e-Informatica Software Engineering Journal
Volume 20, Issue 1, pages: 1–43
DOI: 10.37190/e-Inf260110
BibTeX

Onboarding Newcomers in Large-Scale Agile Projects: A Multiple Case Study

Pavithra Herath*, Muhammad Ovais Ahmad, Tomas Gustavsson
*Corresponding author: pavithra.herath@kau.se

Article info

Keywords:
onboarding
agile
large-scale agile
social debt
process debt
people debt
knowledge debt
organizational debt

Submitted: 31 Mar. 2026
Revised: 12 Jun. 2026
Accepted: 28 Jun. 2026
Available online: 2 Jul. 2026

Abstract

Context: Large-scale agile (LSA) is inherently characterized by socio-technical complexities (e.g., system dependencies, extensive cross-team coordination, and distributed organizational structures). Onboarding of newcomers is a critical challenge in LSA environments and it remains underexplored compared to small-scale agile settings.
Objective: Our study aims to investigate the onboarding processes and challenges within LSA projects.
Method: In this exploratory qualitative study, we conducted 41 semi-structured interviews across four Swedish software companies.
Results: We identified onboarding issues that result in negative outcomes, such as cognitive overload, psychological safety constraints, and cross-team collaboration hurdles. These factors create steep learning curves and knowledge deficiencies that contribute to long-term socio-technical debt.
Conclusions: We provide actionable recommendations for practitioners to improve their onboarding processes. In particular, we recommend designing competency-based rather than time-based onboarding plans, expanding social integration through structured cross-team exposure, and addressing the gap between textbook agile and real-world LSA practices through explicit expectation setting.

1. Introduction

Agile methodologies were originally designed for small, co-located teams, but their growing significance in practice led to their application in large-scale organizations which requires significant adaptations [1–5]. Software companies are increasingly adopting various Large-Scale Agile (LSA) frameworks (e.g. Scaled Agile Framework (SAFe) and Large-Scale Scrum (LeSS)) to manage projects involving dozens of people across multiple, interconnected teams [1]. In LSA, onboarding is a challenging task because new hires must integrate technically and socially into a network of teams, shared resources, and all required company meetings.

Successful onboarding establishes better relationships, increases satisfaction, clarifies expectations, and provides support to reduce unwanted turnover [6]. A few studies explore onboarding in agile teams [7,8], but are lacking in the LSA context. LSA projects experience unique challenges related to task complexity, skills alignment, and cross-team coordination. However, successful onboarding demands learning, confidence building, early mentoring, early task involvement, training and socilalization [7, 9, 10]. This highlights the need for

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

a flexible and effective onboarding process in LSA projects. To bridge this research gap, our study aims to explore how LSA organizations manage the integration of new members. Our study aims to answer the following research questions:

RQ1: How do LSA projects onboard newcomers?

RQ2: How do onboarding challenges contribute to socio-technical debt in LSA projects?

This paper extends our earlier work [11]. We use Bauer's onboarding model [6] as a theoretical lens to map case-specific practices across two primary dimensions: (A) Onboarding Functions (Including recruitment, orientation, support tools, coaching, training, and feedback mechanisms), (B) New Employee Adjustments (Focusing on self-efficacy, role clarity, social integration, and cultural knowledge). We also identify specific socio-technical challenges connected to onboarding within LSA context. In this extended version, we provide a more in-depth analysis and comprehensive insights into the topic, and present additional empirical results focusing on onboarding functions and adjustments. We include "technical integration'' as an adjustment dimension in LSA context, extending beyond Bauer's onboarding model. Furthermore, we identify onboarding challenges, present a categorization of these challenges: organizational, individual and contextual and discuss how these challenges influence socio-technical debt in LSA projects.

The remainder of paper is organized as follows: Section 2 provides background to LSA and onboarding. Section 3 describes our research methodology, including research context, data collection and analysis procedures. Section 4 presents the results, followed by a discussion in Section 5. Section 6 offers recommendations, and Section 7 concludes the paper.

2. Background

In this section, we provide an overview of LSA and an introduction to onboarding. We present a detailed description of Bauer's model, explaining the onboarding functions and new employee adjustments.

2.1. Large-scale agile

Agile methodologies were originally designed for small co-located teams [1], but their success has led to their adoption in LSA contexts [1,2,12]. LSA projects are highly complex, and this complexity tends to increase with the project size [13]. Larger scope and high level of complexity introduce significant management and project integration challenges. LSA projects touch on more parts of the organization, posing a risk when something goes wrong. Missing business focus, technical complexity, unaligned teams, and lack of skills stand out among the causes of most project failures [1,14].

LSA comprises multi-team collaboration, typically involving 2–9 teams and 50 or more personnel who handle complex interdependencies [2]. However, LSA frameworks have "struggled to deal with the exponentially greater complexities and interdependencies of large-scale, organization-wide development" [1]. These complex inter-team interactions and organizational structures produce challenges that traditional methods struggle to address [1,2]. Furthermore, focusing excessively on adapting framework elements (such as SAFe roles and artifacts) often distracts the organization from its core business needs. Scaling agile to non-development units presents a persistent challenge across both the SAFe and LeSS frameworks. LSA organizations must cope with concurrent changes to team

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

structures, processes and tools [1]. At this level, standard knowledge-sharing approaches (e.g., co-located retrospectives) tend to break down at scale [2]. Consequently, LSA involves contingencies on how and when specific practices are applicable, which change with the project size, business domain, and team configurations [2].

Scaling out beyond traditional agile development requires extensive social and technical integration. The degree of complexity in inter-team interactions, development processes and organizational structures makes LSA projects highly susceptible to socio-technical debt. While technical debt (TD) occurs due to technical compromises in development that bring short-term benefits but create long-term system instabilities [15], non-technical debt (NTD) encompasses people, social, process and organizational debt [15–17] which arises from complex social interactions that subsequently accumulate TD. One specific source of NTD is "newbie free-riding" [15, 18]. When newcomers do not receive adequate support, guidance, or information, they may struggle to contribute effectively, placing an additional burden on team members and generating NTD in the form of knowledge gaps, social issues, and organizational inefficiencies.

2.2. Onboarding

"*Onboarding refers to the process that helps new employees learn the knowledge, skills, and behaviours they need to succeed in their new organizations*" [19]. Onboarding is also known as organizational socialization [6]. Onboarding helps "new hires adjust to social and performance aspects of their new jobs quickly and smoothly" [6, 19]. Several onboarding models are discussed in the literature, including Maanen and Schein model [20], Jones' model [21] and Bauer's model [6].

Maanen and Schein [20] define socialization as the "process by which an individual acquires the social knowledge and skills necessary to assume an organizational role " within their organizations. This ranges from self-guided trial and error to formal education and apprenticeship [20]. Maanen and Schein offered a theoretical explanation regarding role orientation in onboarding, showing six dimensions: (1) Collective vs. individual, (2) Formal vs. informal, (3) Sequential vs. random, (4) Fixed vs. variable, (5) Serial vs. disjunctive, and (6) Investiture vs. divestiture. Among these, three of which are particularly relevant to LSA:

- Formal vs. Informal: Formal onboarding isolates newcomers within structured programs (e.g., internships and apprenticeships), whereas informal onboarding has no clearly defined process to guide newcomers, and efforts are not made to distinguish them from experienced team members [7, 20].
- Collective vs. Individual: Collective onboarding occurs when a group of newcomers is recruited and faces a common set of experiences together. Individual onboarding occurs when newcomers face experiences separated from a group of newcomers [7, 20]. Each newcomer's onboarding journey is different from others, shaped by a "more or less unique set of experiences". Typical examples of individual onboarding include Apprenticeship programs, intern or trainee assignments and on-the-job training, where newcomers are expected to learn the knowledge and skills on their own accord.
- Fixed vs. Variable: Fixed onboarding occurs when there is a fixed timetable to complete set of activities. The newcomers are given precise knowledge of the time on how long the onboarding or any transition process will take. Various career paths within the same organization may have different time tables. In contrast, variable onboarding does not include any fixed timetable upon completion [7, 20].

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Building on this, Jones model [21] categorizes onboarding as either institutionalized (structured orientation and mentoring) or individualized (learning norms and expectations "on the fly") [22]. Empirical studies have further characterized these experiences. Ju et al. [9] conducted a case study at Microsoft to understand and characterize the experiences and expectations of developers during onboarding in relation to the tasks they performed. Their findings identified three core themes in developer onboarding: learning, confidence building, and socialization [9]. Regarding technical integration, Yates et al. [23] conducted a study on programme comprehension during onboarding, based on in-situ onboarding sessions across eight organizations. They focused on the type of information that experts share with newcomers during onboarding and how valuable newcomers find that information. Their findings reveal that experts use four different views to explain the code: abstraction of the structure, algorithmic, temporal and rationale-based views. Among these, the rational-based view, which includes both function-based rationale and design-decision rationales, was particularly valued by newcomers [23]. Similarly, Fagerholm et al. [24] study onboarding in open source software projects on the role of mentors and project characteristics. Their results show that developers who receive deliberate onboarding support through mentoring become active at an earlier stage than developers who enter the project through conventional methods. In addition, project size and life time are significant factors that influence onboarding [24].

In the context of distributed agile teams Moe et al. [22] found that effectiveness of onboarding is affected by domain complexity, team type, and the availability of existing members. Additionally, it is important to note that "onboarding outcomes cannot always be predicted by management" [22]. Peggy et al. [7] extended Bauer's model for agile environments. Their findings indicate that while mentorship practices are universally helpful, both adapted general practices and unique agile techniques like pair programming are crucial for successfully integrating newcomers in teams [7]. To operationalize these concepts, Buchan et al. [8] identified eleven key goals for agile onboarding: (1) Understanding company culture, (2) Understanding team norms, (3) Knowing the responsibilities, expertise and authority of other team members, (4) Understanding what others expect from the role and meeting your responsibilities, (5) Understanding what tasks are assigned and when to do them, (6) Understanding the project structure, aims and implications, (7) Understanding how to code and test to meet team expectations, (8) Understand the teams' standards and quality, (9) Understand and gain the agile mindset, (10) Know how to use agile artifacts and techniques uses by team in software development process, (11) Understand the project domain knowledge and terminologies. However, achieving these goals in large-scale settings is challenging. Britto [10] investigated onboarding of software developers in LSA projects from Ericsson perspective. Their results show that onboarding performance was hampered by distant mentors, mismatched training, large early tasks, and team instability. Onboarding in such projects are requiring extensive advance planning and long-term mentoring by experienced members, which is very expensive [10].

Steinmacher et al. [25]found that newcomers in open source software projects encounter numerous barriers that hinder their initial contribution and lead to dropping out. They identified 58 barriers, which were grouped into six categories: cultural differences, newcomers' characteristics, reception issues, orientation, technical hurdles, and documentation problems [25]. Stol et al. [26] examined onboarding across settings on a hybrid work continuum to understand how onboarding shapes newcomers' sense of belonging. Their findings indicate that belonging develops through parallel interpersonal and intrapersonal adjustment processes, while the work setting acts as a critical factor and intentionality moderates

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

the extent to which organizational efforts are translated into employee adjustment [26]. Šmite et al. [27] examined resignation patterns among software professionals from Ericsson perspective between 2016 and 2025, focusing on how the transition from onsite work to fully remote and hybrid models influenced employee attrition. Their findings show that remote onboarding fails to foster organizational attachment and a sense of belonging necessary for long-term retention. Selective return-to-office practices for new hires and their teams are recommended for building social cohesion and reducing early attrition [27].

2.2.1. Bauer's onboarding model

According to Bauer [6] onboarding model, organizations must first determine whether they are best served by formal or informal onboarding. In informal onboarding, newcomers learn their new jobs without an explicit organizational plan. In contrast, formal onboarding process, newcomers receive assistance in adjusting to their new jobs both task-wise and socially. Additionally, organization provides a written set of policies and procedures to support them. Bauer's onboarding model tactics align with institutionalized and individualized onboarding, respectively. Newcomers who undergo formal onboarding with well-planned programs, including awareness on their roles and company norms, are more effective than employees who experience informal onboarding. Bauer identifies four building blocks of successful onboarding, known as the "Four C's": compliance, clarification, culture and connection. Onboarding has short-term outcomes associated with new employee adjustments, which organizations can use to assist newcomers in maximising their onboarding success. Short-term outcomes relate to four adjustments:

- *Self-efficacy* – The self-confidence in performing the job roles.
- *Role clarity* – Determines how well the employees understand their job roles and the expectations.
- *Social integration* – Starting to work with organizational insiders. This supports feeling socially comfortable and accepted by their peers and seniors.
- *Knowledge of culture* – Helps the newcomers understand the organizational culture and their roles within it.

Additionally, Bauer [6] presents six interconnected functions designed to facilitate newcomer support:

- *Recruiting process* helps newcomers to fit into the organizational environment and get to know the insiders and stakeholders. Recruiting is unrelated to onboarding plans. It is important to consider it as a step to begin onboarding.
- *Orientation* helps newcomers to understand important aspects of their job and organization, including its culture, values, goals, history and power structures. Orientation helps newcomers to feel welcome and introduces them to their co-workers and other individuals. This can be conducted either in person or online. Orientation includes discussions, lectures, videotapes, written material and online sources.
- *Support tools and processes* include written onboarding plans; a formal document that outlines the specific timeline, goals, responsibilities and support. Stakeholder meetings or "touching base " meetings help newcomers get the information they need on time. This includes onboarding online, and technology is used to follow the onboarding process.
- *Coaching and support* includes senior employees serving as role models. Mentors help newcomers learn about the organization, offer advice, assist with job instruction, and provide support in social and political contexts. Informal colleagues also provide adequate information. Coaching helps newcomers to follow a successful onboarding.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

– *Training* includes hard skills, soft skills and onboarding skills. Training helps newcomers to support their own adjustments.
– *Feedback tools* provide newcomers with feedback and guidance. This is a two-way process during onboarding in which newcomers seek and receive information. Organizations vary in how effectively they use feedback tools.

The models proposed by Van Maanen and Schein [20], Jones [21], and Bauer [6] provide a comprehensive theoretical foundation for understanding how newcomers integrate into organizations. While these models identify what needs to be achieved (e.g., self-efficacy and culture knowledge), there is a gap in understanding how these functions are adapted within LSA context.

3. Method

We employ an exploratory qualitative approach to investigate onboarding within LSA context across four companies in Sweden. Qualitative research is suited for revealing underlying processes within individuals and teams, capturing not only what individuals experience but also how they interpret those experiences [28] [29]. The case study provides a flexible approach for a thorough examination of the complex LSA context [28]. Following an exploratory multiple case study design [28], this research focuses on four real-world projects in LSA context. Using multiple case studies enables broader exploration of research questions and theoretical elaboration. The evidence from multiple cases is often considered more compelling and robust, providing greater confidence in the overall study [30].

3.1. Research context

We conducted our study at Swedish offices of four global software and technology firms, hereafter denoted by these identification codes: *E*, *S*, *T* and *A*. (See Table 2). These cases provide a comparative view of different LSA frameworks, specifically SAFe (Companies E and S) and LeSS (Companies T and A).

3.2. Data collection

Interviews are a highly efficient way to gather rich, empirical data [31], and semi-structured interviews in particular allow direct contact with participants and enable real time data collection [28]. To ensure the depth and diversity of insights, we used purposive sampling [32] to select our interviewees, based on their LSA knowledge and experience. Engaging numerous knowledgeable informants who can view the phenomena from diverse perspectives across functional areas yields meaningful data [31]. We conducted a total of 41 semi-structured interviews: 7 from case E, 8 from case S, 15 from case T, and 11 from case A, representing a diverse range of experience levels and roles (See Table 1).

All interviews were conducted through Zoom, each lasting for 45-60 minutes. Each interview was recorded and stored in a secure repository. In our interviews, the guidelines covered a diversity of questions; the Interviewees' educational and professional background, LSA experience, teamwork practices, onboarding process and training processes. We cross-validated transcripts with the recordings for possible transcription errors. Moreover, we provided the interviewees with finalized transcripts to facilitate member checking.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Table 1. Interviewees background

Case E			**Case S**		
ID	Role	Professional experience	ID	Role	Professional experience
EP1	Integration specialist	1 year	SP1	Tester	7 years
EP2	Project manager	2 yeras	SP2	Developer	20 years
EP3	Scrum master	7 years	SP3	Tester	12 years
EP4	Developer	20 years	SP4	Project manager	5 years
EP5	Business owner	10 years	SP5	Designer	5 years
EP6	Scrum master	5 years	SP6	Designer	5 years
EP7	Analyst	3 years	SP7	Designer	2 years
			SP8	Designer	8 years
Case T			**Case A**		
ID	Role	Professional experience	ID	Role	Professional experience
TP1	Product owner	10 years	AP1	Developer	7 years
TP2	Security master	3 years	AP2	Team lead	25 years
TP3	Developer	1 year	AP3	Developer	23 years
TP4	Developer	10 years	AP4	Developer	4.5 years
TP5	Developer	20 years	AP5	Test lead	16 years
TP6	Developer	25 years	AP6	Developer	1 year
TP7	Project manager	20 years	AP7	System engineer	15 years
TP8	Product guardian	3 years	AP8	Project manager	14 years
TP9	Security master	1 year	AP9	Developer	8 years
TP10	Architect	16 years	AP10	Developer	12 years
TP11	Developer	1 year	AP11	Developer	21 years
TP12	Senior Software Engineer	7 years			
TP13	Developer	10 years			
TP14	Developer	12 years			
TP15	Tester	2 years			

Table 2. Case company descriptions

Case company	**Description**
E	Medium-sized software and technology company comprising cross-functional agile teams practising SAFe, specialized in Enterprise Resource Planning (ERP) and digitalization.
S	Large institution providing financial and insurance solutions, comprising cross-functional agile teams practising SAFe.
T	Large Nordic software and service provider that comprises cross-functional, self-organized agile teams developing a test tool for telecommunication operators practising LeSS.
A	Medium-sized software and services company consisting of X and Y project teams. Team X is developing a car rental software, whereas team Y is working on a retail chain with web management modules. Comprised of cross-functional agile teams practising LeSS.

3.3. Data analysis

We used thematic analysis [33], because it is recognized as a valid technique for a transparent and systematic procedure for producing codes and themes from qualitative data. Thematic

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Table 3. Phases of thematic analysis

Phase	Description of the process
1. Familiarising ourselves with data	Repeatedly reading the data to search for meanings and patterns. Highlighting key information and taking the initial notes.
2. Generating initial codes	Coding interesting features of the data systematically across the entire dataset. Match code with data extracts that demonstrate the code and collated together within each code.
3. Searching for themes	Refocusing the analysis at the broader level of themes. Collating different codes into potential themes and collating relevant data extracts within the identified themes.
4. Reviewing themes	Checking if the themes align with the coded data extracts (Level 1) and the entire data set (Level 2). Generating thematic mapping of the data analysis.
5. Defining and naming themes	Defining and further refining the themes that will be presented for analysis and analysing the data within themes. Identifying the essence of what each theme is about, and what aspects of data each theme captures. Identifying the story that each theme tells, and considering how it fits into the broader overall story that we are telling about our data.
6. Producing the report	Providing sufficient evidence of the themes within the data. e.g., using enough data extracts to demonstrate the prevalence of a theme. Choosing vivid data extracts as examples to capture the essence of the point being discussed. Relating the analysis back to the research question and literature.

analysis [33] follows a six-phase process (i.e., familiarising ourselves with data, generating initial codes, searching for themes, reviewing themes, defining and naming themes and producing the report) to systematically identify, analyze, and interpret patterns in data, moving between raw interview transcripts and coded extracts to prepare a description of the onboarding phenomenon (See Table 3).

Thematic analysis identifies and interprets the key features of the data guided by research questions and broader theoretical assumptions. The significance of thematic analysis lies in its flexibility to summarise the key features of large amounts of data, providing a "thick description" of the dataset [33]. Thematic analysis involves deep and prolonged immersion in the data, requiring researchers to engage in thoughtful and reflective analysis. [34]. It helps to identify patterns within and across data relating to participants' lived experience, views, perspectives, behaviour and practices, to understand what they think, feel and do. [33]. Analysis involved a "recursive" process that constantly moves back and forth between the entire dataset, the code extracts of the data being analysed, and the analysis of the data produced [33]. Moreover, Braun and Clarke [33] propose fifteen criteria to ensure good qualitative analysis, highlighting the processes of transcription, coding, analysis, overall completion, and written report. We used NVivo (Version 14) software to systematically identify the patterns grounded within the data. We mapped generated codes to the onboarding functions and employee adjustments in Bauer's model (see Table 4).

4. Results

This section presents key findings related to onboarding in LSA context based on our qualitative analysis. This directly answers our research questions which are presented in Section 1. We mapped case companies practices with the lens of Bauer's onboarding model [6], containing six onboarding functions (i.e., Recruiting process, Orientation, Support tools and processes, Coaching and support, Training, and Feedback tools) and new employee adjustments (i.e., Self-efficacy, Role clarity, Social integration, and Knowledge of culture). Building on this model, we introduce "technical integration" as an adjustment during

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Table 4. Themes, sub-themes and codes

Theme	Sub-themes	Codes
Onboarding functions	Recruiting process	Onboarding in recruitment process
	Orientation	Introductory meetings, System overview presentation, Team-led grooming sessions
	Support tools and processes	Onboarding template/checklist, System documentation, Contact support, Standup meetings, Tools accessibility, Provisioning accounts, Environmental setup
	Coaching and support	Dedicated mentors, Scrum master, Product owner support, Informal peer support, Innovation sprint, Question-asking culture
	Training	Learning materials, Online courses, Task-based, Case-based and Competency-oriented learning, Pair-programming, Code reviewing
	Feedback tools	Continuous feedback, Retrospectives
Adjustments	Self-efficacy	Adaptation to work
	Role clarity	Role expectations
	Social integration	Blending with teams, Ad hoc-meetings, Morning-talks, One-on-one rotating conversations, Planning poker, Team-building meetings
	Knowledge of culture	Agile training, Team norms
	Technical integration	Technical meetings, Tech Fika/Coffee time

LSA onboarding. We mapped the presence of these onboarding functions with "Y" (Yes) indicating their presence in each case company (See Table 5) and adjustments in Table 6.

4.1. Functions

This section describes the onboarding functions of case companies, organized into six categories. Together, these functions explain the factors involved in the onboarding of newcomers in LSA projects.

4.1.1. Recruiting process

Recruiting process is the stage where newcomers receive their first impression and create expectations for their specific role. A well-planned onboarding process creates a motivating and captivating environment for newcomers. *" We have a structured onboarding process that is led by our recruitment person. Depending on when a new employee starts, they will have a tailored introduction plan"* (EP5). The recruitment process needs to be consistent and make adequate efforts to get the right person for the role, because new recruits in an established organization fill the gap left by an existing employee in a specific area of expertise. Therefore, a precise balance of technical expertise and personality must be considered during the onboarding process. *"There will be a gap that needs to be filled. I can either say I need someone with much experience or it's okay with someone directly from the university"* (TP1).

The careful consideration of a person's attitudes and behaviours is emphasised as a driving force to be considered, given their effect on team dynamics when working across multiple teams. *"You need to have the basic skills to join the team, but then personality is everything."* and *"They sit on the first meeting, speaking about techniques [...] and then make the first assessment. Is this a person that can fit into the team? Then the team meets*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Table 5. Onboarding functions in four case companies

Onboarding functions	**E**	**S**	**T**	**A**
Recruiting process				
Onboarding in recruitment	Y	Y	Y	
Orientation				
Introductory meetings	Y			
System overview presentation				Y
Team-led grooming sessions		Y		
Support tools and processes				
Onboarding template/checklist			Y	
System documentation			Y	Y
Contact support				Y
Standup meetings	Y		Y	Y
Tools accessibility		Y	Y	Y
Provisioning accounts				Y
Environmental setup			Y	Y
Coaching and support				
Dedicated mentors	Y	Y	Y	
Scrum master support				Y
Product owner support				Y
Informal peer support	Y	Y	Y	Y
Innovation sprint		Y		
Question-asking culture			Y	
Training				
Learning materials	Y		Y	
Online courses			Y	
Task-based learning		Y	Y	Y
Case-based learning				Y
Pair-programming		Y	Y	Y
Competency-oriented learning			Y	
Code-reviewing	Y		Y	Y
Feedback tools				
Continuous feedback	Y			
Retrospectives			Y	

Note: "Y" indicates the presence of these functions in each case company. Each company represented with a code name: E, S, T and A.

this person. I [role in team] only ask about a kind of personal question to check what kind of person this is?" (SP1). Personality tests are used as one of the evaluation criteria in hiring processes. *"We have a set of questions and we have different profiles and during those interviews, to ensure that we check the personality"* (SP2).

This indicates that the recruitment process is linked with structured planning, even before onboarding activities begin. The recruitment process does not end just after hiring newcomers, but continues until newcomers successfully integrate into their work environment both technically and socially.

4.1.2. Orientation

Orientation is an essential step in integrating newcomers into their designated roles and tasks. Case E has a comprehensive 100 days plan which includes two weeks focus on introductory meetings to help newcomers adjustment in their roles. "*Our onboarding*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

process is quite rigorous. We have a lot of meetings for new employees to make sure they feel welcome" (EP5).

Whereas case A has a system overview presentation, which is technical. Newcomers get high-level information on the system and its integrations, which are expected in their LSA projects. "*I take a brief overview presentation about the system, how things are connected and everything.*" and *"We give them some overview knowledge about the systems"* (AP11).

Team-led grooming sessions are another strategy employed to facilitate newcomer learning. This provides the opportunity for the newcomers to engage with the team and understand their technical expertise and business values. This is seen as a team's collective effort in helping newcomers to integrate into their workflow. Case S has an onboarding process between three and six months. "*I think it's an important session for a new person to be together with the team in the grooming sessions*"(SP2).

4.1.3. Support tools and processes

Support tools and processes help newcomers gradually integrate into their tasks. Case T creates an onboarding template in Jira, including all the steps required for a formal introduction. This serves as a checklist for newcomers and includes information regarding tools, different accesses, training videos, e-learning and security rules. This onboarding story is cloned every time a newcomer joins. "*They have like an onboarding story in Jira, which they have gathered, like all steps on the introduction*" (TP2) and "*We have a template for an onboarding story. So it's a newcomer, I clone that story*" (TP1). Case T adopts a more intentional approach to integrating newcomers, improving their onboarding experience and making the process smoother and more effective for all involved. *" We realized that it would be really good to have a ready program that you can put in front of a new person"* (TP6).

In cases T and A, newcomers receive distinct system documentation for reference. This guides newcomers through the ways of working, provides a list of things to download, and initiates learning procedure. "*We have multiple documents and pages to guide them through the environment setup*" (TP12) and "*Here is the documentation page where you can read up on all the documentation*" (AP6). In case A, the information is documented on dynamic pages in Confluence, listing the information of team members whom newcomers can contact for support. Newcomers can contact this person to address their system-related issues. This is a dedicated role assigned with adequate time to guide newcomers on technical inquiries. *"This is the dedicated person, and they should have time to actually answer any questions you have regarding the system"* (AF11). The approach reflects active learning through real work, with adequate support available as needed. *"Here are people that you can ask for help"* (AP3). Moreover, this establishes a more practical approach than long, overwhelming lectures. The documentation emphasizes ways of working, including essential processes and practices, without providing extensive details. *"We are pretty good at documenting how things work, how you set up your environment, roadmap, and onboarding processes"* and *"I think we are trying to be as pragmatic as possible and, try to have a very condensed page in the Confluence specifying ways of working"* (AP2).

In cases E and A, daily standups are used for synchronisation and transparency, helping newcomers understand team's status and challenges. "*The best way to learn the system is through a big project to work with, but it is not always that easy. So we make it work by helping each other through standups*" (EP7) and "*Straight away, we have let them into these standup meetings. So they can get some understanding of how things would work*" (AP4). In case T, teams participate newcomers in standup meetings from the beginning, helping

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

themselves immerse in the team's routine work while providing a regular checkpoint to monitor their onboarding progress. standup meetings function as a supportive mechanism, allowing teams to identify and address any obstacles that newcomers encounter. *"We are checking how it is going with their onboarding, as sometimes they have problems like they don't get any access, get stuck somewhere and have code issues"* (TP5). standup meetings also offer a cooperative platform to discuss ongoing work, gather feedback, check on the progress of other teams, and understand and resolve technical discrepancies in greater detail. This is especially helpful for newcomers as they acclimate to their responsibilities. "*They discuss what they are doing and are now in this step, and they want feedback from another team*" (TP15). standups provide an interactive platform for distributed teams to connect and meet co-workers. This is a significant opportunity for new employees to integrate into the team. *" [...] Especially for new employees, it's a great way to get help and support in the beginning"* (EP6).

Newcomers should gain access to the required tools, configure their accounts and set up their working environment as soon as they get to know the team. Different cases have their own unique ways of integrating tools. In cases S and T, newcomers are first provided access to the tools, followed by an introduction to the areas of work or the domain. "*We have a program helping him first to have the necessary tools to be able to start up, then we would present the area or doing"* (SP2) and "*There are lots of tasks in them for setting up the environment and doing the first well, sandbox environments and so on"* (TP8). In case A, newcomers are initially provided with an overview of their system; their devices, accounts and environment are configured, and they are guided to start their learning. "*We will set up the computers and environments and all the accounts and access rights that they need, or at least put in orders for it*" (AP10) and "*We give them some overview knowledge about the systems and then we try to get their environments running up and they can start to work"* (AP11). Case A has implemented substantial technical improvements to its onboarding process, eliminating bottlenecks that delayed newcomers' ability to begin their work. *"We automated most of our installation processes and things that previously took one or two weeks just to get set up with the environment"* (AP9).

4.1.4. Coaching and support

Coaching and support is an essential element of successful onboarding. This ensures that newcomers are systematically integrated into their roles and teams. Each case assigns dedicated mentors to their newcomers. These mentors assist newcomers in understanding team routines, addressing emerging questions, and gradually helping them adapt to their work environment. *"Their [mentor] responsibility is just to support, and make sure that the new person has access to the team"* (EP1) and *"They are someone you can go to with any questions"* (EP6). Mentors invite newcomers to regular meetups, supporting them in assessing their progress as they settle into their roles. This practice continues until newcomers can confidently manage their responsibilities. "*I got a mentor when I started, and we had meetings once every two weeks at first. Now we don't have those meetings any more. We had them like the first six months*" (EP2).

When a new employee joins, both cases T and S ensure that they receive a dedicated mentor as the first point of contact for any clarifications. Assigning a mentor helps new employees feel supported and monitored as they transition into their roles. Mentorship not only assists them in their work but also helps build their confidence over time, which is a significant benefit of mentoring. "*Each new person gets a mentor in the team and*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

they [newcomers] can come and ask questions, like the first person" (TP2) and *"Mentors are supposed to help the newcomer to be a little bit more concerned"* (TP6). Mentors collaborate with newcomers on all the cases they involve. Regular assistance and substantial attention enhance newcomer's personality and foster collaboration with peers. *"They are working on all the cases that he [newcomer] is doing"* (SP3) and *"The mentor is mostly like a buddy at your new work, someone who makes you feel welcome"* (EP6). The frequency of mentoring meetings is tailored to each individual's progression within the team and system integration. Mentoring fosters familiarity and engagement for newcomers within the team. *"The frequency of these mentor meetings is different for each person"* (EP4).

Case A employs a product owner and scrum master to provide coaching and support. In LSA settings, it is equally important to maintain both technical and business perspectives. *"We [teams] prolonged a meeting, and he [product owner] will go through his view of the system."* and *"He [scrum master] spends some time getting them on board and getting them up to speed, and introducing things, the applications to them*" (AP4).

In all cases E, T, S and A teams willingly support their newcomers in need. Team members collaborate closely, providing informal support to help newcomers overcome adaptation challenges, even in the presence of a formal mentor. "*We all try to help new employees in team and involve them in projects as well. The mentor doesn't have to learn everything for new employee, but we all try to help with the learning process*" (EP6) and "*You [newcomer] have a mentor in the team that is supposed to help newcomer to be a little bit more concerned. But we try to help each other in team*"(TP6). Teams provide ongoing support and assistance to help newcomers assume more responsibilities and integrate successfully. *"New member work together with someone else in the team, taking on a work item for himself with the support from the other person in the team"* (SP2) and *"They [newcomers] can do some parts of their own in some parts together with some of the older guys in the team, but we try to help all each other in the team"* (TP6).

Team members voluntarily stepped in to assist in the absence of a mentor. They support each other not because they are assigned to do so, but because the team values mutual support. The success of onboarding relies on an informal culture of helpfulness, with team members proactively filling gaps whenever needed. *"The newest guy, he got one mentor, and he was away. So I just took over because he needs help. I will help"* (TP2) and *"We were four new people in the group that came simultaneously. We worked on more minor bug cases, and had much help from the team"* (AP6). Team members also make an extra effort to physically meet and help new employees early on. Experienced team members adjust their physical presence to support newcomers despite their remote work preferences. Face-to-face interactions are recognized as important for building social connections. *"Often they are newly employed, and tend to be at the office more in the beginning, and then we [experienced team members] tend to be at the office more in the beginning as well, to be there to physically meet the person [newcomer] and to sit down with the person. But for me, we can choose from three variants"* (AP3).

Case S teams operate on a structured timeline known as "product increment". This is a twelve-week cycle, with the final two weeks dedicated to an innovation sprint. During this period, team members, including newcomers, are given several (three to five) days to work on their technical interests. This time is intentionally structured and time-bound. Newcomers who regularly participate in structured onboarding tasks gain a sense of freedom, which allows them to deepen their understanding, explore broadly and enhance their confidence. Even though newcomers might not expect to be involved in these activities early, team continues the practice because they believe it is important for learning and innovation.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

"They [newcomers] ask: Well, what should I do?. You can do what you want! They will have three, four, five days that they can do whatever they want and it works good" (SP1).

Case T teams encourage a question-asking culture for newcomers. Team dedicates a focused hour to addressing newcomers' questions, thereby reducing newcomers' anxiety, and minimizing potential distractions for experienced team members. *"If you have any questions, just write them down and we can have an hour when we talk about it."* (TP2) and *"They can ask questions or have a session with questions with someone in the team or the whole team."* (TP5). The structured question sessions serve a dual purpose. Beyond answering newcomers' questions, these sessions help maintain a constructive discussion and mutual understanding between newcomers and team regarding the tasks being performed. This reflects an active learning behaviour, as newcomers gradually grasp the context before delving into more complex works. *"Trying to start with minor commits and asking questions. I'm trying to get the idea. What is we really doing here? What am I doing? Really? What is this? Where? What is possible to do and so on? "* (TP15). Asking questions presents a fundamental team value. This reflects a conscious effort to build psychological safety by encouraging newcomers to feel comfortable, avoiding hesitations and embarrassment. *"We [team] always say that to the newcomers. It's always okay to ask questions. No question is too silly to be asked, and we always try to say yes"* (TP6). Team members openly acknowledge that even experienced team members can make mistakes, emphasizing the natural consequences of complex work, thereby encouraging transparency and continuous improvement. *"It is a complex system. So we do mistakes as well"* (TP6).

4.1.5. Training

Training plays an important role in the integration of newcomers into software and technology companies. Formal training (learning materials and online courses) provides a structured, systematic approach to integrate into workflows, while informal training (case-based learning) provides practical, case-based learning.

Cases E and T offer a package of learning materials to newcomers. This consists of recorded knowledge-sharing sessions and courses that newcomers may need to repeat several times. "*We have started to build up with some learning material and have a lot of knowledge sharing sessions that we record*" (EP6) and "*There are some recorded courses that we are able to look at, as part of the introduction*" (TP3).

Case T offers online courses to newcomers to acquire technical skills and knowledge. This indicates a more standard learning approach, which is consistent. *"There are a number of e-learning training courses that they [newcomers] need to participate in during the first weeks"* (TP1) and *"The onboarding process is very focused on setup and going through different courses online"* (TP15).

In Cases S, T, and A, newcomers integrate into the team through task-based learning. In case S, newcomers work together with another team member and take responsibility for a task item while receiving guidance and support. "*[...], taking on a work item for himself with the support of the other person"* (SP2). In cases T and A, newcomers start with narrow tasks and progressively familiarise themselves with the workflow. "*They [team] assigned me some simple tasks and provided on-the-job training to get familiar with how we work in this team*"(TP13). Starting with manageable tasks helps newcomers build their basic competencies before they progress to more complex development work. *"The team members created some tasks for me to build up my basic competence to work in this team"* (TP13).

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Focusing on a narrow task streamlines development, facilitating a smooth technical transition. This further reduces the cognitive load of newcomers by limiting their initial scope of understanding and preventing them from becoming overwhelmed by the system's complexity. *"We [team] assign a task to the person that may be quite narrow tasks, so the person doesn't need to know everything, but just a part of the system"* (AP1). Case A views onboarding as a continuous process that gradually integrates into the domain and development environment. *"It's an ongoing process after that, starting with more manageable tasks"* (AP2). Newcomers are assigned to explore and implement a new tool that integrates learning with practical contribution, enabling them to engage early in projects. *"One person very recently joined. He's just stepped out of college and he's working on those scopes. We [team] kind of wanted to try that tool within the application"* (AP4).

Teams do not expect newcomers to have comprehensive system knowledge; however, they do expect responsible software development accompanied by an adequate contextual understanding. *"It's not mandatory for everyone to know everything. But they should at least know what they're doing and what they are affecting by their changes"* (AP2). Team progressively build newcomers' understanding through "small steps", providing sufficient context for each task. *"When we're specifying the issues or tasks to be done, I try to fill in a little bit more about how it works in small steps"* (AP2).

Given the importance of understanding internal and external dependencies and architecture perspectives in LSA, case A emphasises a case-based learning approach. Such tasks are given as an assignment, enabling newcomers to learn and expand their domain knowledge while moving forward in the workflow. "*There is a need to explain the dependencies between that application and other components, internally or externally, on the architecture*" and "*We start from there, a narrow part of the system, an assignment, a task*" (AP3). Team strategically assign newcomers to a safe learning environment. *"We have identified one product that is low-hanging fruit. So to say it's not used that much. We've identified that and slightly introduced this person to the code and the actual code base, and how all the systems are kind of interconnected"* (AP10). The term "Low-hanging fruit" refers to the team's effort to create opportunities for newcomers to learn and experiment in a low-stakes environment. This approach allows them to gradually become familiar with system architecture. By offering learning opportunities that have a real impact, newcomers can enhance their competence and confidence before moving into more complex systems.

Developing competency-oriented ramp-up program to serve and adjust the competencies of every newcomer is a beneficial approach. *"Some people need a good foundation and ramp-up very fast, while others can't do long studying and need to be integrated into the development cycle faster"* (TP12). Case T implements competency-building sessions, demonstrating a formal commitment to continuous professional development. Newcomers are being supported through this initiative to address their competency gaps. *"We [team] can spend one hour on anything, any competence that we want to build up or learn something"* (TP15) and *"I think everyone is lacking competence in general. But if you compare me with other people, I lack competence because I'm very new"* (TP14). By facilitating more opportunities, case T connects newcomers with employees who have recently completed their onboarding tasks. This allows newcomers to receive guidance and support based on the experiences of those who have recently gone through the onboarding process. *"What we are trying to do is to direct them to someone that has recently done it [onboarding]"* (TP5). Experienced employees are willing to assist newcomers with tasks that align with their specific experience. *"Everyone has a different background. Maybe the newcomer is more*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

experienced than me. So, if I encountered a problem during onboarding that the newcomer did not face, then I would suggest that they refer to my onboarding process" (TP13).

Pair programming is a structured practice in agile, but learning emerges naturally through collaboration and communication. In case S, T, and A newcomers paired with co-workers. Newcomers observe the works of experienced colleagues, collaborate with them and receive guidance until they are capable of working independently. *"First sitting together with another one with the pair programming"* (SP2), "*We sit down with this new person a lot at the beginning and pair programming* (AP3)", *"The new person will do pair programming, shadowing with someone who is developing until they are ready to develop independently*" (TP12) and "*Always have someone they can code with, the newest member of the team, he was paired with him [senior] for a little while*" (AP9).

Case A identifies code reviewing as a dual-purpose mechanism intended to enhance newcomers' learning while promoting knowledge sharing within the team to support high-quality code development. *"We need to do more coding reviews because the other two people in the project are quite new. So for them, it's more important to have coding reviews"* (AP1). Instead of assigning code review to newcomers during their onboarding tasks, case A ensures that newcomers receive reviews from the entire team. This approach helps newcomers improve their coding skills and understand the standards and practices effectively. *"You [newcomer] won't be code reviewing for a while. So many people should code review your code"* (AP6). Similarly, case E views code reviewing as a useful approach and is designed to be implemented, given the benefits of learning from each other and providing a platform to review newcomers' code more concisely. *"It might be a good idea to start doing code reviewing. When we have a new employee, I can review his code in the beginning and learn from each other"* (EP6).

Moreover, case T employs a structured code review process during development, prior to deployments. Multiple reviews are encouraged, allowing developers to invite their team to review their code. *"I was developing this code with my colleague and was reading through this code many times, and then we say to the team, you should also review it"* (TP3). Team identifies code reviewing as a strategic approach to enhance analytical skills and improve understanding of each other's code. *"Code reviewing will help us learn how other people approach a problem and how we approach a problem "* (TP15). Even though code review seems challenging for newcomers, case T encourages them to actively participate in order to improve code quality. *"I find the code review process to be a challenge, but it's necessary for maintaining code quality "* (TP15). However, feedback from experienced developers assists newcomers in producing clearer and more readable code. *" Some more experienced people can give very good comments about the code. For example, our new team member S has less experience, and his code needs more improvement. We will leave comments about how to write the code in a better and more readable way, and he will fix the comments"* (TP13).

4.1.6. Feedback tools

Feedback tools determine whether continuous assistance is required. From the very beginning, newcomers engage in regular meetings with their mentors. These meetings function as a feedback loop, ensuring both learning and adaptation. *"We had them in the first six months, and then we both felt the need. Like, don't need them more, and I would contact her if I have any questions"* (EP2). Teams often collaboratively integrate newcomers, being approachable and open to requests for clarifications and guidance. *"We try to help each*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

other in the team, so they [newcomers] can just reach out to anyone if they have some questions" (EP6).

Case T uses retrospectives as a platform for sharing collective feedback at the team level. Retrospectives facilitate collective reflection and process improvements while providing a psychologically safe environment for newcomers to ask questions and suggest improvements within teams. *"We get feedback during our scrum meetings, we have the retrospective where we give feedback, but that's more on a team level"* (TP11) and *"But we do have team members who have their work shared or at least tested by some other teams. In that case, they do write the feedback in the retrospectives"* (TP15).

4.2. Adjustments

This section describes the employee adjustments in case companies, organized into five categories, along with "technical integration" as an adjustment in LSA. Together, these adjustments describe how case companies support newcomers in maximizing their onboarding success. We mapped the presence of these onboarding adjustments with "Y" (Yes) indicating their presence in each case company (See Table 6)

Table 6. Onboarding adjustments in four case companies

Onboarding adjustments	**E**	**S**	**T**	**A**
Self-efficacy				
Adaptation to work	Y		Y	
Role clarity				
Role expectations		Y		Y
Social integration				
Blending with teams			Y	Y
Ad hoc-meetings			Y	
Morning-talks				Y
One-one-one rotating conversations			Y	
Planning poker				Y
Team-building meetings	Y		Y	Y
Knowledge of culture				
Agile training	Y	Y	Y	Y
Team norms		Y		
Technical integration				
Technical meetings			Y	Y
Tech Fika/ Coffee time			Y	

Note: "Y" indicates the presence of these adjustments in each case company. Each company represented with a code name: E, S, T and A.

4.2.1. Self-efficacy

Self-efficacy focuses on newcomers' confidence to learn and adapt to the team workflow. Successful onboarding creates a supportive and adaptable work environment, fostering competence. *"The department manager and I still have talks every two weeks to discuss all my development and my plan forward*" (EP3) and "*When I started working in this team, it was relatively smooth. It took about one or two months to get familiar with development environment and team*" (TP13). Moreover, case T Team helps its members set goals. Every

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

member is provided with an allocated time to focus on skill development, improve their expertise, and pursue their interests. *"You can focus your competence development on this [allocated time], and use that for going in that direction, or if it's something that you think is interesting, then you can go in that direction"* (TP11).

4.2.2. Role clarity

Role clarity assesses newcomer's role, centered around their knowledge, skills, workflow integration and teamwork spirit. *"We get a new team member, and I can see how good teamwork is flowing and the level of skills the person has*" (AP1). This highlights the team's expectations and desires about the newcomer's performance and their role within the team. Additionally, Case s team expects new employees to value cultural expectations as much as technical expertise. Roles are defined by how newcomers should engage and contribute. Newcomers are expected to pursue their defined responsibilities with appropriate attitudes and behaviours. *"You have to be able to work in this type of team, you need to be curious, and want to develop yourself, want to take on a new assignment, and be interested both in the business and in what the team do with you"* (SP2).

4.2.3. Social integration

Social integration incorporates newcomers into teams, facilitating collaborative participation. Blending newcomers with teams brings higher engagement and motivation. *"If you get a lot of new people in, you can not just put a new team with all new people. It is better to spread them out and then form a new constellation*" (TP8) and *"When I wasn't quite familiar with debugging errors when running test cases, my team members helped me"* (TP13). A supportive team culture fosters understanding, respect and patience, enabling newcomers to integrate smoothly. This integration enhances the effectiveness of formal onboarding activities. *"I think it comes down to everyone respecting each other, giving each other additional time, and being patient"* (AP9).

Ad-hoc meetings serve as an excellent opportunity for newcomers to connect. These informal gatherings foster relationship-building and cultivate a sense of belonging, both of which can significantly enhance collaboration among team members. *"Unplanned, informal meeting just when bumping into each other"* (TP1).

Case A values personal connections as much as professional collaborations and establishes a daily routine to reflect this priority. Teams intentionally allocate time for team mingles called morning talks, fostering a psychologically safe environment that helps newcomers overcome the fear of embarrassment when asking questions. *"Every morning, we join Person xyz, and talk about anything that is not work-related"* (AP10) and *"We have morning talk 15 minutes every morning before the standup and you cannot talk work-related stuff there"* (AP3).

Moreover, case A, T and E organize frequent team-building activities to provide regular breaks in ongoing work, fostering a relaxed and exciting environment where employees can enjoy and get to know one another. Team members, along with newcomers, engage in fun activities such as online games, scribble drawing, and identifying geographic locations. These activities have been appreciated by teams for making regular work more effective. *"We also have added team building meetings each week just to get a break in the work and talk about other stuff"* (AP11). Cases T and E involve various physical team-building activities such as team lunches, after-work drinks and team outings like bowling or mini-golf in regular

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

intervals outside of their formal routine. *"We are going to places, eat something, and there are more interactions*" (TP12) and *"We do a lot of fun activities and after work together"* (EP6). In addition to these activities, case T teams also maintain a digital social platform, such as team chat applications, where team members share entertaining discussions. These informal interactions foster a sense of belonging, contributing to a cohesive team culture. *"We also have a team chat where we can socialize and share non-work-related things like funny memes or interesting articles"* (TP9).

Case A teams use planning poker as a collaborative tool, allowing each member to contribute their insights into the task planning process. This method not only promotes a shared understanding among the team members, but also cultivates a sense of collective responsibility. *"We try to always do some activity that engages everyone. So, for estimation meetings, we do Scrum poker"* (AP11).

Case T organizes one-on-one rotating conversations. This in-person event provides participants with the opportunity to engage in discussions with one another. Newcomers are welcomed and encouraged to share their insights with the team, as they will have the opportunity to conduct a five-minute discussion with each attendee. Although the limited time can create stress, this event is still considered a valuable opportunity for newcomers to connect with the team members with whom they have not interacted. *"They got call from each other for 5 minutes and then it was a new person that they should call, and so on [....]"* (TP5).

4.2.4. Knowledge of culture

Knowledge of culture is necessary for shaping newcomers' adaptation into teams and workflow practices. For instance, onboarding should communicate why specific practices are adopted and how newcomers use a specific workflow. Using agile practices helps to improve collaboration, feedback loop and adaptability. Case S conducts flexible SAFe training, in the form of a continuous program called "Leading within SAFe" which lasts for several days or a week. Newcomers joining the teams are encouraged to participate in the SAFe training from the outset. *"All new people who come here come to work, they will have some kind of education [SAFe]"* (SP1) and *"They [newcomers] will get this SAFe training upfront when they join"* (SP2). Case E conducts introduction sessions in Agile, explaining how the team adopt agile way of working. Agile culture fosters a sense of innovation and motivation among employees. *"As a part of the introduction, there are advanced courses in Agile*" (EP1) and *"I love the Agile process because it forces me to think about things I never thought about before"* (EP4).

Moreover, scrum master, in case A team also assists newcomers in conducting agile scrum ceremonies, *"About how the sprint works"* (AP4). Employees have gained experience and learned the agile way of working over time. *"When we started, we had no experience. So, it started with baby steps"* (AP1). Case T conducts agile courses in which all its employees participate. The full-day course explains the standard agile practices and how the team utilizes them. *"You went through how to use and how we are using Scrum"* (TP3). Though it takes time for new employees to adapt to agile, case T appreciate the agile way of working. Implementing agile has enabled the teams to compress their tasks into flexible sprints instead of long-yearly planning. *"I think it's nice to get a better overview of all the work"* (TP11).

Team norms play a significant role in shaping team practices, and newcomers are expected to adhere to certain ethical guidelines and boundaries. These team practices are

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

negotiable and flexible, provided that there is clear communication. This approach helps newcomers develop trust, accountability and respect for team norms, facilitating their successful integration into the teams. *"I have to get them to adopt the rules and accept them because you're not allowed to look into your mobile phone during a meeting. But if they tell the others in the team, then it's okay, but it's not okay if you don't communicate it"* (SP2).

4.2.5. Technical integration

This is an emerging theme that we identified beyond Bauer's model, as the complexity of large-scale systems creates challenges that neither role clarity nor cultural knowledge can adequately address. Technical integration is particularly important in an LSA environment, where merely defining role expectations, adapting to team workflows, and relying on self-directed learning are insufficient to support newcomer integration. Technical integration addresses how newcomers develop a sufficient understanding of interconnected systems that even experienced members do not fully master.

Technical integration involves regular team discussions and continuous updates on system know-how. Case A teams participate in technical meetings, report on and share information on what they implement. These open meetings are conducted regularly to keep all team members updated and provide an overview of their ongoing tasks. *"We can take up things that maybe we have made a new tool that we know you also should do"*(AP1).

In case T experienced team members lead technical discussions, which function as a platform for newcomer knowledge sharing.*"Explain things deeply for our team because we have different backgrounds"* and *"It's a very specific area to get newcomers into team"* (TP5). Moreover, in case T, all team members participate in "cluster meetings" in which they discuss the achievements made in different areas of their sub programmes. The team also hold regular informal domain knowledge sharing sessions known as tech fika or tech coffee time, in which they share information on specific topics. *"It could be that one person reads up on the subject and tells the other about it"* (TP8). Although these sessions proactively produce knowledge, newcomers perceive them as demanding because they require preparation for active participation. *"It pushes me a bit because I also have to read up before and actually have something with me"* (TP11). However, they also recognize this as a motivation for learning. *"That kind of motivates me as well"* (TP11).

4.3. Onboarding challenges

This section presents a systematic classification of onboarding hurdles into organizational, individual and contextual challenges, describing the difficulties encountered during the onboarding of newcomers (See Table 7).

4.3.1. Organizational challenges

Lack of an organized onboarding process: While newcomers anticipate a formal introduction to team members and other relevant stakeholders, a lack of a structured onboarding process with adequate orientation limit their access to people and information, thereby hindering their early engagement. "*I don't think there's a formalized way of introducing people, and it was a lot up to me trying to find information*" (AP5).

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

Table 7. Onboarding challenges

Onboarding challenge	**Sub-challenge**	**Description**
Organizational Challenges	Lack of an organized onboarding process	(1) Lack of a formal introduction; inadequate access to people and information. (2) Learning opportunities depends on availability of newcomers and ongoing projects. (3) Lack of a common understanding on onboarding approaches among team. (4) Sprint time not deliberately protected for onboarding. (5) Lack of an organized knowledge-sharing approach across applications. (6) Lack of a flexible onboarding approach to accommodate individual needs (7) Lack of clarity regarding prioritization and decision-making. (8) Lack of the supervision of a mentor.
	Recruitment difficulties	(1) Hiring skilled personal from diverse backgrounds. (2) Finding recruits with combined (technology and domain) skills. (3) Recruitment not aligned to team requirements
	Team-facing onboarding challenges	(1) Lower availability of experienced team members vs higher availability of newcomers. (2) Team instability due to frequent newcomer onboarding. (3) Assigning inexperienced newcomers to new customer projects. (4) Newcomer training amid align rapid organizational growth. (5) Newcomer's integration pace affect team effectiveness. (6) Team cost effort, time and energy for training newcomers with little experience on projects.
Individual Challenges	Cognitive overload	(1) Sheer volume of information presented in tightly scheduled introductory meetings. (2) Maintaining a working pace parallel to peers. (3) Frequent switching between tasks, projects and meetings.
	Barriers to psychological safety	(1) Newcomers been observed and reported to progress to management. (2) Newcomers' limited ability to self-advocate unlike experienced team members. (3) Newcomers rarely engage in team discussions. (4) Newcomers remain silent: Lack knowledge to ask questions. (5) Newcomers feeling comparatively inexperienced.
	Knowledge gap and retention	(1) Gaps in domain knowledge (2) Difficulties in integrating highly specialized/narrowly focused expertise into teams
Contextual Challenges	System complexities	(1) Domain complexity. (2) Insufficient system understanding due to limited exposure to components and processes. (3) Numerous application and unclear naming conventions. (4) Difficulties in acquiring input from experiences team members. (5) Multiple team involved in development.
	Remote onboarding	(1) Difficulties to integrate with cross-team collaboration and planning. (2) Gaps in availability and visibility among team members.
	Cultural sensitivity	(1) Adaptation to different working styles. (2) Adaptation to different organizational structures; hierarchical vs flat. (3) Adaptation to different decision-making styles.
	Agile-related issues	(1) Gap between textbook agile values and actual agile practices. (2) Transparency induce performance expectations.

Case E considers it is valuable for newcomers to be included in the projects from the very beginning, as this provides them with a gradual learning process and an opportunity to learn through observation. However, this approach can be challenging, as learning the system through a larger project is not always straightforward. *"It's all about timing. The best way to learn the system is through a big project to work with, but it's not always that easy"* (EP7). A lack of adequate projects limits newcomers' opportunities to learn and contribute. *"Sometimes we don't have so many projects going on. So, I think it's harder for them when they start to like, join and learn and be a part of something"* (EP1). This indicates a lack of an organized onboarding procedures, where newcomers' learning depends on being available at the right time and having access to projects. A lack of such organized tasks creates tense situations, causing newcomers to feel stressed. *"We don't have a specific task for them to work on right away, and that can be stressful for some"* (EP3). Moreover, in the lack of a common understanding of onboarding approaches across teams, each developer follows their own personal onboarding approach. *"Everyone has their own sort of structure"* (AP11). Although a default structure theoretically existed, it was not consistently applied, and employees loosely followed it. *"Tries to follow the default one, and just that it was sort of cluttered"* (AP11).

In Case A, team contributes to setting up the environment early and allocating sprint time for onboarding, thereby avoiding overloading sprints. However, they lacked a structured onboarding process where time was not deliberately protected for the onboarding process, despite their sprint commitments demanding time. *"We haven't really had a good plan, and we didn't have time to actually create an onboarding process"* (AP10). Although the teams had good intentions, they lacked a proper strategy. Teams acknowledge their need for change to accomplish a dedicated onboarding approach. Moreover, providing adequate knowledge of the most frequently used applications enables individuals to become familiar and adapt effectively. However, the lack of an organized knowledge-sharing approach prevents the intended knowledge from being effectively conveyed. *"I think there should be a better process for not giving them knowledge about every application because I have myself felt that I haven't gotten that knowledge from the beginning"* (AP11).

Although newcomers are given time to learn the system, their understanding remains limited. They lack clarity regarding prioritizations and transparency in decision-making, indicating a need for a more organized onboarding approach. *"I have only been six months at S, but for me, it's not clear how everything is being prioritized when it comes to planning"* (SP8). Environment setup remains an unresolved challenge and is acknowledged as an overhead which requires improvements. Individuals learn and integrate differently, with differences in their onboarding needs. The onboarding process should be flexible enough to accommodate these differences concerning newcomers' individual needs. *"Flexible onboarding planning, the environment setup is an overhead that could be optimized, but it takes time. Some people need a good foundation and ramp up very fast"* (TP12).

Moreover, working without the supervision of a mentor is challenging for newcomers. *"He has been working by himself"* (SP3). While being involved in meetings is beneficial for team cohesion, a lack of clear direction and motivation hinders effective newcomer integration. *"He has a tough journey"* (SP3). Self-directed onboarding processes place excessive responsibility on newcomers. When integration into the system is treated as the individual's responsibility, with limited structured support, it may result in stress and reduced confidence. Getting to know and work with multiple teams requires timely learning. *"Responsibility sometimes. Ask a colleague, how can I join you on this? Can I join you on*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

this and just be a part of it?" (EP1) and *"It's not just one person they are working with, but need to know those teams, and they are rotating"* (SP3).

Recruitment difficulties: Recruiting skilled personnel with required competence is a significant challenge for software companies. Most of skilled personnel are hired from a diverse culture background. This brings a constraint from business-related processes, rather than technical skills. *"You can't recruit good people. It's hard. It's too long from X. It's too long from Y. It's hard to find the right competence, and they have struggled with recruiting people for ideas and for my product"* (AP8). Moreover, a noticeable knowledge gap exists, as candidates have computing backgrounds but lack domain expertise, making it difficult to find recruits with combined skills. *"I have noticed that it's very difficult to find such persons who have both [domain knowledge and programming], these two skills"* (TP4) and *"It is not programming, and it is the business requirements, and then we have a scale-out issue instead"* (AP8).

Attrition over the past few years further indicates that recruiting and retaining the required competence has been a persistent challenge. Whereas, replacing a "head" does not immediately replace the knowledge, contextual understanding and team commitments. *"We hired new people, but you can't replace heads, but you can replace experience in a certain project"* (AP3). Additionally, attrition of long-standing employees makes it more difficult to retain critical competencies in the short time.*"I would say our largest problem is that we lose people and replace them with newcomers"* (TP6). and *"For a couple of years, our main challenge has been knowledge about the systems because the architecture is so vast and, at the same time getting new persons up to speed on the project"* (AP3).

Recruitment should be aligned with the team's requirements. Focusing solely on hiring to achieve organizational goals can create challenges if teams do not acquire the appropriate competence and experience required to support their work. *"They are always trying to get new people from the right, directly from the university, but it's always the people that have studied, like computer engineering and stuff, and maybe that's not enough"* (TP3).

Team-facing onboarding challenges: Team faces a challenge regarding the availability of experienced team members. Although there are many new team members, attrition causes low availability of experienced team members. This type of structure reduces their time for questioning, and the team requests expertise in order to better establish learning opportunities. *"We have quite many young people in my team or new people [..] Have a little more time to ask questions or something more personal to take list problems with"* (EP1).

Similarly, team struggles with frequent onboarding of newcomers, which disrupts team stability, making it difficult to proceed. *"It's only been half a year now, like in a couple of months. I would say that the project is suffering from too much changes in people and in teams setups"* (TP5). In addition, case T encounters challenges in allocating team members to new customer projects, because of the limited experience of the newly hired employees. *"It is not possible to put these inexperienced people right into the program of a new customer. So I will lose some of my best people"* (TP1). Newcomers' pace of integration into the system affects the team's effectiveness in pursuing novel solutions. Team relies on the knowledge of experienced team members drawn from previous projects. *"It's always like a newcomer coming instead. That is, of course, something that at least for a while, slows down your team"* (TP6).

Rapid organizational growth brings onboarding challenges. Senior consultants in case E are responsible for training newcomers in their roles while still facilitating rapid organizational demands. This type of situation creates stressful conditions, even for experienced

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

employees. *"This puts a lot of responsibility on the senior consultants to train the new ones and also do their job"* (EP3). Moreover, when a new employee with little experience of the functionalities and the product joins a team, supporting and training the newcomer requires additional resources from the existing team. *"The existing team needs to take care of that new person and train him or her, which will cost effort and time and energy"* (TP1).

4.3.2. Individual challenges

Cognitive overload: The sheer volume of information presented in tightly scheduled introductory meetings creates tense situations for newcomers, resulting in mental fatigue and minimal reflection. "*We had two and a half days full of onboarding classes or introduction meetings where we got to learn everything. It was very stressful, a very tight schedule*" (EP2), *"I thought maybe it was quite hard because we have a specific list of things that we can do the first couple of weeks or several weeks"* (TP3) and "*But it's pretty much information overflow*" (AP2). Newcomers anticipate a smoother transition from onboarding training to their actual job tasks. *"From those tasks to doing the real job or some real task, maybe I should have a smaller step in between that and then"* (TP3).

For newcomers, it is challenging at first to maintain a working pace comparable to their peers. Over time, newcomers are introduced into a network of teams and artifacts that they must continuously engage with to accomplish their work. In a LSA project, this naturally develops work pressure for newcomers. *"I feel more pressure, because everybody in our team is working very hard. I try to catch up as a new team member, and there are a lot of things need to learn. So maybe have some pressure"* (TP13) and *"It's challenging for me because I had my own tasks to finish, and I think it's a priority"* (TP14). Ensuring successful onboarding is a challenge in LSA because it involves multiple teams, scaling of agile practices, and integrated software development. "*Onboarding is a big issue that we have to work on and maintain a nice spirit even as the company grows bigger*" (EP3).

As newcomers enter working life, they gradually learn to manage tasks and navigate through system complexities. However, frequent switching between different tasks, projects or meetings is mentally demanding and challenging for newcomers to manage. *"I've only been in the work life for one year, I'm getting used to a lot of things and the context switching can be a bit difficult"* (TP9).

Barriers to psychological safety: In case A, newcomers are particularly observed by experienced team members who report their progression to the management. Though this is done to inform newcomers' continued presence within the team, this can consequently create a stressful atmosphere for them. *"I give feedback to the management, on how things are going, whether we can keep this person in, or the person should be replaced"* (AP1). Additionally, newcomers are less able to advocate for themselves on questions or decisions compared to experienced team members. *"I [a senior employee] have no problem telling the manager or the customer that I want to do this"*. It is not that they are incapable, but rather they need time to build confidence. The confidence that experienced employers have built up over years of work is difficult for recent graduates to achieve. *"A lot of new people are directly from school, and it's not easy for them to do the same"* (AP3).

Team members tend to notice that newcomers rarely engage with team discussions and share their perspectives. This lower level of engagement creates confusion regarding the tasks these individuals are performing, as the team remains unaware of their conditions. *"I can notice when there are new ones or people with lower energy personalities. They don't want to ask for help"* (AP6). In a highly technical or specialized environment, newcomers

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

remain silent because they lack the knowledge needed to raise queries. *"It's hard to know when you don't know what they're not asking"* (TP8). This substantial knowledge gap creates uncertainty. Even in a psychologically safe environment, newcomers struggle to grasp the context without their confidence in technical understanding. *"Once you've become used to the forums and feel a bit more self-sure of yourself, then I think most people would dare to speak up"* (TP8). In addition, newcomers recognize that their co-workers are more experienced and feel comparatively inexperienced. This makes it difficult for them to follow the discussions, and scared to admit their confusion. This creates tension between their rational thinking and emotional states, making them emotionally insecure even though the work environment is safe. *"No one is going to be angry or something, because I don't understand, I'm just scared sometimes"* (TP3).

Though team needs to maintain a balanced environment in which everyone feels comfortable in asking questions, this can be challenging. Because the information should not be presented in a way that makes it seem acceptable, nor should it make newcomers feel intimidated. *"We tend to have a friendly atmosphere, but it can sometimes be a bit blunt, no need to sugarcoat things, but no need to make them inadvertently aggressive either"* (TP9). Nevertheless, case T teams encourage newcomers to ask questions wherever they encounter issues. Although newcomers face difficulties and make mistakes within a complex system, this leads them to improve and feel more comfortable in a psychologically safe environment. *"No question is too silly to be asked and we always try to say yes"* (TP6).

Knowledge gap and retention: Although case A comprise strong expertise in certain areas of technology, it has identified domains in which their competencies are lacking. These gaps should be considered during the onboarding process to ensure that newcomers acquire the necessary knowledge and help reduce existing knowledge deficiencies. *"[...], There is one area where the competence is non-existent and there is no one out there that knows it either"* (AP2) and *"We do have areas in which we are severely lacking competency and we definitely have a people debt in that sense"* (AP3). Therefore, team strives to hire newcomers into specialized areas to make the experience more engaging. *"I'm sure that's very important. It's a very specific area to get newcomers into team"* (TP5). In addition, experienced team members expect new starters to familiarize themselves with the ongoing project works. *"More fresh new knowledge from school on how you're supposed to code and but does not have this [experience]. I mean, they will gain it"* (TP5). For instance, an experienced developer who understands the entire code base will feel satisfied when they remove redundant code. In contrast, a newcomer might feel satisfied when they add more code. Therefore, newcomers are introduced to a code base that requires careful handling. *"When you are to go into a test or a piece of code and improve something, make a piece of code more readable"* (TP8).

Apart from this, integrating highly specialized expertise into a complex system is challenging. *"It's challenging when someone has 15 years of experience in a bit piece of software and no expertise whatsoever in any kind of other software"*. To utilize such resources, the organization needs to invest further in their development. *"To utilize that kind of resource, you either need to educate them, and he needs to be open to being educated"* (AP7).

4.3.3. Contextual challenges

System complexities: The complexities in the system present challenges in certain applications where even experienced team members lack prior exposure. This hinders the

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

learning process of newcomers and slows their integration into the team. *"Sometimes we receive an error in an application that we haven't even heard about, new people come into the project. It's more complicated than it sounds"* (AP1). Working within a large, complex system does not lead to a full understanding without sufficient exposure to its components and processes. *"I am still pretty new, and it's an extensive system with many applications. So it could just be that I haven't gotten a clear grasp of it yet"* (AP6). These systems include numerous applications and unclear naming conventions that newcomers are not sufficiently familiar with. This creates a lack of clarity in accountability when a problem occurs. These types of situations are challenging for a newcomer to encounter. *"When problems occur, it could be a little hard to see where the responsibility lies"* (AP6).

The complexity of the domain poses substantial challenges for newcomers. Although their onboarding agenda includes initiatives aimed at improving domain knowledge, newcomers perceive that their understanding is insufficient. This calls for input from experienced team members. *"I think there might be some people who really grasp the code and the whole architecture of the system to the level where the newbies and some junior people might need some more input from those people. It is a problem, being the domain is so complex"* (AP2). Additionally, in case T, newcomers are expected to contribute as regular team members once their onboarding process is completed. Nevertheless, team proposes involving experienced employees with strong domain knowledge to train newcomers and junior employees. However, given the system complexities and responsibilities of senior employees, implementing such an approach is difficult in practice. *"We have some people who have 20 years of experience in this domain. How can we use them to train people that have one or two years of experience? But I think it's challenging to make it"* (TP1).

Beyond that, system complexity and barriers to domain knowledge acquisition create steep learning curves for newcomers. Even though newcomers receive introductions to tasks, they require adequate time to adapt to complex systems. *"I have not done anything code-wise during this time. Take time for newcomers to adapt to complex systems"* (TP3) and *"It will be very difficult because the procedures and the way of working is entirely different in the systems team and to get familiarized with all these things"* (TP4). LSA organizations comprise many teams whose members collaborate with each other to perform their tasks. Newcomers require adequate time to familiar with different project teams, as they are primarily assigned to a single team. *"It takes time, to get into the project because first you focus on only your own team on the tasks there"* (TP5) and *"In my experience, which is very little, like five months, I have never had something that was needed from another team or manager"* (TP15).

Remote onboarding: In a hybrid work setting, team members can split their time between remote work and on-site work. It is challenging for remote newcomers to integrate into cross-team collaborations and synchronised planning. "*It would probably have been easier to have something more formalised, especially since we're remote, or at least I am remote. Some of the other guys are sitting together in X, but I am in Y, and it is all between teams*" (AP5).

A remote work setting creates gaps in availability and visibility among team members. Communication through MS Teams platform can reduce the opportunities for in-person social interactions. This absence of frequent interpersonal contacts creates psychological distance among the team members. *"If we don't hear from each other on MS Teams, maybe it feels a bit isolated"* (AP1). As a result, newcomers face difficulties in building strong relationships with their co-workers and are unaware of whom to reach out to, particularly in the absence of a structured communication process. and *"There's also some informal*

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

information going on there that you miss out on" and *"It is maybe not that obvious how to contact other teams and when and who in the other groups"* (AP5).

Cultural sensitivity: Welcoming newcomers from diverse cultural backgrounds requires careful consideration because working cultures and norms differ from region to region. Without careful individual assessment and planning, newcomers experience isolation and find it difficult to integrate into their teams. *"I faced some difficulties during the initial days when I started here. I have heard a lot from others that they are not providing enough support during onboarding"* and "*It is mainly because of the Swedish culture that they think that the other guy knows more than I do*" (TP4). Some working cultures practice self-learning onboarding approaches, whereas others follow well-defined onboarding procedures, providing a sequence of instructions. New employees from diverse cultural backgrounds embrace these changes differently. "*I know that you have to learn it yourself and it depends on. In Sweden, there is no such formalities. But in some other countries"* (TP4).

Different cultural backgrounds associate with distinct working styles. Newcomers from other countries who integrate into a different cultural environment observe variations in working pace, time management, and team practices, which highlight need for careful team discussions and task management. *"Some people, they were against calling for a meeting on Friday afternoon. There are a lot of problems"* (TP4).

Employees from diverse cultural backgrounds are accustomed to different types of organizational structures. Some have experience in hierarchical systems, where authority flows from top down through clear chains of command. *"If they are managers, Indians, trainees, their way of working is different. It's purely hierarchical"*. Others are more familiar with flat structures, characterized by more autonomy and collective decision-making. The statement *"They always force us to do something"* reflects a cultural mismatch resulting in tension arising from differences between top-down management and flat organizational structure. New hires are particularly affected by such structural constraints. *"They are employing more Swedish, Chinese and Indian managers. because Swedes cannot force others"* (TP4). Moreover, in a flat organizational culture, a new employee being productive is not only possible but also acknowledged and accepted. *"A junior employee will be more productive than a senior employee"*. In such an environment, performance and output are valued more than seniority. *"I'll get senior to junior. I haven't seen anything like that"* (TP4).

A team composed of members from diverse cultural backgrounds also face challenges in both decision-making and execution. *"They are not used to Sweden is quite special when it comes to like decision making"* (TP6). Although newcomers are proficient in English, differences in their decision-making styles still create misunderstandings. In Sweden, where orgnizations are embedded in a flat structure, employees make their own decisions once they feel confident in their competence, while seeking clarifications when necessary. However, newcomers from more hierarchical backgrounds are used to following instructions from their managers. *"They are not at all used to that. They are more used to managers actually telling them what to do"* (TP6).

Furthermore, long-term exposure to top-down hierarchical structures leads junior employees to expect those in higher positions to take the lead in discussions and apparently seek only their specific support. As reflected in the statements *"It's always like it's directed to the person higher up in their hierarchy"* (TP11) and *"I'm not a native English speaker, and I mean in the education in X, you must follow the teacher, right? So that's the challenge for me in some meetings"* (TP14), communication tends to flow upward rather

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

than being shared collectively. This creates challenges within teams, as not everyone feels comfortable expressing their insights during meetings. Consequently, this hinders collective decision-making, which is essential for a collaborative team environment. *"There could be some like hierarchy issues, like people aren't really speaking up because they expect the person higher up in the hierarchy to speak up"* and *"We're not used to speak out or myself or even didn't think about it. But we have an opinion, so that's the challenge"* (TP11).

Agile-related issues: Newcomers perceive a contradiction between the intended agile mindset and the formal, task-intensive reality of daily agile practices. These process-related complexities can be overwhelming. *"There has to be a place for innovation and creative thinking and new ideas as well. But maybe it contrasts against the very formal reality of the Scrum Board, which is a little bit crazy because, if you're looking at the agile mindset and what that represents, it should not be that kind of a hassle"* (AP3). While transparency in agile supports task accountability and coordination, it also exposes workload saturation and limits the opportunities for innovation. *"But if it's transparent that we should do these things, it shows that we don't have time to do other stuff"* (AP3).

Integrating into a physically co-located team with a limited development focus and immediate support is straightforward for newcomers. *"I would say, easier for a newcomer coming in because then it was a little bit more limited scope"* (TP6). However, newcomers face significant complexity with LSA projects, involving multiple interconnected systems with broader functionalities. "Higher threshold coming in now" indicates that newcomers need comprehensive knowledge for effective system integration, which is challenging. Agile development also involves trade-offs that are not always acknowledged *"When you have agile teams that are covering so much more functionality"* (TP6).

5. Discussion

Our findings expand understanding of newcomer onboarding in LSA projects. First, while Bauer's organizational socialization model [6] provides a relevant and broadly applicable framework, its direct application to LSA contexts is insufficient without augmentation by agile-specific practices that are structurally embedded in how LSA teams work. Second, the complexity of LSA systems generates a distinct onboarding dimension (i.e., technical integration) that existing onboarding frameworks do not recognize and that social integration practices alone cannot address. Third, the challenges that characterize LSA onboarding are not merely experiential difficulties for individual newcomers. They are organizational compromises that accumulate as forms of socio-technical debt. Together, these three findings suggest that LSA onboarding must be treated not as a peripheral HR activity but as a core risk management function.

Successful LSA onboarding is not a discrete event but begins conceptually from the recruitment phase, requiring a structured and coordinated integration plan. This active approach necessitates connecting newcomers with relevant stakeholders early to ensure they fit LSA environment, aligning well with the initial phases of Bauer's model [6].

Formal orientation and support are crucial in LSA to manage complexity; these practices include mandatory introductory meetings and detailed system overview presentations. These are critical for grasping LSA system integrations. We identified different orientation techniques, including introductory meetings, system overview presentations, and team-led grooming sessions, which are either socially or technically oriented. Using onboarding templates or checklists standardizes timelines and access procedures. Providing such

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

templates that specify the required steps is considered a best practice in onboarding. [6]. Companies consider institutionalized (formal) onboarding to be a successful approach [7, 10] and recommend that "most onboarding activities should be required and offered formally" [35]. Reference to the system documentation provides an efficient and coherent way of familiarizing oneself with the tools and technologies, the system domain, feature dependencies and security measures, which is particularly important in LSA development. Moreover, our case companies employed all or some of the techniques, such as providing access to tools, provisioning accounts, and assisting with system setup, to facilitate newcomer integration into the workflow.

Training is often a hybrid approach, combining formal learning (materials, online courses) with individualized practical methods like task-based and case-based learning. Our results support the finding of [6, 7] that training can be formal or informal. The hands-on methods are particularly supportive in LSA as they help newcomers gradually navigate system architecture, complex dependencies, and essential domain knowledge. This is essential for minimizing subsequent rework caused by incomplete understanding. By conducting code reviews individuals familiarize themselves with the project's codebase, standards and conventions, while promoting knowledge transfer, team awareness and transparency, particularly among junior developers.[36].

Furthermore, coaching and support effectively leverage mentors, peers, scrum masters and product owners, demonstrating a necessary institutionalized onboarding approach characterized as formal and sequential. All four of our case companies position mentors to facilitate a smooth technical and social transition for newcomers. Newcomers who receive mentoring-based education report being active earlier than those who undergo conventional onboarding [24]. Mentoring is claimed to be a central onboarding approach facilitating newcomers' integration into their team, technology, agile ways of working, and occurs informally through peer interactions. Mentoring helps newcomers enhance their skills by providing someone to ask questions. [7,37]. Pair programming provides informal peer support by integrating newcomers into daily collaborative work alongside experienced colleagues [7, 37]. The team being approachable to questions from new employees is considered an effective onboarding practice, as it allows newcomers to seek help from any team member [37]. Additionally, seeking ongoing feedback is a recognized onboarding practice that helps newcomers to actively gather information to improve their performance and adapt to teams.[6, 19]. The incremental and iterative nature of agile [38] treats feedback as a continuous and essential practice, both within teams and across other stakeholders. In LSA settings, however, feedback extends beyond a team practice to serve as a critical coordination and transparency mechanism [1], becoming increasingly challenging to maintain as the number of teams and organizational layers increases.

Our findings align with adjustment dimensions [6] which include self-efficacy, role clarity, social integration, and knowledge of culture. Moreover, we propose "technical integration" as an adjustment for LSA, extending Bauer's model by emphasizing the importance of smooth technical integration. Self-efficacy, reflecting an individual's belief in their ability to perform their responsibilities effectively, is linked to positive workplace outcomes, including organizational commitment, job satisfaction and reduced staff turnover [19]. This is particularly true for newcomers who feel capable of mastering their work, as they tend to adapt and integrate successfully. A higher level of role clarity is indicated when new employees have a clear understanding of their roles and responsibilities. In two of the cases we studied, teams expect newcomers not only to understand the cultural and technical expectations of their job roles but also to engage actively and contribute

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

meaningfully. Role clarity is identified as one of the most important predictors of job satisfaction and organizational commitment [19]. It is also important that newcomers feel socially comfortable and are accepted by their peers and supervisors. [19]. Case companies organize various social integration activities to help newcomers feel settled and psychologically safe within their work environments. A psychologically safe environment allows the belief to emerge that the context is safe for interpersonal risk-taking, in which speaking up with ideas, questions, concerns, or mistakes is welcomed and valued [39–41]. Whereas "fear" undermines cognitive diversity, including analytical skills, creativity and problem solving and further reduces learning behaviours such as asking questions, seeking help, and openly discussing mistakes [39].

Beyond emphasizing people and culture-related adjustments, the complexity of the LSA context calls for "technical integration" as a new adjustment dimension. Case companies facilitate technical integration through various means, such as technical meetings and tech know-how sessions. Providing an opportunity to learn technologies while enjoying a coffee together with peers seems to be beneficial. Specifically, "tech fika" provides an opportunity to learn and raise questions regarding technical and domain knowledge in a relaxed, discussion-friendly setting. These technical meetings become important when the team has more new technical developers, as they can spend more time on learning and understanding system complexities [9].

Notably, onboarding functions are fundamentally applicable to LSA projects when augmented with specific agile practices. Daily standup meetings are a short, time-boxed meeting widely used by agile teams and are particularly useful for newcomers [42]. Daily standup meetings improve communication and coordination within the project, providing an important opportunity for the agile team to share information and increase team awareness. Junior developers tend to view daily standup positively, as they find these meetings valuable for receiving information and getting support. [42]. Majority of project teams use pair-programming for a defined period as an initial approach to technical integration. Pair programming provides useful support for newcomers to familiarise themselves with the code base [7]. However, the relative benefits of pair-programming compared to solo programming are not precise, and it depends on the programmer's expertise, complexity of system, and tasks performance[43].

Innovation sprints provide newcomers with hands-on experience in developing innovative thinking. No work is planned during the Planning innovation (PI) iteration in PI Planning. Instead, this time is set aside, allowing teams to be creative [44]. Retrospectives are one of the other standard practices developed for agile teams, in which they reflect and improve their work process [7, 45]. Retrospectives involve discussing team obstacles, sharing feelings, action points, root causes and future actions [45]. Retrospectives provide a platform for newcomers to exchange feedback and receive guidance on their progression, opportunities and weaknesses. Agile training is included in introduction sessions, where newcomers are encouraged to participate and tailored to their specific roles. Agile training helps to bridge the gap between traditional and agile practices. It allows the agile teams to absorb and retain the agile values and principles [46].

The challenges identified in LSA onboarding expose the projects to various forms of socio-technical debt (e.g., people, social, process, organizational, knowledge, documentation and code). (See Figures 1, 2 and 3 for the organizational, individual and contextual challenges and their association with socio-technical debt accumulation). These associations between onboarding challenges and socio-technical debt are interpretive propositions, inferred from the qualitative data.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

The lack of an organized onboarding process slows the integration of newcomers. In particular, limited access to people and information is associated with social debt, requiring newcomers to seek information and establish relevant connections. Consequently, integrating newcomers into projects solely on newcomers' and project availability slows their growth and performance, interpreted as potentially leading to people debt. The presence of people debt is directly related to productivity and employee satisfaction [18]. Furthermore, failure to allocate dedicated time for onboarding within working sprints results in inconsistent onboarding practices and stressful situations for newcomers, particularly associated with the emergence of organizational debt. Organizational debt represents an accumulated burden that prevents people from achieving exceptional results [15].

The absence of an organized knowledge-sharing approach for newcomers is interpreted as potentially contributing to knowledge debt [47], whereas insufficient knowledge distribution represents another dimension of technical debt [47]. Allocating time for system learning alone is not merely sufficient. Newcomers also require adequate knowledge and skills in planning, prioritisation and decision-making; otherwise, knowledge debt may be incurred. Insufficient time allocated to developing support tools and processes tailored to individual needs and learning pace is interpreted as potentially contributing to organizational debt, reflecting the "things that should've been done to ensure health efficiency, but weren't [15]." A newcomer who does not have the opportunity to be led by someone towards other teams and their structures finds it difficult to perform and get along with team members, associates with organizational debt [15]. Assigning challenging tasks during the early stages of onboarding has been linked to lower confidence and anxiety among newcomers, as reported in a study of onboarding in Microsoft software teams [9]. Similarly, finding the right piece of code to work with and a task to start with has been noted as a challenge in open source software projects [25]. These challenges become particularly important in LSA environment, where domain complexity makes the selection of suitable onboarding tasks even more difficult.

Recruitment is challenging due to inherent knowledge gaps and complex business requirements. In particular, recruiting employees with a combination of IT and business knowledge, especially domain experts can help resolve the root causes of the people-related issues [46]. When recruitment and onboarding are poorly executed, it is associated with people debt [18]. People debt stems from insufficient expertise or lack of effective onboarding support [18].

In addition, attrition of experienced employees, coupled with the simultaneous onboarding of newcomers, changes the teams' structure and task management. Insufficient learning opportunities thus created are interpreted as potentially contributing to the accumulation of knowledge debt [47]. Frequent team switching, challenges of handling new customer projects with inexperienced employees, and training of newcomers in a dynamic environment require significant effort and time. In LSA environments, these challenges are often exacerbated by the coordination and cross-team dependencies. Consequently, these challenges that teams encounter during the onboarding process are associated with organizational debt, which reduces organizational effectiveness and slows adaptation [15]. To address these challenges, organizations could adjust team composition by including more experienced members and fewer newcomers, creating a balance that enhances the team's ability to continuously improve [48].

The issue of cognitive overload, resulting from tightly scheduled introductory sessions and fast-paced development, creates significant mental fatigue and pressure on newcomers. Cognitive overload occurs at a point where the amount of information or the ability to

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

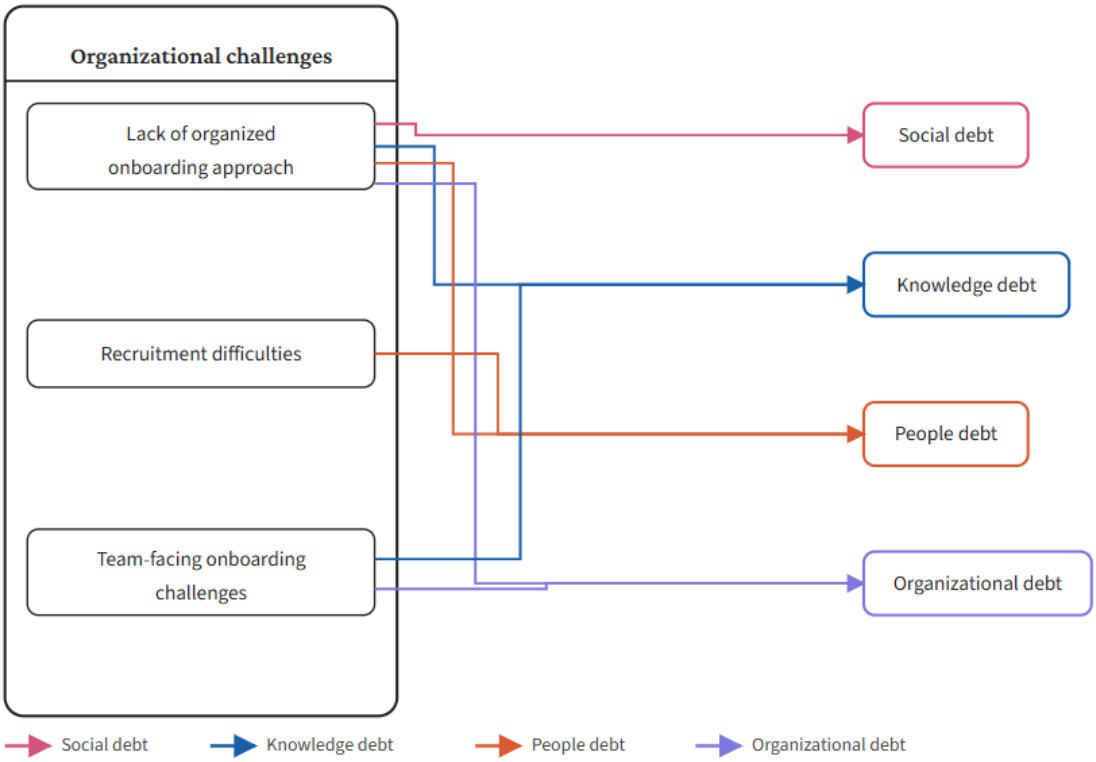

Figure 1. Organizational challenges associated with socio-technical debt

process information exceeds an individual's capacity [49]. Task completion urgency builds pressure, reducing engagement capacity, flexibility and innovation [15]. Characteristics present in contributing files such as branching workflows, hidden dependencies, scattered instructions, and project conventions increase the cognitive burden for newcomers in open source software projects [25, 50], particularly amplifying the impact on systems developed by multiple teams. This highlights the trade-off between culture and management practices interpreted as potentially contributing to organizational debt. "Organizational debt is all the people/culture compromises made to "just get it done" in the early stages of a startup [15]. " Introducing a gradual stepwise transition would help newcomers' adaptation to the real tasks.

We identified both individual and structural barriers to psychological safety. Human instincts can prevent people from working productively in a dynamic environment [39]. When individuals experience high psychological safety and high performance standards, they are more likely to operate in a learning zone [39, 51]. Cultivating psychological safety during onboarding can be supported through clear expectations, encouraging open dialogue, and providing continuous feedback within the first 90 days [52]. This is particularly important in the LSA environment, where newcomers navigate across multiple teams. Newcomers who rarely participate in team discussions exhibit a low level of engagement and awareness, which is interpreted as potentially contributing to social debt [16]. Fear of asking questions and lower confidence arising from experience gaps are associated with people debt [18]. Newcomers face organizational-level challenges that can reduce their confidence and perceived comfort within the workplace. To prevent the accumulation of potential organizational debt [15], teams need to invest substantial efforts in creating an

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

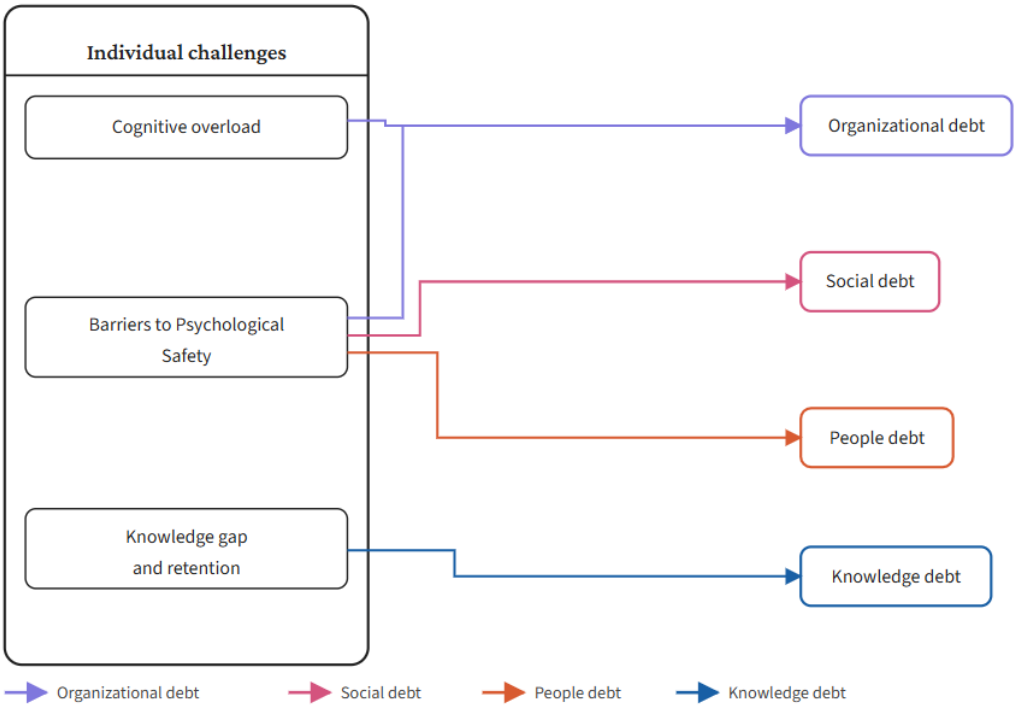

Figure 2. Individual challenges associated with socio-technical debt

environment where newcomers can thrive, ensuring that issues are neither sugar-coated nor expressed aggressively. Moreover, the inherent complexities of legacy systems make it challenging to achieve a comprehensive understanding of the system is potentially interpreted as contributing to knowledge debt [47].

Learning a large legacy codebase makes onboarding challenging for newcomers, requiring a significantly longer period of mentoring and coaching compared to a typical software project. Furthermore, code written by former team members complicates knowledge transfer [53]. These challenges are particularly pronounced in a large-scale development environment. Moreover, inadequate mentoring on the system and workarounds subsequently associate with code debt [54]. The presence of a large, extensive system with many applications and unclear naming conventions suggests that standards are needed to prevent the accumulation of documentation debt. Although companies maintain an intranet, wikis and production documents, these resources could be outdated, incomplete and insufficiently detailed. Consequently, onboarding becomes challenging and requires newcomers to engage in a continuous interaction with mentors to acquire knowledge [53]. Accessing a complex domain requires expert support. Failure to embed domain knowledge through documentation and knowledge-sharing sessions is interpreted as potentially leading to knowledge debt and documentation debt [47].

Similarly, difficulties with remote onboarding complicate newcomers' seamless integration into cross-team collaborations and planning activities. Developing shared organizational values, skills, attitudes, abilities, behaviours, and relationships becomes challenging in a virtual onboarding setting [55]. Limited informal communications between newcomers and their reduced visibility within the team result in newcomers being excluded from discussions [56]. Such challenges are prevalent in LSA environments, where coordination across multiple teams exacerbates communication and availability gaps. When the needs

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

and expectations of these remotely working newcomers are unmet, the resulting lack of effective social and process integration constitutes a compromise in social processes [7, 15]. These results may potentially contribute to the accumulation of social debt and exacerbate team isolation. Process debt arises when a discrepancy exists between a job role and the responsibilities defined in the organizational structure [17].

The issues related to cultural sensitivity are interpreted as potentially contributing to social and organizational debt. A failure to foster an inclusive and collaborative learning culture undermines the project network and requires future organizational effort to rectify [15, 16]. For instance, cultural misunderstanding arising from a failure to respect and adapt to different working styles, organizational structures, decision-making strategies and communication styles associate with potential exacerbation of social and organizational debt.

Lastly, issues related to agile development are associated with process and organizational debt [15]. Process complexities across both ideal and real-world agile practices, coupled with insufficient time for innovations, pose challenges for newcomers. These challenges are further reflected in difficulties in conveying agile principles, as established team members hold high expectations of newcomers and struggle to communicate them effectively [48]. A workflow refers to a series of repeated activities carried out to complete a particular task while a process consists of repeatable activities performed to achieve organizational goals [18]. "Inadequate adaptations" within the organizations leads to ineffectiveness [15].

We offer insights for practitioners to improve LSA onboarding by integrating agile-related practices and adopting a practical training approach that systematically combines formal, task-based, and case-based learning. In particular, we recommend incorporating technical integration to facilitate a psychologically safe adoption of technical systems and to address organizational compromises that extend beyond individual experiences. Our goal is to guide a process that is structured, flexible and context-aware, helping to reduce the accumulation of socio-technical debt.

5.1. Threats to validity and limitations

To evaluate our study limitations, we apply validity criteria defined by Runeson and Höst [28].

Construct validity refers to the extent to which our study measures what it is intended to measure. This is ensured by forming an interview guide to ensure that the data produced through interviews answers our research questions. The researcher's perceptions, along with the interview guide, were discussed with company contacts to potentially build meaningful insights and prevent any misinterpretations. Throughout the study, debriefing sessions supported researchers in avoiding potential bias in data collection and interpretation. The interviews were recorded so that they could be transcribed later and all researchers could have the same understanding of what was said. Spending sufficient time with the cases and engaging in prolonged interaction helped researchers to build trustworthy relationships with interviewees, reducing the likelihood of them providing any biased information.

Internal validity concerns the credibility and robustness of our findings. We conducted a multiple case study using interview data collected in different organizational contexts. We selected participants based on their differing roles in order to capture diverse perspectives. We sent the transcripts back to the interviewees for member checking. Additionally, we produced mind maps during the qualitative analysis stages to ensure the accurate interpretation of the interviewees' perspectives. The researchers collectively reflected on

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

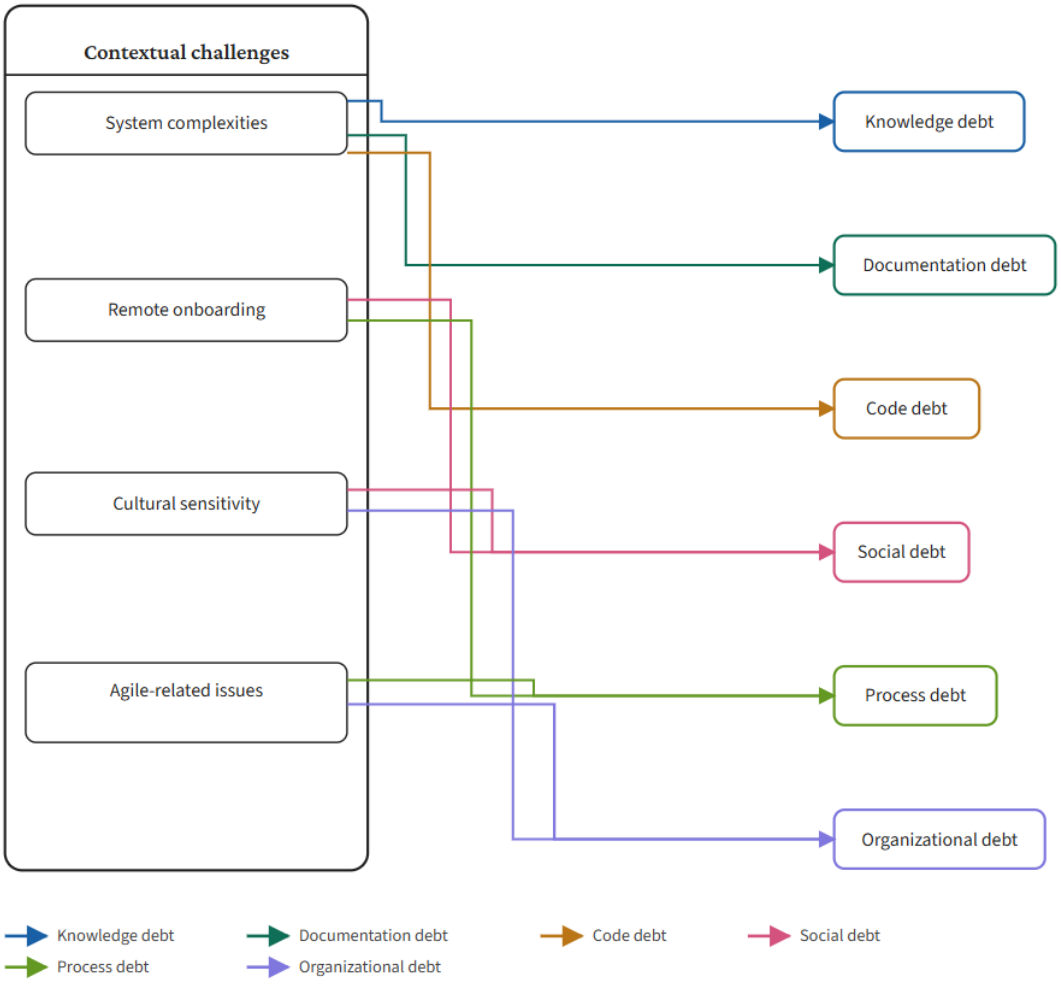

Figure 3. Contextual challenges associated with socio-technical debt

initial coding and assigned it to themes, potentially reducing the bias of the individual researchers when analysing the data. In relation to data adequacy, data saturation was not formally assessed as a distinct step within the methodology. However, with 41 interviews conducted across four different companies, data analysis continued until no substantially new themes were emerging in the later part of analysis. Nevertheless, we acknowledge this as a methodological limitation. In the Methods section, we provided a detailed description of our findings, supported by quotations from the data, and maintained audit trails comprising the interview methods, transcripts and data collection.

External validity concerns the generalizability of findings beyond the studied cases. Our four case companies represent a range of organizational sizes and domains, which strengthens transferability. Regarding data collection, participants' experiences reflect their overall professional experience and not their tenure at the specific organization studied. The participants also included a range of career stages across all four companies. Furthermore, most of our participants are team members holding a variety of roles, including developers, testers, designers and architects. However, we acknowledge that the onboarding experience encompasses multiple perspectives across working levels, including mentors and management. Beyond that, all four of our companies are located in the same geographic

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

region, which introduces cultural homogeneity and limits the applicability of findings to organizational cultures. Additionally, as a qualitative case study, our work does not permit causal claims. The associations we draw between onboarding practices, challenges, and socio-technical debt accumulation are interpretive and theoretically grounded, but not empirically tested in a quantitative sense. Our findings should be treated as propositions that needs further validation rather than established generalizations.

Reliability concerns the repeatability of research process. To uphold reliability, we used the thematic mapping approach by Braun and Clarke [33], showing how we coded collected data. Our methodology includes detailed reporting on case context, data collection techniques and data analysis, ensuring consistency. The interview guide was discussed in detail, and any unclear questions were avoided to ensure that interviews were clear.

5.2. Recommendations

We propose five key recommendations for practitioners to improve LSA onboarding. The goal is to provide guidance toward a process that is flexible, context-aware, and structured to mitigate the accumulation of socio-technical debt.

R1: Design competency-based onboarding plans rather than time-based ones: Across cases, a recurring challenge was the mismatch between fixed onboarding schedule and variable prior experience of newcomers. Rather than prescribing a uniform, fixed-schedule process, onboarding duration and content should be customizable based on newcomer's existing domain knowledge and technical skills. This approach ensures that experienced hires are not over-burdened by redundant training, while those requiring more foundational support are given a stronger, customized learning path. This will support long-term retention and reduces risk of incurring people debt [18] from misaligned roles.

R2: Integrate agile ceremonies into onboarding from day one as structured learning mechanisms: Our findings show that agile ceremonies were among the most consistently reported and effective onboarding tools across all four cases. However, their use was often informal and dependent on team initiative rather than deliberate design. In Case E, standup meetings were explicitly used to monitor newcomer progress and surface blockers. In Cases S, T, and A, pair programming provided technical scaffolding during critical early weeks. Where these practices were structured and newcomers reported faster integration; where they were ad hoc, newcomers were more likely to remain passive observers. This means practical training approach that systematically blends formal instruction with hands-on learning. This method should leverage essential Agile practices (e.g., pair programming, daily standup meetings) from the first day to provide immediate technical and social exposure. Furthermore, teams should deliberately employ task-based learning for simple, focused assignments alongside case-based learning for assignments that require newcomer to explore complex system architecture, multiple dependencies, and demanding business requirements. This deliberate progression from simple to complex tasks ensures gradual increase in cognitive load while building comprehensive system understanding. This results in reducing likelihood of committing errors that contribute to technical debt.

R3: Expand social integration beyond the immediate team through structured cross-team exposure: A distinctive challenge in LSA onboarding, observed across all cases, is that newcomers must integrate not into a single team but into a network of interdependent teams. Cases T and A shows that newcomers who were confined to their immediate squad during onboarding struggled with cross-team collaboration and had difficulty identifying right people to contact for system-related support. This produced

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

social debt [15, 18] in the form of communication breakdowns and duplicated effort. Social integration requires structured effort. We recommend that onboarding plans in LSA contexts explicitly include structured cross-team exposure from early in the onboarding period. For example, newcomers should be spread out. Avoid grouping newcomers all together. Place them on existing, established teams. This maximizes their exposure to diverse organization units and helps them to gain peer knowledge. Robust support must be guaranteed which needs both internal and external stakeholders. Internal support can come from a dedicated mentor or scrum master. External support from the product owner is required to ensure clear guidance on business priorities. This is a common blind spot for newcomers and is noticeable in LSA context.

R4: Address gap between textbook agile and real-world LSA practice through explicit expectation setting: Across all four cases, newcomers consistently encountered a gap between agile principles they had learned through training or prior experience and actual practices of their LSA teams. Fourthly, bridging the gap between textbook Agile values and actual Agile practices requires cognitive and cultural sense-making. For example, implementing a customised agile at the organizational level reduces the sense of being overwhelmed by newcomers due to process complexities, thereby enhancing understanding process orientation and expectation setting. In addition, a limited scope for innovation can demotivate newcomers. For instance, if newcomers expect creativity but encounter time pressure and a focus on delivery, they may feel less engaged. Therefore, realistic innovation opportunities should be facilitated. Transparency within agile reveals workload intensity and performance expectations, which can generate stress and fear of underperforming among newcomers. Therefore, newcomers require a gradual workload ramp-up and an acceptable learning curve to avoid overwhelming them during these critical periods.

R5: Restructure orientation to prevent cognitive overload through phased information delivery: Cognitive overload was one of the most frequently cited individual-level challenges across cases. Such overload stems from a high volume of initial information, tight scheduling, expectation to maintain peer-equivalent working pace, and frequent switching between onboarding tasks, meetings, and project work. In LSA contexts, this challenge is structurally amplified: the system complexity, number of teams, and breadth of domain knowledge required, which means that the information load placed on newcomers is inherently greater than in smaller projects. When unaddressed, cognitive overload generates organizational debt by eroding newcomers' capacity for reflection, reducing code quality, and increasing the likelihood of early disengagement [15, 49]. This is best addressed by restructuring initial information cascade into a phased and gradual approach to delivery. Orientation materials and introductory sessions should be broken down into smaller, digestible modules and intentionally spread across first few weeks, rather than compressed into first few days. By establishing a clearer, stepwise transition from abstract onboarding training to actual job tasks, organization reduces immediate pressure on newcomer. This allows adequate time for critical reflection, assimilation, and sustainable adaptation to true working pace of LSA environment.

6. Conclusions

This study investigated onboarding processes for newcomers in LSA projects and assessed Bauer's model applicability in this context. Through a qualitative multiple case study spanning four Swedish companies, we have established three principal contributions.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

First, Bauer's model is broadly applicable to LSA onboarding but requires augmentation. Crucially, our findings reveal that the model must be augmented with specific agile-related practices and case-based learning practices to be effective. We identified system overview presentations, team-led grooming sessions, and use of system documentation as meaningful techniques for LSA onboarding. Any application of Bauer's model to LSA must incorporate these agile dimensions to be analytically complete. Second, technical integration constitutes a distinct onboarding dimension specific to LSA that is absent from existing onboarding frameworks. The scale, complexity, and multi-team architecture of LSA systems create a category of onboarding challenge (i.e., learning to navigate a large, interdependent, and continuously evolving technical environment) that social integration activities alone cannot address. The emergence of technical integration suggests that LSA teams have organically recognized this need. We propose that technical integration be formally recognized as an extension of Bauer's model when applied to LSA context. Third, the most significant contribution of this work lies in identifying the LSA-specific onboarding challenges and establishing their link to various forms of socio-technical debt. We demonstrated how issues like cognitive overload from overly dense schedules, large-scale system complexities and difficulties of remote integration become compromises that hinder long-term team efficiency and cultural stability. Successful LSA onboarding is not merely transactional; it is an integrated process that begins with rigorous recruitment and requires a flexible approach to training and integration. This framing reposition onboarding from a peripheral HR concern to a core technical and organizational risk management activity. Poor onboarding is not merely costly in the short term; it creates accumulated deficits that must be repaid with interest as projects scale.

We provided five recommendations to offer practitioners a framework for restructuring LSA onboarding: competency-based planning, agile-integrated learning, structured cross-team socialization, explicit expectation setting around real-world agile practice, and phased orientation delivery. These recommendations will not only improve newcomer experience but to proactively prevent socio-technical debt that accrues when onboarding is treated as an afterthought.

Building upon our findings, we propose several future research directions for LSA environments. First, the associations between onboarding challenges and socio-technical debt require quantitative validation. Future studies should focus on developing and applying metrics to measure the accumulation of specific types of debt in projects that employ varying onboarding strategies. Second, Future research should adopt a longitudinal, multi-perspective approach to interview mentors and management in addition to the team members. Third, longitudinal studies are needed to trace how onboarding quality at individual level translates into team performance over time. Fourth, the effectiveness of task-based and case-based learning relative to other training approaches highlights controlled investigation. Finally, as LSA frameworks continue to evolve and hybrid work becomes the norm, comparative research is needed to explicitly analyse onboarding in different scaling frameworks (e.g., SAFe, LeSS) as well as study specific socio-technical debt implications of fully remote versus hybrid LSA teams.

CRediT authorship contribution statement

P. Herath: Writing original draft, review & editing, visualization, methodology, investigation, formal analysis, data curation, conceptualization.

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

M.O. Ahmad: Writing original draft, review & editing, visualization, supervision, project administration, methodology, investigation, formal analysis, data curation, conceptualization.
T. Gustavsson: writing original draft, review, validation, investigation, conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

This research was carried out within the NODLA project [grant no. 20200253] funded by Knowledge Foundation Sweden, Helge Ax: son Johnsons Stiftelse and Swedish Foundation for International Cooperation in Research and Higher Education (STINT) [IB2020-8720].

References

[1] H. Edison, X. Wang, and K. Conboy, "Comparing methods for large-scale agile software development: A systematic literature review," *IEEE Transactions on Software Engineering*, Vol. 48, No. 8, 2021, pp. 2709–2731.
[2] T. Dingsøyr and N.B. Moe, "Research challenges in large-scale agile software development," *ACM SIGSOFT Software Engineering Notes*, Vol. 38, No. 5, 2013, pp. 38–39.
[3] M. Kowalczyk, B. Marcinkowski, and A. Przybyłek, "Scaled agile framework. dealing with software process-related challenges of a financial group with the action research approach," *Journal of Software: Evolution and Process*, Vol. 34, No. 6, 2022, p. e2455.
[4] A. Joskowski, A. Przybyłek, and B. Marcinkowski, "Scaling scrum with a customized nexus framework: A report from a joint industry-academia research project," *Software: Practice and Experience*, Vol. 53, No. 7, 2023, pp. 1525–1542.
[5] M.O. Ahmad, "5g secure solution development and security master role," in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, 2024, pp. 629–633.
[6] T.N. Bauer, "Onboarding new employees: Maximizing success," *SHRM Foundation's Effective practice guidelines series*, 2010. [Online]. https://www.shrm.org/content/dam/en/shrm/topics-tools/news/talent-acquisition/Onboarding-New-Employees.pdf
[7] P. Gregory, D.E. Strode, H. Sharp, and L. Barroca, "An onboarding model for integrating newcomers into agile project teams," *Information and Software Technology*, Vol. 143, 2022, p. 106792.
[8] J. Buchan, S.G. MacDonell, and J. Yang, "Effective team onboarding in agile software development: techniques and goals," in *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 2019, pp. 1–11.
[9] A. Ju, H. Sajnani, S. Kelly, and K. Herzig, "A case study of onboarding in software teams: Tasks and strategies," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 2021, pp. 613–623.
[10] R. Britto, D. Smite, L.O. Damm, and J. Börstler, "Evaluating and strategizing the onboarding of software developers in large-scale globally distributed projects," *Journal of Systems and Software*, Vol. 169, 2020, p. 110699.

[11] P. Herath, M.O. Ahmad, and T. Gustavsson, "Onboarding newcomers in large-scale agile projects," in *SAC '26: Proceedings of the 41st ACM/SIGAPP Symposium on Applied Computing*, 2026, pp. 1002 – 1009.
[12] M.O. Ahmad, "Effective handling of large scale agile secure solution development teams," *Journal of Decision Systems*, Vol. 34, No. 1, 2025, p. 2458875.
[13] P. Patanakul, "Managing large-scale is/it projects in the public sector: Problems and causes leading to poor performance," *The Journal of High Technology Management Research*, Vol. 25, No. 1, 2014, pp. 21–35.
[14] M. Bloch, S. Blumberg, and J. Laartz, "Delivering large-scale it projects on time, on budget, and on value," *Harvard Business Review*, Vol. 5, No. 1, 2012, pp. 2–7.
[15] M.O. Ahmad and O. Al-Baik, "Beyond technical debt unravelling organisational debt concept," in *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, 2024, pp. 802–809.
[16] D.A. Tamburri, P. Kruchten, P. Lago, and H.v. Vliet, "Social debt in software engineering: insights from industry," *Journal of Internet Services and Applications*, Vol. 6, No. 1, 2015, p. 10.
[17] T. Gustavsson, M.O. Ahmad, and H. Saeeda, "Job satisfaction at risk: Measuring the role of process debt in agile software development," *Journal of Systems and Software*, Vol. 222, 2025, p. 112350.
[18] M.O. Ahmad and T. Gustavsson, "The pandora's box of social, process, and people debts in software engineering," *Journal of Software: Evolution and Process*, Vol. 36, No. 2, 2024, p. e2516.
[19] T.N. Bauer and B. Erdogan, "Organizational socialization: The effective onboarding of new employees," in *APA Handbook of Industrial and Organizational Psychology, Vol. 3: Maintaining, Expanding, and Contracting the Organization*, S. Zedeck, Ed. American Psychological Association, 2011. [Online]. http://dx.doi.org/10.1037/12171-002
[20] J.E. Van Maanen and E.H. Schein, "Toward a theory of organizational socialization," 1977.
[21] G.R. Jones, "Socialization tactics, self-efficacy, and newcomers' adjustments to organizations," *Academy of Management journal*, Vol. 29, No. 2, 1986, pp. 262–279.
[22] N.B. Moe, V. Stray, and M.R. Goplen, "Studying onboarding in distributed software teams: a case study and guidelines," in *Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering*, 2020, pp. 150–159.
[23] R. Yates, N. Power, and J. Buckley, "Characterizing the transfer of program comprehension in onboarding: an information-push perspective," *Empirical Software Engineering*, Vol. 25, No. 1, 2020, pp. 940–995.
[24] F. Fagerholm, A.S. Guinea, J. Münch, and J. Borenstein, "The role of mentoring and project characteristics for onboarding in open source software projects," in *Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement*, 2014, pp. 1–10.
[25] I. Steinmacher, A.P. Chaves, T.U. Conte, and M.A. Gerosa, "Preliminary empirical identification of barriers faced by newcomers to open source software projects," in *2014 Brazilian Symposium on Software Engineering*. IEEE, 2014, pp. 51–60.
[26] K.J. Stol, V. Stray, N.B. Moe, and T. Sporsem, "Onboarding software professionals in a hybrid world," *IEEE Transactions on Software Engineering*, 2026.
[27] D. Šmite, F. Zieris, and L.O. Damm, "A wave of resignations in the aftermath of remote onboarding," *Journal of Systems and Software*, 2026, p. 112872.
[28] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical software engineering*, Vol. 14, No. 2, 2009, pp. 131–164.
[29] P. Bansal and K. Corley, "The coming of age for qualitative research: Embracing the diversity of qualitative methods," 2011, pp. 233–237.
[30] A. Dubois and L.E. Gadde, ""systematic combining"—a decade later," *Journal of Business Research*, Vol. 67, No. 6, 2014, pp. 1277–1284.
[31] K.M. Eisenhardt and M.E. Graebner, "Theory building from cases: Opportunities and challenges," *Academy of management journal*, Vol. 50, No. 1, 2007, pp. 25–32.

[32] I. Etikan, S.A. Musa, R.S. Alkassim et al., "Comparison of convenience sampling and purposive sampling," *American journal of theoretical and applied statistics*, Vol. 5, No. 1, 2016, pp. 1–4.
[33] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative research in psychology*, Vol. 3, No. 2, 2006, pp. 77–101.
[34] V. Braun and V. Clarke, "Reflecting on reflexive thematic analysis," *Qualitative research in sport, exercise and health*, Vol. 11, No. 4, 2019, pp. 589–597.
[35] H.J. Klein, B. Polin, and K. Leigh Sutton, "Specific onboarding practices for the socialization of new employees," *International Journal of Selection and Assessment*, Vol. 23, No. 3, 2015, pp. 263–283.
[36] A. Bacchelli and C. Bird, "Expectations, outcomes, and challenges of modern code review," in *2013 35th international conference on software engineering (ICSE)*. IEEE, 2013, pp. 712–721.
[37] T. Enberg, S. Alander, and M. Paasivaara, "Onboarding for an agile software development company," in *International Conference on Agile Software Development*. Springer, 2024, pp. 144–162.
[38] M. Fowler, J. Highsmith et al., "The agile manifesto," *Software development*, Vol. 9, No. 8, 2001, pp. 28–35.
[39] A.C. Edmondson, *The fearless organization: Creating psychological safety in the workplace for learning, innovation, and growth*. John Wiley & Sons, 2018.
[40] M.O. Ahmad, "Psychological safety, leadership and non-technical debt in large-scale agile software development," in *2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 2023, pp. 327–334.
[41] M.O. Ahmad, "Strengthening large-scale agile teams: the interplay of high-quality relationships, psychological safety, and learning from failures," *Journal of Software: Evolution and Process*, Vol. 37, No. 1, 2025, p. e2759.
[42] V. Stray, N.B. Moe, and G.R. Bergersen, "Are daily stand-up meetings valuable? a survey of developers in software teams," in *International Conference on Agile Software Development*. Springer, 2017, pp. 274–281.
[43] J.E. Hannay, T. Dybå, E. Arisholm, and D.I. Sjøberg, "The effectiveness of pair programming: A meta-analysis," *Information and software technology*, Vol. 51, No. 7, 2009, pp. 1110–1122.
[44] I. Scaled Agile, "Safe studio," ScaledAgile,Inc., 2025, accessed: 24 February 2026.
[45] Y. Andriyani, R. Hoda, and R. Amor, "Reflection in agile retrospectives," in *International conference on agile software development*. Springer, 2017, pp. 3–19.
[46] K. Conboy, S. Coyle, X. Wang, and M. Pikkarainen, "People over process: key people challenges in agile development," 2011.
[47] E. Tom, A. Aurum, and R. Vidgen, "An exploration of technical debt," *Journal of Systems and Software*, Vol. 86, No. 6, 2013, pp. 1498–1516.
[48] P. Gregory, D.E. Strode, R. AlQaisi, H. Sharp, and L. Barroca, "Onboarding: how newcomers integrate into an agile project team," in *International conference on agile software development*. Springer, 2020, pp. 20–36.
[49] A.F. Rutkowski and C. Saunders, *Emotional and cognitive overload: the dark side of information technology*. Routledge, 2018.
[50] Z. Feng, P. Tandan, I. Steinmacher, M.A. Gerosa, and A. Sarma, "Restructure this: Using ai to restructure onboarding documents to reduce cognitive overload," *arXiv preprint arXiv:2605.19174*, 2026.
[51] M.O. Ahmad and T. Gustavsson, "Nexus between psychological safety and non-technical debt in large-scale agile enterprise resource planning systems development," in *Conference on Practical Aspects of and Solutions for Software Engineering*. Springer, 2023, pp. 63–81.
[52] M. Hartley, "Enhancing employee engagement and retention: the critical role of psychological safety in the onboarding process," *Strategic HR Review*, Vol. 24, No. 3, 2025, pp. 100–105.
[53] R. Britto, D.S. Cruzes, D. Smite, and A. Sablis, "Onboarding software developers and teams in three globally distributed legacy projects: A multi-case study," *Journal of Software: Evolution and Process*, Vol. 30, No. 4, 2018, p. e1921.
[54] Z. Li, P. Avgeriou, and P. Liang, "A systematic mapping study on technical debt and its management," *Journal of systems and software*, Vol. 101, 2015, pp. 193–220.

[55] J.W. Yarbrough and L. Ramos Salazar, "Virtual onboarding and socialization, an exploration of employee discussions and experiences," *Corporate Communications: An International Journal*, Vol. 28, No. 5, 2023, pp. 707–723.
[56] L. Hemphill and A. Begel, "Not seen and not heard: Onboarding challenges in newly virtual teams," Microsoft white Paper, 2016. [Online]. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/Not20Seen20and20Not20Heard20MSR-TR-2011-136.pdf

Appendix A. Interview guide

The interview guide used during the semi-structured interviews is presented below.

Appendix A.1. Background and introduction

- Would you like to tell me a about yourself, your background and your current assignment?
- How long have you been in the industry?
- Could you tell me a little about this software, this product or the system that you're working with?

Appendix A.2. Development practices and quality

- If we start talking about development, do you follow any specific coding standards or quality rules when you develop a system?
- What do you think about this way of having high quality?
- Can you describe how code reviews are conducted within your team?

Appendix A.3. Cross-team collaboration and communication

- What does it look like in your team and the history of your team? How has it changed?
- Do you think that is a good team composition that you have, or would you need other skills in your team?
- What about cooperation with other teams?
- Can you describe any challenges or disagreements that arise within your team or between teams?
- What kind of dependencies do you have, and how do you coordinate stuff with other teams?
- What channels do you use for communication, or do you have rules on that?
- What about knowledge sharing to get to learn from other teams?

Appendix A.4. Onboarding

- What does it look like with onboarding when you get new people into the team?
- When you do this onboarding, do you have any formal process behind that, or is it more informal?
- When a new member joins, how do you get them across the domain and technical aspects?

This article has been accepted for publication in e-Informatica Software Engineering Journal (EISEJ).
This is the author's version wich has not been fully edited and content may change prior to final publication.
Citation information: DOI 10.37190/e-Inf260110

- Did you get help from the client side or other teams during the onboarding process, or was it primarily from your colleagues?
- How much do a newcomer's personality and team fit play into your decision when hiring a new team member?
- Can you describe any challenge you experienced during the onboarding process?
- What improvements will you suggest for the current onboarding process?

Appendix A.5. Agile ceremonies and practices

- What about other ceremonies or meeting structures that you do?
- Do you have sprint planning? standup meetings, demos or any retrospectives?
- Do you think the ceremonies work well, or is there something that should be changed in the way you do these?
- Did you get any kind of formal training related to Agile?

Appendix A.6. Organizational context

- What are the different roles supporting development activities?
- How are these different roles integrated into the team?
- What roles exist outside the development team, and how they contribute to coordination across teams?
- If you look at the rest of the organization, is there anything that's hindering you or stopping you from doing a good job?

Appendix A.7. Closing

- If you could change one thing about the way your organization works, what would it be?
- Is there anything else you would like to add?

Authors and affiliations

Pavithra Herath
e-mail: pavithra.herath@kau.se
ORCID: https://orcid.org/0009-0007-3510-1433
Department of Computer Science, Karlstad University, Sweden

Muhammad Ovais Ahmad
e-mail: ovais.ahmad@kau.se
ORCID: https://orcid.org/0000-0002-7885-0369
Department of Computer Science, Karlstad University, Sweden

Tomas Gustavsson
e-mail: tomas.gustavsson@kau.se
ORCID: https://orcid.org/0000-0002-1512-6592
Karlstad Business School, Karlstad University,Sweden