L'Erario *et al. Journal of the Brazilian Computer Society* (2020) 26:7
https://doi.org/10.1186/s13173-020-00101-7

Journal of the
Brazilian Computer Society

RESEARCH Open Access

CFDSD: a Communication Framework for Distributed Software Development

Alexandre L'Erario*† , José Antônio Gonçalves†, José Augusto Fabri†, Tiago Pagotto† and Rodrigo Henrique Cunha Palácios†

*Correspondence:
alerario@utfpr.edu.br
†Alexandre L'Erario, José Antônio Gonçalves, José Augusto Fabri, Tiago Pagotto and Rodrigo Henrique Cunha Palácios contributed equally to this work. Computing Department, UTFPR-Federal University of Technology - Paraná, Avenida Alberto Carazzai, 1640, 86300-000 Cornélio Procópio, Brazil

Abstract

Due to geographical and/or temporal dispersion, communication between teams in distributed software projects is a critical factor for success. Notably, distributed teams suffer adverse physical and temporal dispersion effects during an information exchange. To mitigate problems arising from interactions, it is important to understand the communication structure of teams during distributed projects. The objective of this work is to present the Communication Framework for Distributed Software Development. This framework groups a set of distributed projects communication concepts and enables a unified view of all stakeholders' intercommunications that use many interaction technologies. The goal of this framework is to portray the dynamics of the interaction between distributed teams in a multi-tier structure, and each tier approaches a single function in the intercommunication process. We analyzed all interactions from distributed teams after developing an experimental IT Project, using the content analysis method to validate the Communication Framework for Distributed Software Development. The main contribution of this work is the framework specification, and the investigation, which has the potential to help mapping communication patterns in DSD. Moreover, this framework comprises interfaces for communication assessment and includes intercommunications from automated engineering tools as bots.

Keywords: Distributed software development, Communication, Project management, Collaboration

Introduction

The distributed software development (DSD) is common in companies. The growth of this scenario is a reality. A Gartner report explained by [1] shows that more than 90% of the companies in the Fortune Global 500, a list that ranks the top 500 corporations worldwide according to their revenue, used external resources to deliver Information Technology (IT) services. Lee [1] reports that an engineer of a large IT company said the following: "It is nearly impossible these days to find a software team that is completely collocated." Several companies adopted DSD to improve their customer relationships, reach new markets, enhance the quality of products and processes, attract specialized labor, reduce time to market, and reduce costs.

According to Prikladnicki [2], DSD is characterized as a scenario in which people working on a software project are geographically and/or temporally distant. The temporal distance, according to [3], refers to the difference in hours between one site and another. For example, a site located in Sao Paulo (Brazil) is 2 h away from New York (USA) and 12 hours away from Tokyo (Japan). A closer temporal distance is better for the synchronous interaction and a farther temporal distance is better for follow-the-sun projects.

These dispersions are a result of the configuration made by the project management company, which generates independent production units, called sites [2]. Prikladnicki [4] considers natural the presence of physical dispersions between countries or continents. This condition implies that people involved in distributed projects are located at such vast distances and that their schedules of assigned activities are affected by different time zones [5]. This work scenario is classified as Global Software Development (GSD) [4]. Although it is presented as an environment in which activity management tends to be more complex, there are several reasons to use DSD as demand for skilled, lower-cost labor, and proximity with local markets [6, 7].

These factors generate an environment with new conditions for competitiveness and, consequently, require high-technology companies to introduce new and more collaborative production models. According to [8], companies may also employ these production models when they do not dominate the techniques or technologies required for their projects, inducing them to seek partnerships to fill the gap. According to Ehrlich [9], communication management between people and processes is one of the predominant factors when working with DSD. Problems associated with communication in DSD are explored in many aspects. For example, after a systematic literature review, [10] identified the inappropriate use of synchronous and asynchronous communication tools as one of the significant problem associated with these environments. Communication is addressed as a critical factor for the success of a software project according to [11] who created categories related to global software development, effective teamwork, and project effectiveness.

Although GSD can be profitable, according to [12], to maintain an effective and unambiguous communication between teams is critical for project quality. This author investigates a social switching strategy to improve communication in this regard. Due to the difficulties found in a DSD environment [5], the inefficiency of the communication between work teams can generate anomalies during the production and integration of the final product [13, 14]. During a software project execution, a site may receive an excessive volume of interactions and interpretation errors and loss of information may occur, as well as other problems that can jeopardize the project execution. Several studies stress the importance of communication in a DSD environment. Some authors [15–23] approach this subject and emphasize the relevance of communication between development teams, developers, and software users.

The objective of this work is to present the Communication Framework DSD. A framework can be defined as a set of elements and their relationships used to characterize a series of actions for a specific purpose [24] in distributed software development applied to DSD projects. This framework illustrates the structure and flow of communication in a distributed software project. CFDSD (Communication Framework for Distributed Software Development) assembles a set of concepts related to communication in DSD projects.

The CFDSD framework encapsulates many concepts related to communication in software projects, for example, the research showed by [25] and [26] were compiled in this framework structure. In this sense, CFDSD does not depend on software implementation elements and provides a high-level overview on how messages are exchanged. This work contributes to research by presenting a framework that represents the communication structure in projects with the purpose to identify message exchange patterns between teams. The main contribution of this work is the consolidation of many concepts of software communication projects in a structure. The issues of DSD communication require rich communication, maintain software design knowledge, keep teamwork, assure project effectiveness, and prevent unambiguous communication. The CFDSD creates a structure that enables to store and analyze content, including a mechanism for evaluating the communication. The main contributions of this work are:

- A UML specification of communication in distributed projects;
- An abstract evaluation model of intercommunication;
- The creation of a "communication unit (CU)" that is an interaction group with the same purpose;
- An interface specification that links communication with software process.

The present work was organized as follows: the next section approaches the fundamentals, related works, and key concepts. The third section presents a detailed description of CFDSD, the subject of this paper. The fourth section exposes the methodology and research procedures employed. Finally, the last section presents the results, an interpretative analysis of the collected data and the conclusions, respectively.

Communication in DSD and related works

Fuks [15] bases his work on the definition of communication as the act or effect of sending, transmitting, and receiving messages using stipulated methods and/or processes, both through spoken or written language or other signs, signals, or symbols or using specialized technological, sound or visual devices.

The media synchronicity theory presented by [25] aggregates a set of essential concepts related to project communication. In the first concept, the author affirms that the success of completing many tasks that involve more than one person is associated with communication performance. Under these terms, the performance of communication is associated with convergence and conveyance [25]. The development of shared knowledge is a success factor for the project in these situations. In the same work, [25] presents a set of attributes associated with the media synchronicity theory, such as synchronism, communication process, speed, capability, parallelism, symbol set, rehearsability, reprocessability, and appropriation.

The media can influence communication performance and include underlying communication processes [27, 28], and both, transmission of data and individual cognitive processes, to develop information. When addressing the technology-related challenges of DSD, [10] and [17] state that the most evident configuration is the synchrony in communication. Synchronous communication occurs when the participants have access to transmitted messages in real time. Examples of this are person-to-person conversations or meetings or phone calls or even software with real-time exchanging of texts, audios, and videos. An asynchronous communication, according to [10], occurs when participants

use the same method of communication, but not simultaneously. Examples of this are physical correspondence, e-mail, and posts on social networking websites.

Besides, according to [29], communication media has different properties and capabilities depending on the challenges imposed by the distance between the sites in DSD environments. . Research have addressed the media theme, such as [25, 26] and corroborating them also with works dealing with specific types of media, e.g., social media [30].

Literature provides works related to tools employed to optimize communication, as in the case of [16, 18, 20]. Some studies report research on ontology and conflict resolution in communication process [17, 19, 21]. Studies [18] and [31] expose a theoretical approach to outline problems and solutions of communication in DSD.

The works of [15, 16, 22] propose models to optimize communication, while the author of [20] shows a strategic approach to communication. Work [32] shows an organizational communication approach, including connection, concurrency, comprehension, communication, conceptualization, collaboration, and collective intelligence.

This work approach is complementary to the others mentioned above and we present how communication can be disassembled in a logical structure. Although the related works presented in this section address theoretical concepts of communication in distributed projects, none of them presents a structure capable of relating them. This approach, it becomes possible in future work, applying optimization algorithms.

Key concepts

The above-related works approach essentials elements for communication in a DSD environment. This set includes works with a focus on social studies, communication process, synchronicity, maturity, and others. Each work approaches a key communication concept in a distributed software development environment, and CFDSD aggregates many of these concepts. This section explains these concepts.

Table 1 shows a set of concepts about software projects communication. The first column in this table is the ID, the second the name, and the third includes a list of authors that approach the concept. We approach the elements in Table 1 as below.

The concept site (1) in Table 1, for this research, is an autonomous software production organization. Sites are organizations that develop software and have at least one person (collaborator). The site defined as home-office has only one person and this person is a collaborator of an organization with a process and pre-defined tools. However, [33, 34] address another kind of relation between the person and an organization called crowdsourcing. In this case, the development is task-centered, and the person executes many tasks for many organizations. Work [35] presents an essential approach to sites.

According to [36], a software process is a set of activities and results that the purpose to create software. This element is essential for creating a software product or service and many authors such as [37–39] highlight the process improvement.

In software development, according to [40], many stakeholders use many tools (Table 1 item 3). These tools are, for instance, IDE (integrated development environment), project management tools, text editors, spreadsheets, communications tools, bots, UML modelers, and others. Each tool, combined with the work process, can generate or update one/or more artifacts (work product).

Table 1 List of key concepts

ID	Concept	Authors
1	Site	[2, 4, 33–35]
2	Software process	[36–39]
3	Tools	[40]
4	Physical distance	[3, 41]
5	Temporal distance	[1, 26]
6	Media	[25, 26]
7	Computer-mediated communication	[30, 42, 43]
8	Interaction	[25, 26, 44]
9	Conveyance	[25, 26]
10	Convergence	[25, 26]
11	Synchronism	[25, 45]
12	Feedback	[25, 46–48]
13	Communication frequency	[26]
14	Communication turn-taking	[26, 49, 50]
15	Parallelism	[25]
16	Rehearsability	[25]
17	Reprocessability	[25]
18	Appropriation	[51]
19	Symbol set	[25]

Physical distance is the geographic range between sites. This feature defines DSD/GSD. Many authors approach this characteristic as [3, 41]. The time zone, in distributed software development, can be associated with temporal distance (item 5 of Table 1) and physical distance. There is a temporal distance when the sites are largely physically distant, or the collaborators have different work schedules. Many authors approach this subject as [1, 26].

Media, item 6 in Table 1, is the transport environment between source and destination. The media synchronicity theory aggregates many properties in this element. Researches [25] and [26] approach this subject.

Computer-mediated communication tools (CMC) allow interaction in many different ways and data exchange between individuals. Some authors who approach this theme, item 7 of Table 1, are [42, 43]. Social media also compose this element, as reported by [30]. The interaction uses a medium and has content. Several authors approach this subject, such as [25, 26, 44].

Conveyance (item 9 in Table 1) is the transport of interaction. The authors of [25, 26] use this definition as the basis for their work. In item 10, Table 1, convergence is the extraction of meaning from the information transmitted by individuals, according to authors [25, 26].

Synchronous or asynchronous communication is recognized as an essential factor affecting interpersonal communication and teamwork [45]. Synchronicity (item 11, in Table 1) is a coordinate shared pattern behavior among individuals as they work together, according to [25].

Feedback, item 12 in Table 1, is associated with the media capacity. A medium with a high degree of synchronicity is able to receive immediate feedbacks [25]. In software projects, feedback is very important to support the identification of problems and the improvement of product/processes, such as in the work of [46, 47]. Feedback is also

crucial in maintaining context awareness. For example, the work of [48] systematically addresses how to reduce delay feedbacks by using continuous analysis.

Based on the definition of [26], communication frequency (item 13 in Table 1) is the intensity of the interactions.

Table 1, item 14, indicates turn-taking. Turn-taking is about switching of interlocutors during a conversation or explanation. This concept is common in a dyadic conversation (communication between two interlocutors) for establishing a commonly shared knowledge among stakeholders. The works of [26, 49] approach this concept. According to [49], the turn-taking is composed by several components and [50] points that turn-taking involves processes for construction, contributions, response to previous comments, and transitioning to a different speaker, using a variety of linguistic and non-linguistic signs.

Parallelism (item 15 in Table 1) is defined by [25] as the set of simultaneous interactions with many senders sending many messages at the same time to many receivers.

The author of [25] defines rehearsability (item 16 in Table 1) as the media capability to improve the content before sending the message. Reprocessability (item 17 in Table 1) is the capability that the medium has to enable a message to be reexamined or processed again. In a set of interactions, different interpretations can exist between the stakeholders, that is, the inherent divergence and ambiguity of communication can lead to different and sometimes conflicting knowledge. Several authors approach the subject, such as [51]. A symbol set (item 19 in Table 1), derived from [25], is the set of symbol types used in interactions.

The CFDSD

These issues are present largely in distributed project stakeholder's communication. The CFDSD creates a structure that enables to store and analyze content, including a mechanism for evaluating the communication. The CFDSD has the following features:

1. A general nature that makes communication process understandable, regardless of its maturity, software-engineering tools used in the project, and context;
2. A structure with interactions generated by people and tools (e.g., automatic bots);
3. An interface that allows the creation of techniques and models to assess communication;
4. It allows the study of communication even in a current project;
5. It creates interactions groups into the same context;
6. It encourages the introduction of software tools to automate communication assessments and structuring in distributed projects.

The CFDSD corroborates with the above-mentioned works in the "Key concepts" subsection. The concepts addressed in the second section corroborates with the conceptual framework that frames communication elements into single units. Table 2 shows how CFDSD combines key concepts. Key concepts are presented in the previous section. This table shows the relationship among them. Figures 1 and 2 show the CFDSD and its details in the next section.

Framework details

To accomplish the previously mentioned objectives, the CFDSD was divided into six tiers. Figure 1 shows a package diagram with these six tiers, named T0_Project, T1_Purpose,

Table 2 Key concepts in CFDSD

ID	Concept	How CFDSD approach
1	Site	Teams, home-office and crowdsourcing, branches, offshore
2	Software process	Abstraction. Create a link between message and project activity.
3	Tools	Software engineering tools and communications tools (including bots).
4	Physical distance	Site Geocoordinate.
5	Temporal distance	Work-schedule and time-zone.
6	Media	Only computer-mediated communication (content and logs)
7	Computer-mediated communication	Synchronous and asynchronous interaction of any CMC tool.
8	Interaction	Unidirectional stream of communication, initiated by a sender and delivered to one or more receivers.
9	Conveyance	Only record-generating interactions and does not address face-to-face communication.
10	Convergence	Evaluation based on artifacts.
11	Synchronism	Site communication synchronism. It does not address the synchrony between individuals (face-a-face).
12	Feedback	Specific unilateral interaction.
13	Communication frequency	Based on the send time.
14	Communication turn-taking	Switching among talkers (dyadic or not) and identify the communication purpose.
15	Parallelism	Record date/time and create a unique id per interaction.
16	Rehearsability	Not applied by the CFDSD.
17	Reprocessability	Store all interaction, but does not look if receiver reprocesses the message.
18	Appropriation	Not applied by the CFDSD.
19	Symbol set	CFDSD encapsulates any set of symbols, but it cannot parse them in this work.

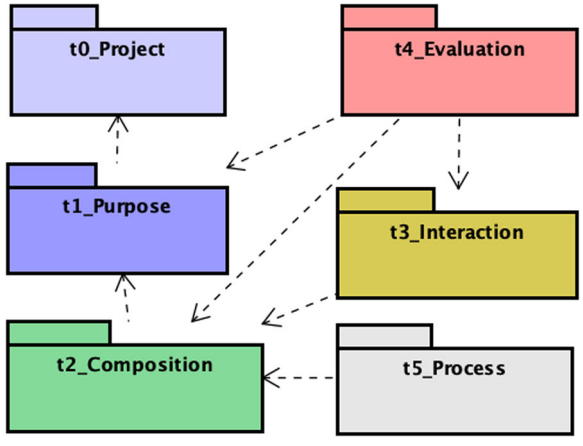

Fig. 1 Tiers of CFDSD

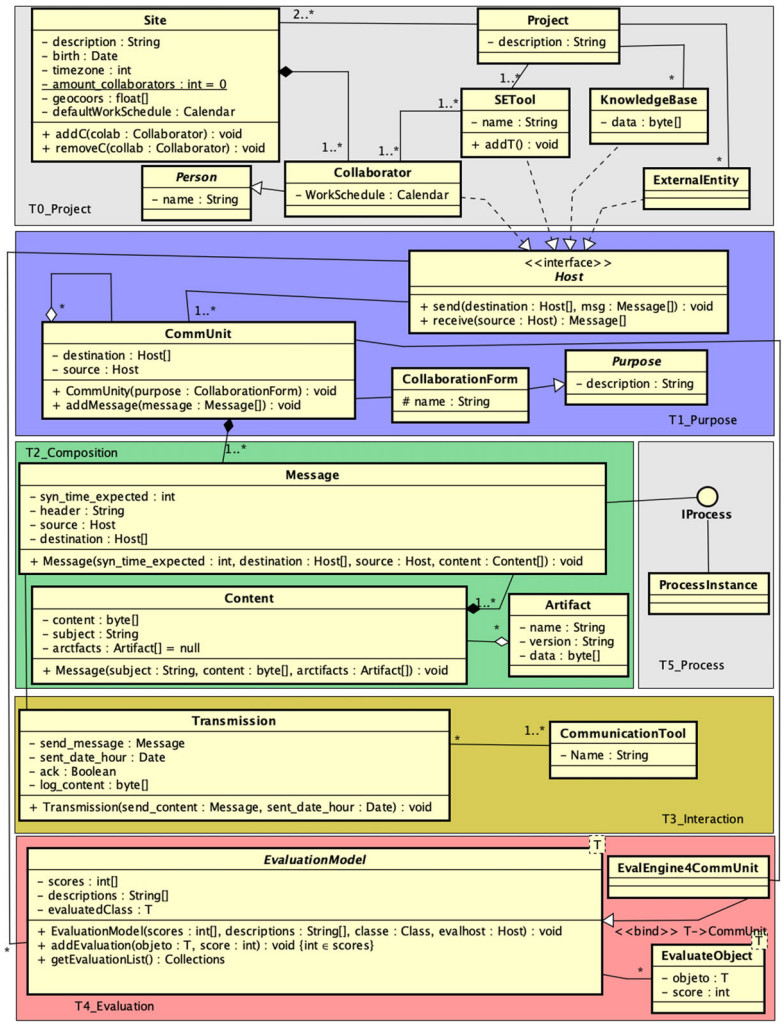

Fig. 2 All classes of CFDSD

T2_Composition, T3_Interaction, T4_Evaluation, and T5_Process. Each tier has a set of common purposes. The next sections of this paper describe each tier in detail.

Figure 2 presents the CFDSD in a class diagram. In this work, the communication in any distributed project consists of two elements: interaction and host. Interaction is a one-way flow of information with origin and destination. The hosts are the agents that send and/or receive this flow, such as developers, managers, customers, software engineering tools, chatbots, continuous integration tools, and other stakeholders.

The t0_Project tier

This first tier, T0_Project (Fig. 2), represents an abstraction of the project under development. This level contains all the elements that generate and receive interactions within a software project. A software project (represented by the Project class) is developed by multiple (two or more) sites composed of one or more collaborators (typified as human beings—Person class). In addition to the sites, a project can contain software engineering tools (SETool class) used to directly or indirectly support the development of the final

product, a knowledge base (KnowledgeBase class)—repository of templates and artifacts generated by the project or through previously collaborator experiences—and external entities (ExternalEntity class)—representation of entities not directly associated with the project.

The T1_Purpose tier

The T1_Purpose tier (Fig. 2) abstracts all the elements that generate and receive messages. It also associates a set of forms of intervention that may occur during project execution. The forms of intervention, mapped in the CollaborationForm class (action, schedule, media, network, adaptation, command/control, among others), were based on the activities defined by Fuks [15] and they are represented by the generic class Purpose. This tier is the start of communication and fragments it into several communication units (CommUnit Class). A CommUnit is a set of interactions between origin(s)/destinations(s) that start and end with same purpose.

The host class initializes the communication by creating and sending/receiving a message. This class is an abstraction of four classes of the Project tier: collaborator, SETool, KnowledgeBase, and ExternalEntity. The CFDSD approaches the host as any stakeholder that send and/or receive messages and includes interactions generated and received between collaborators, tools (software engineering and/or bots), external entities, and the knowledge base.

When elaborating a message, the host creates the CommUnit, which aggregates the purpose and has a destination. Moreover, a CommUnit is composed of a set of interactions ([0, n]) from other CommUnits, characterizing a tree structure.

The T2_Compositions tier

Once the concepts of the T1_Purpose tier have been established, the next tier (T2_Composition Fig. 2) physically addresses message construction physically. In this tier, the Message class defines the transmission method and the synchronism of the message, which has a content (Content class) and can have software project artifacts (Artifact class) attached.

Any message in a software project can be associated in a specific process phase, activity, or discipline. The CFDSD creates a relationship between a message and a specific point in the process.

The T3_Interaction tier

The T3_Interaction tier (Fig. 2) transmits the message generated in the T2_Composition tier (immediately above tier). The classes that make up this tier are Transmission and CommunicationTool. For this purpose, the Transmission class stores submission data and the communication tool (CommunicationTool class) adopted. This tier approaches the conveyance.

The T4_Evaluation tier

Communication evaluation occurs in the last tier, T4_Evaluation (Fig. 2). This tier has an abstract evaluation model (EvaluationModel template class) in which the parameter of this template defines which object will be evaluated. When the subclass (EvalEngine4CU class) is instantiated, the attributes and the collaborator (person responsible for the evaluation) are informed through the constructor method.

The EvalEngine4CU class evaluates CommUnit type objects, as bind indicated in Fig. 2. The evaluation method must also be implemented. It contains a list of objects to be evaluated and their possible evaluations. Finally, the template class EvaluateObject stores the object that was evaluated and your evaluation.

Any object instantiated from the CFDSD classes can be evaluated as long as they are created from the EvaluationModel and inform the bind value; the EvalEngine4CU class, for example, is responsible for evaluating CommUnits.

The T5_Process tier

The T5_Process is the tier to abstract the process. A software process is composed of several components such as phases, activities, and guidance, and the configuration of these components depends on a set of organizational factors. It is common for the same organization to run different process instances for different projects. The ProcessInstance class is for linking the organization process static view, and the interface IProcess associates the message to the process. This structure associates software process structure to messages. The CFDSD can gather interactions to conduct an assessment, for example.

Research methods and procedures

The research method adopted was a qualitative study using content analysis. This method was applied in the works of [52–55] for example. Studies carried out by [56, 57] indicate that this method can be both quantitative and qualitative and inductive or deductive. The inductive content analysis method gets an abstraction from data, and a deductive content analysis is often used for retesting existing data in a new context. This work uses the deductive approach to test CFDSD cohesion in a real environment. A group of productive sector professionals developed a robotic project pre-established in a laboratory environment. All interactions among the distributed groups were stored during project development (this data is the content) and analyzed later by the researchers.

This method was chosen to meet the environmental conditions of DSD and to track sites behavior and all communication generated during the distributed project implementation. The content analyzed here was the interactions (message content and logs). The content analysis intention was to verify if CFDSD is applicable in a real situation, proving its adherence to real interactions generated in a DSD environment. However, interactions are not purely composed of artifacts but also of particular interactions, so it was necessary to analyze casual information among participants of the project.

A physical building with several rooms inside at UTFPR-CP (Federal University of Technology – Cornélio Procópio/PR – Brazil - 23° 11' 13.3" S 50° 39' 23.3" W) was provided for the teams to execute the distributed experimental project. The rooms were far apart enough to ensure that people in different work teams were completely isolated and did not have physical contact with each other.

The participants were professionals with at least 3 years of experience in software production companies. Four sites with four to five participants in each were configured to perform the tasks related to the development of the experimental project. In addition to these four sites, two extra sites were created. The first additional site was named logistics and the second was named orchestrator. The orchestrator in the project is responsible for organizing all the stages before, during, and after the project execution. The logistics site

was used to move physical components between sites to ensure people in different sites did not have face-to-face contact during the project runtime.

Project planning

A problem scenario was introduced in the first face-to-face meeting with all stakeholders. In a second stage, the work teams were physically separated to develop the product. The main objective of the project was deliver a robot with embedded software to solve a specific problem. The teams had to assemble the robot and develop the software. Besides, they followed a macro process and used a set of tools (Eclipse, Astah and LeJOS – Lego Java Operating System) defined earlier by the orchestrator. The macro process (based on a waterfall) was introduced in the first face-to-face meeting with the team members.

Experimental project execution

Figure 3 presents the experimental project execution processes divided into six steps. The first one defines the composition and the physical location of each site. The next stage corresponds to the face-to-face meeting with the stakeholders. This second stage also includes the project presentation, initial resources, and the restrictions of each site. In step three, the sites are isolated in separate rooms. The step four represents the experimental project execution. Step five consists of the final product delivery and, finally, the last step is the completion of the experimental project.

Selecting the unit of analysis

According to [58], the unit of analysis is a the sample of small pieces that the researcher needs to determine; however, [57] points out that there are no fixed criteria to create a the unit of analysis. The interactions between stakeholders were analyzed to validate this work, and we observed the following features: (1) communication purposes. Although

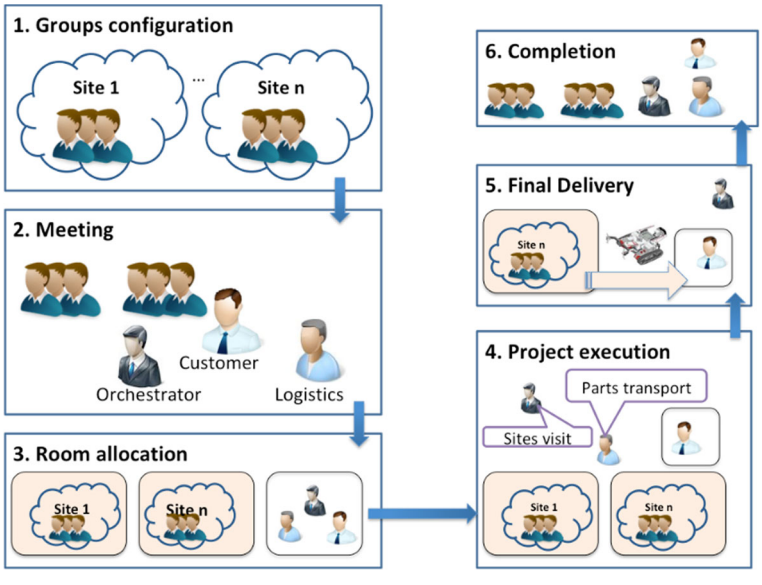

Fig. 3 Project execution

Fuks studies [15] were used in this work, it is understandable that the diversity and the number of purposes are related to the singularities of the project. In this study, the aim was to identify the amount and the types of purposes generated in the experiment and aggregate them into communications unities. (2) Switching sites with the same purpose (communication turn-taking): it measures the amount of the sites with the same purpose instance. (3) Evaluation of communication: uses the generated artifacts to define how communication can be evaluated. The communication flow was monitored during the project execution. The researchers visited the sites randomly to verify whether the project and communication tools provided met the sites interaction needs.

Data collection and instrumentation

To monitor all the interactions, the work teams used previously registered accounts in a groupware environment. This environment provided collaborators with an e-mail, forum, wiki, chat, and videoconference. Communication tools recorded the data collected during the project runtime and Internet interactions data stored in a proxy system.

Results

Project runtime was 7 h and 45 min and it was conducted in two periods. The teams were responsible for the coordination and organization of the activities. They had to define the actions, roles, and assignments for each site. After the project execution, the data extracted from Moodle, Hangouts, Gmail, and Squid proxy were standardized. Table 3 presents the fields of the final worksheet generated after formatting the data and their respective descriptions:

The fields CommUnit and Purpose, described in Table 3, were completed during analysis after building the project. Therefore, these data were generated from the researchers' interpretation of the messages. Although the experiment was performed in Brazil and the language used in these interactions was Brazilian Portuguese, for illustration purposes, some interactions in this document were translated into English.

Data analysis and interpretation

The analysis of the final worksheet resulted in the identification of 152 communication units in more than 2573 interactions. This section presents the structure and sequence of some different interactions. Date and time were omitted to reduce the size of the

Table 3 Standardized fields of the final worksheet

Field	Description
Tool	Indicates the selected medium and whether interaction occurred using Hangouts, Gmail or a registry generated by accessing Moodle or the Internet
CommUnit	Communication unit.
Purpose	Indicates the purpose of each interaction.
Source	What is the site/collaborator that send the interaction?
Destination	What is the site, collaboration or group that receive the interaction?
Time (Br)	Date and time in the DD/MM/YYYY hh:mm:ss format used in Brazil. This field records the time that interaction occurred.
Header	Indicates the message header.
Message	Content of the message.

tables presented below. Tables 4 and 5 are in chronological order. The Table 4 contains a communication unit that was developed in the project between team 1 (g1) and the orchestrator (vl). In addition, the second and third column of Table 4 (Purpose and SubPurpose -S.Purp.) was describe into Table 6.

In Table 4, five interactions were identified in order to establish this CommUnit. Team 1 (g1) misinterpreted the model for the cash flow. This team requested the help of the orchestrator, who explained that the cash flow would be in dollars and the initial resource would be $200. In the sixth row of Table 4, g1 delivers a formal artifact and, it was possible to measure the communication through this artifact.

There was a time delay between interactions in the Table 4. It is important to note that a Communication Unit is not self-exclusive, which means that there could be another CommUnit, composed of other interactions, running at the same time. Interactions 2 and 3 in Table 4 sent the same message, but the communication tool duplicated them. Message 3 was noise, for this work. The same sequence identified in Table 4 can be represented graphically, as illustrated in Fig. 4.

There was also communication between the team and the knowledge base. Table 5 shows the interaction performed for the conclusion of the work breakdown structure (WBS). The g4 (team 4) downloaded and delivered the artifact without the intervention of collaborators from other sites. The communications presented in Table 5 were integralized and achieved their purposes. Figure 5 illustrates the communication flow of this CommUnit.

A single CommUnit can have multiple origins and destinations. In the experiment, there were some situations where the same site sent or received messages from more than one site. Figure 6 illustrates this process.

The purpose of this interaction is to identify potential sellers of a particular physical component. Team 3 (g3) needed a component but was unaware of which team had this component.

Identified purposes

After analyzing the final worksheet, it was possible to identify the purposes of the interactions that occurred during the project execution. These purposes are listed and described in Table 6.

The purpose was associated with the interaction after we collected the data. Three researchers involved in this work analyzed the spreadsheet described in Table 3 and, after each one included the purpose, the results were compared. Each researcher worked on this operation at least 40 hours. Moreover, it was found that purposes can be made up

Table 4 Interaction between group 1 and the orchestrator

Ord.	Tool	Purpose	S.Purp.	Source	Dest.	Message
1	Email	QO	QO	g1	vl	Is the cash flow sheet in Reais (R$ - BRL) ? Is the initial amount of budget only 100.00?
2	Email	QO	RQO	vl	g1	You can use 200.00.
3	Email	QO	RQO	vl	g1	You can use 200.00.
4	Email	QO	QO	g1	vl	Reais (R$ - BRL)?
5	Email	QO	RQO	vl	g1	No, you can use Dollar (U$ -USD)
6	Base	Deliver	QKB	Moodle	g1	Upload

S.Purp subpurpose, *Dest.* destination

Table 5 Communication between a group and the base

MainPurpose	Source	Destination	Message	Header
QKB	Moodle	g4	Query	Main document
DownTempl	Moodle	g4	Download	WBS
Deliver	Moodle	g4	Upload	WBS
DownTempl	Moodle	g4	Download	WBS

from other purposes. This way, at a certain moment, a purpose can be considered the main one, and at another time, it can become a component of another purpose. In this work, all undesired situations that could compromise the communication process are considered a noise, similarly to the computer network area, as explained in [59]. Some noise was detected during the project execution, most of them related to the technology that was being used for communication (for example, duplication of messages in the chat system), and the rest of them was human-generated. As illustrated in Fig. 7, it was possible to map the purposes (P1..Px) of this project in a hierarchical structure consisting of CommUnits.

Figure 7 contains a graph used to divide the project into CUs and split the units into subunits. This figure also presents an evaluation indicative, represented by the ellipse av1, and the association of artifacts with the CUs. The interactions (SP1...SPn) are in the lowest level of this tree. The representative structure shown in Fig. 7 only includes only two communications unities.

Table 6 Identified purposes

CU ID	Purpose	Description
1	Action	Incentive a site to take an action
2	Comm/Control	Command /control
3	Adjustment	Adjustment
4	Network	Network
5	Social	Social
6	Schedule	Schedule
7	QO	Query to the orchestrator
8	Wait	Signal another to wait a moment.
9	QKB	Query to the project knowledge basis
10	RQS	Response for a query from a site
11	RQO	Response for a query from orchestrator
12	DownTempl	Download a project template artifact
13	Deliver	Deliver task product to the project
14	Recalibrate	Request to change the communication technique or technology
15	QS	Query to a site
16	Reaffirm	Reaffirm agreement or commitment, make communication in more formal way
17	QExt	Query or download from external knowledge base
18	Noise	Messages generated by the participants that are not related to the project or duplicated messages
19	Inform	Notify all or a site for any purpose
20	RQEKB	Response to the QEKB
21	QEKB	Query to the external knowledge base

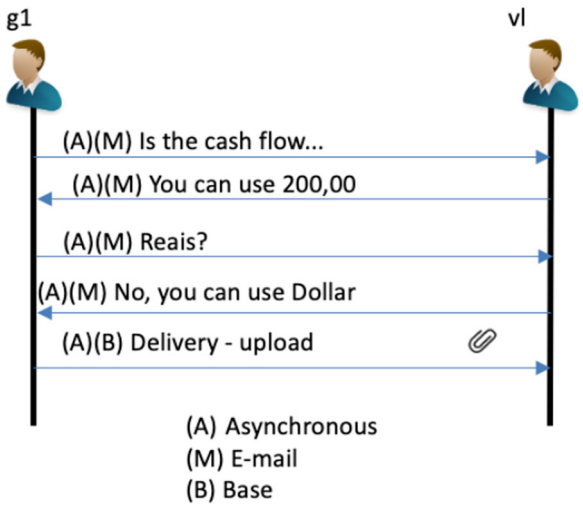

Fig. 4 Graphic representation of Table 4

Unit of analysis

Although we had mapped the communication units after the project execution, it was possible to group a set of interactions into a single purpose (purpose and sub-purpose). Thus, the variable was considered satisfied and 100

For the same purpose/CU, several sites can generate interactions according to interests. A single CommUnit can temporarily contain a variable set of sites and hosts. We have found CommUnits that the communication tool switched from an e-mail (asynchronous) to Google Hangouts (synchronous). This evidence establishes the possibility of using different communication tools and, consequently, different types of synchronism in the same CommUnit.

The analysis of Figs. 4, 5, and 7 allowed the quantification and qualification of the interaction, and this experiment revealed that it is also possible to evaluate a set of interactions, manually, thus qualifying a CommUnit in this scenario.

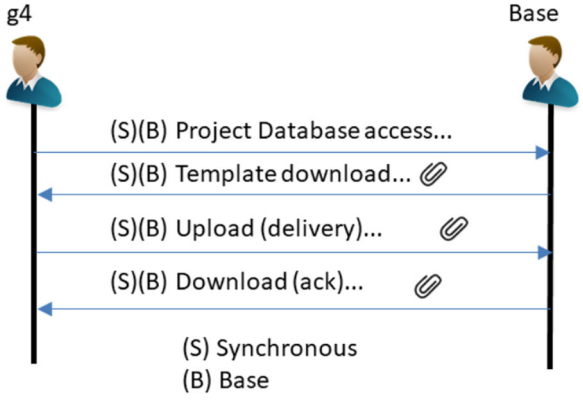

Fig. 5 Flow of interactions

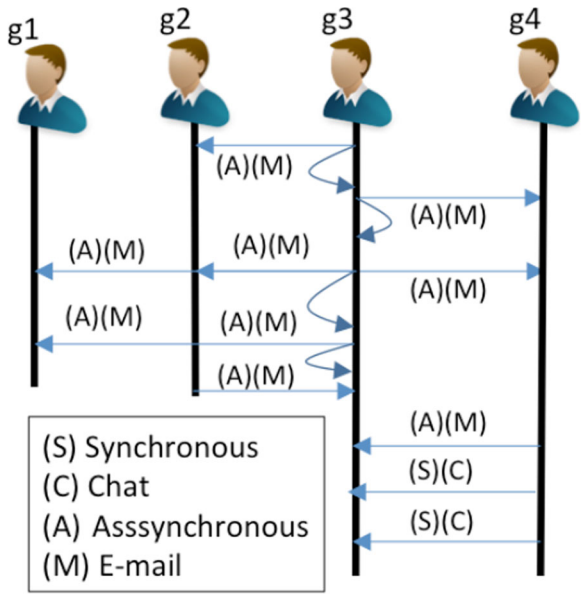

Fig. 6 Multiple sources and destinations in the same CommUnit

The conversation between sites was effective, and the communication tools provided for the project fulfilled all the needs. The project was delivered according to the specifications, and the predefined macro process was precisely followed.

The researchers visited the sites every half hour, with no intervention, to observe the behavior and the work of the teams. Through these visits, the researchers identified some deadlocks caused by different expectations from the sites. During the execution of the tasks, the sites did not communicate with each other. This way, a site could presume that the other site was not executing a given task because it was solving internal conflicts related to the project. This presumption created the false sensation of a setback

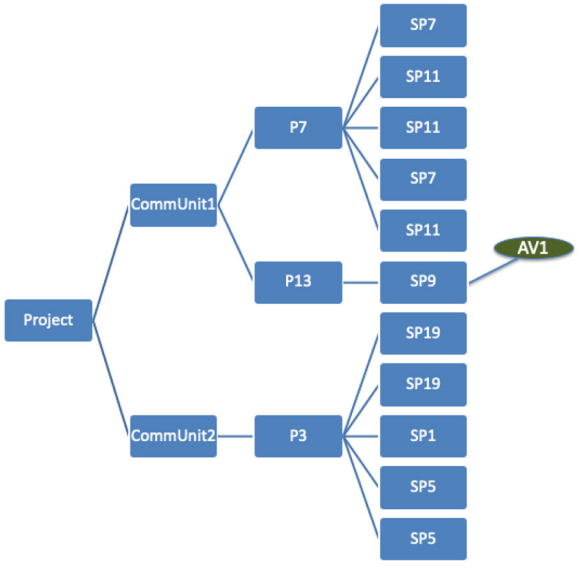

Fig. 7 Hierarchical structure of the two communication units

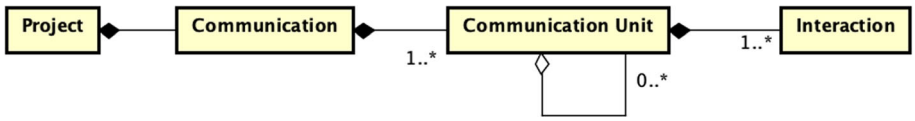

Fig. 8 Communication elements relationship

because a site can create a false sense of conflict when it is unaware of what other is doing. One site perception about the others is reported by the concept of awareness, according to [21, 60, 61].

Threats to validity

Data obtained from the project were collected, processed, and analyzed. The analysis process was very difficult since all contents were analyzed manually, a possible but unfeasible task to be done in real-time. To identify all communication units was difficult. However, the CFDSD was adherent to the project communication set.

Even though we conducted the experiment and stored all interactions, our research method has constraints for generalization since we analyzed the interactions, and, although we had analyzed some variables, we tested them in a controlled environment, and we made the analysis after the project delivery. It is important to highlight that the main research method was content analysis, and this method has constraints for generalization.

Furthermore, all participants were Brazilians with experience in software development, and some spoke a foreign language; therefore, the language adopted was Portuguese. The experiment did not address homonyms (perfect homonyms) problems in communication.

The experimental project was similar to a real project since all participants had similar professional skills. We conducted experimental tests with undergraduate students prior to this IT project, and we found out that participants had problems related to software process (students in the second semester), software development (students in the first semester), maturity, and capability to solve problems (students in the last semester). All students were attending undergraduate IT courses.

During the visits to the sites, the researchers observed that these sites did not always respond to requests from other sites, which caused some anxiety. On the other hand, in some occasions, the stakeholders sent an excessive number of messages. These problems are defined by [21, 61, 62] as awareness in a DSD environment.

Conclusions

This work shows how communication in a distributed project can be structured in CFDSD. For this purpose, we conducted an IT project during which we monitor all the interactions. In this research, the communication generated by the project is composed of communication units, and these units are composed of interactions. This model attests that communication can be structured in a tiered structure, as illustrated in Fig. 8.

Figure 8 shows the communication segment of a project. At the highest atomic level, interactions are identified; a set of these interactions with the same purpose characterizes a CommUnit. The communication of a project consists of a set of CommUnit.

In addition to the explicit knowledge mapped in the CFDSD, other implicit concepts were observed during the project execution. These implicit concepts are discussed briefly below.

The following implicit knowledge was observed in the first tier (T0_Project): internal and external perspectives caused by geographic dispersion, coordination mechanism [1] , and prior knowledge related to project and development.

The perspectives mentioned above refer to the personal perception of each site. In this case, the internal perspective is related to site performance at the project development. The external perspective is associated with the vision of a site in relation to another site during the project development. In this regard, a site may have incoherent perceptions of other sites. An example of this occurs in a project when a site takes a long time to respond to other websites because it is solving some issues related to the project's tasks. Consequently, the other sites believe the site is blocking the project development, which is not the case.

The T1_Purpose tier implicitly treats project demands and prior relations between project stakeholders, which lead to a competition and/or cooperation between them.

The following tier, T2_Composition, implicitly includes language, dialect, and expression capacity, as well as internal denominations used in projects, such as references to specific document names. The T3_Interaction tier implicitly covers different levels of formality, ranging from informal and without record (e.g., phone calls) to formal and auditable (e.g., the use of an e-mail as a contract). Moreover, in this tier, it is implicit that, for the same purpose, stakeholders can switch communication tools.

Finally, the last tier, T4_Evaluation, implicitly incorporates the way that the communication elements, such as an interaction or an artifact, are evaluated and measured.

Although the project executed by the professionals was fictional, we noticed some important factors that influence communication in distributed projects. There is a direct relationship with the stakeholder's coordination mechanism in the communication. How one site can influence (offshore is different by branch, for example) another is subjective and can jeopardize process standardization.

After the implementation of our research method and analysis of the data, we can summarize our conclusion as follows: (1) this is a new UML communication specification in distributed projects based on relevant works; (2) the CFDSD was validated by a similar real project conducted by professionals; (3) the CFDSD works, but not in real-time; (4) the CFDSD provides interfaces to connect to project management and evaluation methods; (5) the communication unit is the new element of this work and its contribution; (6) the abstract evaluation model is a new element, too; and (7) there are unmapped problems in communication yet. These problems include face-to-face communication hidden by CFDSD, message overload, coordination mechanism, and awareness. The future works identified in this paper converge on two aspects: the conceptual and the instrumental. From the conceptual aspect, a future work is to develop metrics based on CommUnit. These metrics could identify problems and may even proactively enhance project performance. The adopted instrumentation was efficient, but it was not agile or effective. Since CommUnit mapping was manual, this process was relatively slow and required too much time and labor. Real projects that demand time and more data will require computer tools

[1] The coordination mechanism was described by MINTZBERG, Henry. Structure in fives: Designing effective organizations. Prentice-Hall, Inc, 1993

to classify the interactions and map the purposes. A mechanism to drive interactions with the automatic identification of CommUnit and purposes without human intervention can be a valuable tool for future experiments or projects.

Acknowledgements
We thank UTFPR-CP (Federal University of Technology - Paraná).

Authors' contributions
Alexandre L'Erario: project manager and design of this work; have drafted the sections introduction, key concepts, and results, and he had made data analysis.
José Antônio Gonçalves: have drafted the methodology, coordinate the experiment and content analysis.
José Augusto Fabri: data analysis and paper review
Tiago Pagotto: Merged all data in one single spreadsheet, data standardization, and support into the experiment.
Rodrigo Henrique Cunha Palácios: interpretation of data and review this paper.

Authors' information
Alexandre L'Erario, José Auguto Fabri, José Antônio Gonçalves, and Rodrigo Henrique Cunha Palácios are professors from Federal Technological University of Paraná and founders of LabInov (Laboratory of Innovation).
Alexandre L'Erario is a professor at the Federal Technological University of Parana—Campus Cornelio Procopio—since 2005, and Ph.D. in Engineering from the Polytechnic School, University of São Paulo (2009). Since then, his efforts are researching issues related to coordination and communication in distributed software development projects. He has researched Software Evolution and Maintenance for 4 years and is the founder and collaborator of LabInov. The Innovation Laboratory (LabInov) at the Federal University of Technology started in 2014 with a mission to provide the opportunity for growth companies in innovative projects. He has been a member in ABNT's Software Lifecycle Management Study and participated representing UTFPR in meetings. He has been improving software processes for startup companies since 2014 and, since 2013, he has been accredited as a permanent professor in the postgraduate computer program. Today, he is a member of ABNT.
Tiago Pagoto was a student in Software Engineering and currently he is Full Stack Software Engineer in the UBsend company.

Funding
Not applicable.

Availability of data and materials
The datasets used during the current study are available from the corresponding author on a reasonable request. A significant analysis is included in this paper.

Competing interests
The authors declare that they have no competing interests.

Received: 27 March 2020 Accepted: 24 July 2020
Published online: 21 August 2020

References

1. Lee G, Espinosa JA, DeLone WH, Lee G, Espinosa JA, DeLone WH, Lee G, Espinosa JA, DeLone WH (2013) Task environment complexity, global team dispersion, process capabilities, and coordination in software development. IEEE Trans Softw Eng 39(12):1753–1771. https://doi.org/10.1109/TSE.2013.40
2. Prikladnicki R, Audy JLN (2010) Process models in the practice of distributed software development: a systematic review of the literature. Inf Softw Technol 52(8):779–791. https://doi.org/10.1016/j.infsof.2010.03.009
3. Carmel E, Agarwal R (2001) Tactical approaches for alleviating distance in global software development. IEEE Softw 18(2):22–29. https://doi.org/10.1109/52.914734
4. Prikladnicki R, Damian D, Audy JLN (2008) Patterns of evolution in the practice of distributed software development: quantitative results from a systematic review. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, EASE'08. BCS Learning & Development Ltd., Swindon, UK. pp 100–109. http://dl.acm.org/citation.cfm?id=2227115.2227126
5. Lings B, Lundell B, Agerfalk PJ, Fitzgerald B (2007) A reference model for successful distributed development of software systems. In: International Conference on Global Software Engineering (ICGSE 2007). IEEE, Munich. pp 130–139. https://doi.org/10.1109/ICGSE.2007.6. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4299847
6. Herbsleb JD, Moitra D (2001) Global software development. IEEE Softw 18(2):16–20
7. Jabangwe R, Smite D (2011) Decision support for offshore insourcing software development. In: 2011 IEEE Sixth International Conference on Global Software Engineering Workshop. IEEE. pp 111–113. https://doi.org/10.1109/ICGSE-W.2011.13
8. Lengnick-Hall CA, Beck TE, Lengnick-Hall ML (2011) Developing a capacity for organizational resilience through strategic human resource management. Hum Resour Manag Rev 21(3):243–255. https://doi.org/10.1016/j.hrmr.2010.07.001
9. Ehrlich K, Cataldo M (2012) All-for-one and one-for-all?. In: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work - CSCW '12. ACM Press, New York. p 945. https://doi.org/10.1145/2145204.2145345. http://dl.acm.org/citation.cfm?doid=2145204.2145345
10. Niazi M, Mahmood S, Alshayeb M, Hroub A (2015) Empirical investigation of the challenges of the existing tools used in global software development projects. IET Softw 9(5):135–143

11. DeFranco JF, Laplante PA (2017) Review and analysis of software development team communication research. IEEE Trans Prof Commun 60(2):165–182
12. Butt AUR, Asif M, Ahmad S, Imdad U (2018) An empirical study for adopting social computing in global software development. In: Proceedings of the 2018 7th International Conference on Software and Computer Applications - ICSCA 2018. ACM Press, New York. pp 31–35. https://doi.org/10.1145/3185089.3185105. http://dl.acm.org/citation.cfm?doid=3185089.3185105
13. Cataldo M, Herbsleb JD (2008) Communication networks in geographically distributed software development. In: Proceedings of the ACM 2008 Conference on Computer Supported Cooperative Work - CSCW '08. ACM Press, New York. p 579. https://doi.org/10.1145/1460563.1460654. http://portal.acm.org/citation.cfm?doid=1460563.1460654
14. Yassine A, Joglekar N, Braha D, Eppinger S, Whitney D (2003) Information hiding in product development: the design churn effect. Res Eng Des 14(3):145–161
15. Fuks H, Raposo A, Gerosa MA, Pimentel M, Filippo D, Lucena C (2008) Inter-and intra-relationships between communication coordination and cooperation in the scope of the 3c collaboration model. In: 2008 12th International Conference on Computer Supported Cooperative Work in Design, vol. 1. IEEE, Xi'an. pp 148–153. https://doi.org/10.1109/CSCWD.2008.4536971. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4536971. http://ieeexplore.ieee.org/document/4536971/
16. Fortaleza LL, Vieira SRC, Junior OOM, Prikladnicki R, Conte T (2013) Using distributed software development in the improvement of communication and collaboration skills in se courses: An observational study. In: 2013 26th International Conference on Software Engineering Education and Training (CSEE&T). IEEE, San Francisco. pp 139–148. https://doi.org/10.1109/CSEET.2013.6595245. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6595245. http://ieeexplore.ieee.org/document/6595245/
17. Rocha RG, Azevedo R, Fernandes R, Cesar Y, Espinhara D, Tavares E, Oliveira A, Franca G, Cassimiro D, Rodrigues C, et al (2015) A system based on ontology and case-based reasoning to support distributed teams. In: 2015 12th International Conference on Information Technology-New Generations. IEEE, Las Vegas. pp 403–408. https://doi.org/10.1109/ITNG.2015.71. http://ieeexplore.ieee.org/document/7113506/
18. Lima AM, Reis RQ, Reis CAL (2015) Empirical evidence of factors influencing project context in distributed software projects. In: 2015 IEEE/ACM 2nd International Workshop on Context for Software Development. IEEE, Florence. pp 6–7. https://doi.org/10.1109/CSD.2015.8. http://ieeexplore.ieee.org/document/7181497/
19. Khan HH, Malik N, Usman M, Ikram N (2011) Impact of changing communication media on conflict resolution in distributed software development projects. In: 2011 Malaysian Conference in Software Engineering. IEEE, Johor Bahru. pp 189–194. https://doi.org/10.1109/MySEC.2011.6140667. http://ieeexplore.ieee.org/document/6140667/
20. Mokashi N, Mahapatra DP, Jha MM (2015) Offshore supplier to preferred partner: A journey. In: 2015 IEEE 10th International Conference on Global Software Engineering. IEEE, Ciudad Real. pp 125–129. https://doi.org/10.1109/ICGSE.2015.15. http://ieeexplore.ieee.org/document/7224490/
21. Chen C, Zhang K (2014) The effects of continuous awareness on distributed software development. In: 2014 IEEE 9th International Conference on Global Software Engineering. IEEE, Shanghai. pp 56–64. https://doi.org/10.1109/ICGSE.2014.28. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6915254 http://ieeexplore.ieee.org/document/6915254/
22. de Farias Junior IH, de Moura HP, Marczak S (2013) Towards a communication maturity model for distributed software development. In: 2013 IEEE 8th International Conference on Global Software Engineering Workshops. IEEE, Bari. pp 81–83. https://doi.org/10.1109/ICGSEW.2013.18. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6613455. http://ieeexplore.ieee.org/document/6613455/
23. L'Erario A, Thomazinho HCS, Fabri JA (2020) An approach to software maintenance: A case study in small and medium-sized businesses it organizations. Int J Softw Eng Knowl Eng 30(05):603–630
24. Soni G, Kodali R (2013) A critical review of supply chain management frameworks: proposed framework. Benchmark Int J 20(2):263–98. https://doi.org/10.1108/14635771311307713
25. Dennis AR, Fuller RM, Valacich JS (2008) Media, tasks, and communication processes: A theory of media synchronicity. MIS Q 32(3):575–600
26. Espinosa JA, Nan N, Carmel E (2015) Temporal distance, communication patterns, and task performance in teams. J Manag Inf Syst 32(1):151–191
27. Fulk J, Boyd B (1991) Emerging theories of communication in organizations. J Manag 17(2):407–446
28. Huang WW, Wei K-K (2000) An empirical investigation of the effects of group support systems (gss) and task type on group interactions from an influence perspective. J Manag Inf Syst 17(2):181–206
29. Jaanu T, Paasivaara M, Lassenius C (2012) Effects of four distances on communication processes in global software projects. In: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '12. ACM Press, New York. p 231. https://doi.org/10.1145/2372251.2372293. http://dl.acm.org/citation.cfm?doid=2372251.2372293
30. Alduaiji N, Datta A, Li J (2018) Influence propagation model for clique-based community detection in social networks. IEEE Trans Comput Soc Syst 5(2):563–575
31. dos Santos ACC, de Farias Junior IHdF, de Moura HP, Marczak S (2012) A systematic tertiary study of communication in distributed software development projects. In: 2012 IEEE Seventh International Conference on Global Software Engineering. IEEE, Porto Alegre. pp 182–182. https://doi.org/10.1109/ICGSE.2012.42. http://ieeexplore.ieee.org/document/6337361/
32. Oinas-Kukkonen H, Oinas H (2004) The 7c model for organizational knowledge creation and management. In: Proceedings of the 5th European Conference on Organizational Knowledge, Learning and Capabilities. University of Innsbruck, Innsbruck. https://warwick.ac.uk/fac/soc/wbs/conf/olkc/archive/oklc5/papers/e-1oinas.pdf
33. Tunio MZ, Luo H, Cong W, Fang Z, Gilal AR, Abro A, Wenhua S (2017) Impact of personality on task selection in crowdsourcing software development: A sorting approach. IEEE Access 5:18287–18294
34. Zanatta AL, Steinmacher I, Machado LS, de Souza CR, Prikladnicki R (2017) Barriers faced by newcomers to software-crowdsourcing projects. IEEE Softw 34(2):37–43
35. Herbsleb JD, Mockus A (2003) An empirical study of speed and communication in globally distributed software development. IEEE Trans Softw Eng 29(6):481–494
36. Sommerville I (2016) Software Engineering, 10 edn. PEARSON EDUCATION, Harlow

37. Khan AA, Keung J, Hussain S, Niazi M, Kieffer S (2018) Systematic literature study for dimensional classification of success factors affecting process improvement in global software development: client–vendor perspective. IET Softw 12(4):333–344
38. Larrucea X, O'Connor RV, Colomo-Palacios R, Laporte CY (2016) Software process improvement in very small organizations. IEEE Softw 33(2):85–89
39. Mesh ES, Burns G, Hawker JS (2014) Leveraging expertise to support scientific software process improvement decisions. Comput Sci Eng 16(3):28–34
40. Desmond C (2017) Project management tools-software tools. IEEE Eng Manag Rev 45(4):24–25
41. Jolak R, Wortmann A, Chaudron M, Rumpe B (2018) Does distance still matter? revisiting collaborative distributed software design. IEEE Softw 35(6):40–47
42. Sayago S, Sloan D, Blat J (2011) Everyday use of computer-mediated communication tools and its evolution over time: An ethnographical study with older people. Interact Comput 23(5):543–554
43. Paul R, Drake JR, Liang H (2016) Global virtual team performance: The effect of coordination effectiveness, trust, and team cohesion. IEEE Trans Prof Commun 59(3):186–202
44. Mesmer-Magnus JR, DeChurch LA, Jimenez-Rodriguez M, Wildman J, Shuffler M (2011) A meta-analytic investigation of virtuality and information sharing in teams. Organ Behav Hum Decis Process 115(2):214–225
45. Burke K, Chidambaram L (1999) How much bandwidth is enough? A longitudinal examination of media characteristics and group outcomes. MIS Q 23(4):557. https://doi.org/10.2307/249489
46. Liao Z, He D, Chen Z, Fan X, Zhang Y, Liu S (2018) Exploring the characteristics of issue-related behaviors in github using visualization techniques. IEEE Access 6:24003–24015
47. Shaikh A, Wiil UK (2018) Overview of slicing and feedback techniques for efficient verification of uml/ocl class diagrams. IEEE Access 6:23864–23882
48. Muşlu K, Brun Y, Ernst MD, Notkin D (2015) Reducing feedback delay of software development tools via continuous analysis. IEEE Trans Softw Eng 41(8):745–763
49. Levinson SC (2016) Turn-taking in human communication – origins and implications for language processing. Trends Cogn Sci 20(1):6–14. https://doi.org/10.1016/j.tics.2015.10.010
50. Drew Paul (2005) Conversation analysis. Handb Lang Soc Interact 1:71–102
51. Mejias RJ, Reinig BA, Dennis AR, MacKenzie SB (2018) Observation versus perception in the conceptualization and measurement of participation equality in computer-mediated communication. Decis Sci 49(4):593–624. https://doi.org/10.1111/deci.12292. https://onlinelibrary.wiley.com/doi/pdf/10.1111/deci.12292
52. Madi T, Dahalin Z, Baharom F (2011) Content analysis on agile values: A perception from software practitioners. In: 2011 Malaysian Conference in Software Engineering. IEEE, Johor Bahru. pp 423–428. https://doi.org/10.1109/MySEC.2011.6140710
53. Carmel E, Abbott P (2006) Configurations of global software development. In: Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner - GSD '06. ACM Press, New York. p 3. https://doi.org/10.1145/1138506.1138509. http://portal.acm.org/citation.cfm?doid=1138506.1138509
54. Akgün AE, Keskin H, Ayar H, Okunakol Z (2017) Knowledge sharing barriers in software development teams: a multiple case study in Turkey. Kybernetes 46(4):603–620. https://doi.org/10.1108/K-04-2016-0081
55. Chen Y, Bharadwaj A, Goh K-Y (2017) An empirical analysis of intellectual property rights sharing in software development outsourcing. MIS Quarterly 41(1):131–161. https://doi.org/10.25300/MISQ/2017/41.1.07
56. Elo S, Kyngäs H (2008) The qualitative content analysis process. J Adv Nurs 62(1):107–115. https://doi.org/10.1111/j.1365-2648.2007.04569.x
57. Bengtsson M (2016) How to plan and perform a qualitative study using content analysis. NursingPlus Open 2:8–14. https://doi.org/10.1016/j.npls.2016.01.001
58. Patton MQ (2014) Qualitative Research & Evaluation Methods: Integrating Theory and Practice, 4th edn. SAGE Publications, Saint Paul. p 832. https://books.google.com.br/books?id=dCNhngEACAAJ
59. Kurose JF, Ross KW (2016) Computer Networking: A Top-Down Approach (7th Edition), 7th edn. Pearson, Boston
60. Lanubile F, Calefato F, Ebert C (2013) Group awareness in global software engineering. IEEE Softw 30(2):18–23. https://doi.org/10.1109/MS.2013.30
61. Damato MAP, L'Erario A, Fabri JA (2016) Awareness, collaboration and comunication's tools for distributed software development: a systematic mapping. In: 2016 35th International Conference of the Chilean Computer Science Society (SCCC). IEEE. pp 1–8. https://doi.org/10.1109/SCCC.2016.7836036. http://ieeexplore.ieee.org/document/7836036/
62. Steinmacher I, Chaves AP, Gerosa MA (2013) Awareness support in distributed software development: a systematic review and mapping of the literature. Computer Supported Cooperative Work (CSCW) 22(2-3):113–158. https://doi.org/10.1007/s10606-012-9164-4

Publisher's Note