The Journal of Systems & Software 190 (2022) 111345

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Maximizing integrative learning in software development teams: A systematic review of key drivers and future research agenda[☆]

Anju Mehta [a], Nikhil Mehta [b,*], Ishaan Bindal [c]

[a] North Carolina A&T State University, Willie A Deese College of Business and Economics, Greensboro, NC 27410, United States of America
[b] University of North Carolina Greensboro, Bryan School of Business and Economics, 516 Stirling St, Greensboro, NC 27412, United States of America
[c] Solvay Brussels School of Economics and Management, Universite Libre de Bruxelles, Campus du Solbosch, Avenue F.D. Roosevelt 42, B-1050 Brussels, Belgium

ARTICLE INFO

Article history:
Received 6 July 2021
Received in revised form 16 April 2022
Accepted 19 April 2022
Available online 26 April 2022

Keywords:
Team integrative learning
Software development teams
Social capital
Boundary spanning
Team design
Literature review

ABSTRACT

Software development is a complex phenomenon that requires software teams to integrate diverse knowledge and expertise to provide innovative solutions. A critical success factor for software teams is their integrative learning capability which is driven by multiple factors. We systematically review and synthesize extant literature to identify critical antecedents of integrative learning in software teams and uncover knowledge gaps to inform future research. Searching multiple databases, 447 papers were identified, of which 32 were selected for the final analysis after a rigorous data extraction process. We performed open and axial coding and affinity diagramming to thematically analyze each study for antecedents associated with integrative learning. We also triangulated our results by content analyzing the studies using a software-based text analysis tool. The findings indicate five crucial drivers of integrative learning in software teams: *social harmony*, *process agility*, *team design*, *technological maturity*, and *project environment*. The study also highlights the value of recognizing the inter-team and intra-team contexts in team learning. Based on the study findings, we propose a model of integrative learning in software teams, discuss future research agendas, and offer practical insights and recommendations to software professionals.

1. Introduction

A firm's ability to adapt quickly to the contextual uncertainties and to innovate depends on its potential to create new knowledge — commonly referred to as the organizational learning capability (Migdadi, 2019). The fundamental premise of organizational learning is embedded in the notion of collective intelligence, which ensues when individual learning is acquired, shared, and transferred through technological, procedural, and social conduits within an organization. When this collective knowledge is preserved over time through organizational systems and culture, organizational learning occurs.

Organizational learning is critical for knowledge-based entities such as software organizations, which rely extensively on their intellectual capital for software development - a highly complex, socio-technical activity (Clarke et al., 2016; Khalil and Khalil, 2019). Software development requires people to interact, share and integrate information and ideas, and use technology and processes to achieve project goals. These activities are often accomplished through software development teams (SDTs)[1] - the primary social conduits for integrating individual knowledge in organizations (Dixon, 2017).

Software teams typically lack necessary intellectual resources and must create new team-level capital by integrating individual knowledge (Newell et al., 2004; Tiwana, 2004), termed *integrative learning*. Prior literature uses multiple terms to reflect learning at the team level, such as team learning (Edmondson et al., 2007), cooperative learning (Felder and Brent, 2007), and collaborative learning (Laal and Ghodsi, 2012). Team learning has been defined in terms of the rate of improvement within a team, an outcome of team coordination, and a group process (Edmondson et al., 2007). Aligning with Edmondson et al. (2007), we define integrative learning as a team-level knowledge-creation process that results in a new, adaptive response or behavior (such as developing a new solution or adopting a new process).

Integrative learning involves creatively assimilating a team's internal knowledge resources (the skills, knowledge, and abilities of individual team members) and enriching it by selectively

☆ Editor: Kelly Blincoe.

* Corresponding author.
E-mail addresses: amehta@ncat.edu (A. Mehta), n_mehta@uncg.edu (N. Mehta), ishaan.bindal@ulb.be (I. Bindal).

[1] The terms 'software team' and 'software development team' are used interchangeably.

https://doi.org/10.1016/j.jss.2022.111345
0164-1212/

absorbing valuable external knowledge resources (competencies available outside the team) (Faraj and Sproull, 2000). This diverse knowledge is a software team's most valuable resource, as opposed to other resources such as technology, budget, and space (Faraj and Sproull, 2000). Thus, integrative learning is a collective process that requires team members to combine unique knowledge and skills from both within and outside the team to create new learning (Weinberg, 2015) for positive outcomes such as collective creative efficacy (Cheng and Yang, 2014) and improved team performance (Dingsøyr et al., 2016).

Despite the critical role of integrative learning in a software team's success, there is no consolidated research on the topic to the best of our knowledge. This is a surprising knowledge gap given the critical role of team learning in a software firm's success. A software development firm relies heavily on the learning capabilities of its software teams, and an understanding of what facilitates team learning could be valuable for such knowledge-based firms. To ascertain the knowledge gaps more precisely, we review the team literature to identify prior review studies on team learning. In the next section, we summarize the related works and discuss study motivation.

2. Related works and study motivation

Appendix A (Table A.1) presents details of prior review studies on team learning. We found 16 literature reviews on team learning. A majority of these studies (68%) were conducted in the nursing, medical, and healthcare education domains. For example, Alberti et al. (2021) conducted a systematic review on the effectiveness of team-based learning in achieving learning outcomes in nursing students. Similarly, other reviews focused on team learning as a pedagogical tool for improving student performance (Lang et al., 2019), learning outcomes (Burgess et al., 2014; Reimschisel et al., 2017), engagement and satisfaction (Dearnley et al., 2018; Sisk, 2011), and nursing knowledge, skill performance, and clinical competence (Zhang and Cui, 2018).

Only 3 review studies out of the 14 focused on team learning in work teams. For example, Widmann et al. (2016) reviewed the impact of team learning behaviors on innovation in work teams and concluded that learning and innovation are interdependent. Similarly, Hannes et al. (2013) conducted a review to understand team learning in work or vocational settings. The authors highlighted the role of several factors such as boundary-crossing, communication, and knowledge sharing in team learning, however, the review included qualitative studies only, published till 2011 (Hannes et al., 2013). Fu and Hwang (2018) reviewed the literature on mobile technology-supported collaborative learning and reported an upward trend for the same.

We found only 2 related works on software development teams, and both focused on knowledge sharing. Ghobadi (2015) identified task- and technology-related drivers of knowledge sharing in a review focused on software teams. Similarly, Habeh et al. (2021) reviewed the literature on knowledge-sharing challenges and the suggested solutions in the context of software development teams. The authors identified four broad categoried of challenged and solutions, namely, individual, technology, project, and organization (Habeh et al., 2021).

Thus, although systematic reviews on team learning exist, most of these examine team learning as a pedagogical tool in nursing, medical, or healthcare education. The two studies that do focus on software teams, examine knowledge sharing and not integrative learning (Ghobadi, 2015; Habeh et al., 2021). Moreover, Ghobadi's (2015) review does not include studies published after 2011 and Habeh et al. (2021) reviewed only 9 studies published between 2013 and 2018.

These knowledge gaps highlight the need for a rigorous literature review of team learning literature in the software development context for a deeper understanding of the phenomenon. Also, existing literature on software team learning is scattered and sometimes contradictory. For example, the role of IT use in integrated learning in software teams is still ambiguous. A systematic review can help consolidate such research to identify distinct patterns of relationships among variables. Finally, there is no established framework of integrated learning in software development teams to guide future research and practice. A conceptual framework based on literature review can serve as an anchor for future research and provide guidelines to software professionals on how to maximize integrative learning in teams. This study aims to fill these gaps by conducting a systematic literature review that summarizes the current literature, identifies critical drivers of integrative learning in software teams and proposes a framework. Thus, broadly, we focus on the state of research on variables associated with integrative learning in software development teams and identify areas for future research.

We focus on software teams for three reasons. Firstly, software development is a complex, knowledge-intensive activity that requires integrating a vast array of expertise to effectively design, construct, test, implement, and maintain software solutions (Otero, 2012). The introduction of agile development methodology has added further layers of complexity. As such, combining individually held knowledge for team-level - learning is critical to successful software development. A deeper understanding of integrative learning in software teams would benefit project leaders and advance research. Secondly, integrative learning is likely context-specific in that the drivers of integrative learning could vary across different teams. For example, team diversity may not be as crucial for functional and production teams as it may be for a software team. Software teams are unique in the number of activities they perform, the degree of flexibility and inter-team and intrateam collaboration they require, and the project uncertainties they face in achieving their goals. Therefore, a review focused on software teams is prudent. Thirdly, such an examination also fills a gap in the existing information systems literature as no previous reviews on the topic exist to the best of our knowledge. Overall, this review contributes to the information systems, project management, and organizational behavior domains since software teams use information technology (IT) and systems, human capital, and social and cognitive processes to accomplish project goals.

We describe how the literature review was conducted in Section 3, which highlights the data extraction process, the coding protocol, and the triangulation of results. In Section 4, we present the results. Our findings are presented and discussed in Sections 5 and 6. The implications of this review for research and practice are underlined in Section 7, followed by study limitations and conclusions in Sections 8 and 9.

3. Methodology

Given the limited number of discussion studies on the relationships of interest, meta-analysis was not feasible. Therefore, we conducted a systematic review, which follows a specific, step-wise process recommended by previous researchers (Cooper et al., 2019). A systematic review allows for a comprehensive evaluation of different types of studies and an assessment of reliability and validity.

In adopting the systematic review process and data extraction and reporting, we paid special attention to the following aspects to ensure study quality, reliability, and validity.

(1) Inclusion and exclusion criteria: Peer-reviewed studies that identified drivers of integrated learning or team learning in software development teams were included in this review. Both qualitative and quantitative studies were included from any country. The search was not restricted to any time frame. Studies were excluded if they were not available in English or if they were not conducted within a software development context. Studies on individual-level and organizational-level learning were also excluded.
(2) Search strategy: Keywords were identified through a scoping review of the literature and were grouped using the 'OR' Boolean term. The search strategy was reviewed by two researchers experienced in conducting a systematic review.
(3) Information sources: Electronic databases were searched between 3 June 2021 to 8 June 2021. An updated search was conducted on 2 December 2021. The electronic databases searched were Scopus, ProQuest, Google Scholar, IEEE Explore, and Web of Science, which were selected based on prior research. For example, Scopus was selected since it is the largest abstract and citation scientific database of peer-reviewed research (Falagas et al., 2008), and prior systematic reviews in the software domain have used it as a major data source (Kitchenham and Charters, 2007). A forward and backward search or snowballing was also conducted, which is recommended for systematic reviews to ensure that any relevant studies are not excluded from the sample (Zhou et al., 2016; Wohlin, 2014). We conducted the searches after the final set of 32 papers had been shortlisted. For backward search, we manually reviewed the reference lists of the 32 shortlisted papers. For forward search, we looked at those papers citing the paper being examined using Google Scholar. We followed the same exclusion and inclusion criteria to identify relevant papers. However, two researchers independently conducted one search each.
(4) Study screening and quality: Two reviewers independently reviewed the abstracts and full-text, peer-reviewed studies. Inter-rater reliabilities were calculated to ensure reliability and validity. If there was any disagreement or ambiguity, a third reviewer evaluated the study, and the consensus was reached on inclusion or exclusion through discussion or majority view.
(5) Data synthesis: We established clear protocols and criteria for coding the studies. Studies were analyzed using open coding, axial coding, and affinity diagramming. The full texts of the studies were reviewed to identify factors associated with integrative learning in software teams, the factors were organized into logical categories, and categories were further synthesized under the relevant themes. The studies were also analyzed in terms of the methodology adopted, keywords, main objectives, and major conclusions.

The details of the steps in the systematic review process are discussed in the following sections and presented in Table 1.

3.1. Planning and organizing

The planning and organizing phase entails setting review goals and determining how these will be achieved. This step also involves setting up applicable review parameters such as the population, interventions, comparison conditions, outcomes, and study design (PICOS). This study focuses on antecedents of software teams' integrative learning, and the study design is a systematic literature review. We focused on the knowledge creation aspect of knowledge management only, characterized by team-level knowledge integration and learning. Other knowledge management processes, such as knowledge sharing, were not included since we identified a substantial knowledge gap concerning knowledge creation and team learning only.

3.1.1. Develop search terms relevant to the research question

After finalizing the review parameters, we reviewed existing literature to identify search terms pertinent to the research objective. The search revealed that multiple terms exist to connote integrative learning and software teams. We used all the terms that reflected the integration of individual knowledge among team members working on software projects (Table 1).

3.1.2. Choose databases

Researchers are increasingly relying on online databases for literature reviews in multiple domains (Mariano et al., 2020). Given that software development is a relatively newer domain, searching online databases is deemed appropriate. We used Scopus, ProQuest, Google Scholar, IEEE Explore, and Web of Science databases for locating relevant research papers.

3.1.3. Develop protocols for data search and extraction

We developed search strategies and clear protocols for conducting the search based on established guidelines (Cooper et al., 2019). Advanced search option in online databases was used to restrict the search to the English language, peer-reviewed papers including journal articles, conference proceedings, book chapters, and dissertations. The time frame was not specified. All search terms related to integrated learning and software teams were used with the Boolean "or" operator. These two categories of terms were searched with the Boolean "and" operator to obtain all possible studies related to the topic. We excluded publications with keywords related to teaching, education, curricula, e-learning, and students. Additionally, we restricted the search to disciplines such as business, management, computers, and social sciences and excluded domains such as life sciences.

3.2. Implementation: Data extraction, coding, and selection

The implementation stage involved a step-wise data search based on a pre-determined search protocol and extraction of relevant papers. Standardized information on publications was collected, and data were exported to reference manager software RefWorks.

3.2.1. Identification

The initial search was conducted in the Scopus database using the 'Title–Abs–Key' function. This search returned 285 studies. This search was further refined to exclude non-English language studies and studies related to the education domain resulting in 155 studies. In Proquest, 59 studies were identified using the search terms. In Web of Science, 109 studies were identified. In Google Scholar, since the title–abstract–keyword functionality is unavailable, we used a combination of terms such as "knowledge integration" and "software development team," and "software team" and "integrative learning." A total of 97 studies were obtained using Google Scholar. The IEEE Explore yielded 27 papers. Thus, the initial search returned 447 studies.

3.2.2. Screening: Exclusion decision based on title, keywords, and duplication

After removing the duplicates, 236 studies remained, which were filtered based on title and keywords. Two researchers reviewed the titles and keywords and excluded the papers that were not team-related and/or that did not examine learning. For example, studies that did not pertain to team-level learning were excluded. This narrowed down the sample to 159 studies.

Table 1
Summary of the systematic literature review process.

Step	Action	Description
Step 1: Planning and organizing	Develop search terms relevant to the research question by reviewing the extant literature	*Integrative learning, group learning, collaborative learning, cooperative learning, collective learning, team learning, knowledge integration, knowledge assimilation, software team, software development team, software development, software project team, software project, software engineering, project team*
	Choose databases/literature sources	Scopus, ProQuest, Google Scholar, Web of Science, and IEEE Explore
	Create search strategies for sources	Scopus: Title, abstract, keyword ProQuest: Title, abstract, keyword Web of Science: Title, abstract, keyword Google Scholar: Keyword search IEEE Explore: Title, abstract, keyword Search for all years
	Determine exclusion criteria	Exclude studies that are not English, not peer-reviewed, not related to software development and teams
	Create coding protocols	Axial coding
Step 2. Implementing	Collect data and extract a final sample	The final sample of 32 studies identified Forward and backward search conducted Shortlisted sample of 32 studies finalized
Step 3. Reporting results	Synthesize data and present findings	Five categories of antecedents identified

3.2.3. Eligibility: Exclusion decision based on abstract

The remaining 159 studies were then screened further based on the abstract. Two researchers independently read the abstracts and assigned each study to the "included" or "excluded" category. The studies that were not related to software teams and team-level learning were excluded. Any disagreements were resolved by discussing the study in question and mutually agreeing to a category. If there was no agreement, the study was excluded. After this step, a total of 71 studies remained.

3.2.4. Inclusion: Final selection based on full text

In the final stage, two researchers independently read and coded the 71 studies based on the inclusion criteria that: (1) the unit of analysis and examination should be a software team; (2) the study should have examined integrative learning in some form as a dependent or mediating variable; (3) the study should not be related to knowledge transfer or knowledge management or processes; and (4) the focus of the study should be team level learning. Researchers resolved any disagreement with mutual discussion. Fig. 1 presents the details of the data extraction process. After this funneling, 32 studies were finally shortlisted. A forward and backward snowballing strategy was used on the final set of studies to ensure that all relevant studies are captured. Two researchers independently conducted the forward and backward search using the exclusion and inclusion criteria followed in extracting the 32 studies. No additional studies were identified. The final sample of 32 studies is presented in Table B.1 (Appendix B).

3.2.5. Ensuring methodological rigor (study quality, reliability, and validity)

We ensured methodological rigor by carefully considering study quality, reliability and validity. Only peer-reviewed publications from established databases and dissertations available via ProQuest were included to address study quality (Valentine, 2019). The reliability was ensured by following well-defined search and selection protocols and using two independent researchers to review and code studies. The inter-rater reliability for study eligibility was .89 and for final inclusion, it was .90. These values are well above the recommended value of .70 (Belur et al., 2021; Landis and Koch, 1977). Validity concerns related to biased assessment were minimized by using clear selection criteria and reporting all relevant components for each study (Sanderson et al., 2007). For example, to ensure that only the studies that focused on team-level learning are included, we read the full-text articles to determine the eligibility for inclusion. If the level of analysis mentioned in the study was individual or organization, it was not included. To avoid limitations caused by a too specific set of search criteria, we used a set of keywords to build the search queries and performed several test runs. During search, extraction, and inclusion, we ensured that the studies examined a specified population, i.e., software teams (Petticrew and Robert, 2008). Content validity was established by two researchers independently reviewing full-text articles.

3.3. Synthesize data and report findings

3.3.1. Categories identified by manual coding

We developed a classification and coding scheme to extract helpful information from the data. Following the open-coding system, all possible variables associated with team learning were recorded to be as inclusive as possible (Strauss and Corbin, 1997). Each study was reviewed carefully to understand its research objectives and antecedent and outcome variables examined. This step resulted in 71 antecedents or variables derived from open coding of the data.

Next, common categories were identified by two independent researchers through axial coding to consolidate the open codes into related sub-categories of antecedents. This step resulted in 17 sub-categories. A publication could contribute to multiple sub-themes if it examined different types of variables. For example, a study examining the impact of relational capital and IT use on knowledge integration would contribute to both 'social capital' and 'IT-use' sub-categories. Any disagreements in coding were resolved by collectively reviewing the relevant studies again. Next, the sub-themes were re-coded using affinity diagramming to identify the main categories associated with integrative learning. This resulted in five antecedent categories. In the next section, we present the results of the review.

4. Results

4.1. Descriptive statistics of studies

Although we did not time-restrict our database search and kept it open, the distribution of our final set of studies by year reveals that all shortlisted studies were published between 2003

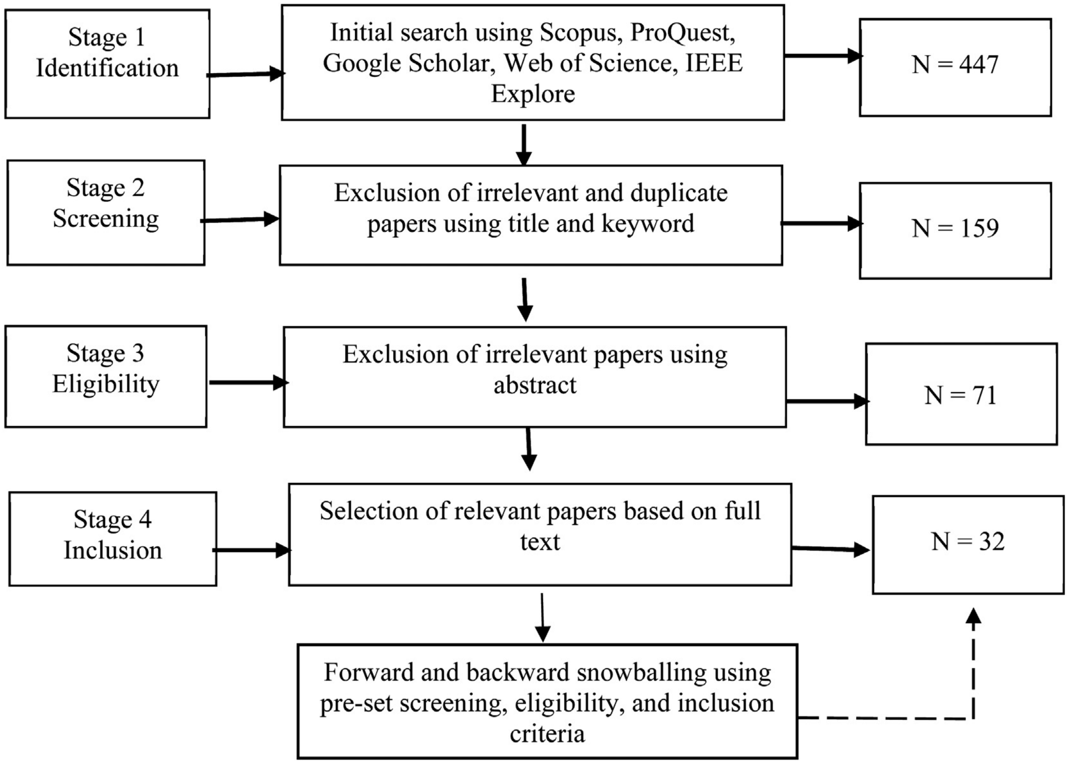

Fig. 1. Data extraction process for study selection.

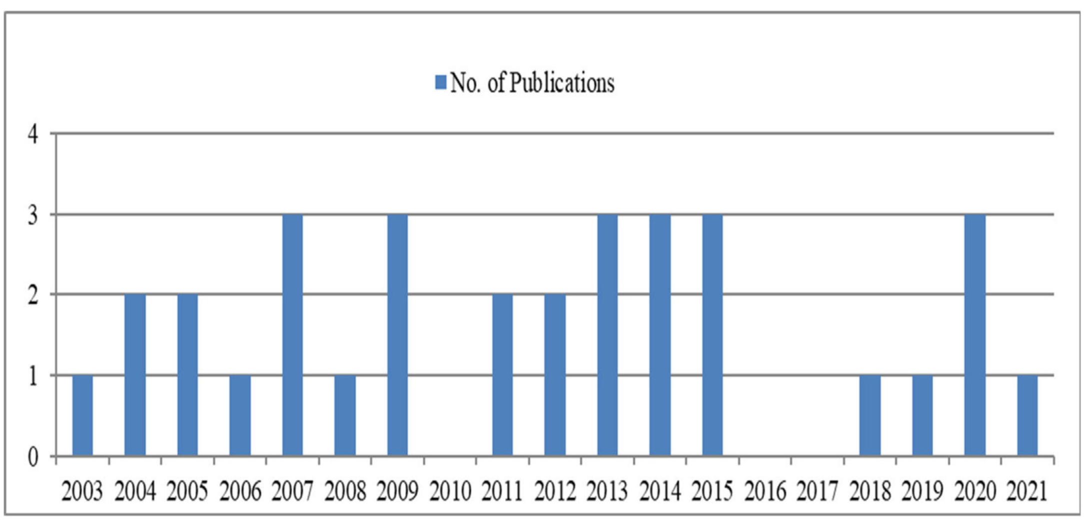

Fig. 2. Year-wise distribution of publications.

and 2021 (Fig. 2). Most publications appeared in information systems (IS) related journals and conferences. In terms of methodology, 14 studies were survey-based, 13 were case studies, one was a review, two were conceptual, and one was an experimental study (Fig. 3).

4.2. Results from the manual coding process

The open coding resulted in 71 factors associated with integrative learning in software teams. Further synthesis of data with axial coding resulted in 17 sub-themes, which were finally clustered into five broad themes, namely people, process, technology, team, and project. Fig. 4 shows the results of the coding process.

Antecedents related to the 'people' category were examined most often, with 75% of studies reporting people-related antecedents. Team-related variables were the second most frequently discussed factors (62.5%), followed by process (31%), technology (19%), and project (15%) related factors. Fig. 5 shows the total number of studies contributing to each category.

4.3. Triangulation of findings using software-based exploratory text analysis

We content-analyzed the studies using Quanteda in R statistical package to triangulate our findings. We created a sparse feature co-occurrence matrix (FCM) measuring co-occurring of

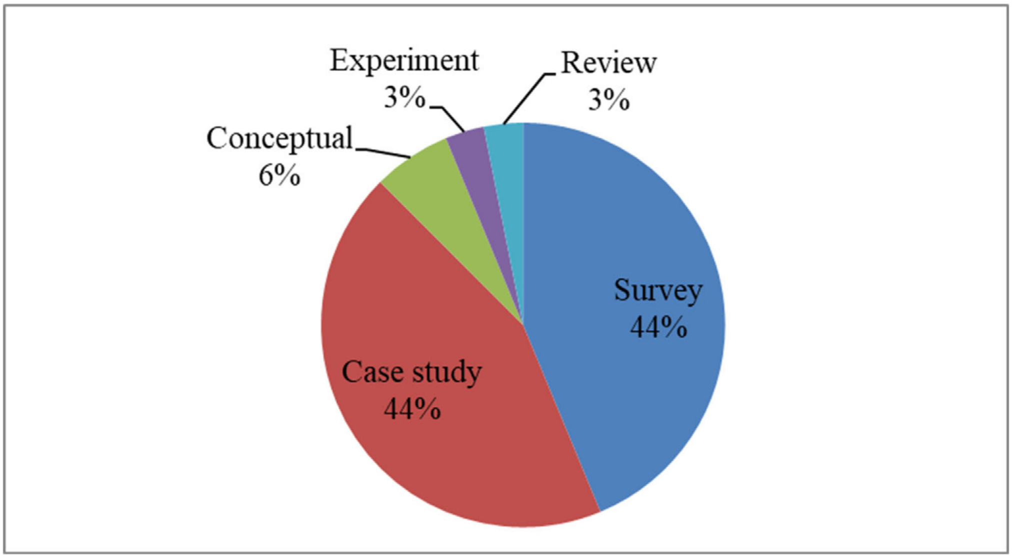

Fig. 3. Classification of studies based on the methodology.

Table 2
Categorization of most frequently occurring words obtained via text-analysis in R.

Classification	Categories	Words obtained from Text-Analysis
Antecedent	People	Sharing, shared, trust, social, support, communication, sharing, network, members, capital, empathy, collective, understanding, interaction, members
	Process	Sentry; guard, process(es), agile, practices, formal, informal, boundaries, lean, quality
	Team	Teams, role, expertise, individual, design, task, work, activities, cognitive, resources
	Technology	Systems, software, digital
	Project	Uncertainty, project, product
Outcome	Integrative learning	Knowledge, learning, exchange, combination, integration, groups, development, creativity, creative, successful
Universal	Nonspecific or generic	Important, future, data, theory, model(s), influence, high construct, review, results, effects, indie, table, IEEE, variance, empirical, variables, research

features. The FCM object was then used to plot the top 80 words (.5 frequency) as a network, where edges show co-occurrences of features (Momtazi et al., 2010). Fig. 6 presents a visual representation of the output.

All the words extracted using the software except generic terms align with the categories identified through manual open-coding, establishing the validity of our findings (Table 2). However, manual coding provides a richer and deeper analysis compared to using a statistical package since contextual and more detailed information can be captured.

5. Study findings

This review identified major factors associated with integrative learning in software teams. Five categories emerged — people, technology, process, team, and project. Of the five categories, we found minimal studies on project attributes, signaling the need for future research in the area. The findings support the traditional people–process–technology framework used across domains such as software development (Haron et al., 2013) and IS adoption (Soja and Soja, 2017). However, our results show that the people–process–technology framework fails to account for all categories of antecedents affecting software team learning. Based on the findings, we propose a *people–process–team–technology* (PPTT) framework for software project development. Fig. 7 presents the proposed PPTT framework. We discuss the results in detail next.

5.1. People — social harmony

The 'people' category includes factors related to the social and relational fabric of the team. As presented in Fig. 2 presented earlier these factors fall under four sub-categories — social exchange, social capital, collaborative teamwork, and group cohesion. Individual-level factors such as team member skills, personality, and expertise were not included in this category. Only the factors related to the functioning of the team as a social entity were included. Research related to the 'people' theme was most popular as nearly 75% of studies reviewed examined social factors. Overall, this category highlights the critical role of the social aspect of teamwork in team learning.

Research in this domain pertains to social capital, including relational, structural, and cognitive capital. Both survey and case-based research have concluded the key role of social capital in software team learning (e.g., Dovey and White, 2005; Mehta and Bharadwaj, 2015). For example, Akgün et al. (2015) reported that high social capital generated commitment from members to act for the benefit of the team and improved team learning. Robert et al. (2008) also showed that all dimensions of social capital: relational, cognitive, and structural, were directly related to team knowledge integration in both face-to-face and digital environments. Mehta et al. (2014) confirmed the positive association of relational capital with knowledge integration processes in software teams. The crucial role of social capital, including

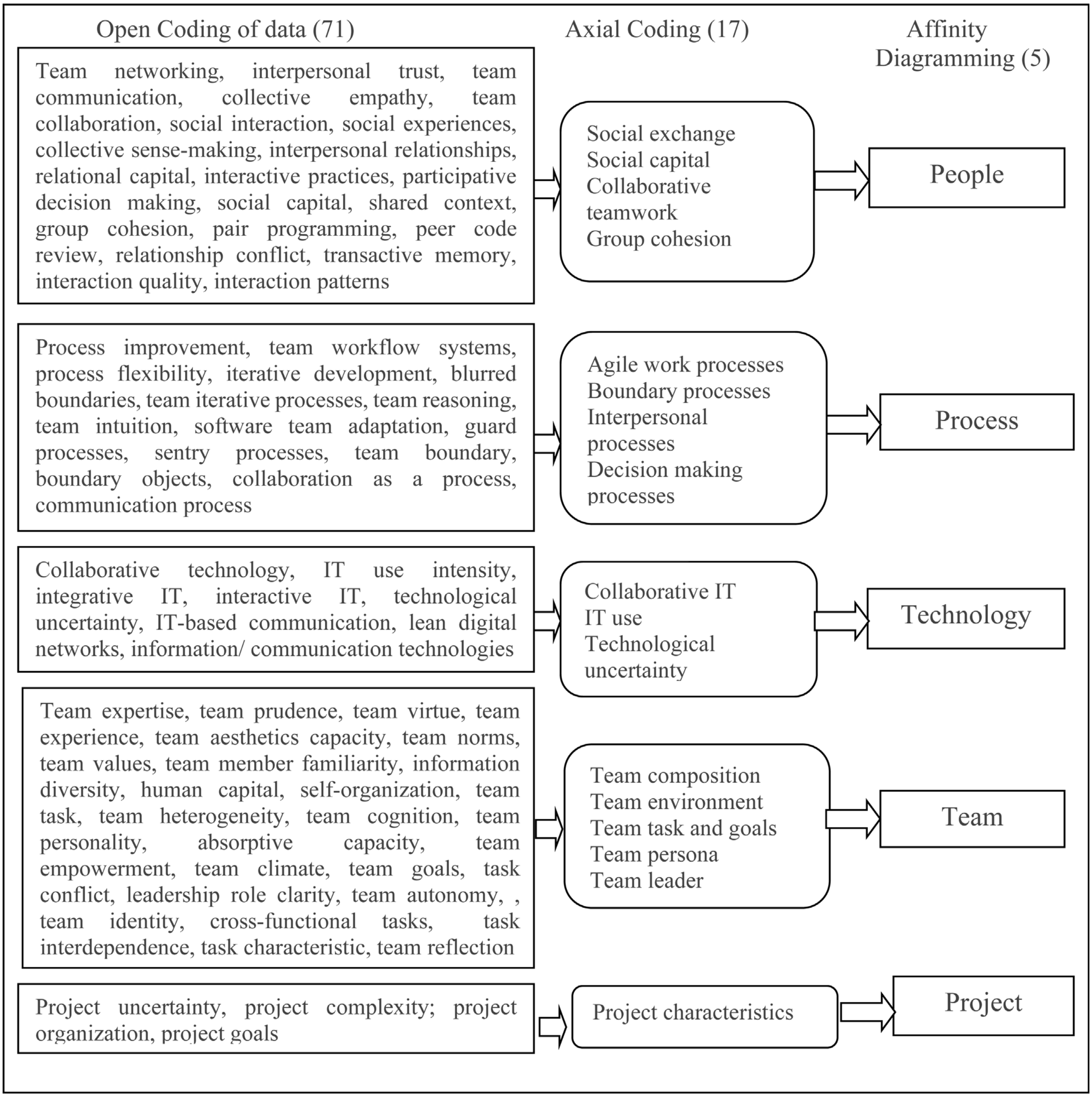

Fig. 4. Main antecedent categories and sub-categories identified.

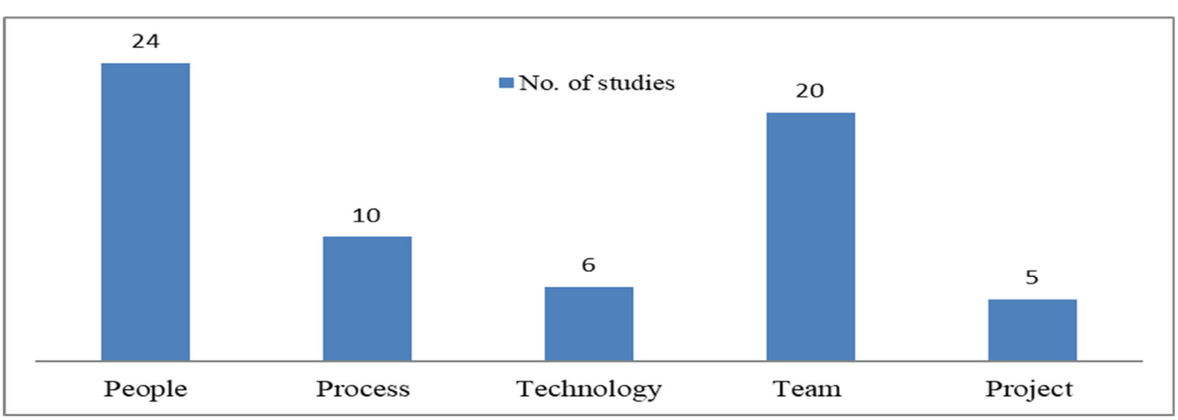

Fig. 5. Number of studies related to each category.

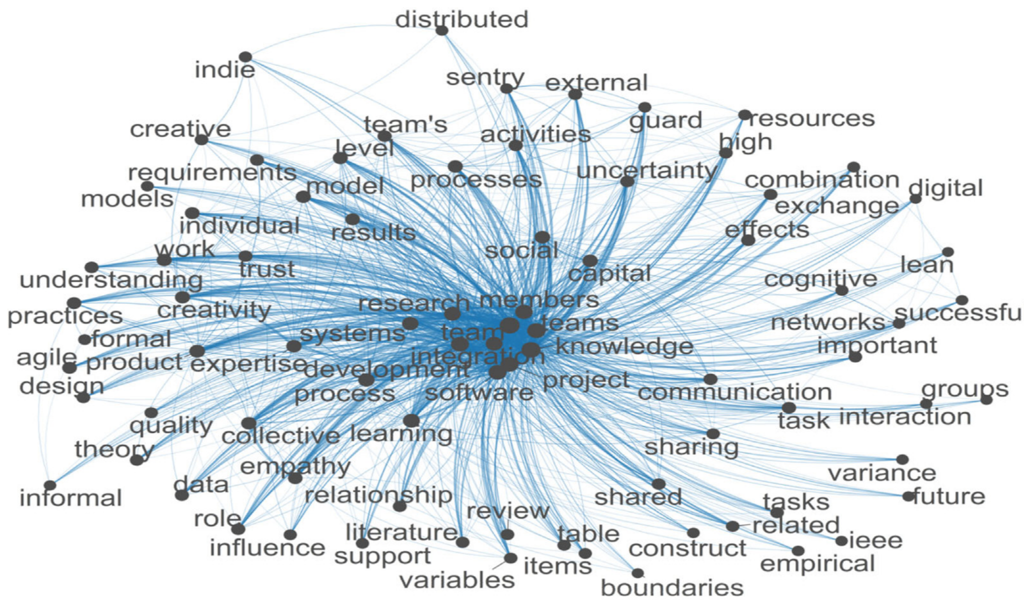

Fig. 6. The feature co-occurrence matrix of top 80 words at .5 frequency.

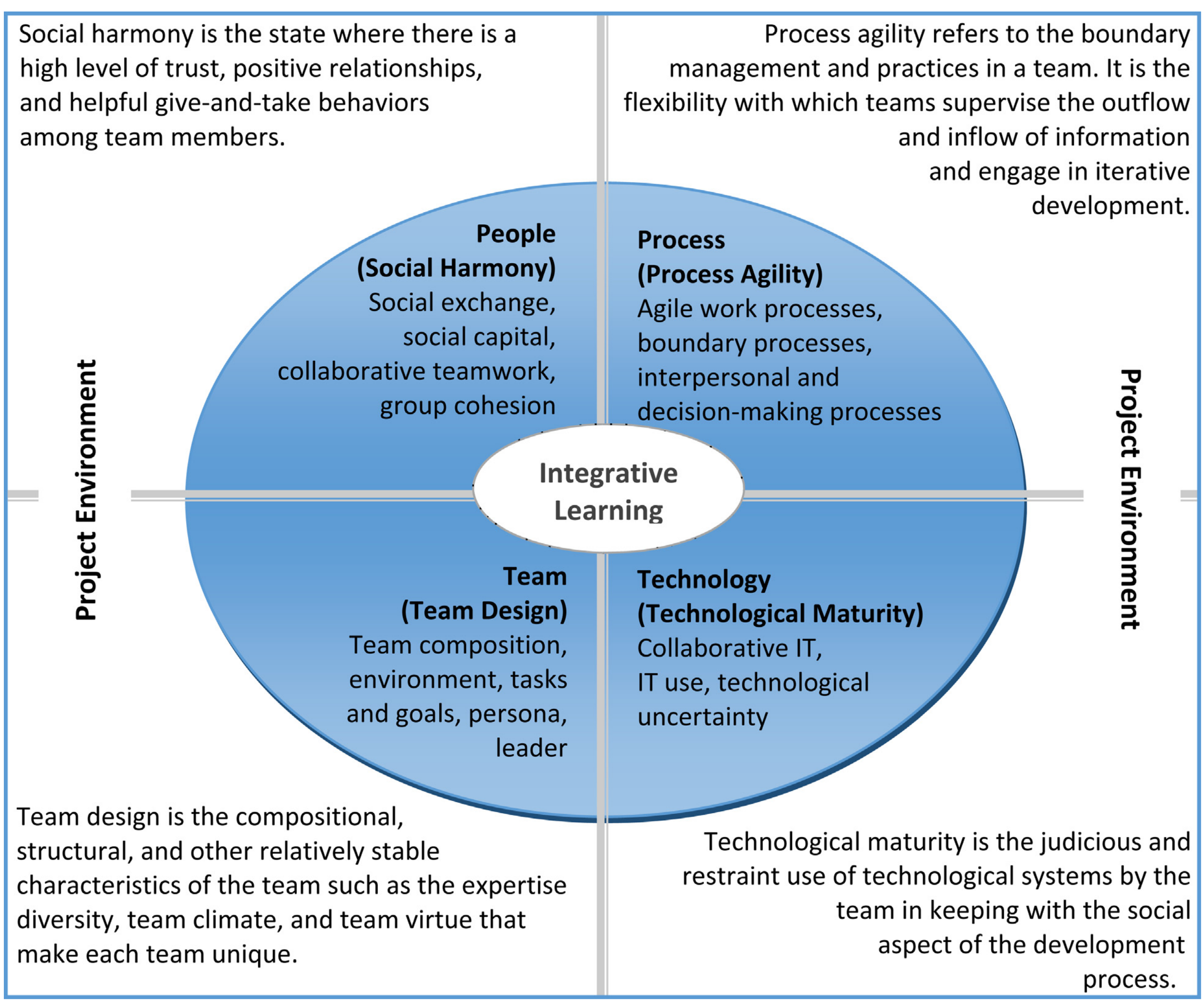

Fig. 7. The People–Process–Team-Technology (PPTT) framework of integrative learning in software development teams.

relational and identity resources, was also underscored by White and Dovey (2004). The authors found that social capital fostered collaborative learning practices through which project-specific knowledge was constructed and re-constructed as the context changed.

Researchers in this category have also examined social exchanges, collaborative practices, and shared cognition. For example, research on social networking shows that it enables shared understanding among team members regarding events, issues, or solutions (Fuhse, 2009), improving the team's joint epistemic actions, and in turn, team learning (Akgün, 2020). Team collaborative practices such as peer code review and pair programming are also shown to improve team learning by collectively enabling "socialization, externalization, combination, and internalization processes" (Spohrer et al., 2013, p. 11). Highlighting the role of collective emotions, Akgün et al. (2015) stressed collective empathy as an essential driver of team learning. However, the authors warned against having too much or too little empathy in a team. Similarly, research has found support for the role of shared memory or group memory (Spohrer et al., 2012), collective sense-making (Fuller, 2019), and shared context (Tiwana et al., 2003) in integrative learning software teams.

Based on the literature, we argue that a team characterized by trust, high social capital, collaboration, and positive social interactions would likely enjoy *social harmony*. The term 'social harmony' embodies a state of togetherness where team members value, express, and promote trust, admiration, and respect for each other and experience positive social exchanges, high participation, and close collaborations. The concept of social harmony extends beyond social capital. It encompasses a dynamic state where the teams display "balance, alignment, mutual support, and flourishing" and team members feel safe and valued (Ip, 2014, p. 730). For software teams, it is an intrateam social dynamic that has a bearing on integrative learning. The notion of "harmony" enacts the belief that creating a positive foundation of well-being, trust, and helpful reciprocal behaviors increases team members' willingness to integrate knowledge. For example, the trust would make developers less apprehensive of each other's opportunistic behavior, and they would openly share their unique ideas and expertise to improve team learning (Akgün et al., 2015).

Similarly, close interpersonal relationships would improve the quality of interactions and the likelihood of knowledge exchange and integration (Patnayakuni et al., 2007). Prior research also demonstrates the role of network tie strength in improving knowledge transactions (e.g., Yuan et al., 2010). This indicates that relational dynamics generated by social harmony likely strengthen the social network ties, resulting in improved knowledge exchange, problem-solving, and learning (De Montjoye et al., 2014; Dokko et al., 2014; Yuan et al., 2010). These intellectual synergies, driven by social harmony, would promote team learning. Social harmony would serve as the necessary foundation for other factors too. The group processes, utilization of technology for software development tasks, and team design dynamics can only be effective if these are embedded in a positive social context. Thus, this study affirms the critical role of people-related factors in facilitating software team learning across traditional and agile development.

5.2. Process — process agility

The 'process' category of the PPTT framework includes factors related to *how* the outcomes would be achieved. It includes formal and informal processes as well as various inter-team and intrateam processes critical to integrative learning. As presented in Fig. 4, prior literature on process-related drivers of integrative learning has focused mainly on four types of processes namely, boundary-spanning processes, agile work processes, interpersonal processes, and decision-making processes. About 31% of studies included in this review examined 'process' factors, signifying the importance of these processes in team learning.

An important field of inquiry in this domain is team boundary spanning (Akgün, 2020; Mehta and Bharadwaj, 2015; Fuller, 2019; Patnayakuni et al., 2007; Richter et al., 2006). Developing boundary-spanning processes to manage the inflow and outflow of knowledge to facilitate integrative learning is critical to software teams (Akgün, 2020; Mehta and Bharadwaj, 2015). The concept of boundary also indicates that a porous boundary that allows a seamless flow of information and resources between teams may not always be beneficial to the team's integrative learning (Mehta and Bharadwaj, 2015). Experienced teams pay explicit attention to this belief by enacting boundary-management processes, such as sentry and guard, to control the inflow and outflow of its resources.

Given the predominantly transactional and iterative nature of boundary spanning (Akgün, 2020), teams need to display *process agility* for optimal benefit. For example, sentry processes are best performed moderately (Mehta and Bharadwaj, 2015). A low level of sentry processes allows too much external inflow, causing more confusion than learning. On the other hand, teams performing a high level of sentry fail to integrate valuable external learning and overprotect their members from improvement-oriented external feedback. For guard processes, keeping them high initially to consciously develop social harmony and team identity may be helpful. Once that happens, the team may deescalate guard processes and perform them selectively to ensure optimal utilization of its resources.

Other studies have emphasized the importance of boundary objects (Patnayakuni et al., 2007) and flexible boundaries (Fuller, 2019) both within and between teams. For example, Patnayakuni et al. (2007) call attention to creating boundary objects to enhance knowledge integration. The authors show how formal and informal organizational integrative practices enhance knowledge integration within and across teams. Fuller (2019) also underscores that the ability to blur the boundaries impacts a team's capacity to grok collectively.

With the popularity of agile software development, iterative processes and practices have also featured in research lately. Iterative development is an integral aspect of agile software development and involves breaking down a large application into smaller pieces and repeatedly testing at each stage. In the context of team learning, Kude et al. (2014) reported that software teams that work iteratively could learn faster and develop new responses to other, comparable events compared to less iterative teams. Dissanayake et al. (2013) emphasized that iterative development helps foster continual rethinking and reevaluation, bringing new learning and insights. The authors propose that iterative development facilitates knowledge integration in software teams (Dissanayake et al., 2013). Similarly, de Araujo and Borges (2007) underline the value of creating a 'process improvement environment' to facilitate learning.

Research also shows a salient role of human processes in integrative learning. Taking a process approach, Akgün (2020) demonstrated the impact of team wisdom processes on team reasoning and intuition, which in turn, impacted team learning. The authors reasoned that through the ongoing interpersonal and cooperative processes, teams develop collective 'we' intentions (Colman et al., 2008) that prompt members to help each other interpret new or unclear information, explore new knowledge, and coordinate to complete the tasks (Yu and Petter, 2014). This results in improved team learning. We assigned communication and decision-making to the 'process' category if these were described from a process perspective in the source studies. However, such processes can also be a part of the 'people' category.

For example, if communication ties or quality are mentioned, it would fall under the 'people' category, but it would fall under the 'process' category if the communication hierarchy is mentioned. This is one of the areas where overlaps between categories may exist.

5.3. Technology — technological maturity

The 'technology' category in the PPTT framework includes the technological tools and techniques that the teams employ to accomplish software development goals. Judicious use of IT can facilitate team learning by improving teams' capacity to access and share knowledge resources. Surprisingly, only 6 out of 32 studies focused on IT-related antecedent factors. As presented in Fig. 4 which shows the coding results, the research in this area was related to collaborative and knowledge management (KM) systems, IT use, and technological uncertainty.

Research shows that software teams use two types of technology systems — collaborative and KM systems (Mehta et al., 2006). Collaborative systems, such as mailing lists, expert directories, and electronic discussion forums, help developers interact with their peers (de Araujo and Borges, 2007; Freeman and McNeese, 2019). These technology-enabled interactions serve as vehicles of information and resource-sharing, ultimately influencing integrative learning. de Araujo and Borges (2007) report on the use of collaborative technologies in creating a process improvement environment and in facilitating process learning. The KM systems aim to manage organizational knowledge (Alavi and Leidner, 2001). These systems enable a firm's knowledge creation, storage, transfer, and application capabilities through tools such as knowledge networks, expertise mapping, and knowledge repositories, impacting team and organizational level learning (Alavi and Leidner, 2001). For example, project repositories influence team learning by offering a centralized collection of fragmented knowledge stocks to the team from within and outside the team boundary (Desouza, 2003). Another popular category of IT-based systems identified in the literature is the lean digital networks, which, as a mode of communication, have a moderating effect on knowledge integration (Robert et al., 2008).

Typically, technology is considered essential to organizational success. However, our findings reveal the varied nuances of IT use in software teams. Research shows that technology use in software teams is best exploited when its contingent effects on other capabilities are clearly understood (Mehta et al., 2014, 2006). In different contexts, IT use can have complimentary, substitutive, negative, or no effects on integrative learning. Complimentary effects happen when the use of IT enhances the effects of other capabilities on learning, and substitutive effects refer to a situation when it can be used instead of another antecedent. For example, when a software team is interdependent on other teams for project completion, knowledge integration is already high due to the interdependence, and greater IT use may not result in additional learning. However, if project interdependence is low, IT use may increase a team's integrative learning. Past studies have indicated that IT use could be detrimental or ineffective (Ayyagari et al., 2011; Tarafdar et al., 2015). For example, IT use can create 'technostress' resulting in adverse organizational outcomes such as decreased job satisfaction and commitment (Ragu-Nathan et al., 2008). Similarly, in co-located teams, IT use may not be significantly associated with integrative learning since it is just a means to extend informal conversations (Gupta et al., 2009).

Thus, research shows that IT use does not have an absolute positive impact on integrative learning. Various contingencies impact this relationship. In light of the findings, we argue that software teams must exercise *technological maturity*, representing a team-level understanding that using IT-based systems intelligently accrues compounded benefits. Technological maturity entails that the teams understand the technology and assume that the more the teams know about technology the better they can apply it to suit their needs (Nolte, 2008). Thus, although technology is an integral part of the PPTT framework, its efficacy in improving a software team's integrative learning is dependent upon how wisely the teams exploit it. For software teams, the technology dimension should represent a calculated, strategic use of IT, referred to as a firm's technological maturity.

5.4. Team — team design

The 'team' category in the PPTT framework includes factors related to various aspects of the team. The coding results presented in Fig. 4 show that team-related factors fall under five subcategories namely, team composition, team environment, team task and goals, team persona, and team leader. This category was created since the data revealed several factors that could not be accounted for in the traditional 'people' category, such as team goals, norms, self-organization, etc. Thus, the factors related to social capital and team members' embeddedness in the social system were included in the 'people' category. Those related to team composition, task, goals, environment, leadership, and other emergent dynamics were assigned to the 'team' category. Sixty-two percent of studies in the sample mentioned team-related factors, which is second only to people-related factors. This is an important finding since the 'team' category is missing from the traditional people–process–technology framework, often used in the IS domain (Haron et al., 2013; Soja and Soja, 2017).

One of the most critical team composition variables to contribute to integrative learning is team diversity. Researchers have used a myriad of terms to describe the variety of knowledge inputs, such as 'team diversity' (Akgün, 2020), 'informational diversity' (Liang et al., 2009), 'expertise diversity' (Tiwana and Mclean, 2005), 'human capital' (Zorgios et al., 2009), and 'team heterogeneity' (Mehta and Bharadwaj, 2015). Generally, a diverse team is characterized by members from various educational and functional backgrounds and with different skills and capabilities. Overall, a positive association between team diversity and learning has been reported (Liang et al., 2009; Mehta et al., 2006). However, Tiwana and Mclean (2005) did not find expertise heterogeneity important in expertise integration.

The team environment, team leader, and team task and goals also emerged as significant. Various attributes of team environment facilitate team learning directly or indirectly, especially in agile development and self-organized teams. For example, team autonomy impacts how often members engage in cooperative learning behaviors (Janz and Prasarnphanich, 2009). Similarly, member familiarity indirectly facilitates team learning via its impact on collective empathy (Akgün et al., 2015), and team climate (of transparency and trust) promotes collaborative learning (Gholami and Heinzl, 2013). Team leaders, however, can also create confusion and dissatisfaction if the role and responsibilities of the leader are unclear (Gholami and Heinzl, 2013).

Learning in software teams is also determined by the nature of the task and team goals. Research shows that task characteristics impact team learning by influencing members' motivation to learn (Gholami and Heinzl, 2013). For example, a team may be willing to learn for developmental tasks, but maintenance tasks may not motivate them to integrate knowledge. Similarly, when tasks and goals are assigned without consultation, integrative learning tends to diminish as teams do not take ownership of the set goals, resulting in little commitment to an overall plan (Stray et al., 2011). This lack of commitment may harm integrative learning by hurting the team's cognitive capital and a clear understanding of the team goals (Stray et al., 2011). The impact of team tasks and goals on integrative learning denotes another area of overlap among categories.

When team members with diverse values, beliefs, and identities come together as one social entity, team-level dynamics emerge from the aggregation of individual parts. These team-level phenomena, such as team virtue, team prudence, and team aesthetics capacity, which we have sub-categorized under 'team persona,' influence software team learning. Researchers have highlighted the virtue-ethic aspect of wisdom and its role in motivating team members to connect and collectively act in a reflective manner during projects, thereby promoting team learning (Akgün, 2020; Berente et al., 2011). The authors also demonstrate the role of team prudence and the team's aesthetic capacity in team learning (Akgün, 2020). Similarly, software teams can benefit by creating entities of homogeneous personalities (Anderson et al., 2018). Tiwana and Mclean (2005) highlighted the role of team absorptive capacity in expertise integration.

Therefore, we argue that project leaders must focus on '*team design*' when creating teams to facilitate integrative learning. Team design refers to the team membership or composition, the structure of a team's task, goals, and members' roles, and the creation of organizational support for the team (Carter et al., 2019; Stewart, 2006). It constitutes the relatively stable compositional, structural, and contextual team characteristics. Given that the teams are the fundamental social entities in software firms through which individual knowledge and expertise are integrated, paying attention to team-related factors while formulating and managing software teams would accrue long-term learning gains. Several team compositional variables, such as team diversity and team tasks, can be actively manipulated by project leaders. The results indicate that firms should proactively manage such team design elements to form viable and productive teams.

5.5. Project — project environment

Under the 'project' category, factors associated with project characteristics were classified (Fig. 4). Only three project-related factors were identified: project uncertainty, project complexity, and project organization (Govindaraju and Arianto, 2015; Mehta et al., 2014, 2006). Both direct and interactional effects of project uncertainty on knowledge integration were reported making it a critical aspect of the software development context. The relationship between project complexity and integrative learning was proposed but not tested (Mehta et al., 2006). Given that the project is, by default, a starting point for all software teams and that project characteristics impact how teams use technology and processes to accomplish project goals, the lack of studies in this area is surprising. We argue that there is a need to understand the *project environment* within which the software teams are embedded to maximize integrative learning. The project environment will impact all other dimensions of the PPTT framework and directly or indirectly influence integrative learning.

6. Discussion

The results, including sample descriptive statistics and thematic analysis, present valuable insights. As evident from the descriptive statistics, all the studies that were included in the sample were published over the last decade. This spurt of research attests to the growing acceptance of team learning as a critical team process in the software development domain. Another interesting aspect is the almost equal concentration of survey-based and case-based studies in the sample. This varied and rich sample allowed us to understand the role of various drivers of team learning more deeply. Although a varied sample in terms of methodology may undermine conclusion validity, we were able to minimize these concerns by having two researchers independently review and code the studies and calculating inter-rater reliabilities.

The fact that people-related factors were most often examined in our sample reiterates the importance of human resources and the social ecosystem in software firm success. We found that software teams that engaged in open communication, shared collaborative practices, developed trusting relationships, and had collective empathy were more successful at integrating learning. Of all the people-related factors, the relationships and interactions matter the most signifying that even for the teams engaged in complex, technical work such as software development, a robust social fabric characterized by social harmony is critical to success.

The second category that was almost as frequently mentioned as people was the team. This is a significant finding since the traditional people–process–technology framework adopted often in the IS domain (Haron et al., 2013) does not account for this key dimension. A broad set of team-related factors such as its composition, goals, tasks, and climate impact integrative learning. The notion that diverse teams integrate more knowledge was confirmed. More importantly, the results highlighted emerging research on the role of factors such as team virtue, prudence, and empathy in integrative learning. Also, developmental tasks seem to facilitate integrative learning more than maintenance tasks. These team-related factors provide a novel 'team-design' perspective for managing and improving integrative learning in software teams.

Our findings on process-related factors indicate the importance of regulating team boundaries, especially the inflow and outflow of knowledge, using boundary processes. Teams that engaged actively in boundary-spanning activities and were flexible were more successful in integrating knowledge. Iterative processes also contributed to team learning. Another salient finding under the process category was the role of human processes in integrative learning. The process category is generally focused on task-related processes such as operations, boundary-spanning, or technological processes. However, our review identifies human processes such as team wisdom processes and decision-making processes as critical to the teams' reasoning ability and intuition, which impact team learning. One common theme underlying this category was the need for flexibility and agility.

Concerning technology, our findings indicate a complex relationship between the use of technology and integrative learning. Although technology use is encouraged and is essential to the functioning of a software development team, under certain situations, less may be more. Overuse of technology can not only create techno-stress but can also impede integrative learning if it comes in the way of the relational capital of the team. Similarly, beyond a certain point, using more IT may not increase integrative learning, as in the case of highly interdependent projects. Thus, to optimize integrative learning, software teams need to carefully think about their IT-use strategy.

Among all the categories, project-related factors were mentioned the least in our sample. This could indicate an unintended knowledge gap in the literature that needs to be filled or a signal that project characteristics are not considered as important to integrative learning as other factors. However, our findings indicate that a software team's ability to integrate knowledge is challenged when the degree of project uncertainty and complexity is high. Understanding project-related factors could be an important starting point for software teams to develop appropriate work strategies, processes, and team composition.

Thus, the findings of our review show that to facilitate integrative learning, software teams have to achieve a unique balance in terms of people, process, team, technology, and project factors. While a strong social foundation and process flexibility are a must, the use of technology and team designing has to be prudent, based on project characteristics.

7. Implications for research and practice

7.1. Implications for research

This study presents the current state of research on integrated learning in software teams. In doing so, we also highlight the gaps in the existing literature that can be addressed by future research. Substantiating the strong sentiment regarding the role of the social aspect in business, 'people-related' factors were found to be most examined. However, opportunities for future investigation exist. For example, we found that although social capital has been extensively researched across multiple domains, the distinct role of different dimensions of social capital and their inter-linkages in facilitating integrative learning in software teams remains understudied. These findings reflect the need to examine social capital in software teams more deeply. Similarly, the review shows that the role of team-member exchange and leader–member exchange, especially in a digital environment, is not clear. Based on these findings, future researchers can examine these social processes and their impact on integrative learning in virtual teams. Our results also indicate that trust, which is a key aspect of relational capital, impacts team learning by fostering psychological safety. However, the existing literature does not establish how psychological safety develops in face-to-face and digital environments.

Additionally, concrescent conversations, composed of disclosing, questioning, and steering verbal behaviors, are also associated with team psychological safety (Akan et al., 2020). In agile software development, concrescent conversations may improve integrative learning and iterative delivery through psychological safety. Finally, research is emerging on the role of team members' interactive style in the software development process (Rangel et al., 2017). The interactive styles would likely influence knowledge sharing and integration also. However, we did not find existing research that has explored this relationship. These knowledge gaps identified in this study present rich avenues for future research.

Among 'team-related' factors, one area that can benefit from future inquiry is team leadership. For example, our findings show that given the need for continuous adaptation and iterative delivery in agile software development (Fogelströtom et al., 2010), the role of a project leader is likely complex and demanding. With geographically distributed software teams, project leaders must also navigate additional issues such as task coordination and knowledge sharing (Šmite et al., 2017), affecting integrative learning. However, this review reveals that research examining team leaders' direct and indirect effects on integrative learning in software teams is scant and needs further inquiry. Another team-related theme worth pursuing is team virtue and ethics. We found that team-level phenomena such as team virtue motivate team members to connect and act collectively during projects, promoting team learning (Akgün, 2020; Berente et al., 2011). However, gaps still exist on how these perceptions about team virtue and ethics develop and impact team outcomes. The notion that team members are influenced by the degree of virtue perceived in the team has not been examined either (Wang et al., 2017, 2018). Also, although task characteristics are shown to influence software team outcomes, only a few scholars (see Huckman et al., 2009) have examined the characteristics of software development tasks in relation to team learning and performance. Similarly, we found studies that examined the role of participative goal setting on integrative learning (Stray et al., 2011), however, understanding how different types of goals benefit team learning and performance in different team configurations and structures remains unexplored.

Our findings demonstrate that existing research on the 'process' category focuses on iterative work processes; however, empirical research is warranted in this area as most of the work is conceptual or case-based. Also, we found that the boundary-spanning processes and objects have been examined often, but cognitive team processes such as decision-making and problem-solving remain under-investigated. As revealed in our review, project teams have formal and informal decision-making and problem-solving protocols to facilitate iterative delivery and improve response time. For example, a team's decision to keep searching for new knowledge resources for creative solutions or to use existing knowledge may depend on time-related protocols, which could affect its integrative learning. Existing literature does not examine these processes in depth. Further research on the possible interaction of team processes with other factors is also warranted. For example, this study shows that boundary-spanning ties and an individual's social identity have a combined effect on creativity in high-technology firms (Dokko et al., 2014). How IT interacts with these processes to impact team learning needs to be investigated.

One interesting observation was the minimal amount of research on the 'technology' category. Our findings also showed that the role of IT is not simplistic and that higher technology use is not always better. For example, this study showed that IT use in software teams is contingent upon other capabilities (Mehta et al., 2014, 2006) and IT can have complimentary, substitutive, negative, or no effect on integrative learning. Also, different types of IT have different outcome implications. These contingencies that may influence the impact of IT use and the type of IT on team learning can be explored in the future. With multiple formal and informal IT platforms available for sharing knowledge (e.g., social media networks, company intranets, and blogs), it would be interesting to examine which IT platforms would enhance team learning in knowledge-intensive activities such as software development. Additionally, does the efficacy of various IT-based systems in facilitating integrative learning differ for co-located versus virtual teams and for small-scale versus large-scale projects? Future research can help clarify the salient role of IT use in software development projects in addressing such questions.

An area for future research is the 'project-related' factors of integrative learning. Only three project-related factors were identified in this review namely, project uncertainty, project complexity, and project organization (Govindaraju and Arianto, 2015; Mehta et al., 2014, 2006). Of these factors, project complexity has not been examined empirically (Mehta et al., 2006). We also did not find any studies that have examined other project-related antecedents of integrative learning such as size and duration. This knowledge gap is surprising given that project characteristics are tied to team outcomes. For example, project size and interdependence may directly impact knowledge integration, as evidenced by the knowledge management and coordination challenges faced by large-scale agile software teams (Khalil and Khalil, 2019). Similarly, project duration (Liu et al., 2015) and project risk (Wallace et al., 2004) may also impact team outcomes by influencing teams' ability to integrate knowledge. These relationships remain unexplored in software teams.

Finally, the findings can apply to other types of teams engaged in similar, highly complex tasks as software teams, such as the research and development (R&D) teams and new product development teams. For example, the product development teams also engage in knowledge-intensive activities under similar project environments and require the integration of knowledge scattered across multiple stakeholders. The findings and the PPTT framework can be applied to understand what drives team-level learning and outcomes in these teams. For example, prior research indicates that social capital fosters innovation in research and development teams (Gu et al., 2013). Based on this review, we can argue that the effect of social capital on innovation in R&D teams is likely mediated through integrative learning.

7.2. Implications for practitioners

7.2.1. Relationships are not overly rated

First, it is essential to know that social harmony is a necessity, not a luxury. Our findings indicate that social harmony is a necessary foundation for effective group processes and utilization of technology in a team. As evident from our results, close interpersonal relationships among team members improve the quality of interactions and strengthen the social network ties (De Montjoye et al., 2014; Dokko et al., 2014). Similarly, we found that trust facilitates openness and sharing of ideas and expertise to improve team learning (Akgün et al., 2015). Thus, project leaders must invest in openness and trust and strive to create shared mental models. This can be done by developing decentralized social networks, having members work closely together in the early team development stages, and building team functioning procedures. Second, understand that social harmony cannot be replaced with technology. Suppose your team faces social complexity in the form of mistrust, weak interpersonal relations, and knowledge-hoarding behaviors. In that case, its adverse effects on integrative learning cannot be overcome by using technology. In other words, using technology-based tools without a solid social foundation would lead to suboptimal results. In fact, IT use must be selective initially to allow team social harmony to emerge.

7.2.2. Managing team boundaries is as important as managing the team itself

Software professionals must be mindful of the inter-team and intra-team contexts of software development. This study identified boundary-spanning as one of the most critical team processes to be managed and the need for flexibility in managing the boundary processes. Additionally, iterative processes and practices are also highlighted. These findings indicate that team leaders need to achieve a delicate balance concerning their boundary processes — the sentry and guard. It would be wise to project an image of a 'helpful team' by keeping your guard process low to amplify your resources. On the other hand, if your team has the key resources it needs, your team would benefit by protecting its resources. Implementing agile practices with flexible boundaries would also help enable team leaders to improve learning.

7.2.3. IT discipline is beneficial

Although technology use is natural to software development, scaling back the use of technology may be required in certain situations. Our findings indicate that IT use has varying effects based on different contingencies. For example, existing research shows that when teams are co-dependent to achieve performance outcomes, integrative learning is already high due to the very nature of this setting, and higher IT use may not have proportionate gains (Mehta et al., 2014). Also, if your team is experiencing low social capital, relying exclusively on technology may not yield the desired benefits. In fact, it may worsen social dynamics (Schultze and Orlikowski, 2004). Thus, it would be helpful to develop a conservative technology use strategy in such situations. On the other hand, in large-scale projects, where the knowledge is dispersed across different stakeholders, data-mining and recommender system technologies may facilitate collaboration (Castro-Herrera et al., 2009) and enable developers to learn from divergent views.

7.2.4. Crafting an adept team pays dividends

Team designing is one of the few activities that software project leaders can undertake proactively and exercise some control. This study identified several team-related factors critical to integrative learning such as team environment, team tasks and goals, and team diversity. For example, expertise diversity is associated with higher knowledge integration. Similarly, a participative goal-setting process and goal clarity are also shown to improve team learning. These findings indicate that paying attention to team composition during team formulation in terms of educational background, expertise, personality, and team skills would go a long way in team success. Further, team leaders must clarify team goals and tasks early on, provide autonomy and opportunity for self-organization to the teams, and provide an open and transparent climate through positive leadership. Team leaders should also convey team expectations and norms related to the task, work values, and goals to help the team develop a collective understanding and identity.

7.2.5. Don't forget the project

The software team has little control over project attributes. It has to essentially adapt its use of technology, people, processes, and team dynamics to suit project requirements and achieve project goals. This study identified project complexity, unpredictability, and organization as three critical project-related factors impacting integrative learning in software teams. For example, project uncertainty is found to impact team outcomes both directly and indirectly. Therefore, it is critical that team leaders clearly convey project attributes such as complexity, project scale, and expected uncertainty to the team members to manage their resources better and have contingency plans. One area where project leaders may intervene is the project organization since a well-organized project facilitates team learning. Table 3 lists the categories discussed above, their explanation, theoretical foundation, and recommendations for practitioners.

7.2.6. Implications for the software industry

In identifying the critical factors known to improve integrative learning in software teams, this study offers insights for the software industry too. The software industry has undergone a huge transformation in recent decades, especially with the popularity of the internet and cloud computing. Being a knowledge-intensive industry, it relies on the creative inputs and intellectual resources held by software firms. Inversely, the rapid organic development the industry has witnessed necessitates that software firms remain agile and creative. Thus, the software industry benefits in direct proportion to the knowledge creation and team learning that ensue in software firms. By highlighting the factors associated with integrative learning in software teams, this study contributes to both the software firms and the industry.

The study also offers practical recommendations (Table 3) that can serve as the industry benchmarks across multiple segments of the software industry. For example, maintaining technological maturity may be critical to firms in face of Industry 4.0. Information and communication technologies form the backbone of Industry 4.0, which is characterized by the bridging of the physical and digital world through cyber-physical systems embedded in an environment of big data, people, processes, services, systems, and other resources (Dalenogare et al., 2018). However, our research suggests that indiscriminate and excessive use of IT (lack of technological maturity) to create and maintain these cyber-physical systems may not be judicious. For example, the proponents of the emerging Industry 5.0 argue that Industry 4.0 lacks the human touch (Nahavandi, 2019), which must be central to any industrial system.

Table 3
Key themes, their theoretical foundations, and recommendations for practitioners.

Theme and explanation	Theoretical foundation	Recommendations for practitioners
Social harmony Represented by the presence of trust, interpersonal relations, and reciprocal behaviors, social harmony shapes a team's positive social fabric. It creates the groundwork for integrative learning but also reduces the social complexity of SDTs.	The social network theory (Liu et al., 2017) - the "strength" of social relationships among the team members determines information exchange and integration. Social capital theory (Nahapiet and Ghoshal, 1998) - relational, cognitive, and structural capital facilitate team outcomes.	✓ Facilitate the development of trustworthy relationships among team members. ✓ Encourage face-to-face interaction instead of overuse of technology-mediated transactions. ✓ Publicize benefits of knowledge reciprocity. ✓ Increase the visibility of members' reciprocity profile.
Process agility Teams regularly perform boundary spanning processes to streamline the inflow and outflow of information and resources. Additionally, the iterative and participatory processes help the teams adapt and learn.	The input-process-output (IPO) perspective (Cohen and Bailey, 1997) - team inputs and team processes determine team effectiveness.	✓ Actively perform sentry activities. ✓ Assign sentry roles to people with extensive interpersonal contacts. ✓ Assign guard roles with clear expectations. ✓ Clearly demarcate information in public vs. confidential domain. ✓ Know the ramifications of 'guarding' too much!
Technological maturity Teams rely on collaborative systems and KM technologies for integrative learning. Contrary to conventional wisdom, reckless use of these systems could sometimes impede a team's learning. The contextual and moderating nuances of IT use must be understood to deploy it effectively.	The socio-technical systems theory (Appelbaum, 1997) - a work unit is a blend of social and technical elements embedded within an environment, and technology must be implemented considering this social aspect.	✓ Improve awareness of IT-based systems available to the team and enhance the usage of KM systems. ✓ Limit excessive use of IT in intrateam settings to support social harmony. ✓ Promote IT use in an inter-team environment to complement sentry processes. ✓ Promote IT use to neutralize the negative effect of guard processes on integrative learning.
Team design SDTs rely on integrative learning, which requires team design features specific to the task, such as diversity, autonomy, a shared understanding of goals and tasks, and an open climate. Crafting such an optimal team would enhance integrative learning in SDTs.	Team effectiveness models (Guzzo and Dickson, 1996) - leverage for enhancing team effectiveness exists through the group's design and context. The I-P-O model - inputs or team design elements enable effective team outcomes (Barrick et al., 1998; Cohen and Bailey, 1997))	✓ Ensure diversity of knowledge, education, experience, and functional background. ✓ Give teams autonomy and flexibility. ✓ Promote an open and transparent climate. ✓ Provide positive leadership. ✓ Promote participation and collective understanding early on during team development.
Project Environment Software teams are embedded in the project environment, which significantly impacts how they use processes, technology, and team dynamics to achieve project goals.	Complexity theory (Curlee and Gordon, 2011) - level of complexity impacts resource utilization and learning. Contingency theory - team learning outcomes are contingent upon project attributes.	✓ Clearly convey project attributes to the teams. ✓ Keep team members abreast of any potential changes in project requirements. ✓ Have contingency plans and promote agile development.

8. Limitations

8.1. Threats to validity

Despite our best effort, this study has some limitations that must be acknowledged. Although we took several measures to ensure reliability and validity, the threats to validity exist. Generally, four types of validity are of interest in a systematic review, namely construct, internal, external, and conclusion validity (Zhou et al., 2016). In this study, a threat to *internal validity* may exist due to sample selection constraints. Internal validity is strong if the conclusions or causal relationships can be trusted and are not attributed to other factors. The study sample was restricted to peer-reviewed works only and did not include non-reviewed, online, and non-English studies. Also, we relied on a title–abstract–keywords search to extract relevant papers from the databases and had to restrict the search to relevant academic domains such as business, management, and computers. For example, a relevant study might have been excluded if the search terms did not match precisely the text in the title, keywords, or abstract or if there was a software development-related study in academic domains such as life sciences. Thus, despite a rigorous search, our search strategy may have resulted in the omission of some relevant publications, which could undermine internal validity (Zhou et al., 2016). The publication bias, which refers to the issue of positive results being published more often, could also impact the internal validity of our results as we only included the published peer-reviewed papers (Kitchenham and Charters, 2007).

Threats to *conclusion validity* cannot be completely ruled out. Conclusion validity refers to how reasonable the research conclusion is based on the data. The studies included in the review had a mix of study designs and reporting formats. For example, about half of the studies were survey-based and the rest were a mix of case studies, conceptual, and review papers. Similarly, some studies included clear outcome measures and some examined the concepts indirectly using qualitative measures. We also could not infer the causal effects of variables or analyze the moderating influences of certain factors such as industry type, project type, and project size due to sample size and insufficient amount of data across available studies on these variables. For example, the software industry consists of different segments such as programming services, system services, and open source. Due to the small sample, we could not analyze the moderating effects of these segments. These constraints pose a threat to conclusion validity due to a lack of clear comparison and consolidation of information.

Finally, sample size and team context limitations may impact *external validity*, which refers to the generalizability of results to another context. We could not distinguish between different types of software teams due to the small sample size. For example, differences may exist in face-to-face versus globally distributed teams and agile versus traditional software development. Similarly, the results may not be generalizable to other types of teams such as marketing or production teams, sports teams, and student teams due to the differences in the nature of their work and role expectations. This impacts the external validity of our results. Thus, further research with a larger

Table A.1
Related work: Existing literature reviews on team learning.

No.	Study	Time frame and sample	Focus	Study objective and conclusion
1	Alberti et al. (2021)	2013–2020 (12)	Nursing education	The objective of this study is to identify, appraise, and summarize primary studies on the effectiveness of TBL in achieving learning outcomes in undergraduate nursing students. Overall, TBL was found to be effective in achieving undergraduate nursing students' learning outcomes.
2	Habeh et al. (2021)	2013–2018 (9)	*Software development*	This paper conducts a systematic literature review of the knowledge-sharing challenges within the software development team and evaluates the suggested solutions. Generally, challenges and suggested solutions across all selected studies were structured under four main pillars: individual, technology, project, and organization.
3	Lang et al. (2019)	2013–2018 (12)	Pharmacy education	The study examines the effect of using the TBL approach in pharmacy education in China. Compared with traditional pedagogy, TBL pedagogy exhibited more effectiveness in improving university and college students' performance.
4	Chen et al. (2018)	Up to 2015 (13)	Medical education	TBL in theoretical teaching of medical education seems to be more effective than LBL in improving the knowledge, attitude, and skill of students in China.
5	Dearnley et al. (2018)	2011–2017 (7)	Nursing and midwifery education	This study examines the relationship between TBL and attainment for nursing and midwifery students in professional higher education. There is evidence that TBL impacts student engagement, satisfaction, attainment, and practice development.
6	Fu and Hwang (2018)	2007–2016 (90)	Mobile technology	This study reviewed the literature on mobile technology-supported collaborative learning. The review found that the amount of research on mobile collaborative learning increased and the connection between new mobile technology and collaborative learning activities became tighter.
7	Zhang and Cui (2018)	1985–2018 (29)	Nursing education	This study examines the collaborative learning implementation process and factors influencing its effectiveness in nursing education. Collaborative learning improves nursing knowledge and skill performance, clinical competency, and student group skills.
8	Reimschisel et al. (2017)	2001–2016 (118)	Healthcare education	The study summarizes the published literature on TBL in health professions education. A substantial body of literature examines the effect that TBL has on traditional measures of achievement, however, gaps still exist.
9	River et al. (2016)	2011–2015 (9)	Healthcare education	This paper identifies how technology has been incorporated into TBL in higher education health disciplines. There is limited evidence that blended-TBL improved student learning outcomes or student preference.
10	Widmann et al. (2016)	1991–2015 (31)	Work teams	Examines the impact of learning behaviors on innovative behavior in work teams. Learning and innovation development are mutually dependent aspects of teamwork.
11	Ghobadi (2015)	1993–2011 (49)	*Software development*	Literature on knowledge sharing drivers in software development teams are reviewed. The analysis identifies task-related and technology-related drivers and recommends exploring new knowledge-sharing drivers, causal relationships, and longitudinal approaches, among other things.
12	Burgess et al. (2014).	2002–2012 (20)	Medical education	The study aims to establish the extent, design, and practice of TBL programs within medical schools to inform curriculum planners and education designers. The majority of the articles reported that TBL provides a positive learning experience for students.
13	Haidet et al. (2014)	Up to 2014 (112)	Education	More rigorous testing and study of additional questions relating to TBL are needed, as well as more accurate reporting of TBL implementation.
14	Stone et al. (2013)	2001–2010 (18)	Nursing education	The study examined various methods of peer learning and their effectiveness in undergraduate nursing education. It concludes that peer learning develops students' skills in communication, critical thinking, and self-confidence.
15	Hannes et al. (2013).	1994–2011 (14)	Work or vocational setting	This study integrates findings on the experiences of employees with team learning in the context of their work or vocational learning setting. The role of boundary-crossing, communication, knowledge sharing, learning environment, authority structures, and recognition of employee authenticity, commitment, and devotion toward team learning is emphasized.
16	Sisk (2011)	2003–2011 (17)	Nursing, medical, and business education	Studies on team-based learning (TBL) in the nursing, medical, education, and business literature are reviewed to examine its impact on student outcomes. TBL promotes student engagement, satisfaction, and class performance.

Table B.1
Summary of the studies used in the review.

No.	Authors	Independent variable	Outcome variable	Main findings
1	Akgün (2020)	Team - diversity; experience; networking; prudence; virtue; reasoning; intuition; aesthetic capability	Team learning; speed to user	SDT's wisdom-related mechanisms are positively related to team prudence and virtue; team prudence and virtue are positively related to SDT joint epistemic actions, and joint epistemic actions are positively associated with team outcomes.
2	Akgün et al. (2015)	Interpersonal trust; team communication; team member familiarity; collective empathy (cognitive; affective, behavioral); group norms	Team learning; speed-to-market; development cost	Collective empathy affects team learning and product speed-to-market and results in lower project development costs.
3	de Araujo and Borges (2007)	Awareness of information mechanisms; collaboration	Team process learning; software process improvement	Process improvement environment, including collaboration support, is an important factor in software process improvement and process learning.
4	Dissanayake et al. (2013)	Agile practices (iterative development; self-organization; process flexibility)	Team knowledge integration (mediator); team creativity	A model is proposed that elucidates how knowledge integration mediates the relationship between agile practices and team creativity.
5	Freeman and McNeese (2019)	NA	Collaborative learning	Team learning is a natural, routine process for developers. They learn by participating in formal workshops, working and coordinating with team members, answering others' questions, sharing and discussing ideas and works in progress with team members, and providing feedback and opinions. Modern technologies significantly facilitate learning.
6	Fuller (2019)	NA	Collaborative understanding	A cross-functional product team's ability to blur the boundaries both within the team, within the company, and between the team and its product domain impacts the team's capacity to collectively grok. For software product organizations, both productivity and innovation are impeded by hardened, sharpened, and specialized boundaries and methods.
7	Gulliksen Stray et al. (2011)	NA	Team learning	Two challenges related to team learning were the problem of transforming experience into action and the lack of paying attention to what was working well.
8	Kude et al. (2014)	NA	Team learning; team adaptation	How iteratively a team works impacts team learning and adaptation. Iterative teams learn faster and develop new responses to events. Less iterative teams learn to a lower extent and may face similar issues repeatedly.
9	Mehta and Bharadwaj (2015)	Sentry processes, guard processes, project uncertainty (moderator)	Team knowledge integration	Complex curvilinear associations exist among sentry and guard activities and knowledge integration. The level of project uncertainty the team faces moderates these relationships.
10	Mehta et al. (2006)	Knowledge heterogeneity; project complexity; usage of IT-based systems (moderator)	Team external and internal knowledge integration	Team knowledge integration is influenced by IT use, knowledge heterogeneity, and project complexity.
11	Mehta et al. (2014)	IT use intensity; project uncertainty; relational capital; team cognition	Team knowledge process (knowledge combination and exchange)	IT use intensity impacts knowledge processes, and the effect is stronger on knowledge exchange compared to knowledge combination. Project uncertainty moderates the relationship between IT use intensity and knowledge processes.
12	Patnayakuni et al. (2007)	Formal and information integrative practices; knowledge	Knowledge integration (mediator); SDT performance	The positive influence of formal and informal organizational integrative practices on ISD performance is partially mediated by knowledge integration.
13	Spohrer et al. (2012)	NA	Team leaning	Three perspectives on team learning applied to the field of ISD are identified: (1) the team learning curve, (2) shared knowledge and group memory, and (3) team learning behavior. Presents a structural and relational perspective to research team learning.
14	Tiwana and Mclean (2005)	Expertise heterogeneity; relational capital; absorptive capacity; technological uncertainty	Expertise integration (mediator); creativity	Project teams' diverse specialized knowledge, the quality of intrateam working relationships, and members' cross-domain absorptive capacity engender creativity through expertise integration.
15	White and Dovey (2004)	NA	Knowledge construction	Social capital (relational and identity resources) plays a critical role in the collaborative learning practices resulting in the construction and re-constructed of knowledge.
16	Zorgios et al. (2009)	Human capital	Team learning	Human capital variability across the population of developer teams impacts the fluctuations in the learning rates.
17	Robert et al. (2008)	Social capital (relational, structural, cognitive); communication environment	Knowledge integration (mediator); team decision quality	Structural and cognitive capitals are more important to knowledge integration when teams communicate using lean digital networks compared to communicating face-to-face. Social capital influences team decision quality in part through team knowledge integration.
18	Tiwana et al. (2003)	Business-IS linkage	Knowledge integration (mediator); ISD capability	IS-business linkages influence team knowledge integration, which in turn influences ISD processes and outcomes.

(*continued on next page*)

Table B.1 (*continued*).

No.	Authors	Independent variable	Outcome variable	Main findings
19	Anderson et al. (2018)	Team personality; group cohesion	Team learning (mediator); team effectiveness	Teams comprised of homogeneous personalities develop greater cohesion and confidence in their ability to execute creative problem solving. Group cohesion improves performance, but the learning does not affect it.
20	Gholami and Heinzl (2013)	Leadership role clarity; team empowerment; team climate; learning goals and task.	Collective learning	Collective learning is impacted by leadership role clarity, team empowerment, team climate, team learning goals, and task.
21	Spohrer et al. (2013)	Pair programming; peer code review	Knowledge creation	PP and PCR complement each other in the case of team-level knowledge creation. Pair rotation (in the case of PP) or multiple reviewers and readers (when applying PCR) further facilitate knowledge creation.
22	Liang et al. (2009)	Informational diversity; task conflict; relationship conflict	Learning and interaction quality (mediators); software quality	Informational diversity leads to task conflict, which promotes learning, which, in turn, improves the software quality. The link between relationship conflict and software quality is mediated by interaction quality and learning.
23	Janz and Prasarnphanich (2009)	Team autonomy (product, people, and planning related)	Cooperative learning (positive interdependence, group process, promotive interaction), performance, and satisfaction.	Autonomous teams engage more frequently in cooperative learning behaviors and consequently perform better and are more satisfied. The type of autonomy present in teams affects elements of cooperative learning and work outcomes.
24	Chou and He (2011)	Relational capital, cognitive capital, and structural capital; technological complexity; task interdependence	Expertise integration; project effectiveness (team size, team effort, task completion)	Reciprocity and centrality affect expertise integration, but commitment and cognitive capital (including expertise and tenure) do not. Expertise integration affects both team size and team effort, which in turn jointly influence task completion.
25	Ekemen and Şeşen (2020)	Social capital	knowledge integration capability	A significant positive effect of structural and relational social capital on knowledge integration but no relationship between cognitive social capital and knowledge integration.
26	Bhandar et al. (2007)	Social capital	Knowledge integration	Social capital can be leveraged as a motivator, an integrator, and a facilitator during the various stages of a collaborative IS project.
27	Newell et al. (2004)	Social capital	Knowledge integration	For effective mobilization of 'weak' social capital bridges for collective purposes, there is a need to create 'strong' social capital bonds to integrate knowledge acquired through members' bridging activity.
28	Govindaraju and Arianto (2015)	Project organization, team environment, formalization and centralization mechanisms, communication medium	Internal knowledge integration, distribution of information	Project organization impacts knowledge integration. The creation of formalization and centralization mechanisms ensure clarity in the distribution of information. A good atmosphere among the team members improves internal integration. The use of an effective communication medium affects knowledge integration.
29	Koh and Lim (2012)	Initial participation	Task performance, team learning, outcome satisfaction	Higher initial participation leads to negative outcomes, which suggests the importance of coordination mechanisms in the initial period, especially in time-limited teams.
30	Babb et al. (2014).	Team reflection	Agile development, team learning	The Reflective Agile Learning Model (REALM) is presented, showing where and how to integrate reflective practice in agile software development.
31	Lamoreux (2005)	Team reflections	Team learning	This paper describes how can teams improve team learning by engaging in regular reflection.
32	Fonseca et al. (2021).	Information and communication technologies	Group learning	Group learning occurs in software teams through information and communication technologies, which allow developers to interact and share their knowledge.

sample and in different contexts is warranted to understand the subtle nuances of team learning in different contexts and sub-contexts.

8.2. Other limitations

Although our review provides insights on a set of critical factors impacting software team learning, we cannot conclude that these factors would be equally important in all contexts and sub-contexts. Moreover, due to the small sample, we could not differentiate between the proximal and distal factors or the intervening processes impacting team learning. Similarly, some aspects could not be completely understood such as the role of project characteristics. The lack of studies in the domain highlights the need for further research. Future studies can examine the impact of software team learning on various team outcomes, which we did not examine in this review. Future researchers can also broaden the scope of their inquiry by including other team contexts and examining specific causal relationships in a review or a meta-analytical investigation in the team learning domain.

9. Conclusions

In conclusion, this study contributes to the information systems, software project management, and organizational behavior domains. Research is still scant and fragmented on integrative learning in software teams. This review identifies the knowledge gaps and proposes future research agenda. The study shows that the key drivers of integrative learning in software teams extend beyond the traditional people–process–technology categories. IT use presents some intriguing contingencies where the role of people and process components is more or less straightforward. Also, team- and project-related factors constitute other key dimensions. Based on the findings a framework is proposed that can guide future research and practice on team learning in a software context. Finally, essential areas of action for software project leaders are uncovered.

CRediT authorship contribution statement

Anju Mehta: Data screening and extraction, Coding, Revisions. **Nikhil Mehta:** Conceptualization, Methodology, Data curation, Writing – original draft. **Ishaan Bindal:** Data extraction and coding, Triangulation of results.

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.

Appendix A

See Table A.1.

Appendix B

See Table B.1.

References

Akan, O.H., Jack, E.P., Mehta, A., 2020. Concrescent conversation environment, psychological safety, and team effectiveness. Team Perform. Manag. Int. J. 26 (1/2), 29–51. http://dx.doi.org/10.1108/TPM-07-2019-0079.

Akgün, A.E., 2020. Team wisdom in software development projects and its impact on project performance. Int. J. Inf. Manage. 50, 228–243. http://dx.doi.org/10.1016/j.ijinfomgt.2019.05.019.

Akgün, A.E., Keskin, H., Cebecioglu, A.Y., Dogan, D., 2015. Antecedents and consequences of collective empathy in software development project teams. Inf. Manag. 52 (2), 247–259. http://dx.doi.org/10.1016/j.im.2014.11.004.

Alavi, M., Leidner, D.E., 2001. Knowledge management and knowledge management systems: conceptual foundations and research issues. MIS Q. 25 (1), 107–136. http://dx.doi.org/10.2307/3250961.

Alberti, S., Motta, P., Ferri, P., Bonetti, L., 2021. The effectiveness of team-based learning in nursing education: a systematic review. Nurse Educ. Today 97, 104721. http://dx.doi.org/10.1016/j.nedt.2020.104721.

Anderson, G., Keith, M.J., Francisco, J., Fox, S., 2018. The effect of software team personality composition on learning and performance: making the dream team. In: Proceedings of the 51st Hawaii International Conference on System Sciences. http://dx.doi.org/10.24251/HICSS.2018.059.

Appelbaum, S.H., 1997. Socio-technical systems theory: an intervention strategy for organizational development. Manage. Decis. 35 (6), 452–463. http://dx.doi.org/10.1108/00251749710173823.

Ayyagari, R., Grover, V., Purvis, R., 2011. Technostress: technological antecedents and implications. MIS Q. 35 (4), 831–858. http://dx.doi.org/10.2307/41409963.

Babb, J., Hoda, R., Nørbjerg, J., 2014. Embedding reflection and learning into agile software development. IEEE Softw. 31 (4), 51–57. http://dx.doi.org/10.1109/MS.2014.54.

Barrick, M.R., Stewart, G.L., Neubert, M.J., Mount, M.K., 1998. Relating member ability and personality to work-team processes and team effectiveness. J. Appl. Psychol. 83 (3), 377–391. http://dx.doi.org/10.1037/0021-9010.83.3.377.

Belur, J., Tompson, L., Thornton, A., Simon, M., 2021. Interrater reliability in systematic review methodology: exploring variation in coder decision-making. Sociol. Methods Res. 50 (2), 837–865, 10.1177%2F0049124118799372.

Berente, N., Gal, U., Hansen, S., 2011. Ethical implications of social stratification in information systems research. Inf. Syst. J. 21 (4), 357–382. http://dx.doi.org/10.1111/j.1365-2575.2010.00353.x.

Bhandar, M., Pan, S.L., Tan, B.C., 2007. Towards understanding the roles of social capital in knowledge integration: a case study of a collaborative information systems project. J. Am. Soc. Inf. Sci. Technol. 58 (2), 263–274. http://dx.doi.org/10.1002/asi.20493.

Burgess, A.W., McGregor, D.M., Mellis, C.M., 2014. Applying established guidelines to team-based learning programs in medical schools: a systematic review. Acad. Med. 89 (4), 678, 10.1097%2FACM.0000000000000162.

Carter, K.M., Mead, B.A., Stewart, G.L., Nielsen, J.D., Solimeo, S.L., 2019. Reviewing work team design characteristics across industries: combining meta-analysis and comprehensive synthesis. Small Group Res. 50 (1), 138–188. http://dx.doi.org/10.1177/1046496418797431.

Castro-Herrera, C., Duan, C., Cleland-Huang, J., Mobasher, B., 2009. A recommender system for requirements elicitation in large-scale software projects. In: Proceedings of the 2009 ACM Symposium on Applied Computing. pp. 1419–1426.

Chen, M., Ni, C., Hu, Y., Wang, M., Liu, L., Ji, X., et al., 2018. Meta-analysis on the effectiveness of team-based learning on medical education in china. BMC Med. Educ. 18 (1), 1–11. http://dx.doi.org/10.1186/s12909-018-1179-1.

Cheng, H.H., Yang, H.L., 2014. The antecedents of collective creative efficacy for information system development teams. J. Eng. Technol. Manag. 33, 1–17. http://dx.doi.org/10.1016/j.jengtecman.2013.12.001.

Chou, S.W., He, M.Y., 2011. The factors that affect the performance of open-source software development–the perspective of social capital and expertise integration. Inf. Syst. J. 21 (2), 195–219. http://dx.doi.org/10.1111/j.1365-2575.2009.00347.x.

Clarke, P., O'Connor, R.V., Leavy, B., 2016. A complexity theory viewpoint on the software development process and situational context. In: Proceedings of the International Conference on Software and Systems Process. pp. 86–90.

Cohen, S.G., Bailey, D.E., 1997. What makes teams work: group effectiveness research from the shop floor to the executive suite. J. Manag. 23 (3), 239–290. http://dx.doi.org/10.1016/j.jbusres.2022.03.056.

Colman, A.M., Pulford, B.D., Rose, J., 2008. Collective rationality in interactive decisions: evidence for team reasoning. Acta Psychol. 128 (2), 387–397. http://dx.doi.org/10.1016/j.actpsy.2007.08.003.

Cooper, H., Hedges, L.V., Valentine, J.C., 2019. The handbook of research synthesis and meta-analysis. Russell Sage Foundation.

Curlee, W., Gordon, R.L., 2011. Complexity theory and project management. Wiley.

Dalenogare, L.S., Benitez, G.B., Ayala, N.F., Frank, A.G., 2018. The expected contribution of industry 4.0 technologies for industrial performance. Int. J. Prod. Econ. 204, 383–394. http://dx.doi.org/10.1016/j.ijpe.2018.08.019.

de Araujo, R.M., Borges, M.R., 2007. The role of collaborative support to promote participation and commitment in software development teams. Softw. Process Improv. Pract. 12 (3), 229–246. http://dx.doi.org/10.1002/spip.314.

De Montjoye, Y.A., Stopczynski, A., Shmueli, E., Pentland, A., Lehmann, S., 2014. The strength of the strongest ties in collaborative problem solving. Sci. Rep. 4 (1), 1–6. http://dx.doi.org/10.1038/srep05277.

Dearnley, C., Rhodes, C., Roberts, P., Williams, P., Prenton, S., 2018. Team-based learning in nursing and midwifery higher education; a systematic review of the evidence for change. Nurse Educ. Today 60, 75–83. http://dx.doi.org/10.1016/j.nedt.2017.09.012.

Desouza, K.C., 2003. Barriers to effective use of knowledge management systems in software engineering. Commun. ACM 46 (1), 99–101. http://dx.doi.org/10.1145/602421.602458.

Dingsøyr, T., Fægri, T.E., Dybå, T., Haugset, B., Lindsjørn, Y., 2016. Team performance in software development: research results versus agile principles. IEEE Softw. 33, 106–110. http://dx.doi.org/10.1109/MS.2016.100.

Dissanayake, I., Dantu, R., Nerur, S., 2013. Knowledge management in software development. In: Proceedings of the Nineteenth Americas Conference on Information Systems, Chicago, Illinois. Available at: https://core.ac.uk/download/pdf/301359277.pdf.

Dixon, N., 2017. Learning together and working apart: routines for organizational learning in virtual teams. Learn. Organ. 24 (3), 138–149. http://dx.doi.org/10.1108/TLO-12-2016-0101.

Dokko, G., Kane, A.A., Tortoriello, M., 2014. One of us or one of my friends: How social identity and tie strength shape the creative generativity of boundary-spanning ties. Organ. Stud. 35 (5), 703–726. http://dx.doi.org/10.1177/0170840613508397.

Dovey, K., White, R., 2005. Learning about learning in knowledge-intense organizations. Learn. Organ. 12 (3), 246–260. http://dx.doi.org/10.1108/09696470510592502.

Edmondson, A.C., Dillon, J.R., Roloff, K.S., 2007. Three perspectives on team learning: outcome improvement, task mastery, and group process. In: Brief, A., Walsh, J. (Eds.), The Academy of Management Annals. Routledge, London, England, pp. 269–314.

Ekemen, M.A., Şeşen, H., 2020. Dataset on social capital and knowledge integration in project management. Data Brief 29, 105233. http://dx.doi.org/10.1016/j.dib.2020.105233.

Falagas, M.E., Pitsouni, E.I., Malietzis, G.A., Pappas, G., 2008. Comparison of PubMed, Scopus, web of science, and google scholar: strengths and weaknesses. The FASEB J. 22 (2), 338–342. http://dx.doi.org/10.1096/fj.07-9492LSF.

Faraj, S., Sproull, L., 2000. Coordinating expertise in software development teams. Manage. Sci. 46 (12), 1554–1568. http://dx.doi.org/10.1287/mnsc.46.12.1554.12072.

Felder, R.M., Brent, R., 2007. Cooperative learning. In: Active Learning: Models from the Analytical Sciences. In: ACS Symposium Series, vol. 970, pp. 34–53.

Fogelströtom, N.D., Gorschek, T., Svahnberg, M., Olsson, P., 2010. The impact of agile principles on market-driven software product development. J. Softw. Main. Evol. Res. Pract. 22 (1), 53–80. http://dx.doi.org/10.1002/spip.420.

Fonseca, L.R.D., Silva, M.R., Silva, S.W., Pereira, G.M., 2021. Group learning: an analysis of individuals in software development teams who perform interdependent tasks. Int. J. Innov. Learn. 29 (1), 89–105.

Freeman, G., McNeese, N.J., 2019. Exploring indie game development: team practices and social experiences in a creativity-centric technology community. In: Proceedings of the 17th European Conference on Computer Supported Cooperative Work. Vol. 28. ECSCW, (3–4), pp. 723–748. http://dx.doi.org/10.1007/s10606-019-09348-x.

Fu, Q.K., Hwang, G.J., 2018. Trends in mobile technology-supported collaborative learning: a systematic review of journal publications from 2007 to 2016. Comput. Educ. 119, 129–143.

Fuhse, J.A., 2009. The meaning structure of social networks. Sociol. Theory 27 (1), 51–73. http://dx.doi.org/10.1111/j.1467-9558.2009.00338.x.

Fuller, R., 2019. Blurring boundaries. In: Cappiello, C., Ruiz, M. (Eds.), Information Systems Engineering in Responsible Information Systems. In: CAiSE 2019. Lecture notes in business information processing, vol. 350, Springer, Cham, http://dx.doi.org/10.1007/978-3-030-21297-1_8.

Ghobadi, S., 2015. What drives knowledge sharing in software development teams: a literature review and classification framework. Inf. Manag. 52 (1), 82–97.

Gholami, B., Heinzl, A., 2013. Leading agile self-organizing teams: a collective learning perspective. In: International Conference on Organizational Learning, Knowledge and Capabilities. Available at: https://gess.uni-mannheim.de/fileadmin/gess_db/Publications/CDSB/Erscheinungsjahr_2013/183.pdf.

Govindaraju, R., Arianto, T.A., 2015. Knowledge integration and its impact on offshore IT outsourcing success. In: Proceedings of the 3rd International Congress on Interdisciplinary Behavior and Social Science 2014 (ICIBSoS 2014). CRC Press, Bali, Indonesia, p. 385.

Gu, Q., Wang, G.G., Wang, L., 2013. Social capital and innovation in R & D teams: the mediating roles of psychological safety and learning from mistakes. R D Manag. 43 (2), 89–102. http://dx.doi.org/10.1111/radm.12002.

Gulliksen Stray, V., Moe, N.B., Dingsoyr, T., 2011. Challenges to teamwork: a multiple case study of two agile teams. In: Proceedings of the International Conference on Agile Software Development. Springer, Berlin, pp. 146–161. http://dx.doi.org/10.1007/978-3-642-20677-1_11.

Gupta, A., Mattarelli, E., Seshasai, S., Broschak, J., 2009. Use of collaborative technologies and knowledge sharing in co-located and distributed teams: towards the 24-h knowledge factory. J. Strateg. Inf. Syst. 18 (3), 147–161. http://dx.doi.org/10.1016/j.jsis.2009.07.001.

Guzzo, R.A., Dickson, M.W., 1996. Teams in organizations: recent research on performance and effectiveness. Ann. Rev. Psychol. 47 (1), 307–338. http://dx.doi.org/10.1146/annurev.psych.47.1.307.

Habeh, O., Thekrallah, F., Salloum, S.A., Shaalan, K., 2021. Knowledge sharing challenges and solutions within software development team: a systematic review. In: Recent Advances in Intelligent Systems and Smart Applications. pp. 121–141.

Haidet, P., Kubitz, K., McCormack, W.T., 2014. Analysis of the team-based learning literature: TBL comes of age. J. Excell. College Teach. 25 (3-4), 303.

Hannes, K., Raes, E., Vangenechten, K., Heyvaert, M., Dochy, F., 2013. Experiences from employees with team learning in a vocational learning or work setting: a systematic review of qualitative evidence. Educ. Res. Rev. 10, 116–132.

Haron, A., Sahibuddin, S., Harun, M., Zakaria, N.H., Mahrin, M.N., 2013. The important role of people, process, and technology during software project requirement. Int. J. Mach. Learn. Comput. 3 (1), 24. http://dx.doi.org/10.7763/IJMLC.2013.V3.266.

Huckman, R.S., Staats, B.R., Upton, D.M., 2009. Team familiarity, role experience, and performance: evidence from Indian software services. Manage. Sci. 55 (1), 85–100. http://dx.doi.org/10.1287/mnsc.1080.0921.

Ip, P.K., 2014. Harmony as happiness? Social harmony in two Chinese societies. Soc. Indic. Res. 117 (3), 719–741. http://dx.doi.org/10.1007/s11205-013-0395-7.

Janz, B.D., Prasarnphanich, P., 2009. Freedom to cooperate: gaining clarity into knowledge integration in information systems development teams. IEEE Trans. Eng. Manage. 56 (4), 621–635. http://dx.doi.org/10.1109/TEM.2009.2023451.

Khalil, C., Khalil, S., 2019. Exploring knowledge management in agile software development organizations. Int. Entrepreneurship Manag. J. 1–15. http://dx.doi.org/10.1007/s11365-019-00582-9.

Kitchenham, B.A., Charters, S., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical Report EBSE-2007-01, School of Computer Science and Mathematics, Keele University.

Koh, E., Lim, J., 2012. Too early, too bad: uncovering and understanding the initial participation paradox in technology-mediated learning teams. IEEE Trans. Prof. Commun. 55 (1), 55–84. http://dx.doi.org/10.1109/TPC.2011.2172122.

Kude, T., Bick, S., Schmidt, C., Heinzl, A., 2014. Adaptation patterns in agile information systems development teams. In: Proceedings of the Twenty-Second European Conference on Information Systems. Tel Aviv. https://aisel.aisnet.org/ecis2014/proceedings/track13/11/.

Laal, M., Ghodsi, S.M., 2012. Benefits of collaborative learning. Procedia-Soc. Behav. Sci. 31, 486–490. http://dx.doi.org/10.1016/j.sbspro.2011.12.091.

Lamoreux, M., 2005. Improving agile team learning by improving team reflections [agile software development]. In: Agile Development Conference. ADC'05, pp. 139–144. http://dx.doi.org/10.1109/ADC.2005.29.

Landis, J.R., Koch, G.G., 1977. The measurement of observer agreement for categorical data. Biometrics 159–174.

Lang, B., Zhang, L., Lin, Y., Han, L., Zhang, C., Liu, Y., 2019. Team-based learning pedagogy enhances the quality of chinese pharmacy education: a systematic review and meta-analysis. BMC Med. Educ. 19 (1), 1–12.

Liang, T.P., Jiang, J., Klein, G.S., Liu, J.Y.C., 2009. Software quality as influenced by informational diversity, task conflict, and learning in project teams. IEEE Trans. Eng. Manage. 57 (3), 477–487. http://dx.doi.org/10.1109/TEM.2009.2033049.

Liu, L., Kong, X., Chen, J., 2015. How project duration, upfront costs, and uncertainty interact and impact on software development productivity? A simulation approach. Int. J. Agile Syst. Manag. 8 (1), 39–52. http://dx.doi.org/10.1504/IJASM.2015.068605.

Liu, W., Sidhu, A., Beacom, A.M., Valente, T.W., 2017. Social network theory. Int. Encycl. Media Eff. 1–12. http://dx.doi.org/10.1002/9781118783764.wbieme0092.

Mariano, S., Casey, A., Olivera, F., 2020. Organizational forgetting part I: a review of the literature and future research directions. Learn. Organ. 27 (3), 185–209. http://dx.doi.org/10.1108/TLO-12-2019-0182.

Mehta, N., Bharadwaj, A., 2015. Knowledge integration in outsourced software development: The role of sentry and guard processes. J. Manage. Inf. Syst. 32 (1), 82–115. http://dx.doi.org/10.1080/07421222.2015.1029381.

Mehta, N., Byrd, T., Hall, D., 2006. Knowledge integration in software teams: an assessment of team project and it-related issues. In: Proceedings of the Americas Conference on Information Systems. p. 212, Available at: https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1760&context=amcis2006.

Mehta, N., Hall, D., Byrd, T., 2014. Information technology and knowledge in software development teams: The role of project uncertainty. Inf. Manag. 51 (4), 417–429. http://dx.doi.org/10.1016/j.im.2014.02.007.

Migdadi, M.M., 2019. Organizational learning capability, innovation, and organizational performance. Eur. J. Innov. Manag. 24 (1), 151–172. http://dx.doi.org/10.1108/EJIM-11-2018-0246.

Momtazi, S., Khudanpur, S., Klakow, D., 2010. A comparative study of word co-occurrence for term clustering in language model-based sentence retrieval. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 325–328.

Nahapiet, J., Ghoshal, S., 1998. Social capital, intellectual capital, and the organizational advantage. Acad. Manag. Rev. 23 (2), 242–266. http://dx.doi.org/10.5465/amr.1998.533225.

Nahavandi, S., 2019. Industry 5.0—A human-centric solution. Sustainability 11 (16), 4371. http://dx.doi.org/10.3390/su11164371.

Newell, S., Tansley, C., Huang, J., 2004. Social capital and knowledge integration in an ERP project team: the importance of bridging and bonding. Br. J. Manag. 15 (S1), 43–57. http://dx.doi.org/10.1111/j.1467-8551.2004.00399.x.

Nolte, W.L., 2008. Did I ever tell you about the whale? In: Or Measuring Technology Maturity. IAP.

Otero, C., 2012. Software Engineering Design: Theory and Practice. CRC Press.

Patnayakuni, R., Rai, A., Tiwana, A., 2007. Systems development process improvement: a knowledge integration perspective. IEEE Trans. Eng. Manage. 54 (2), 286–300. http://dx.doi.org/10.1109/TEM.2007.893997.

Petticrew, M., Robert, H., 2008. Systematic Reviews in the Social Sciences, a Practical Guide. Blackwell Publishing.

Ragu-Nathan, T.S., Tarafdar, M., Ragu-Nathan, B.S., Tu, Q., 2008. The consequences of technostress for end users in organizations: Conceptual development and empirical validation. Inf. Syst. Res. 19 (4), 417–433. http://dx.doi.org/10.1287/isre.1070.0165.

Rangel, N., Torres, C., Peña, A., Muñoz, M., Mejia, J., Hernández, L., 2017. Team members' interactive styles involved in the software development process. In: Stolfa, J., Stolfa, S., O'Connor, R., Messnarz, R. (Eds.), Systems, Software and Services Process Improvement. EuroSPI 2017. Communications in Computer and Information Science. Vol. 748. Springer, Cham, http://dx.doi.org/10.1007/978-3-319-64218-5_56.

Reimschisel, T., Herring, A.L., Huang, J., Minor, T.J., 2017. A systematic review of the published literature on team-based learning in health professions education. Med. Teacher 39 (12), 1227–1237. http://dx.doi.org/10.1080/0142159X.2017.1340636.

Richter, A.W., West, M.A., Dick, R., Dawson, J.F., 2006. Boundary spanners' identification, intergroup contact, and effective intergroup relations. Acad. Manag. J. 48 (6), 1252–1269. http://dx.doi.org/10.5465/amj.2006.23478720.

River, J., Currie, J., Crawford, T., Betihavas, V., Randall, S., 2016. A systematic review examining the effectiveness of blending technology with team-based learning. Nurse Educ. Today 45, 185–192. http://dx.doi.org/10.1016/j.nedt.2016.08.012.

Robert, Jr., L.P., Dennis, A.R., Ahuja, M.K., 2008. Social capital and knowledge integration in digitally enabled teams. Inf. Syst. Res. 19 (3), 314–334. http://dx.doi.org/10.1287/isre.1080.0177.

Sanderson, S., Tatt, I.D., Higgins, J.P., 2007. Tools for assessing quality and susceptibility to bias in observational studies in epidemiology: a systematic review and annotated bibliography. Int. J. Epidemiol. 36, 666–676. http://dx.doi.org/10.1093/ije/dym018.

Schultze, U., Orlikowski, W.J., 2004. A practice perspective on technology-mediated network relations: the use of internet-based self-serve technologies. Inf. Syst. Res. 15 (1), 87–106. http://dx.doi.org/10.1287/isre.1030.0016.

Sisk, R.J., 2011. Team-based learning: systematic research review. J. Nursing Educ. 50 (12), 665–669. http://dx.doi.org/10.3928/01484834-20111017-01.

Šmite, D., Moe, N.B., Šāblis, A., Wohlin, C., 2017. Software teams and their knowledge networks in large-scale software development. Inf. Softw. Technol. 86, 71–86. http://dx.doi.org/10.1016/j.infsof.2017.01.003.

Soja, E., Soja, P., 2017. Exploring root problems in enterprise system adoption from an employee age perspective: a people-process-technology framework. Inf. Syst. Manag. 34 (4), 333–346. http://dx.doi.org/10.1080/10580530.2017.1366218.

Spohrer, K., Gholami, B., Heinzl, A., 2012. Team learning in information systems development - a literature review. In: Proceedings of the Twentieth European Conference on Information Systems, Barcelona, Spain. https://aisel.aisnet.org/ecis2012/223.

Spohrer, K., Kude, T., Schmidt, C.T., Heinzl, A., 2013. Knowledge creation in information systems development teams: The role of pair programming and peer code review. In: Proceedings of the Twenty-First European Conference on Information Systems, Utrecht, Netherlands. http://aisel.aisnet.org/ecis2013_cr.

Stewart, G.L., 2006. A meta-analytic review of relationships between team design features and team performance. J. Manag. 32 (1), 29–55. http://dx.doi.org/10.1177/0149206305277792.

Stone, R., Cooper, S., Cant, R., 2013. The value of peer learning in undergraduate nursing education: a systematic review. Int. Sch. Res. Notices 1–10. http://dx.doi.org/10.1155/2013/930901.

Strauss, A., Corbin, J.M., 1997. Grounded theory in practice. Sage.

Stray, V.G., Moe, N.B., Dingsøyr, T., 2011. Challenges to teamwork: a multiple case study of two agile teams. In: Proceedings of the International Conference on Agile Software Development. Springer, Berlin, Heidelberg, pp. 146–161.

Tarafdar, M., Darcy, J., Turel, O., Gupta, A., 2015. The dark side of information technology. MIT Sloan Manag. Rev. 56 (2), 61–70, https://sloanreview.mit.edu/wp-content/uploads/2014/12/b104172b57.pdf.

Tiwana, A., 2004. An empirical study of the effect of knowledge integration on software development performance. Inf. Softw. Technol. 46 (13), 899–906. http://dx.doi.org/10.1016/j.infsof.2004.03.006.

Tiwana, A., Bharadwaj, A., Sambamurthy, V., 2003. The antecedents of information systems development capability in firms: a knowledge integration perspective. In: ICIS 2003 Proceedings. Vol. 21. https://aisel.aisnet.org/icis2003/21.

Tiwana, A., Mclean, E.R., 2005. Expertise integration and creativity in information systems development. J. Manage. Inf. Syst. 22 (1), 13–43. http://dx.doi.org/10.1080/07421222.2003.11045836.

Valentine, J.C., 2019. Incorporating judgements about study quality into research syntheses. In: Cooper, H., Hedges, L.V., Valentine, J.C. (Eds.), The Handbook of Research Synthesis and Meta-Analysis, third ed. Russel Sage Foundation, pp. 129–140.

Wallace, L., Keil, M., Rai, A., 2004. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. Decis. Sci. 35 (2), 289–321. http://dx.doi.org/10.1111/j.00117315.2004.02059.x.

Wang, J., Keil, M., Oh, L.B., Shen, Y., 2017. Impacts of organizational commitment, interpersonal closeness, and confucian ethics on willingness to report bad news in software projects. J. Syst. Softw. 125, 220–233. http://dx.doi.org/10.1016/j.jss.2016.12.004.

Wang, N., Sun, Y., Shen, X.L., Zhang, X., 2018. A value-justice model of knowledge integration in wikis: the moderating role of knowledge equivocality. Int. J. Inf. Manage. 43, 64–75. http://dx.doi.org/10.1016/j.ijinfomgt.2018.07.006.

Weinberg, F.J., 2015. Epistemological beliefs and knowledge sharing in work teams: a new model and research questions. Learn. Organ. 22 (1), 40–57. http://dx.doi.org/10.1108/TLO-11-2013-0067.

White, R., Dovey, K., 2004. Knowledge construction in an Australian software development enterprise: developing the knowledge bases for innovative renewal. Int. J. Learn. Intell. Cap. 1 (4), 405–415. http://dx.doi.org/10.1504/IJLIC.2004.005991.

Widmann, A., Messmann, G., Mulder, R.H., 2016. The impact of team learning behaviors on team innovative work behavior: a systematic review. Hum. Resour. Dev. Rev. 15 (4), 429–458. http://dx.doi.org/10.1177/1534484316673713.

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering. pp. 1–10.

Yu, X., Petter, S., 2014. Understanding agile software development practices using shared mental models theory. Inf. Softw. Technol. 56 (8), 911–921. http://dx.doi.org/10.1016/j.infsof.2014.02.010.

Yuan, Y.C., Fulk, J., Monge, P.R., Contractor, N., 2010. Expertise directory development, shared task interdependence, and strength of communication network ties as multilevel predictors of expertise exchange in transactive memory work groups. Commun. Res. 37 (1), 20–47, 10.1177%2F0093650209351469.

Zhang, J., Cui, Q, 2018. Collaborative learning in higher nursing education: a systematic review. J. Prof. Nursing 34 (5), 378–388. http://dx.doi.org/10.1016/j.profnurs.2018.07.007.

Zhou, X., Jin, Y., Zhang, H., Li, S., Huang, X., 2016. A map of threats to validity of systematic literature reviews in software engineering, In: Proceedings of the 2016 23rd Asia-Pacific Software Engineering Conference (APSEC), p. 153–160, https://dx.org/10.1109/APSEC.2016.031.

Zorgios, Y., Venieris, G.G., Vlismas, O., 2009. A learning curve explanatory theory for team learning valuation. VINE: J. Inf. Knowl. Manag. Syst. Available at SSRN: https://ssrn.com/abstract=1786355.

Anju Mehta is an assistant professor in the Willie A. Deese School of Business and Economics at North Carolina A&T University at Greensboro. Her research interests include knowledge integration in teams, team processes and effectiveness, diversity and inclusion, and human resource management.

Nikhil Mehta is an assistant professor in the Bryan School of Business and Economics at University of North Carolina at Greensboro. His research interests include socio-technical aspects of IS development, socio-behavioral impacts of IS, IS success, and strategic management of IS.

Ishaan Bindal is a graduate student at Universite Libre de Bruxelles. His research interests include Natural Language Processing (NLP), applied econometrics, big data, advanced analytics, and artificial intelligence.