Information and Software Technology 54 (2012) 1067–1078

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

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

Impact of physical ambiance on communication, collaboration and coordination in agile software development: An empirical evaluation

Deepti Mishra [a,*], Alok Mishra [b,c,d], Sofiya Ostrovska [e]

[a] Department of Computer Engineering, Atilim University, Incek, Ankara 06836, Turkey
[b] Department of Software Engineering, Atilim University, Incek, Ankara, Turkey
[c] Univ S Pacific, Sch Comp Informat & Math Sci, Suva, Fiji
[d] U21Global, Singapore, Singapore
[e] Department of Mathematics, Atilim University, Incek, Ankara, Turkey

ARTICLE INFO

Article history:
Received 28 June 2011
Received in revised form 6 March 2012
Accepted 22 April 2012
Available online 14 May 2012

Keywords:
Collaboration
Communication
Coordination
Physical settings
Agile software development

ABSTRACT

Context: Communication, collaboration and coordination are key enablers of software development and even more so in agile methods. The physical environment of the workspace plays a significant role in effective communication, collaboration, and coordination among people while developing software.
Objective: In this paper, we have studied and further evaluated empirically the effect of different constituents of physical environment on communication, coordination, and collaboration, respectively. The study aims to provide a guideline for prospective agile software developers.
Method: A survey was conducted among software developers at a software development organization. To collect data, a survey was carried out along with observations, and interviews.
Results: It has been found that half cubicles are 'very effective' for the frequency of communication. Further, half cubicles were discovered 'effective' but not 'very effective' for the quality/effectiveness of communication. It is found that half-height cubicles and status boards are 'very effective' for the coordination among team members according to the survey. Communal/discussion space is found to be 'effective' but not 'very effective' for coordination among team members. Our analysis also reveals that half-height glass barriers are 'very effective' during the individuals problem-solving activities while working together as a team. Infact, such a physically open environment appears to improve communication, coordination, and collaboration.
Conclusion: According to this study, an open working environment with only half-height glass barriers and communal space plays a major role in communication among team members. The presence of status boards significantly help in reducing unnecessary communication by providing the required information to individuals and therefore, in turn reduce distractions a team member may confront in their absence. As communication plays a significant role in improving coordination and collaboration, it is not surprising to find the effect of open working environment and status boards in improving coordination and collaboration. An open working environment increases the awareness among software developers e.g. who is doing what, what is on the agenda, what is taking place, etc. That in turn, improves coordination among them. A communal/discussion space helps in collaboration immensely.

1. Introduction

Software development vendors as well as corporate Information Technology (IT) departments are under increasing pressure to deliver more, faster, and better [1]. Several surveys [2,3] tend to indicate that current levels of achievement are below expectations, and that users and customers are dissatisfied. Recently, agile software development methods have become popular because software needs to be developed in a short period [4]. In agile methods, developers should achieve short development periods by dint of eliminating development documents; detailed design documents, detailed specification documents; detailed quality reports, and progress reports [5]. Therefore, communication among developers becomes very significant in agile methods. Good communication results in high quality products and good progress [5].

The processes of communication, coordination, and collaboration are at the heart of, and key enablers of, software development processes [6]. Communication is the imparting or interchanging of thoughts, opinions, or information by speech, writing, or signs. Coordination is the harmonious adjustment or interaction of different people or things to achieve a goal or effect. In a well coordinated project, every team member is well aware of the tasks

* Corresponding author. Tel.: +90 312 5868379; fax: +90 312 5868091.
E-mail addresses: deepti@atilim.edu.tr (D. Mishra), alok@atilim.edu.tr (A. Mishra), ostrovska@atilim.edu.tr (S. Ostrovska).

0950-5849/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.infsof.2012.04.002

he/she is expected to fulfill and the timeline to do so. He/She also understands the relationship between what they do and what the coordinated whole achieves. Coordination among team members ensures efficiency of the project by minimizing the redundancy and overlapping of the work carried out by them. In turn, collaboration is working together to accomplish a task and discussing with each other the different approaches to solve difficult problems. Effective collaboration includes both individual-focused tasks and interactive group work. In collaboration, people with complementary skills come together to solve a complex problem which none of them could do individually. Given this background, communication seems to be an essential component of all software development coordination and collaboration practices and processes.

Face-to-face communication is found to be the most effective among various channels of communication as it provides instant feedback and multiple cues like expression, emotions, and personal focus [7,8]. The knowledge acquired through face-to-face communication can be contained for only a limited time, after which it starts to diminish gradually. Therefore, tools such as papers, whiteboards, etc., may be used to store information planned for future use. These tools are also useful to access information about the project when many individuals are working together on one project, or when multiple teams (consisting of many individuals) are simultaneously working on different parts of the same project and coordination among them, hence, becomes paramount. Small teams are more effective in coordination, communication, and collaboration than large ones [9,10].

It has been commonly acknowledged that the physical environment of the workspace can affect communication, coordination, and collaboration in many different ways. To maximize efficiency, e.g. individuals working in the same team can be placed adjacent to each other so that they can interact effectively when required, to achieve clarification. Also, workspace should be designed in such a way that it should not hinder the need to work effectively by oneself. It means that, workspace environment should not only be designed to improve communication, coordination and collaboration, but also to provide an environment to enable individuals to work with ease. There have been few studies carried out so far that focus on the significance of the physical design environment at workspace for effective communication, coordination, and collaboration – which are the backbone of software development, especially those using agile methods [10–13]. The tight social and technical cohesion found in mature agile teams is not disputed, but understanding how and why the set of practices support each other, or the effect of implementing only one or a few practices is little understood [15]. A collaborative work environment requires spaces, furnishings, and technologies that support both individual focus and group interaction, while also facilitating transitions between these activities [13].

In their work on collaboration and co-ordination among mature eXtreme Programming (XP) teams, Sharp and Robinson [14] studied the significance of the physical medium for story cards and the wall in an XP team and discussed the considerations that need to be taken into account for the design of technology to support the teams. Later on, Sharp et al. [15] in another study observed the role of physical artefacts in agile software development and explored the use of story cards and the wall from notational and social perspective that how they complement each other. Recently, Sharp and Robinson [16] discussed the three 'Cs' (communication, coordination, and collaboration) of agile development and their support through story cards and the wall. In this paper, the authors empirically investigate the physical workspace environment of a two complex projects – supply chain management and customer relationship management – in a software development organization, and explain how different individuals (developers, business experts, customers, etc.) working on these projects communicate, coordinate, and collaborate effectively without disrupting other individual focused tasks for the purpose of successfully completing the project by adopting agile methodologies. Thus, the present work is intended to complement and extend previous studies by Sharp and Robinson to a certain extent. To the best of our knowledge, the most related work in this field has been carried out by Mishra and Mishra [17,18] who studied the significance of the workplace environment on communication, coordination, and collaboration collectively. The study found that appropriate workspace environment has a positive effect on communication, collaboration, and coordination in small organizations developing softwares using eXtreme Programming (XP) [18]. The present study extends Mishra and Mishra's previous work on the effect of physical environment on agile software development. The main distinctions between the former study and the present one are (i) The impact of physical environment on different constituents of communication, collaboration, and coordination are evaluated individually; (ii) different questions are formulated in order to study the different aspects of communication, coordination and collaboration exclusively in agile software development; and (iii) studies are performed on software projects in different organizational setting than earlier one. As mentioned earlier, regarding the motivation behind this work, the authors intend to answer the following questions for each of the three 'Cs':

- What is the impact of different constituents of the physical environment on communication itself?
- How do different factors within the physical environment affect coordination itself?
- Which factors comprising physical environment affect collaboration among developers itself?

The paper has been organized in the following way: In the next section, literature review has been provided as to the importance of communication, coordination, collaboration, the role of high-awareness environment, artifacts, and physical set-up towards improved communication, coordination and collaboration. The following section illustrates the case study conducted in software organization, followed by the section presenting research methods, data gathering, and analysis. Section 5 presents the discussions and the final section infers the conclusions in this area.

2. Literature review

2.1. Significance of communication

Software developers spend their considerable time in communicating [19]. Communication is essential to transfer critical project information [20]. Patel et al. [21] found face-to-face communication as primary means (79%) in comparison to email (17%) or telephone (4%). This is further supported by Cockburn [10] that the most effective form of communication (for transmitting ideas) is interactive and face-to-face communication. Also, Korkala et al. [22] described that face-to-face communication is identified as the most efficient means of communication between the participants. Moreover, the daily collaborative work of business people and developers demands efficient verbal communication between the customer and developers [22]. In complex situations, communication effectiveness is particularly critical to project success where multiple and integrated stakeholder teams are involved and where 'time to market' and project efficiency are key drivers [23]. This is supported by many researchers in their studies [11,24]. Kraut and Streeter [25] argue that main characteristics of software development like size, interdependence and uncertainty also need extensive support for informal or interactive communication. Software developers face problems when integrating different components from heterogeneous environments, developers engage in direct or indirect communication, either to coordinate their activities or to acquire knowledge of particular aspects of the software [26].

Focusing on skills, communication, and community allows the project to be more effective and more agile than focusing on processes. People's individual skills and collaboration, conversations, and communications between them enhance flexibility and innovation [27]. Patel et al. [21] in their study of data collected from 62 organizations found that agile values and principles related to communication, collaboration and team involvement are perceived as important and are being practiced widely. Agile methods work with volatile requirements and embrace personal communication between the participants [28]. Since the level of ambiguity can be considered high in the form of unstable requirements, it seems that personal communication is a natural choice for agile development [22]. Communication in agile development is both crucial but also tacit, informal and predominantly verbal [16]. Melnik and Maurer [29] discussed the role of conversation and social interactions as the key elements of effective knowledge sharing in an agile process. They concluded that the higher the level of abstractionism (complexity), the more the need for interactive knowledge sharing via direct verbal communication is [29]. Intuitively collaboration and co-ordination depend on communication, and communication – in one form or another – is central to successful software development [10,30].

2.2. Relevance of co-ordination

Research in software engineering coordination has examined interactions among software developers [31,32] and how they acquire knowledge [33]. Prior studies have investigated ways for effective coordination in software development [34,20]. Nelson et al. [20] summarize communication, shared knowledge, and mutual influence as three determinants of coordination. The ability to co-ordinate has been shown to be an influential factor in customer satisfaction [25] and improves the capability to produce quality work [35]. Annu et al. [34] suggest four factors for success intergroup coordination based on the practical experiences in a CMM-5 company: stakeholder commitment, effective communication, milestone hitting, and mutual support. Mutual influence in software development is mainly derived from the relationship between customers and developers [36]. Intergroup coordination is a key process area in the Capability Maturity Model for software (CMM-SW) [37] and is critical for the success of software projects [38]. Curtis et al. [9] consider communication and coordination breakdowns to be in the top three salient problems in software design process. Herbsleb and Grinter [39] investigated the influence of coordination on integrating software modules through interviews, and found that processes, as well as the willingness to communicate directly, helped teams integrate software. de Souza and Redmiles [40] found that implicit communication is important to avoid collaboration breakdowns and delays. Empirical studies suggest that both formal communication, such as status review meetings, design review meetings, code inspections, and informal communication (such as group meeting) all play an important role in intergroup coordination [25].

2.3. Purport of collaboration

Software engineering (SE) by its very nature is a complex cooperative process, which requires the collaboration of stakeholder teams (e.g. managers, end-users, designers, etc.) [41]. As a matter of fact, formal and informal communication require more than 50% of software engineer's time [19] and collaborative activities take up to 70% of software developer's time [42]. Collaboration among participants has two major forms: one is concrete working together to accomplish a task, the other is discussing with each other to solve some difficult problems [43]. Collaboration has become the essential part of software development [44], and knowledge collaboration has become an essential activity in the development of modern complicated software systems due to not only the required human hands in coding but also the many kinds of knowledge [45]. Dube and Pare [46] find that most teams rely to some extent on face-to-face meetings to ease the process of collaboration and coordination. Face-to-face meetings are also seen as more appropriate for new ideas and strategic thinking [8]. Agile practice is strongly collaborative in its outlook, favoring individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan [47]. The need for agile developers to work collaboratively is repeated in several agile seminal works e.g. in [27,48,49] but there has been limited research in the area [16]. Ambler [50] considers agile quality to be a result of practices such as effective collaborative work, incremental development and iterative development as implemented through techniques such as refactoring, pair-programming, test-driven development, modeling, and effective communication techniques. Bryant et al. [51] examined collaboration in pair programming and concluded that pairing is highly collaborative with both partners contributing new information to almost every subtask. The reliance on collaboration and communication within an agile team goes beyond pairing and is a fundamental characteristic found throughout the approach [14]. Agile software development approaches tend to rely on effective personal communication and collaboration between the project participants [28,10]. Members of an agile team collaborate on wide range of activities: estimating stories, implementing stories, developing and running tests, planning an iteration, talking to customers, solving problems, and so on. In fact many agile practices promote a collaborative environment [16].

2.4. Influence of awareness in communication, coordination and collaboration

Information arises directly out of each person's activity, rather than having to be managed explicitly; awareness information does not need to be sought out [52]. There is nothing paradoxical in being engrossed in one line of action and simultaneously making sense of and taking heed of what goes on beyond one's immediate line of action [53]. Competent practitioners are able to align and integrate their activities because they know the setting, they are not acting in abstract space but in a material environment which is infinitely rich in cues [54,55]. In fact, by somehow displaying his or her actions, the actor is always, in some way and to some degree, intending some effect on the activities of colleagues [53]. Actors display those aspects of their activities that may be of relevance to their colleagues [53]. Awareness, a key concept provides a mechanism that facilitates the coordination of the work among social actors [56]. Strong awareness helps coordination [57]. Empirical evidence from previous work reveals the collaborative nature of the process of designing and building software systems. For instance, Perry et al. [19] investigated how software development activities were influenced by the technology. Goncalves et al. [58] in their studies observed that formal and informal communication encompass over 50% of a software developer's time, and much of this interaction among developers aims to allow one to be aware of their colleague's work.

2.5. Effect of artifacts in communication and coordination

Although verbal interaction is the primary channel for communication [59], it is only one of a number of channels used for communication. According to Finsterwalder [60] verbal communication is not always well thought-out and the contents of the conversation can be forgotten in time. Also, unnecessary verbal interaction can result in loss of important time of the people who are interacting and may create disturbance and distractions for other people present in the same room. So, it is effective to use face-to-face communication for acquiring knowledge and clarifying issues related to the current task. But if the information gathered is going to be used in the future or when the requested information does not require further analysis (ex: facts such as status of project), it can be acquired effectively without using verbal communication with the use of information-rich tools like whiteboards, electronic display, flipcharts, etc. Also, when number of people are working in a team and/or many teams concurrently working on the different part of a product, they must have continual access to the latest information related with the project for effective coordination.

Traditionally, handwritten whiteboards [61], electronics displays and other such tools play a significant role in inter and intra group communication and coordination. Important information can be made visible or available to everyone with the help of these tools. Their contribution to other fields such as hospitals/trauma centers is stated in number of studies [62–66] where communication and coordination between different medical staff is utmost important.

2.6. Small teams and their impact on software productivity and quality

The issue of software productivity and software quality haunts the software community perpetually with no single solution [67]. Boehm [68] found that the effect of tools is relatively small, whereas the impact of people and organization is significant on the software productivity. Curtis et al. [9] have studied software development using a layered behavioral model and concluded that systems must be treated as learning, communication, and negotiating process. The common denominator of all these three parts is a small team. They also found that design collaboration is more effective when small teams are formed, and this collaborative problem solving is related to productivity. The communication remains manageable in small teams and small teams suffer less from conflicting views. A quality product also results when the application knowledge is widely spread across the team and with smaller teams, it is easier to propagate this knowledge [67].

2.7. Physical set up in organization

Office layout or physical space also plays an important role in effective communication. If all team members are physically scattered or if a business expert is not available physically in person to development teams, then communication will suffer and this will affect the quality and productivity of the product. There are very few studies done so far that observed the importance of physical space for effective communication in software development organization which is the backbone of software development, especially, those using agile methods. Cockburn [10] described three separate effects that office layout has on communication costs within a project:

- The reduction in cost when people discover information in background sounds (osmotic communication)
- The overall cost of detecting and transferring information (erg-seconds)
- The lost opportunity cost of not asking questions

These three magnify the effects of distance in office seating. People sitting close by each other benefit in all three effects, people sitting in separated locations suffer in all three [10]. Deliberate movement to engage someone in conversation drops off dramatically after 30 meters [13]. Sharp and Robinson [12] described physical setting of a project using extreme programming. Beck [11] described the ideal work space for XP is to have one big room with little cubbies around the outside and powerful machines on tables in the middle. He also stressed that different teams should be separated from each other. Cubicle walls should of half-height or eliminated entirely [11]. A common workplace solution to enhance collaboration is the provision of informal group spaces adjacent to or interspersed among personal workstations. The spaces have moveable furnishings, are located in the open and are often shared by different work groups. The intent of the space is to support spontaneous meetings and informal work [13].

3. Case study

Case study is a suitable research methodology for software engineering research since it studies contemporary phenomena in its natural environment. The case study methodology is well suited for many types of software engineering research, as the objects of study are contemporary phenomena, which are hard to study in isolation [69]. The term case study is used in parallel with terms like field study and observational study, each focusing on a particular aspect of the research methodology. For example, Lethbridge et al. [70] use field studies as the most general term.

In this section, the physical environment of a small scale software development organization is represented. It is illustrated how this environment facilitates communication, collaboration and coordination among team members. There were around 19 members in the first project (Supply Chain Management) consisting of 2 teams: Team 1 (8 members) and Team 2 (11 members) respectively. Supply Chain Management, which is complex software, was developed by using agile methodologies. There was an existing architecture which was open to evolve as a result of customer's feedback from future iterations. Another project was related with customer relationship management consisting of around 18 members in the project working in two teams in the same organization. This project was also done in the similar physical environment and by using agile methodologies.

There were three rooms situated on the same floor as shown in Fig. 1. There were two development teams, Team 1 and Team 2, simultaneously working on different parts of the project and developing software that was going to be launched in the market. There were no real customers but there were some prospective customers. In general, the software developers, except one, did not have any prior experience in the application area. This person acted as business expert for this project. Initially, prospective customers were interviewed to gather the scope and initial requirements of the product in the form of stories. Later, both development teams, prospective customers and a business expert took part in a series of workshops. These workshops helped in defining a common vision among all team members and also to collect additional requirements and refine old ones.

Each development team occupied a separate room but they were close enough in adjacent rooms to communicate easily whenever required. Beck [11] also supports the separation of different teams. The third room was the meeting room. In the first room, Team 1 of technical developers sat in the cubicle desks grouped in pairs. Every cubicle was separated by another cubicle with half-height glass barriers. In the second room, a business expert, another team of developers and other non-technical engineers were available. The third room was used for critical release, iteration and other meetings. This was the meeting room where the two development teams could discuss something together if required. This was also the room of the product owners.

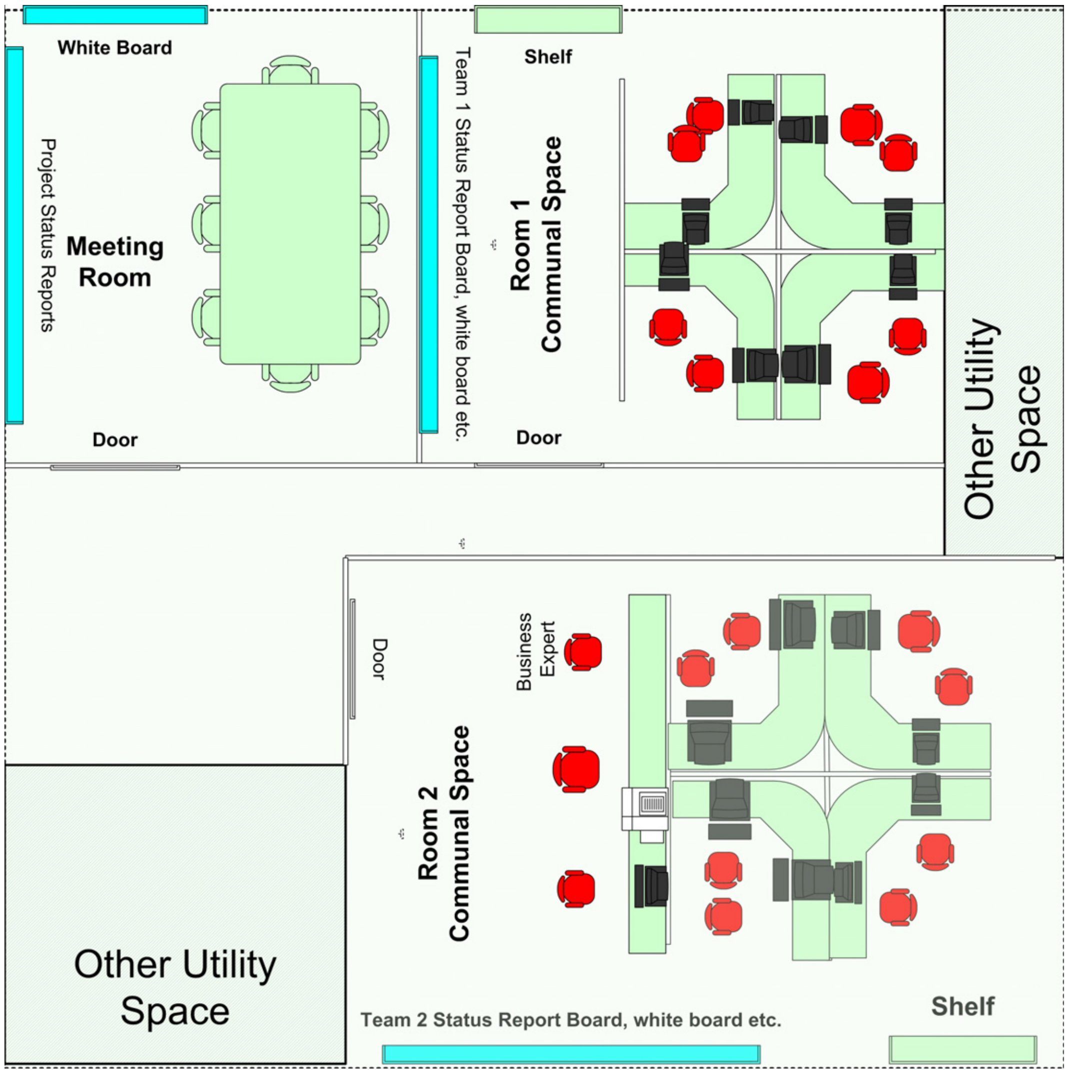

Fig. 1. Office layout.

In the first room as shown in Fig. 1, near the entrance is the communal area which was used for standup meetings by Team 1. In the communal area, three boards (one white board, one board showing the status of work of Team 1 and one extra board for sticking other important notes on) were located on one wall. At the far end after the communal area, each pair of developers sat at the desks that were separated just by a half height glass barrier. This arrangement increased team member's awareness of what is going on around them without using focused attention, since visual and aural accessibility are key environmental contributors to workspace awareness [71]. According to Gutwin and Greenberg [71], although overhearing conversations can be distracting, they maybe most valuable when work is highly interdependent and when the collocated people are working on the same or similar projects. The structure and seating arrangement of room 2 was similar to room 1 except a business expert was sitting near the communal area at the entrance of the second room. In this way, Team 1 members could talk to the business expert without disturbing Team 2 members.

Key problems of high-awareness environments include loss of privacy, loss of confidentiality, distractions and interruptions [72]. However, there is some indication that a highly open environment might lead to reduced interruptions and distractions due to greater availability of non-verbal and behavioral cues that modulate interaction [73]. When people are focused on an individual task, their posture, eye gaze and demeanor indicate that they are not available for conversation. However, if they look up, make eye contact or walk around, others are more likely to perceive them as available for interaction. Availability is largely determined by whether or not the person appears to be involved in focused work [13]. So, with this physical arrangement of workspace, Team 1 members could communicate with each other whenever the need arose but they also had their private space where they could concentrate on their individual task without interruption. Each one can see whether the other person is busy doing something or free before interrupting. The availability of individual workspaces that aid focused attention and reduce distractions and interruption has numerous benefits [13], including increased time on individual tasks [74], reduced stress [75], improved performance on mental tasks [76], and the ability to maintain one's line of thought and cognitive flow [77].

Each cubicle was shared by two development team members according to their roles and responsibility in the project (i.e. these two developers were working on the same part of the product or they are probably pair programmers for critical modules). The furniture was arranged in such a way that development teams can have a stand-up meeting or discussion whenever they wanted onto a white board and also could use the walls efficiently to put

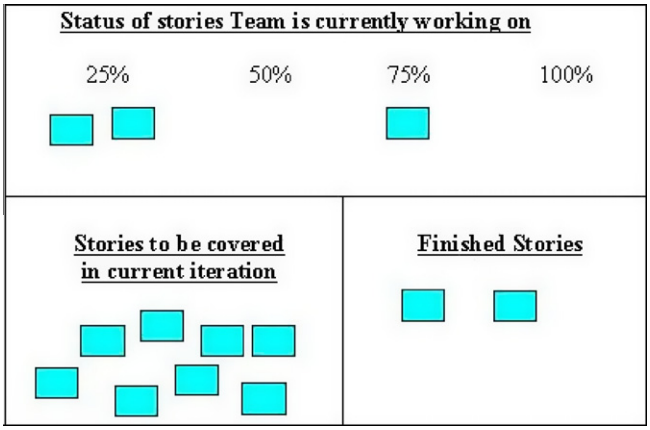

Fig. 2. Status of current iteration of a team.

diagrams, notes, module specifications, working plans and any other documents that needed to be seen by other team members. It may be increasingly important to have artifacts and visual displays of information to aid coordination, memory and understanding the work as a whole because knowledge work activity is largely cognitive and does not leave physical traces [78,79]. Sharp and Robinson [14] further analyzed the teams and highlighted the role of physical artifacts in co-operation and collaboration activities.

In the communal area, one white board, one more board showing the status of work which Team 1 was doing, plus one extra board for sticking other important notes needed by Team 1 were fixed. The white board was erasable and was used for discussion or elaborating any story. Other board as shown in Fig. 2, showing the status of the current iteration had three sections. User stories, which Team 1 was working on in current iteration, were written on a small piece of paper and stuck on this board. User stories are the mechanism in agile development by which user requirements are captured [15]. Each story was written on a small piece of paper to ensure that it did not grow too large and could be implemented in a single iteration.

On the upper part of the status board, there was a range from 0% to 100%. Stories, this team was currently working on, were stuck on this part according to the percentage of work done on them so far. The bottom left corner had an area displaying stories that need to be included in this iteration but the team had not yet started working on them. The bottom right corner contained stories that were already completed in current iteration. The other team in the second room also had these three boards showing Team 2s' work. Anyone could find out about the status of the work of these two teams without disturbing the team members.

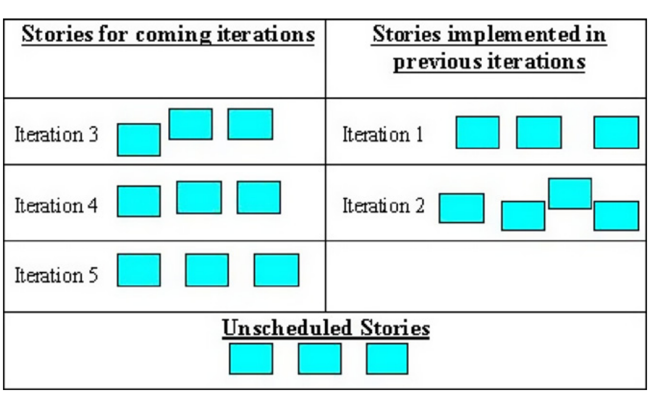

Fig. 3. Project status board.

The meeting room as shown in Fig. 1 was used for critical release, iteration and other meetings. Here both teams' members, a business expert and other stakeholders met before each iteration or release or whenever a demo version was shown to prospective customers. This room was also used when both teams along with a business expert wanted to discuss something important altogether. In this meeting room, there was a whiteboard for brainstorming and another big board showing the complete information about the status of the whole project. This board as shown in Fig. 3, was also divided into three parts – one part containing stories stuck in rows, each row representing subsequent iterations. The second part contained stories completed so far along with the iteration number they were implemented in. There is one section for emergent or unscheduled stories which were needed to be handled urgently.

4. Research methods

The present study performs a systematic analysis to investigate the impact of selected constituents of physical environment on communication, coordination and collaboration among software developers using agile software development. In terms of research methods, the present study combines survey based on questionnaire along with case study in the organization. To a certain extent, it has also adopted ethnographically informed approach in which the nature of practice is not known a priori, and the researchers avoid any form of control, intrusion or experimentation [80]. In this way the researchers did not intervene in the teams' day-to-day working, but instead joined the project as a passive observer-attending meetings, interacting with developers and other team members, having informal discussions, etc. [16]. Later on, survey was conducted to understand respondents' views regarding effect of physical environment on communication, coordination, and collaboration. This was followed by semi-structured interviews in order to gain further clarification.

A survey instrument aimed to determine how small organization's physical environment facilitates communication, collaboration and coordination among team members has been developed. This pilot instrument was distributed to a group of software professionals from different organizations to get their suggestions and clarifications. Several software professionals were also interviewed in this regard. We included these suggestions in the research instrument as much as possible. Finally, 37 copies of the revised questionnaire were distributed to the software developers in a software development organization. 30 people out of 37 responded. So, the response rate was 81 percent. The questionnaire was based on Likert scale ('strongly agree', 'agree', 'neutral', 'disagree', and 'strongly disagree'). The following constituents of physical environment have been examined: half cubicles, status boards, whiteboards, communal/discussion space, and the distance between multiple teams. Their impact on factors under study, i.e. communication, coordination, and collaboration, has been described in terms of such categories as 'very effective', 'effective', 'no effect' and 'of negative effect', which are defined below. The research models shown in Figs. 4–6 present the association between constituents of physical space and the affected factors of the software development process. Research questions are shown in table 1.

To derive conclusions on the impact of various constituents of physical environment on the factors under study, the tests on the population proportions of the respective affirmative answers have been conducted. More precisely, we suggest that a selected constituent is '*very effective*' for a specified factor if it is strongly supported (i.e. the corresponding answers are 'strongly agree') by more than 50% of the population of software developers. More formally, if the probability p of the 'strongly agree' response exceeds

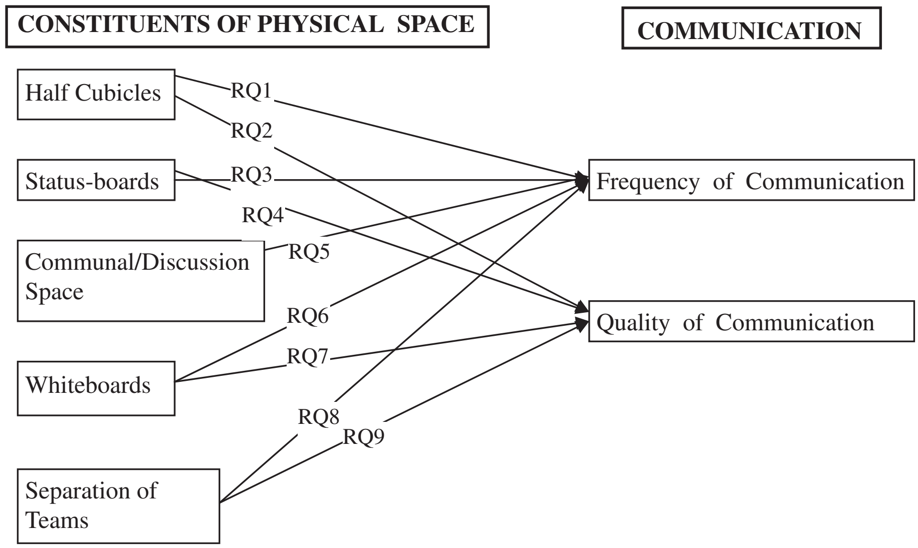

Fig. 4. Research model to study communication.

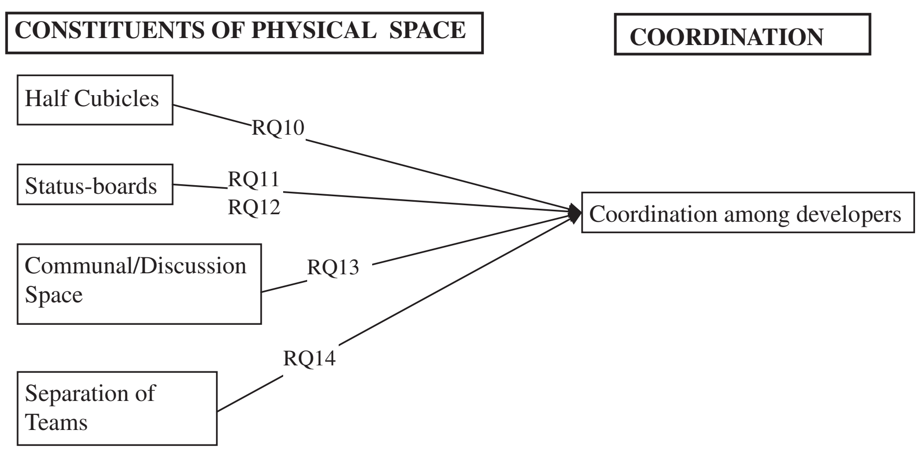

Fig. 5. Research model to study coordination.

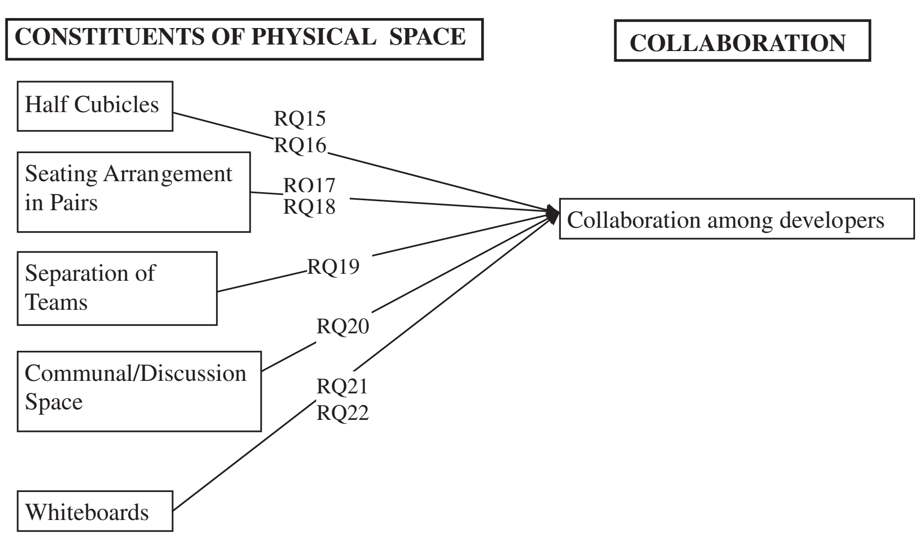

Fig. 6. Research model to study collaboration.

0.5. The respective procedure requires testing the null hypothesis $H_0 : p = 0.5$ against the one-sided alternative $H_1 : p > 0.5$. Whenever the test confirmed a constituent to be 'very effective', investigation was stopped. However, if, according to the test, this has not

Table 1
Research questions.

Q.	Research question	Constituents	Factors
1	Is half cubicle *effective* for the frequency of communication?	Half cubicles	Communication
2	Is half cubicle *effective* for the quality/effectiveness of communication?		
3	Are status Boards *effective* for the frequency of communication?	Status boards	
4	Are status Boards *effective* for the quality/effectiveness of communication?		
5	Is communal space *effective* for the frequency of communication?	Communal space	
6	Do team members very often use whiteboards while communicating with each other?	Whiteboards	
7	Are whiteboards *effective* for the quality/effectiveness of communication?		
8	Does separation of teams make *negative effect* on the frequency of communication?	Physical separation of teams	
9	Does separation of teams make *negative effect* on the quality/ effectiveness of communication between teams?		
10	Is half cubicle *effective* for the coordination among team members?	Half cubicles	Coordination
11	Are status boards *effective* for the coordination among team members?	Status boards	
12	Are team members *positive* about updating the status boards because benefits outweigh efforts of updating?		
13	Is communal/discussion space *effective* for the coordination among team members?	Communal space	
14	Does separation of teams make *negative effect* on the coordination among different teams?	Physical separation of teams	
15	Is half cubicle *effective* for the collaboration among team members?	Half cubicles	Collaboration
16	Does half cubicle make *negative effect* on the collaboration among team members due to additional obstacle of noise?		
17	Is seating arrangement in pairs *effective* for working on joint tasks?	Seating arrangement in pairs	
18	Is seating arrangement in pairs *effective* for working on individual tasks?		
19	Does separation of teams make *negative effect* on the collaboration among different teams?	Physical separation of teams	
20	Is communal space *effective* during problem solving activity together as a team?	Communal space	
21	Do team members often use whiteboards during problem solving activity together as a team?	Whiteboards	
22	Are whiteboards *effective* during problem solving activity together as a team?		

been established, we try to check a weaker statement. It has been suggested that a selected constituent is '*effective*' for a specified factor variable if it is supported (with answers 'agree' or 'strongly agree') by more that 50% of the population of software developers. In other words, it is the probability $\tilde{p}$ that a person admits but not necessarily strongly the positive affect of a selected constituent on the specified factor (communication/coordination/collaboration). In this case, the rigorous procedure requires testing the null hypothesis $H_0 : \tilde{p} = 0.5$ against the one-sided alternative $H_1 : \tilde{p} > 0.5$. Due to the fact that $\tilde{p} \geqslant p$, the conclusion $p > 0.5$ makes the test on $\tilde{p}$ redundant. Apart from characterizing a constituent to be '*effective*' or '*very effective*', it may be appropriate to categorize it as either having '*no effect*' or possessing '*negative effect*'. To be specific, we state that a constituent is having '*no effect*' if it is supported by half of the software developers' population. Furthermore, a constituent is said to be of '*negative effect*' if the population proportion of developers who recognize its malign influence on the software development process exceeds ½.

Mostly, the sample size allows us to use *Z*-test since the conditions for the applicability of the Moivre-Laplace Theorem have been satisfied (see Ovedovitz [81], Ch. 5.5, p. 184 and Trivedi [82], Ch. 4.7, p. 241). However, as we disregard all "neutral" responses to the questionnaire and, as a result, decrease sample size, it seems reasonable to carry out the test based on the binomial distribution along with *Z*-test with the same level of significance $\alpha = 0.05$. The corresponding critical value for *Z*-test is $z_\alpha = 1.645$, while critical values for the binomial test are given in Table 2 below. The table represents the statements of hypotheses relevant to each of the research question, necessary sample data (sample size and sample proportion) and the results of the testing procedure.

Meanwhile, the study of RQ 8, 9, 14, 18, and 19 should be organized in a slightly different manner. These questions are concerned with the impact of separation of teams and seating arrangements on the dependent variables. The obtained responses reveal that an essential part of the software developers are 'neutral' about the subject, while the number of supporters and opponents are rather close. Therefore, an appropriate procedure for the situation is testing an alternative hypothesis $H_0 : \rho = 0.5$ against the two-sided alternative $H_1 : \rho \neq 0.5$, where ρ is the probability that the separation of teams (or seating arrangements in RQ18) has negative but not necessarily very negative effect. After we disregard the 'neutral' answers, the sample size for the rest gets rather small. Therefore, we did not refer to the limit theorem here. Rather, the tests are based directly on the binomial distribution, where the assumed value $p_0 = 0.5$.

We summarize our finding in the following table 3.

5. Discussion

The following are the results of the study:

5.1. Effect of the physical environment on communication

According to Table 2, it is found that half cubicles are 'very effective' for the frequency of communication. Further, they are considered 'effective' but not 'very effective' for the quality/effectiveness of communication. These half-height glass barriers were considered to make the development team members more visible and reachable by each other which increase the frequency of communication. Also, they can exchange notes, see each other's computer screens, etc. – which was believed by the individuals to make their communication much richer.

Status boards were considered to help in the non-verbal communication of information and found to be 'very effective' for the frequency of communication. However, these boards are only 'effective' but not 'very effective' for the quality/effectiveness of communication.

Discussion space is found to be 'effective' but not 'very effective' for the frequency of communication. As the space for discussion has been made available within the working room, individuals can have a stand up meeting anytime and discuss matters if and when required without disturbing people from other teams. It also was considered to be beneficial in informal communication as that is where

Table 2
Hypotheses and observed values of the test statistics.

RQ	Hypothesis (very effective)	Sample size n	Observed value of k	Sample proportion $\bar{p}$	Critical value k_α	Z_{obs}	Hypothesis (effective)	Observed value of k	Sample proportion $\bar{p}$	Critical value k_α	Z_{obs}	Result
RQ.1.	H_{01}: $p = 0.5$ H_{11}: $p > 0.5$	28	23	0.821	20	3.397	–				–	Reject H_{01} in favor of H_{11}
RQ.2.	H_{02a}: $p = 0.5$ H_{12a}: $p > 0.5$	26	14	0.538	17	0.38	H_{02b}: $\tilde{p} = 0.5$ H_{12b}: $\tilde{p} > 0.5$	25	0.961	17	4.70	Accept H_{02a} against H_{12a} Reject H_{02b} in favor of H_{12b}
RQ.3.	H_{03}: $p = 0.5$ H_{13}: $p > 0.5$	28	22	0.786	20	3.016	–				–	Reject H_{03} in favor of H_{13}
RQ.4.	H_{04a}: $p = 0.5$ H_{14a}: $p > 0.5$	24	9	0.375	17	0.853	H_{04b}: $\tilde{p} = 0.5$ H_{14b}: $\tilde{p} > 0.5$	20	0.833	17	3.26	Accept H_{04a} against H_{14a} Reject H_{04b} in favor of H_{14b}
RQ.5.	H_{05a}: $p = 0.5$ H_{15a}: $p > 0.5$	22	13	0.591	16	0.853	H_{05b}: $\tilde{p} = 0.5$ H_{15b}: $\tilde{p} > 0.5$	21	1	16	$+\infty$	Accept H_{05a} against H_{15a} Reject H_{05b} in favor of H_{15b}
RQ.6.	H_{06a}: $p = 0.5$ H_{16a}: $p > 0.5$	27	16	0.592	19	0.961	H_{06b}: $\tilde{p} = 0.5$ H_{16b}: $\tilde{p} > 0.5$	24	0.889	19	4.0426	Accept H_{06a} against H_{16a} Reject H_{06b} in favor of H_{16b}
RQ.7.	H_{07}: $p = 0.5$ H_{17}: $p > 0.5$	29	23	0.793	20	3.1557	–				–	Reject H_{07} in favor of H_{17}
RQ.10	H_{010}: $p = 0.5$ H_{110}: $p > 0.5$	28	22	0.786	20	3.026	–				–	Reject H_{010} in favor of H_{110}
RQ.11	H_{011}: $p = 0.5$ H_{111}: $p > 0.5$	27	23	0.852	19	3.63	–				–	Reject H_{011} in favor of H_{111}
RQ.12	H_{012a}: $p = 0.5$ H_{112a}: $p > 0.5$	28	17	0.607	20	1.13	H_{012b}: $\tilde{p} = 0.5$ H_{112b}: $\tilde{p} > 0.5$	27	0.964	20	4.91	Accept H_{012a} against H_{112a} Reject H_{012b} in favor of H_{112b}
RQ.13	H_{013a}: $p = 0.5$ H_{113a}: $p > 0.5$	25	14	0.560	17	1.00	H_{013b}: $\tilde{p} = 0.5$ H_{113b}: $\tilde{p} > 0.5$	24	0.960	17	4.6	Accept H_{013a} against H_{113a} Reject H_{013b} in favor of H_{113b}
RQ.15	H_{015}: $p = 0.5$ H_{115}: $p > 0.5$	26	20	0.769	17	2.65	–				–	Reject H_{015} in favor of H_{115}
RQ.16[a]	H_{016}: $p = 0.5$ H_{116}: $p > 0.5$	20	18	0.900	14	3.57	–				–	Reject H_{016} in favor of H_{116}
RQ.17	H_{017}: $p = 0.5$ H_{117}: $p > 0.5$	27	25	0.926	19	4.41	–				–	Reject H_{017} in favor of H_{117}
RQ.20	H_{020}: $p = 0.5$ H_{120}: $p > 0.5$	28	19	0.679	20	1.88	–				–	Reject H_{020} in favor of H_{120}
RQ.21	H_{021}: $p = 0.5$ H_{121}: $p > 0.5$	29	21	0.724	20	2.369	–				–	Reject H_{021} in favor of H_{121}
RQ.22	H_{022}: $p = 0.5$ H_{122}: $p > 0.5$	28	24	0.857	20	3.78	–				–	Reject H_{022} in favor of H_{122}

[a] p is the population proportions who do not agree that half cubicle make *negative effect* on the collaboration among team members due to additional obstacle of noise.

different issues and concerns can be talked about. Also, status boards were situated in this space which helped in the non-verbal communication of important information. A majority of the software developers stated that they often used whiteboards while communicating with their team members. Whiteboards were also believed in illustrating their thoughts in a much better way. The presence of whiteboards in discussion space further enriched verbal communication, and face-to-face communication with whiteboards helped the developers to clarify some complex issues that could not be addressed merely by means of only verbal face-to-face communication. It is also stated by Ambler [83], face-to-face communication at whiteboard is a ''hotter'' communication media than face-to-face communication. As a result, whiteboards are regarded as 'very effective' for the quality/effectiveness of communication.

Surprisingly, separating the teams proved to have 'no effect' on the frequency and effectiveness of communication among the teams. The reason possibly being that the projects carried out so far had been modular in nature and these projects had well-defined parts that were not closely interrelated. Also, the rooms where these teams were located were close enough and, as such, did not impede communication among them. An additional third formal meeting room was used for meeting between these teams

Table 3
All tests are: H_0: $\rho = 0.5$, H_1: $\rho \neq 0.5$, with the level of significance $\alpha = 0.05$.

Question	Sample size n	Critical values k_1 and k_2	Observed value k	Decision
RQ8	21	4 and 17	13	Accept H_0 against H_1
RQ9	18	5 and 13	12	Accept H_0 against H_1
RQ14	24	5 and 19	14	Accept H_0 against H_1
RQ18	13	3 and 10	6	Accept H_0 against H_1
RQ19	16	4 and 12	10	Accept H_0 against H_1

but in the case of projects with low modularity and highly inter-related parts, separation of teams is possible to pose challenges in communication, specifically if the rooms of these teams are not adjacent (ex, on different floors).

5.2. Effect of the physical environment on coordination

According to Table 2, it is found that half-height cubicles and status boards are 'very effective' for coordination among team members. Software developers reported that these cubicles increased their awareness about the status of the inter-related tasks different team members are performing, while status boards increased their understanding of inter-related tasks and each person's role/responsibility. The status board showed the information about each team's work for a single iteration and also information about the project as a whole. It was added that such information helped to minimize redundancy and overlapping of work, while also informed developers about their responsibilities and schedules to coordinate their tasks so as to achieve their final goal.

As part of survey and regarding whether the benefits outweighed the efforts to update the status boards since, eventually, developers themselves were doing this updating, a majority of the software developers agreed that the benefits were far more when compared to the efforts spent in updating the status boards.

Communal/discussion space is found to be 'effective' but not 'very effective' for coordination among team members. Prior to every iteration, each team conducted a stand-up meeting at this communal space to discuss the tasks to be undertaken throughout the iteration. The work to be done and task division was set out and agreed upon during such gatherings. Therefore, this communal area was believed to play a very important role in the coordination of tasks prior to an iteration. Also, this information regarding such decisions and agreements was reflected on status boards in this area which further helped in coordination during an iteration.

According to Table 3, it has been observed that locating the teams in different rooms has 'no effect' on their coordination. This may be due to the fact that every team had a status board to show the work to be done in the current iteration and the work finished so far. These iterations had short cycles, typically ranging from 7 to 15 days. After each iteration, a meeting was held to decide and reset new targets and delegate individuals for the tasks. Consequently, the status boards played more fundamental and prominent role in coordination during a particular iteration, and direct verbal communication was required to coordinate the work of different developers before an iteration initiated.

5.3. Effect of the physical environment on collaboration

According to the analysis, it is observed that half-height glass barriers are 'very effective' during their problem-solving activity together as a team; such an open environment in fact helped to improve communication. Open communication and conversations among team members played a very significant role in improving collaboration. Also, this open environment was key in creating a feeling of togetherness which is important for collaboration. Since collaboration involves both group tasks and individual focused work, it is important for the environment to provide some means by which people can also do their individual work whenever required. Contrary to what was anticipated, these half-height glass barriers do not increase the noise due to individuals' proximity to one another which may adversely affect their individual focused tasks. In short, these glass barriers, being minimal, helped in group tasks but at the same time did not pose any obstacle in the form of noise during individual focused tasks.

It can be inferred from the study that sharing the same cubicle is 'very effective' during tasks that required working together; for example, when programmers are trying to code a complex problem together or when they are working using pair programming. However, there is no evidence that this arrangement of sharing a cubicle has any effect on their work when they are not collaborating.

Regarding the question whether placing different teams in different rooms made it difficult to solve problems together, it has been observed that separation of teams in different rooms has 'no effect' on their collaboration.

It was noted that communal/discussion spaces are 'very effective' for collaboration among developers. Communal space supports open communication, open-space meetings, brainstorming and conversations (formal or informal) which are very significant factors to improve collaboration. What is more, that developers very often use whiteboards during problem-solving activity together as a team as these are deemed 'very effective' during this activity. Whiteboards played an important role during brainstorming which is an important component of collaboration. They were also believed to be beneficial during stand-up meetings to solve complex issues which is part of collaboration.

6. Conclusion

According to this study, an open working environment with only half-height glass barriers and communal space plays a major role in communication among team members. Status boards help in non-verbal communication of important information and, therefore, reduce unnecessary distractions a team member may face in their absence. There is no impact on communication detected as a result of separating teams from one another.

As communication plays a major role in improving coordination and collaboration, it is not surprising to find the effect of open working environment and status boards in improving coordination and collaboration. An open working environment increases the awareness among developers that, in turn, improves coordination among them. Status boards contribute significantly in the coordination of tasks within a team and among different teams. According to this study, locating teams to different rooms to work does not have any effect on their coordination.

This study showed that an open working environment and a communal space help in collaboration immensely. Whiteboards situated in the communal space also have a major role in improving collaboration. The separation of teams again has no effect on their collaboration as both teams used a third meeting room for solving problems together. Also, both teams' rooms were adjacent to each other and, therefore, were convenient to contact one another within a few seconds. This may be another reason why the teams did not notice any effect of their separation on communication, coordination, and collaboration.

Because of the context-sensitive nature of socio-technical issues, especially with respect to the construction of communication and dependencies, it is difficult to consider any one case study in software development as a benchmark for communication, coordination, and collaboration. As Kwan et al. [57] have stated having an understanding of the context of the project is extremely important when interpreting the results pertaining to such studies [57].

Finally, this study is a single case study and teams being agile teams, the sample size is smaller in comparison to other surveys. Hence, the results can not be generalized, nor can they be directly compared to the existing studies due to the differences in the projects under examination. This study does not include distributed teams, nor does it engage cognitive issues and patterns related with communication, collaboration, and coordination. However, the authors believe that this study advances the theoretical and the empirical examination of the impact of different constituents of physical environment on communication, collaboration, and coordination individually.

References

[1] G. Shroff, A. Mehta, P. Agarwal, R. Sinha, Collaborative development of business applications, in: International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2005.

[2] C. Schwaber, Corporate software development fails to satisfy on speed or quality, Forrester Research 11 (2005) 1–4.

[3] C. Ferruri Ross, 2005 IT service provider scorecard, Forrester Research 28 (2005) 1–7.

[4] L. Williams, A. Cockburn, Agile software development: it's about feedback and change, Computer, IEEE Magazine (6) (2003) 39–43.

[5] N. Hanakawa, K. Okura, A project management support tool using communication for agile software development, in: Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04), 2004.

[6] L. Layman, L. Williams, D. Damian, H. Bures, Essential communication practices for extreme programming in a global software development team, Information and Software Technology (489) (2006) 781–794.

[7] G.M. Olson, J.S. Olson, Distance matters, Human–Computer Interaction 15 (2000) 139–178.

[8] K. Crowston, J. Howinson, C. Masango, The role of face-to-face meetings in technology-supported self-organizing distributed teams, IEEE Transaction on Professional Communications 50 (3) (2007) 185–203.

[9] B. Curtis, H. Curtis, N. Iscoe, A field study of the software design process for large scale systems, Communications of the ACM 31 (11) (1988) 1268–1287.

[10] A. Cockburn, Agile Software Development, Addison-Wesley, Indianapolis, 2002.

[11] K. Beck, Extreme Programming Explained: Embrace Change, Addison-Wesley, Upper Saddle River, New Jersey, 2000.

[12] H.C. Sharp, H.M. Robinson, An ethnography of XP practice, in: Proceedings of the co-located 15th Annual Psychology of Programming Interest Group Workshop and Empirical Assessment of Software Engineering Conference, Keele, April 2003, pp. 15–27.

[13] J.H. Heerwagen, K. Kampschroer, K.M. Powell, V. Loftness, Collaborative knowledge work environments, Building Research & Information 32 (6) (2004) 510–528.

[14] H.C. Sharp, H.M. Robinson, Collaboration and co-ordination in mature extreme programming teams, International Journal of Human–Computer Studies 66 (2008) 506–518.

[15] H.C. Sharp, H.M. Robinson, M. Petre, The role of physical artifacts in agile software development: two complementary perspectives, Interacting with Computers 21 (1–2) (2009) 108–116.

[16] H. Sharp, H. Robinson, Three 'Cs' of agile practice, in: T. Dingsoyr et al. (Eds.), AgileSoftware Development: Current Research and Future Directions, Springer, Berlin, 2010, pp. 61–85.

[17] D. Mishra, A. Mishra, Workspace Environment for Collaboration in Small Software development Organization, in: Yuhua Luo (Ed.), 5th International Conference on Cooperative Design, Visualization and Engineering (CDVE 2008) held at Mallorca, Spain on 21st–25th September 2008, LNCS 5220, 2008, pp. 196–203.

[18] D. Mishra, A. Mishra, Effective Communication, Collaboration & Coordination in eXtreme Programming: Human Centric Perspective in Small Organization, Human Factors and Ergonomics in Manufacturing, vol. 19, John Wiley, 2009. Issue 5, pp. 438–456.

[19] D.E. Perry, N.A. Staudenmayer, L.G. Votta, People, organizations, and process improvement, IEEE Software 11 (4) (1994) 36–45.

[20] K.M. Nelson, D. Armstrong, M. Buche, M. Ghods, Evaluating the CMM level 3 KPA of intergroup coordination: a theory-based approach, Information & Management 1 (3) (2000) 171–181.

[21] C. Patel, M. Lycett, R. Macredie, S. Cesare, Perceptions of agility and collaboration in software development practice, in: Proceedings of the 39th Hawaii International Conference on System Sciences, IEEE Computer Society Press, 2006.

[22] M. Korkala, P. Abrahamsson, P. Kyllonen, A case study on the impact of customer communication on defects in agile software development, in: Agile Conference, 23–28 July 2006, pp. 76–88.

[23] J.J. Elliott, Design of a product-focused customer-oriented process, Information and Software Technology 42 (14) (2000) 973–981.

[24] A. Edstrom, User influence and the success of MIS projects: a contingency approach, Human Relations 30 (1997) 580–607.

[25] R.E. Kraut, L.A. Streeter, Coordination in software development, Communications of the ACM 38 (1995) 69–81.

[26] K. Nakakoji, Y. Ye, Y. Yamamoto, Supporting Expertise Communication in Developer-Centered Collaborative Software Development Environments, Springer Verlag, 2010.

[27] J. Highsmith, Agile Software Development Ecosystems, Addison Wesley, 2002.

[28] P. Abrahamsson, O. Salo, J. Ronkainen, J. Warsta, Agile Software Development Methods: Review and Analysis, VTT Publications, Espoo, 2002. pp. 107. <www.inf.vtt.fi/pdf/publications/2002/P478.pdf>.

[29] G. Melnik, F. Maurer, Direct verbal communication as a catalyst of agile knowledge sharing, in: Proceedings of the Agile Development Conference (Adc'04) – vol. 00, June 22–26, 2004, pp. 21–31.

[30] J.O. Coplien, N.B. Harrison, Organisational Patterns of Agile Software Development, Pearson Prentice Hall, New Jersey, 2005.

[31] S. Carter, J. Mankoff, P. Goddi, Building connections among loosely coupled groups: Hebb's rule at work, Computer Supported Cooperative Work 13 (3/4) (2004) 305–327.

[32] S. Marczak, D. Damian, U. Stege, A. Schröter, Information brokers in requirement-dependent social networks, in: Proc. Int'l Conf. Requirements Eng., September 2008.

[33] K. Ehrlich, K. Chang, Leveraging expertise in global software teams: going outside boundaries, in: Proc. Int'l Conf. Global Software Eng., 2006, pp. 149–158.

[34] G. Annu, S. Srividya, L. Balamurali, Intergroup coordination: a strategic approach, <http://www.qaiindia.com/Conference/pml_pre_publishing.htm>.

[35] S. Faraj, L. Sproull, Coordinating expertise in software development teams, Management Science 46 (12) (2000) 1554–1568.

[36] K.M. Nelson, J.G. Coorider, The Contribution of shared knowledge to IS group performance, MIS Quarterly 20 (4) (1996) 409–432.

[37] M.C. Paulk, B. Curtis, C. MeryBath, C. Weber, Capability maturity model for software ver. 1.1, Technical Report, CMU/SEI-93-TR-24, Software Engineering Institute, 1993.

[38] M. Yuan, D. Vogel, Cultural impact on intergroup coordination in software development in china: a qualitative analysis, in: Proceedings of the 39th Hawaii International Conference on System Sciences, IEEE Computer Society, 2006.

[39] J.D. Herbsleb, R.E. Grinter, Architectures, coordination, and distance: conway's law and beyond, IEEE Software 16 (5) (1999) 63–70.

[40] C.R.B. de Souza, D. Redmiles, The awareness network: to whom should i display my actions? And, whose actions should i monitor? in: Proc. European Conf. Computer Supported Cooperative Work, September 2007.

[41] B. Al-Ani, E. Trainer, R. Ripley, A. Sarma, A. van der Hoek, D. Redmiles, Continuous coordination within the context of cooperative and human aspects of software engineering, in: Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE '08), ACM, New York, NY, USA, 2008, pp. 1–4.

[42] I. Vessey, A.P. Sravanapudi, CASE tools as collaborative support technologies, CACM 38 (1) (1995) 83–95.

[43] L. Wu, H. Sahraoui, Accommodating software development collaboration, in: Proceedings of the 12th Asia-Pacific Software Engineering Conference (Apsec'05) – vol. 00, December 15–17, APSEC, IEEE Computer Society, Washington, DC, 2005, pp. 33–42.

[44] J. Herbsleb, A. Mockus, An empirical study of speed and communication in globally-distributed software development, IEEE Transactions on Software Engineering 29 (2003) 1–14.

[45] Y. Ye, Dimensions and forms of knowledge collaboration in software development, in: 12th Asia-Pacific Software Engineering Conference (ASPEC 05), 2005.

[46] L. Dube, G. Pare, The multifaceted nature of virtual teams, in: D.J. Pauleen (Ed.), Virtual Teams: Projects, Protocols and Processes, Idea Group, Hershey, PA, 2004, pp. 1–39.

[47] K. Beck, M. Beedle, A.V. Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, S. Mellor, K. Schwaber, J. Sutherland, D. Thomas, Manifesto for Agile Software Development, 2001. <http://agilemanifesto.org/>.

[48] A. Cockburn, J. Highsmith, Agile software development: the people factor, IEEE Computer 34 (11) (2001) 131–133.

[49] K. Beck, Extreme Programming Explained: Embrace Change (2nd edition), Addison- Wesley, San Francisco, 2005.

[50] S. Ambler, Quality in an agile world, Software Quality Professional 7 (4) (2005) 34–40.

[51] S. Bryant, P. Romero, B. du Boulay, The collaborative nature fo pair programming, in: Proceedings of XP2006, LNCS 4044, 2006, pp. pp 53–64.

[52] P. Dourish, Extending awareness beyond synchronous collaboration, in: T. Brinck, S.E. McDaniel (Eds.), CHI'97 Workshop on Awareness in Collaborative Systems, Atlanta, Georgia, 22–27 March 1997.

[53] K. Schmidt, "The problem with 'awareness': introductory remarks on 'awareness in CSCW'", Computer Supported Cooperative Work (CSCW), The Journal of Collaborative Computing 11 (3–4) (2002) 285–298.

[54] C. Heath, M.S. Svensson, J. Hindmarsh, P. Luff, D. Vom Lehn, Configuring awareness, Computer Supported Cooperative Work: The Journal of Collaborative Computing 11 (3–4) (2002).
[55] T. Robertson, The public availability of actions and artefacts, Computer Supported Cooperative Work: The Journal of Collaborative Computing 11 (3–4) (2002).
[56] C. Gutwin, R. Penner, K. Schneider, Group awareness in distributed software development, in: Proceedings of the 2004 ACM Conference on Computer Supported Cooperative work (CSCW '04), ACM, New York, NY, USA, 2004, pp. 72–81.
[57] I. Kwan, A. Schröter, D. Damian, Does socio-technical congruence have an effect on software buil success? A study of coordination in a software project, IEEE Transactions on Software Engineering 37 (3) (2011) 307–324.
[58] M.K. Goncalves, C.R.B. de Souza, V.M. Gonzalez, Initial findings from an observational study of software engineers, in: Proceedings of the 2009 13th International Conference on Computer Supported Cooperative Work in Design, IEEE Computer Society, 2009, pp. 498–503.
[59] R. Spencer, E. Coiera, P. Logan, Variation in communication loads on clinical staff in the emergency department, Annals of Emergency Medicine 44 (2004) 268–273.
[60] M. Finsterwalder, Does XP need a professional customer, in: XP2001 Workshop on Customer Involvement, XP2001, Cagliari, Italy, 2001.
[61] R. Wears, S. Perry, S. Wilson, J. Galliers, J. Fone, Status boards: user-evolved artefacts for inter- and intra-group coordination, Cognition Technology & Work 9 (3) (2007).
[62] E. Boger, Electronic tracking board reduces ED patient length of stay at Indiana Hospital, Journal of Emergency Nursing 29 (2003) 39–43.
[63] D.J. France, S. Levin, R. Hemphill, K. Chen, D. Rickard, R. Makowski, I. Jones, D. Aronsky, Emergency physicians' behaviors and workload in the presence of an electronic whiteboard, International Journal of Medical Informatics 74 (10) (2005) 827–837.
[64] J. Jensen, United hospital increases capacity usage, efficiency with patient-flow management system, Journal of Healthcare Information Management 18 (2004) 26–31.
[65] H.A. Marinakis, F.L. Zwemer Jr., An inexpensive modification of the laboratory computer display changes emergency physicians' work habits and perceptions, Annals of Emergency Medicine 41 (2003) 186–190.
[66] Y. Xiao, S. Schenkel, S. Faraj, C.F. Mackenzie, J. Moss, What whiteboards in a trauma center operating suite can teach us about emergency department communication, Annals of Emergency Medicine 50 (4) (2007) 387–395.
[67] A.K. Jain, P.D. Ting, Software quality via rapid prototyping, in: Global Telecommunications Conference, 1989, and Exhibition. 'Communications Technology for the 1990s and Beyond'. GLOBECOM '89, IEEE, vol. 1, 1989, pp. 642–646.
[68] B.W. Boehm, Improving software productivity, Computer 20 (9) (1987) 43–57.
[69] P. Runeson, M. Host, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164.
[70] T.C. Lethbridge, S.E. Sim, J. Singer, Studying software engineers: data collection techniques for software field studies, Empirical Software Engineering 10 (3) (2005) 311–341.
[71] C. Gutwin, S. Greenberg, A Descriptive Framework of Workspace Awareness for Real Time Groupware, CSCW'01, Kluwer Dordrecht, 2001.
[72] M. Brill, S. Weidemann, BOSTI Associates, Disproving Widespread Myths About Workplace Design, Kimball International, Jasper, IN, 2001.
[73] F. Becker, W. Sims, Offices That Work: Balancing Communication, Flexibility, and Cost, in: International Workplace Studies Program, Cornell University, Ithaca, 2001. <http://iwsp.human.cornell.edu>.
[74] L.A. Perlow, The time famine: toward sociology of work time, Administrative Science Quarterly 44 (1999) 57–81.
[75] R. Kaplan, Urban forestry and the workplace, in: P.H. Gobster (Ed.), Managing Urban and High-use Recreation Settings, General Technical Report NC-163, North Central Forest Experiment Station, USDA Forest Service, Chicago, 1992.
[76] D.P. Wyon, Indoor environmental effects on productivity, in: Proceedings of IAQ'96, 'Paths to Better Building Environments', 6–8 October, Baltimore, MD, USA, 1996.
[77] M. Csikszentmihalyi, Flow: The Psychology of Optimal Experience, Harper & Row, New York, 1990.
[78] S. Lahlou, Observing cognitive work in offices, in: N.A. Streitz, J. Siegel, V. Hartkopf, S. Konomi, (Eds.), Proceedings of the Second International Workshop on Cooperative Buildings, Integrating information, Organization, and Architecture, Lecture Notes in Computer Science, vol. 1670, Springer-Verlag, 1999, pp. 150–163.
[79] J. McGee, Knowledge work as craft work, McGee's musings, 2002. <www.mcgeesmusings.net/stories/2002/03/21/KnowledgeWorkAsCraft.html>.
[80] H.M. Robinson, J. Segal, H. Sharp, Ethnographically-informed empirical studies of software practice, Information and Software Technology 49 (6) (2007) 540–551.
[81] A.C. Ovedovitz, Business Spastitics in Brief, South-Western College Publishing, 2001.
[82] K.S. Trivedi, Probability and Statistics with Reliability, Queueing, and Computer Science Applications, second ed., J. Wiley and Sons, 2002.
[83] S.W. Ambler, Validating agile models, Cutter IT Journal 15 (2002) 33–39.