap in the communications that occur. So, it's kind of every time we've had a retro everyone has said, communications need to improve. It's like you've said it, but you're not actually doing it." [TL].* A part-time newcomer commented on the difficulty of finding out what had happened in the project after an absence, *"because they'll just talk to each other and just figure something out, and then you won't find it documented anywhere, or it won't even be in the [TFS]" [NC3].*

#### 5.2.2.3 Communities of Practice

However effective an agile team is, it is also part of the wider organisation. If the rest of the organisation is not familiar with agile working practices, members of an agile team do not feel they are part of a wider community. Communities of practice are "groups of people who share a concern or a passion for something they do and learn how to do it better as they interact regularly" (Wenger, 2011) . There was only one other agile team in the IT function, so there was little opportunity for newcomers to learn from others, to get institutional agile training, or to feel part of a wider community of agile practitioners. There was no time or budget for staff to fulfil those needs outside the organisation. A shared community is what the team was missing, and an adjustment was needed to either find or build a community of practice. This observation confirms findings from previous research that doing agile in a non-agile environment is a challenge (Gregory *et al.,* 2016), and having a critical mass of staff who understand agility is essential to make agile work (Kuusinen *et al.,* 2016 p.3).

*Agile community of practice*: The team were one of only two agile teams in the University's ISS department. The other team were always extremely busy and didn't use Scrum so there was almost no communication with them. There was therefore no wider 'agile' environment that new team members could learn from or feel part of … *"it's kind of a desert. It would be great if there were more practitioners that you could share experiences with.*" [TL].

## 6 Discussion: answering the research questions

This study explored the onboarding of newcomers into a co-located agile software development project team. This is of interest to practitioners who want to sustain their agile teams over the long term. We asked two research questions: *How do newcomers integrate into an ongoing agile project team?* and *How do agile practices particularly assist with onboarding?*

*Table 7: Onboarding practices found this study – general and agile-related – organised by Bauer's functions (2010)*

| Onboarding Activities | General onboarding practices | Agile-related onboarding practices |
|---|---|---|
| **Recruiting** | *Follow legal recruitment requirements*<br>*Consider long-term recruitment* | *Evaluate agile knowledge and give resources* |
| **Orientation** | *Provide new staff pack*<br>*Provide working with client pack*<br>*Socialise with newcomers* | *Provide agile method pack* |
| **Support tools and processes** | *Introduce all communication tools* | *Display information radiator* |
| **Coaching and support** | *Provide mentoring*<br>*Provide role-modelling*<br>*Encourage learning*<br>*Exercise empathy* | *Take part in agile ceremonies – explain prior to use*<br>*Encourage teamwork*<br>*Use pair programming*<br>*Take part in daily stand-ups*<br>*Provide support through co-location* |
| **Training** | *Offer formal courses* | *Immersion into the agile team from day one*<br>*Self-study about agility* |
| **Feedback tools** | *Offer one-to-ones with senior staff*<br>*Provide immediate feedback during immersion*<br>*Offer small tasks* | *Participate in informal code reviews*<br>*Provide feedback through testing and TDD*<br>*Participate in sprint retrospectives*<br>*Participate in sprint reviews*<br>*Participate in sprint refinements* |

First, we address the research question, *how do newcomers integrate into an ongoing agile project team?* Within the case study we found a rich and complex set of processes, practices, and adjustments were used to help newcomers integrate into the team. By mapping our findings to Bauer's model (2010) we showed that these practices and approaches covered all six onboarding sub-categories. In Table 7 we highlight how the team used a combination of *general* and *agile-related* onboarding practices. The *general* practices, such as providing a mentor, might be used in any job role or organisation. However, we found that some aspects of agile practice helped even when the team used general onboarding practices. For example, it was always possible for the team to give newcomers small tasks because they always had a set of reasonably small tasks on the backlog. The *agile-related* practices were context-specific and fall into two groups. Some were general practices that had simply been adapted to help in the agile context, such as providing an agile method pack. Others were agile-specific practices with characteristics that helped onboarding, such as pair programming and sprint retrospectives (discussed in answer to the second research question below). The use of agile-related practices for onboarding was important as the team wanted to sustain their agile approach and could only do so if newcomers readily adapted to it.

In our adapted model (Figure 1) we found two adjustment categories aided newcomer integration, newcomer adjustments and workplace adjustments (Table 6). Our case study findings provide evidence to support Bauer's contention that newcomers integrate better if

they experience self-efficacy, role clarity, social integration, and knowledge of culture. In addition, we found that mindset change, team composition, team communication, and communities of practice were important adjustments. There is a resonance between some of our adjustments and challenges that have been identified in the research literature about adopting the agile mindset (van Manen and van Vliet, 2014), communication and developing people in agile teams (Stray, Viktoria Gulliksen, Moe and Dingsøyr, 2011; Conboy *et al.,* 2011), and building community of practice in agile workplaces (Kahkonen, 2004). We discuss challenges of onboarding in this case study in our previous paper (Gregory *et al.,* 2020).

Second, we address the research question, *how do agile practices particularly assist with onboarding?* We found the team used agile practices to support onboarding in two ways. First, they used immersion and introduced newcomers to agile practices from day one. This meant that newcomers started productive work immediately and were introduced to agile practices and ceremonies as they encountered them during the sprint. For example, by attending daily stand-ups newcomers learnt the purpose and the structure of those meetings, and by doing pair programming, they learnt that agile practice. After two weeks newcomers had been through a whole sprint cycle and were familiar with several agile practices and the major parts of the team's process. Second, the team deliberately used some agile practices as onboarding mechanisms when they had new team members. For example, they used test-driven development with newcomers during their first few weeks in the team because it was an effective way to learn about the team's coding and testing approach. Other agile practices that were useful during onboarding were informal code reviews (to learn aspects of the code base, see different coding practices and discuss quality), sprint retrospectives (to develop awareness of continuous improvement and the agile mindset), and sprint refinements (to learn about the project and practise empowered decision-making). Agile practices therefore had a double purpose during onboarding: they were practices that newcomers had to learn because they were used by the team, but also the practices themselves were useful teaching mechanisms. For example, the team used pair programming during onboarding so that newcomers learnt how it worked. But they also used it because it was a very effective way to transfer knowledge about the product requirements, the technology being used, coding styles and conventions, and the people in the team. This finding is backed up by Plonka *et al*. (2015) who found that pair programming in expert-novice pairs is a form of knowledge transfer that benefits both novices and experts. We found a similarity between the onboarding process observed in this case study and the concept of situated learning, in which learning takes place in the same context in which it is applied, and the concept of legitimate peripheral participation, in which newcomers become community members by taking part in low-risk, productive tasks (Lave and Wenger, 1991).

Our findings resonate with those of Buchan *et al.* (2019), Sharma and Stol (2020), and Britto *et al.* (2018). Our list of onboarding practices is similar to that of Buchan *et al.* (2019). Additional practices in our work are immersion, information radiators, co-location, one-to-ones, immediate feedback, role modelling and empathy. Practices in Buchan *et al.* that we did not find are using online communities, formal training courses, a local knowledge database, attending conferences, a location map, and checklists. One key result from Sharma and Stol (2020) was that support was important for onboarding success, and our findings indicate this agile team used many practices for coaching and support. Compared to Britto *et al.* (2018) we did not find there was a need for formal training in the code base. Nobody in our case mentioned difficulties with learning the code base or how it was conveyed to them. This could have been because the team were in the early stages of a new product when we interviewed them. But it is also likely that formal training was not required because newcomers became familiar with the code base through code reviews and

pairing, and they could always ask questions because they were pairing and co-located. Our study adds to previous literature as it is based on an empirical, contextual study of onboarding practices in an agile team and provides more nuanced understanding than prior studies on the onboarding challenges and practices of an agile team. In addition, we extend previous findings by identifying some additional practices that support onboarding in agile teams, including the need to encourage empowerment in newcomers to an agile team. Also, we identify new agile-related onboarding adjustments indicating that organisations must adjust as well as individuals. These findings are related the commonly recognised issue that it is difficult for agile departments to successfully operate within non-agile organisations (Gregory *et al.,* 2016).

We provide three recommendations for agile practitioners synthesised from our findings:

1. incorporate the agile-related practices shown in Table 7 that support onboarding,
2. focus on training, explaining, and modelling empowerment when onboarding staff, and
3. regularly reflect and consider the workplace and newcomer adjustments where necessary to support onboarding.

For theory, our major contribution is a new model of onboarding in agile teams that is an adaptation and extension of Bauer's onboarding model (2010) to which we contribute new concepts for the agile context. These are useful findings as they indicate areas of difficulty that agile teams need to pay attention to. This is an example of case study as theory elaboration (Ketokivi and Choi, 2014) as we have expanded an existing model to explain onboarding in agile teams.

## 6.1 Limitations

To discuss the limitations of this case study we consider study validity according to Robson and McCartan (2016)'s discussion about the trustworthiness of qualitative research: *threats to validity in flexible designs*, *bias and rigour* and *generalizability.*

### *6.1.1 Threats to validity in flexible designs*

Three types of understanding may lead to threats in flexible research: description, interpretation, and theory.

The *description* of this case was based on a significant amount of data, including individual narratives from each interviewee and observations. All this data was used to identify the original codes (Gregory *et al.,* 2020), and the method used is described in detail. The main threat regarding *interpretation* relates to imposing a framework or meaning on the data we collected. Our initial analysis was inductive, focusing on the perspective of the participants. We employed member checking to ensure validity by providing reports to the project team that summarised our findings and asking for confirmation and feedback. Regarding the use of *theory*, although the Bauer model is used to frame the findings, and as a basis for identifying new adjustments, it did not guide the initial analysis. Other theories relating to onboarding could be used in this way.

### *6.1.2 Bias and rigour*

The study took place over three months, with ten site visits to the team under study (*prolonged involvement*). This supports a deep understanding of the case but avoids the danger of familiarity bias. We collected insights from a range of sources including interviews and observation; participants with a variety of longevity in the team, e.g. very new staff, staff with one year of experience, and long-established staff were involved. Hence, some *data triangulation* was achieved. Further *triangulation* was achieved through a) collecting

interview and observational data through which we corroborated certain practices and behaviours, and b) discussion of data and analyses between authors.

This research method relies on researcher expertise and intuition, which may result in bias. However, *peer debriefing* with members of the author team, and *member checking*, as described above, were used to counter this form of bias. The reliability and rigour of this case is addressed via a careful *audit trail* of activities, data collection and analyses.

### *6.1.3 Generalisability*

*Internal generalisability* in this case may be threatened because only a selection of the team agreed to be interviewed, so some perceptions are missing. However, the sampling intention was to interview the whole team. While some data triangulation between interviews and observations was achieved, this was limited to three visits. The *external generalisability* of our findings is limited by the particularities of this single case, which has a complex staffing model and a distinct university context in which most newcomers are novices to agile working and professional software development. However, we would argue that it is quite common to have complex team compositions, and that although our study setting was distinct it is not unique in employing new graduates. Our single case study does not aim for generalisability but aims to provide detailed insights into a phenomenon (Eisenhardt and Graebner, 2007; Yin, 2018). We used those insights to elaborate an existing theory to account for the agile software development context, with each new sub-category being well-supported with literature from prior studies. Further, Leung (2015) suggests that a pragmatic approach to generalisability in qualitative case studies includes using systematic sampling, triangulation, and theory, all of which we have attempted. We see this work as an initial step towards supporting onboarding in agile teams and encourage others to challenge and extend our work to date.

## 7 Conclusion

In this paper, we claim that onboarding newcomers to co-located agile software development projects might differ from onboarding in general. We found general onboarding practices are used in agile project teams and that certain agile practices taught using immersive learning also support onboarding. We also identified individual and workplace adjustments that need to be made while onboarding to an agile project team.

This paper makes four contributions, 1) it provides in-depth insights into onboarding in an established co-located agile project team, 2) it illustrates how a combination of general practices, agile-related practices, and technical agile practices are used to onboard newcomers in an agile team, 3) it makes recommendations for agile practitioners synthesised from our findings, and 4) it provides an extended model for agile onboarding based on Bauer's (2010) onboarding model.

In future we recommend research to develop a comprehensive onboarding model that fully elaborates the factors in onboarding in an agile team. That research should encompass onboarding in all agile environments, co-located, distributed, and large-scale. A more detailed exploration of the specific needs of different types of newcomers such as full-time/part-time, permanent/temporary, and experienced/inexperienced. In the light of the Covid-19 pandemic, onboarding in virtual agile teams should also be investigated.

## Acknowledgements

We acknowledge the contributions of the research participants, Dr Raid AlQaisi who helped with data collection, and the co-funding provided by the Agile Business Consortium (ABC Ltd.) for the Agile Research Network (agileresearchnetwork.org).

## References


Andriyani, Y., Hoda, R. and Amor, R. (2017) *Reflection in agile retrospectives.* Springer, Cham, pp. 3.

Babb, J., Hoda, R. and Nørbjerg, J. (2014) 'Embedding reflection and learning into agile software development', *IEEE Software,* 31(4), pp. 51-57.

Bandura, A. (1982) 'Self-efficacy mechanism in human agency.', *American psychologist,* 37(2), pp. 122.

Barroca, L., Gregory, P., Kuusinen, K., Sharp, H. and AlQaisi, R. (2018) 'Sustaining agile beyond adoption'*In 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 29-31 Aug. 2018* Prague, Czech Republic: IEEE, pp. 22-25.

Bauer, T.N. (2010) *Onboarding new employees: Maximizing success.* VA, USA: The Society for Human Resource Management Foundation (SHRM).

Bauer, T.N. and Erdogan, B. (2011) 'Organizational socialization: The effective onboarding of new employees', in S. Zedeck (ed.) *APA handbook of industrial and organizational psychology, Maintaining, expanding, and contracting the organization* Washington, DC, US: American Psychological Association, pp. 51-64.

Beck, K. (2000) *Extreme programming explained: embrace change.* addison-wesley professional.

Britto, R., Cruzes, D.S., Smite, D. and Sablis, A. (2018) 'Onboarding software developers and teams in three globally distributed legacy projects: A multi-case study', *Journal of Software: Evolution and Process,* 30(4), pp. e1921.

Britto, R., Smite, D., Damm, L. and Börstler, J. (2020) 'Evaluating and strategizing the onboarding of software developers in large-scale globally distributed projects', *Journal of Systems and Software,* 169, pp. 110699. doi: https://doi.org/10.1016/j.jss.2020.110699.

Buchan, J., MacDonell, S.G. and Yang, J. (2019) 'Effective team onboarding in agile software development: Techniques and goals'*ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* IEEE, pp. 1-11.

Calefato, F., Giove, A., Lanubile, F. and Losavio, M. (2020) *A case study on tool support for collaboration in agile development.* pp. 11.

Cockburn, A. (2006) *Agile software development: the cooperative game.* Pearson Education.

Cohn, M. (2004) *User stories applied: For agile software development.* Addison-Wesley Professional.

Conboy, K., Coyle, S., Wang, X. and Pikkarainen, M. (2011) 'People over Process: Key Challenges in Agile Development', *IEEE Software,* 28(4), pp. 48.

da Camara, R., Marinho, M., Sampaio, S. and Cadete, S. (2020) 'How do Agile Software Startups deal with uncertainties by Covid-19 pandemic?', *arXiv preprint arXiv:2006.13715,* .

Dagenais, B., Ossher, H., Bellamy, R.K., Robillard, M.P. and De Vries, J.P. (2010) 'Moving into a New Software Project Landscape '*Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, Vol 1* IEEE, pp. 275-284.

Davis, V.D. and Kleiner, B.H. (2001) 'How to orient employees into new positions successfully', *Management research news, .*

Dingsøyr, T. and Lindsjørn, Y. (2013) *Team performance in agile development teams: Findings from 18 focus groups.* Springer, pp. 46.

Eisenhardt, K.M. and Graebner, M.E. (2007) 'Theory building from cases: Opportunities and challenges', *Academy of management journal,* 50(1), pp. 25-32.

Fagerholm, F., Guinea, A.S., Borenstein, J. and Münch, J. (2014) 'Onboarding in open source projects', *IEEE Software,* 31(6), pp. 54-61.

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

Gandomani, T.J. and Nafchi, M.Z. (2015) 'An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach', *Journal of Systems and Software,* 107, pp. 204-219.

Gregory, P., Barroca, L., Sharp, H., Deshpande, A. and Taylor, K. (2016) 'The challenges that challenge: Engaging with agile practitioners' concerns', *Information and Software Technology,* 77, pp. 92-104.

Gregory, P., Strode, D.E., AlQaisi, R., Sharp, H. and Barroca, L. (2020) 'Onboarding: How Newcomers Integrate into an Agile Project Team'*Agile Processes in Software Engineering and Extreme Programming* Lecture Notes in Business Information Processing, vol 383. Springer, pp. 20-36.

Handke, L., Costa, P.L., Klonek, F.E., O'Neill, T.A. and Parker, S.K. (2020) 'Team perceived virtuality: an emergent state perspective', *European Journal of Work and Organizational Psychology,* , pp. 1-15. doi: https://doi.org/10.1080/1359432X.2020.1806921.

Hoda, R. and Murugesan, L.K. (2016) 'Multi-level agile project management challenges: A self-organizing team perspective', *Journal of Systems and Software,* 117, pp. 245-257.

Kahkonen, T. (2004) *Agile methods for large organizations-building communities of practice.* IEEE, pp. 2.

Ketokivi, M. and Choi, T. (2014) 'Renaissance of case research as a scientific method', *Journal of Operations Management,* 32(5), pp. 232-240.

Klein, H.J. and Polin, B. (2012) 'Are organizations on board with best practices onboarding', *The Oxford handbook of organizational socialization,* 54, pp. 267-287.

Kuusinen, K., Gregory, P., Sharp, H. and Barroca, L. (2016) *Strategies for doing agile in a non-agile environment.* pp. 1.

Lave, J. and Wenger, E. (1991) *Situated learning: Legitimate peripheral participation.* Cambridge university press.

Leung, L. (2015) 'Validity, reliability, and generalizability in qualitative research', *Journal of family medicine and primary care,* 4(3), pp. 324.

Miler, J. and Gaida, P. (2019) *On the agile mindset of an effective team–an industrial opinion survey.* IEEE, pp. 841.

Mordi, A. and Schoop, M. (2020) 'Making IT tangible-Creating a Definition of Agile mindset.'*Proceedings of the 28th European Conference on Information Systems*.

Pavlina, K. (2020) *Assessing best practices for the virtual onboarding of new hires in the technology industry.* Pepperdine University.

Plonka, L., Sharp, H., Van der Linden, J. and Dittrich, Y. (2015) 'Knowledge transfer in pair programming: An in-depth analysis', *International journal of human-computer studies,* 73, pp. 66-78.

Salas, E., Sims, D.E. and Burke, C.S. (2005) 'Is there a “big five” in teamwork?', *Small group research,* 36(5), pp. 555-599.

Saldaña, J. (2015) *The coding manual for qualitative researchers.* 3rd edn. LA: Sage Publications London.

Schwaber, K. and Sutherland, J. (2020) *The Scrum Guide.* Available at: https://scrumguides.org/ (Accessed: .

Sharma, G.G. and Stol, K. (2020) 'Exploring onboarding success, organizational fit, and turnover intention of software professionals', *Journal of Systems and Software,* 159, pp. 1-16.

Sharp, H. and Robinson, H. (2010) 'Three ‘C’s of agile practice: collaboration, co-ordination and communication', in Dingsøyr, T., Dybå, T. and Moe, N.B. (eds.) *Agile software development* Springer, pp. 61-85.

Sharp, H., Robinson, H. and Petre, M. (2009) 'The role of physical artefacts in agile software development: Two complementary perspectives', *Interacting with Computers,* 21(1-2), pp. 108-116.

Stavru, S. (2014) 'A critical examination of recent industrial surveys on agile method usage', *Journal of Systems and Software,* 94, pp. 87-97.

Steinmacher, I. and Gerosa, M.A. (2014) 'How to support newcomers onboarding to open source software projects'*In IFIP International Conference on Open Source Systems,* Springer, pp. 199-201.

Stray, V.G., Moe, N.B. and Dingsøyr, T. (2011) *Challenges to teamwork: a multiple case study of two agile teams.* Springer, pp. 146.

Stray, V., Moe, N.B. and Bergersen, G.R. (2017) *Are daily stand-up meetings valuable? A survey of developers in software teams.* Springer, pp. 274.

Timmermans, S. and Tavory, I. (2012) 'Theory construction in qualitative research: From grounded theory to abductive analysis', *Sociological Theory,* 30(3), pp. 167-186.

Van Maanen, J.E. and Schein, E.H. (1977) *Toward a theory of organizational socialization.* MIT Alfred P. Sloan School of Management. Available at: https://dspace.mit.edu/bitstream/handle/1721.1/1934/?sequence=1 (Accessed: Sept 2020).

van Manen, H. and van Vliet, H. (2014) *Organization-wide agile expansion requires an organization-wide agile mindset.* Springer, pp. 48.

Version One (2019) *13th annual state of agile report.* Version One Inc. Available at: https://stateofagile.com/ (Accessed: Sept 2020).

Wenger, E. (2011) *Communities of practice: A brief introduction.* National Science Foundation (US). Available at: http://hdl.handle.net/1794/11736 (Accessed: .

Whitworth, E. and Biddle, R. (2007) *The Social Nature of Agile Teams.* pp. 26.

Yates, R., Power, N. and Buckley, J. (2020) 'Characterizing the transfer of program comprehension in onboarding: an information-push perspective', *Empirical Software Engineering,* 25(1), pp. 940-995.

Yin, R.K. (2018) *Case study research and applications: Designs and Methods.* 6th edn. Thousand Oaks, CA: Sage Publications Ltd.

**APPENDIX – Interview Process & Guide**

- **Pre-receipt of Information Sheet,** aim of study to explore different perspectives of how individuals adapt to agile working practices when joining an agile team
- **Sign Consent Form**
- **Introduction:**
    - How long have you worked here?
    - How many years of work experience before that?
    - What is your role in the team?
- **Learning about agile when you started and over time:**
    - What was your experience with agile when you joined this team?
    - Did you have any training or induction when you first joined the team? What type of training?
    - Did you get any mentoring?
    - How did you learn about this team's approach to agile?
    - Do you think you still need/require training? Could you please explain?
    - Do you practice pairing? Could you please explain? (agile games/mobbing)
- **How do you know you're doing what is expected?**
    - How do you know that you are doing what is expected?
    - If you weren't sure what to do, what would you do?
- **Understanding of team culture (or way of work?):**
    - How would you describe the team ethos?
    - If you don't understand something, what would you do?
    - Do you think there are good levels of trust in the team?
- **Ways of engaging with the team:**

    - Could you describe how you interact with other team members while you're working?
    - How do you know what others are doing in the team?
    - Do team members give each other feedback? What about? Could you please describe that?
    - Do you do social activities together as a team? For example, do you celebrate success together?
- **Pulling together**
    - If a new team member joined your team, in your opinion, what is the best way for them to learn about how your team works?
    - Based on your experience, what techniques or approaches would improve learning and adaptation to agile when new members join a team?
    - How well do you think agile approaches work for the team? Could you please explain?
    - Is there anything else you would like to add that you think is interesting in this context, but not covered by the questions asked?