We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

2

Globalization of Software Development Teams

M. Rita Thissen, Susan K. Myers, R. Nathan Sikes,
Jean P. Robinson and Valentina Grouverman
RTI International, Research Triangle Park, North Carolina
USA

1. Introduction

The effects of globalization reach into almost every aspect of the software development business. Computers have become nearly ubiquitous in technically advanced societies, whether embedded in household appliances or supporting international data libraries. In parallel with the demand for new computer systems and functions, there is a growing need for highly productive software development teams who can work in the global market. For this reason, there are practical benefits to examining the factors of globalization that influence the success of software teams.

Predictably, advances in communication technologies and transportation draw the business sector into ever greater interaction around the globe (Vardi, 2010), a phenomenon termed "globalization." As commercial ventures, nonprofit organizations, and other agencies establish offices worldwide, the teams that develop software for their use have changed to accommodate the global viewpoint. Many teams experience a need for staff, advisors, and customers from around the world, creating a "globalized" team, and some aspects of software teamwork are changing as a result of the new pressures. What makes a good software team? What makes a good globalized software team? This chapter examines the characteristics of these technical teams, the obstacles to their success, and the various tools available for their use in coping with the demands of globalization.

We focus here on the effects of globalization on team composition and performance, considering technological aids to counteract teamwork challenges that are introduced or heightened by globalization.

1.1 Categorizing development teams by their composition

Not all software teams are alike in composition. We can classify the majority of them into one of three groups, and we use these terms throughout the chapter:

a. **traditional teams**, located in proximity to each other and with knowledge of each other as individuals, sometimes called "collocated teams" (Teasley et al., 2002);
b. **distributed teams**, working from multiple locations, but aware of the identities of team members, sometimes called "virtual teams" (Andres, 2002); and
c. **crowds**, unknown to each other as individuals and located anywhere, such as open-source development teams (crowds may also be considered to be virtual teams because their communication is largely electronic).

Our discussion centers on the first two types. Traditional and distributed teams are formed deliberately for the most part, and stakeholder organizations have some degree of control

over composition and expectations for performance, as opposed to crowds, which may form haphazardly. Because the approach to "crowdsourcing" (see Howe, 2006) differs sharply from traditional or distributed teams, we leave that discussion for another publication. Like crowds, distributed teams may sometimes form spontaneously or from a need expressed by team members rather than by managers. Such grassroots teams soon take on the characteristics of teams formed by directive, and so we do not treat them separately.

Aside from geographic location, a second important dimension of teams is that of affiliation. Traditional and distributed teams may comprise individuals from a single organization or from many. When multiple employers are involved, the teams may be described as **collaborative**, indicating a common set of goals but not a common source of support. When all members belong to a particular organization, the team may be designated as **proprietary**, indicating a unity of purpose and support. For team members and leaders, the collaborative arrangement may add organizational diversity to an already heterogeneous team.

A third important component is the mix of skills, experience, and abilities that members provide. In the following sections, we discuss the interaction of globalization and technical composition, such as the increased opportunity for selecting team members according to capabilities as opposed to geographic proximity.

1.2 Categorizing success factors and barriers to team performance

Not all teams perform alike, even when tasked with similar jobs under similar conditions. How does globalization affect the performance of software development teams, and what tools exist to help the team perform better? Consider what makes a traditional software development team productive, without regard to the effects of globalization or location. Based on our experience, we identify the critical success factors as follows for both traditional teams and for those that are distributed regionally or globally:

a. selection of the people who make up the team,
b. team management,
c. effectiveness of communication,
d. adequacy of tools, and
e. control of external factors.

The topic of the first two items, the selection of team members and their management, can determine the effectiveness of any team. Yet the "right" people for a distributed team may be different from the "right" people for a traditional team, and project management can be more challenging because of the distributed nature of the team. To ensure the success of distributed teams, one must consider explicitly the pressures and barriers to teamwork that occur when the team members are not located in proximity to one another and may have very different cultural and working environments. New global challenges relate to the behavior and expectations of the people assigned to the team, their management, the work itself, and the tools and technologies routinely employed in their tasks.

The next two items seem equally important. In our experience, software team managers who promote communication and provide appropriate aids in the form of software tools are more likely to reach their goals and maintain good working relationships. We explore some of the reasons underlying this statement in the sections on management and tools, with particular emphasis on software tools. The same technological advances that promote globalization on a grand scale can facilitate globalization within a distributed team, particularly in a technically adept group such as software developers.

The last item bears examination in that it differs from the others in character. Regardless of the makeup of the team, success may be influenced by external factors, such as network disruptions, poorly defined requirements, budgetary pressures, and the nature of the work itself. Managers of any team, distributed or collocated, encounter these constraints and events, and successful managers achieve control through various means. It is not our intent to examine team management in general, only in the context of globalization.

In the sections that follow, we offer the knowledge that has been gained by the authors and by others who have reported their experiences in the technical literature. Our goal is to encourage examination and discussion of globalization as it affects the practice of software development, along with the technologies available to improve results.

2. Team formation and management

2.1 Selection of team members

The needs or mission of a project will often dictate the choice of whether to have a distributed or traditional software development team. Software development managers may be more comfortable with traditional or collocated teams when the end product is to be used in a local or in-house environment at a single location. If the software is run within the confines of a smaller organization, then the development team may also take on the same characteristics. Thus, when developing software for use within a limited geographic area, common sense often calls for a team from that same area.

At times, team formation may need to break with tradition to help resolve some difficulty. For example, an essential skill may be needed that cannot be found within the same locale, or deployment of the software may be planned outside the confines of the organization or to other countries, thereby requiring adherence to laws of a specific locale, a software interface design that incorporates specific cultural or language features, or technical support across widespread time zones. A manager of a project or a loosely organized group of people with a common goal may determine the need for the formation of a distributed team to provide the sought-after skills, cultural awareness, or presence in multiple time zones. A recent and highly publicized example of this was formation of the international team that won the "Netflix prize," a computing challenge to improve the probability of anticipating Netflix subscribers' interests. The Netflix prize was awarded to a group of teammates from many academic backgrounds, including computer science, machine learning, and engineering, bringing into play different perspectives and skills. The prize was first offered in 2006 for $1 million U.S. dollars and awarded in 2009 to a team that included Americans, Austrians, and Canadians (Bell et al., 2010).

However, solving one problem can initiate another. For example, a person with a highly unique skill-set who can address a project need may work and live in another time zone, which is inconvenient for other team members and may not easily communicate in the same language as others on the team. Distributed software development teams may encounter impediments to productivity resulting from the same reasons that led to their formation in the first place. Forming a distributed team to provide skills or cultural knowledge not available locally may force team members to work with people who have different technical backgrounds or languages. They may be accustomed to various work environments, legal restrictions, and political situations, and they may use disparate methods or styles of communication or rely on different technical infrastructure. As the physical distance grows between team members, often these cognitive challenges grow as well. Members of a

distributed team who are located in the same geographic area or within a single country may readily become comfortable with each other's cultural, language, and communication nuances. Members of a distributed team that crosses geographic boundaries, especially at great distances, and particularly from multiple organizations in different countries, may find a significant challenge in understanding each other. Thus, forming a team based solely on members' knowledge may introduce operational difficulties as the work gets under way.

Aside from the practical benefits of assembling necessary skills within a team, there is another bright side to including distributed participants. Members may bring a diversity of skills, attitudes, personalities, and even cultures to a project, creating an interesting, exciting, and educational mix. The specialized attributes of the team members can be used to advantage in creating a software product with wide appeal because each team member contributes the point of view of his or her own background. This diversity can foster a natural mind-set of "thinking outside the box" because each team member is already "outside of the box" compared with the others (Hinds & Bailey, 2003). In an atmosphere of openness and acceptance, a programmer in a different geographic region often brings a different viewpoint, which in turn improves the universality of product design and can offer multiple avenues of problem-solving. Each team member can be encouraged to embrace the differences and strengths of other team members to maximize successful results for the team.

2.2 Team management

Many books have been written about team management in general and software team management in particular. See, for example, Covey (1989), DePree (1989), Guaspari (1991), Harragan (1977), Hill (1992), and Humphrey (2010) for general discussions of management, and Brooks (1975), Chrissis et al. (2004), Humphrey (2000), Krames (2005), McConnell (1998), and Wiegers (1999) for ways to manage software development. Many of the principles, practices, and advice can be applied to distributed software development teams although written with traditional teams in mind. Yet some special circumstances involved in directing globalized teams may take a team manager by surprise or cause the team to function with less than full efficiency and productivity. In the following paragraphs, we have merged the cited writers' general principles with some of our own experiences and comments about managing distributed teams.

As mentioned earlier, distributed and traditional teams have commonalities that should be exploited to full advantage. In both types of teams, the team leader brings focus to the group by firmly establishing the mission of the team's project. The leader directs and enforces the use of written specifications for the software with specific protocols for establishing boundaries for the software and assigning each person to specific tasks on the project. Those who lead by consensus seek agreement from all team members as to the scope of their work. Communication guidelines set by the team leader foster a cooperative environment and favorable work relationships. The team leader manages not only the team members and scope of work, but also the hardware and system environment, according to the project's specifications or other factors. The team leader is the final decision maker when judgment calls are needed by the team. Additionally, the team leader manages common tasks, such as project scheduling, labor forecasting, expense reporting, time reporting, and acquisition of needed software or hardware. These responsibilities and roles can be seen in both traditional and distributed teams. Table 1 gives an overview of the similarities and differences between traditional and distributed teams that may influence their management.

Additional responsibilities fall to the manager of a distributed team. The effective team leader feels comfortable communicating with team members whom he or she cannot see face to face. Because of the likelihood of cultural or lifestyle differences in a distributed team, and the low level of personal contact, global development team managers must be results-oriented, judging on clearly defined criteria, such as expected quality level, adherence to schedule and budget, and the degree to which individuals and their products integrate well with the work of the team as a whole.

Trait/aspect	Traditional team	Distributed team
Location	The traditional team is commonly located in same locale and often within same building.	The distributed team can traverse many different locales, time zones, and countries.
Familiarity with team members	Traditional team members often know one another and may be used to seeing one another in casual or business settings.	Members of distributed teams usually only engage one another for the sake of the project goals and may never meet one another face to face. The only contact they may have with one another is for the sake of the project.
Work environment and communication	Traditional team members work in close proximity to one another allowing for face-to-face communication. E-mails, telephones, and shared project documentation supplement the communication environment.	Face-to-face communication is rare. In-person meetings are replaced with telephone conferences and electronic white board sessions. E-mails and phone calls are likely more frequent, and project documentation can be more crucial in defining goals for the project.
Development (hardware, software, network) environment	The development environment is often set by the organization and is rigid and often cannot be easily adapted for software projects out of the ordinary.	The development environment may be as diverse as its team members. Members may be responsible for their own hardware, software, and network setup, requiring creative solutions for integration.
Project management	The manager of traditional teams has the ability to visually oversee work as it is being conducted by team members.	The manager of distributed teams must rely on review of deliverables, status reports, and other nonvisual communication to oversee projects. A greater degree of trust of team members is essential.

Table 1. Comparing traits and aspects of the traditional team versus the distributed team

The able manager of a distributed team seeks strength and advantage in the team members' differences. For example, sometimes hardware and software differences are likely among widely distributed teams as specific hardware, operating systems, or software tools are required by the project (e.g., an application needs to be migrated from the Windows platform to Linux or to the Apple OS environments). In this scenario, the needs of the software product dictate the hardware and software environment needed by its members, yet it may be possible to divide the work such that one individual works on one platform, creating a platform-independent or portable package to be tested by another team member. The team leader will recognize the talents and environments of his or her team members and exploit such differences to successfully reach the team's goals. Effective distributed team leadership requires that the team leader recognize and value the varied characteristics of team members. To overcome the challenges of diversity, the team leader fosters teamwork regardless of differences, promotes the mission of the project, and ensures that each team member benefits from successful completion of the goals. The team leader monitors work fairly regardless of individual personalities or cultural differences to ensure that results meet schedule milestones, relying on a focus on the work to address and overcome any issues between team members.

The leader of the distributed team can assure success by maintaining an atmosphere of responsibility for work results and schedule. Rewards, however, may be challenging to define. Cultural differences may influence whether rewards for meeting milestones would best be monetary or otherwise or whether the reward ought to come as public praise for work performed well or private comments. Section 3 goes into more depth on some of the cultural differences that may affect teamwork and productivity.

2.3 Practical guidelines for monitoring distributed teams

Effective management of global software development teams occurs best if the manager proactively maintains a watchful presence. The need for oversight can be more pronounced while working with distributed teams than with traditional teams because the isolation of one individual from another may allow irritation to fester unnoticed. At the same time, it can be more difficult.

For the greatest probability of success, try to do the following: (a) select staff who will adapt well to the distributed environment; (b) maintain accountability for the team as a whole and for individual staff members; (c) build an attitude of teamwork, trust, and collegiality; and (d) monitor staff availability and work progress.

We recognize that the personal characteristics needed for staff to be effective in a global team may differ from the characteristics of those who do well in traditional teams (Baruch & Nicholson, 1997; Guimaraes & Dallow, 1999). Traditional team members and managers are normally located in close proximity, and the manager or team leader can ascertain the presence of the team member at most any time by dropping in on them physically to see the progress of their work. Although managers of traditional teams can literally "stand over" their team members and their work, managers of distributed teams are deprived of this luxury. Consequently, the manager of a distributed team needs to select team members who can work independently and responsibly and who take it upon themselves to proceed with subsequent steps in a project once they have completed milestones if moving ahead is appropriate. Often the best team members are those who are self-reliant and are good problem-solvers who take the initiative to address challenging issues in their physical or technical environment before asking the team leader to intervene. These team members also

recognize when to ask and receive help if the need arises. In general, such people are experienced because a novice team member requires more oversight and direction or training. Although these characteristics are valuable in any teamwork, they become even more important when team members must work independently at widespread locations, coming together on occasion virtually or indirectly to achieve the team's goals.

The effective manager maintains accountability by setting milestone goals and quality expectations for both the team and for its individual members. The goals should be announced internally so that the team can keep them in mind and self-monitor while developing their software. Each team member then knows the expectations for all and can plan accordingly. The individual team member should be encouraged to communicate with the manager and affected team members when they may not be able to meet the schedule with the desired results or if they are not available to work at the expected pace. Managers can then intervene with additional resources or other alternatives in order to meet the project goals. Being proactive in maintaining accountability is essential to good management of distributed teams.

The effective manager imparts a sense of camaraderie to the team and fosters a reliance on one another to stay on schedule and produce the desired results in a timely fashion. For some teams, it is effective to praise or reward individuals and the team as a whole when they have achieved their goals. The manager can set the tone by encouraging and welcoming praise for one another in communication among team members. These actions will foster desired camaraderie and even help maintain accountability of the team. Underscoring the need for cohesiveness as an antecedent to success, Denning et al. (1989) identified seven universal values and associated practices for coordination in a diversified team: (1) proficiency, (2) capacity to articulate a vision of the team's value, (3) capacity to enter into binding commitments and fulfill them, (4) capacity to spot and eliminate waste, (5) capacity to share real-time assessments of performance, (6) capacity to observe one's own history and how it interacts with others, and (7) capacity to *blend* (i.e., to align with others). These values must be promoted by the team manager to be fully adopted by the team as a whole.

It is essential for the manager to maintain knowledge of the progress of the work assigned to team members in nontraditional teams. For the team to achieve milestone dates and maintain a schedule, the manager should review software results on a periodic basis. This can be achieved by having team members post their work to a common storage location for the manager to review. Work results can be reviewed by using screen-sharing tools between the manager and the team member, as described in Section 6. While reviewing the software, a good practice is for the team leader to be in direct communication with the team member so that the manager can provide real-time feedback and explanation of the software deliverable being reviewed. It should also be a practice to turn the software deliverable directly over to one or more team members chosen specifically as testers to review and test the software for accuracy and completeness of the software as expected. Trust in the team member is always desired, but accountability for the results using software testers ensures the desired results.

3. Cultural effects on teamwork

Cultural diversity has both positive and negative effects on distributed team effectiveness and success. By employing team members from anywhere in a world, organizations can

have access to a larger pool of skills and combine the best expertise available in the field regardless of members' geographic locations, thereby improving team quality and reducing development time. However, as noted earlier, distributed teams face greater communication challenges than traditional teams, especially teams that are not homogeneous with respect to cultural composition. Understanding the impact of cultural differences is one of the keys to distributed-team success in the global environment.

In one study, participants in global teams described challenges associated with intercultural communication and positive effects due to a potential for better decision-making (Shachaf, 2008). The negative impact came from increased complexity of communication, due in part to team differences and in part from working at multiple locations. Cultural and language differences often resulted in miscommunication, which undermined trust, cohesion, and team identity. Study participants mentioned challenges, such as lack of accuracy in both written and spoken communications, requiring team members to invest more time and effort in producing and understanding messages.

Yet cross-cultural diversity can enhance project team experience as a source of innovative thinking that improves the project design and enhances its chances for success through different approaches to solving problems. Because of a wider range of perspectives, cultural diversity can increase creativity and generate innovative ideas and alternative solutions.

Increased globalization is forcing a growing number of managers and employees to interact across linguistic and cultural boundaries that have demonstrable but unnoticed impact. Language differences are generally obvious, but other differences also affect teamwork. Often without people's realization, culture influences how closely they stand, how loudly they speak, how they make decisions, how well they handle conflict, or even their styles of participating in meetings. These differences present a unique challenge to management and teamwork, potentially reaching throughout the organizational structure (El Guindi & Kamel, 2003).

Consider the differences within a single language. In the English-speaking world, American, English, Canadian, Indian, Nigerian, Australian, and other cultures have their own differences of accents and vocabulary. Spoken language versions may range from easily understandable to incomprehensible among a group from around the world, if some team members have very strong accents. There are also regional differences of expression, and it may not be easy to understand colloquialisms. For example, an Australian may say "sticky beak" for a nosy person, and Americans may refer to someone having "horse sense," meaning a practical view of the world. Also, based on the context and tone used, a word or phrase can have several different meanings. A common language helps to overcome cultural barriers and sort out misunderstandings, but it is not necessarily enough; for those who learned a language as nonnative speakers, a common language can be very puzzling (Noll et al., 2010).

Although Western culture currently dominates much of the business world, Asian markets are expanding rapidly. This is particularly true in the software industry, in which the Indian communities of Bangalore and Hyderabad now play a strong role (Glaeser, 2010), but also in China, the Middle East, and other areas. Differences exist in body language, attitude toward age and rank, directness of speech, and attitude toward the passage of time. For example, Americans say, "Time is of the essence," which means that time is of the utmost importance in American business. In the United States, delay or slow pace may be seen as a lack of respect for one's employer, team members, clients, and business partners, and it is very important to get to the point, particularly during business meetings.

Some norms of body language accepted in the West convey the exact opposite of meaning in the East. For example, direct eye contact during conversation in the West is a sign of honesty, while in many Eastern countries it is disrespectful and can be even seen as a threat or hostile behavior. In a distributed development team, body language messages are muted because of the rarity with which team members may encounter each other. However, it is valuable for team members to recognize such differences if they do meet and to be open with each other about their expectations and understanding. Differences in Western and Eastern communication styles may cause dilemmas because so much of a distributed team's interactions depend on written or spoken communication. Western people often prefer clear instructions and direct talk. On the other hand, many Eastern cultures use "coded" speech (Krishna et al., 2004), and a lot is left to the intellectual "decoding" by team members, which enables them to correctly interpret a vague comment that is full of nuanced meaning. Straightforward statement from some Western team members could be misjudged by others, especially by those who are from an Asian background and who might find it disrespectful to their knowledge and ability to understand the underlying meaning.

Cultural background influences how people express themselves and with whom they are willing to share personal issues. For example, although many people in some nations may feel comfortable talking openly about their personal problems, people in Middle Eastern and Asian cultures generally discuss such personal only issues with very close friends. One study found that an American-Israeli team struggled at first with this concept. The Americans reported that Israeli team members were "all business" and never spoke of personal matters (Quinones et al., 2009).

Apart from the issues inherent in interpersonal communication, culture also has a major impact on approaches to problem-solving and strategic decision making. A lengthy consensus-building process is usual in Eastern cultures for making a decision that would be acceptable to all team members. They all would share the responsibility for this decision, and every member of the team needs to feel comfortable with a proposed way to move forward. There is an obvious benefit of such an approach that team members implementing this decision are actively involved in the project design, a concept sometimes referred to as "ownership." The converse also requires adaptation: Asian team members may need to realize that statements by Americans or other Westerners that seem abrupt or excessively pointed may actually be intended as an efficient use of time (Salacuse, 1998).

It is also very important for the entire development team to understand communications correctly, especially when a decision is being made. In Japan, for example, people are reluctant to say "no" or disagree with others, especially those who outrank or are older than themselves, because it is a sign of disrespect. It can be very difficult to be completely confident that a decision or agreement has been finally reached with support from all team members because of this. On the other hand, verbal agreements in Japan would have as much weight as written and signed contracts, while in the United States, they may simply be an indication of willingness to pursue a question further (Chui, 2005).

In today's globalized world, travel restrictions in the form of entry or work visas play an important role in controlling the movement of foreign nationals across borders, which may prevent multinational software development team members from meeting each other in person. Almost all countries now require visas from certain nonnationals who wish to enter their territory. Although visa restrictions are primarily based on citizenship, the holding of a residence permit may also be of importance. For example, a resident of any European Union country that is part of the Schengen zone may travel visa-free throughout that zone

(European Union website, http://Europa.eu). Therefore, on some occasions, team members working within that zone might be able to meet with each other in spite of working across national boundaries.

Geographic location, physical distance, and government policies may separate management and employees within a global team. The separation may require a change in management practices and also a different approach to treating transfer of services, products, and tools across international borders. In many cases, this type of activity may require interaction with government agencies. A manager may need a great deal of information related to international export/import restrictions in order to supply his or her team with materials and equipment. Security concerns are evolving along with the global workforce. In third-world countries, a shortage of hardware and abundant labor may promote hardware sharing, raising the risk of loss or damage to work products stored on the same machine. Data repositories generally cannot be located within countries whose governments may compromise confidentiality or security. To address security concerns, global teams may be required to follow enhanced security procedures, such as using a virtual private network, encryption, and antivirus or anti-malware software.

4. Tools for communication

4.1 The right tools for the team

As noted in the team formation and management section, communication is paramount for good teamwork, but not necessarily easy for distributed teams. For software teams, the specialized languages of programming, platforms, and tools may reduce misunderstanding
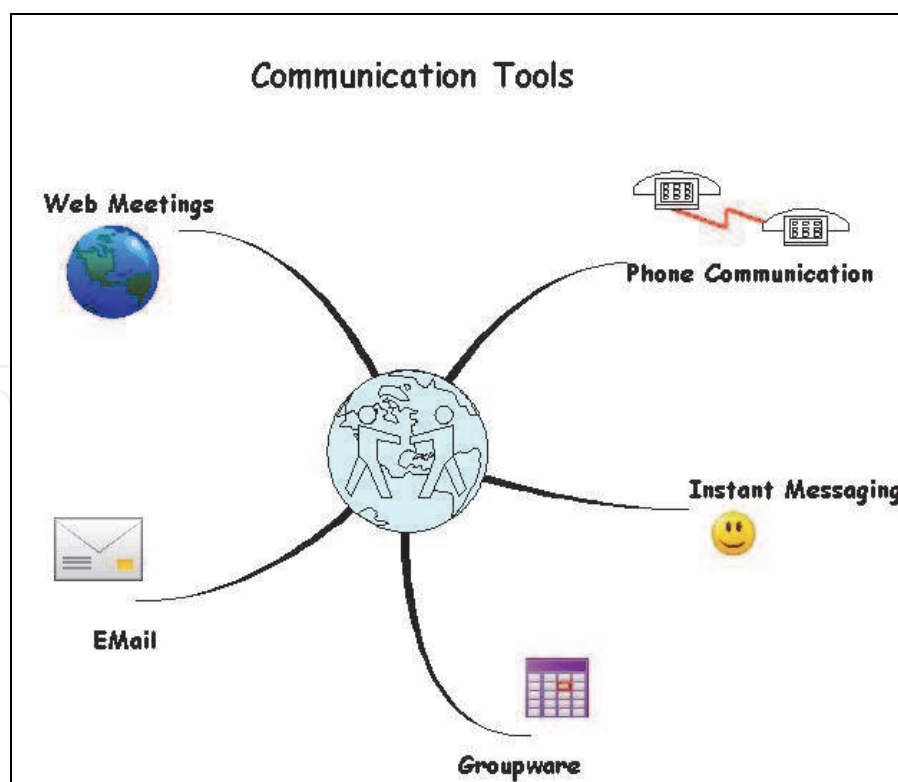

Fig. 1. Conceptual view of global communication tools (used with permission from Thissen et al., 2007)

because of the precise nature of the terms, but the process of communicating over long distances may still be daunting. For this reason, most teams rely on tools and products to aid communication.

Technology-mediated social participation supports closer coordination among larger groups of people, making it possible to address the problems of distributed workgroups in new ways. Kraut et al. (2010) described such team formations as "collaboratories" where collections of researchers who are not collocated work together to address a common problem, coordinating work through technology and social practice.

Communication tools can be divided into three groups: (a) synchronous, (b) asynchronous, and (c) knowledge transfer. Synchronous tools are dynamic or "real time." Asynchronous tools allow information to be transferred or received over a period of time, not requiring simultaneity. A knowledge transfer tool can be a collection of information or a tool that aids in using a collection of information. Depicted conceptually in Figure 1 (Thissen et al., 2007), all of these types of tools contribute to successful distributed teamwork, each in its own way.

4.2 Synchronous communication

Synchronous communication tools for software developers are much the same as those for any other type of far-flung working group. Some forms of communication take place with all participants involved at the same time, such as a telephone conversation or other real-time interactions. Tools that aid this form of communication are called synchronous. We describe here how these tools can help teams share thought processes and therefore boost quality and productivity among a software team.

Table 2 provides examples of the types of synchronous tools that may be of use to such teams. Some of these tools are physical devices, but most are software. Also, most of them

Tool	Examples	Features	Primary advantage
Telephone	"Plain Old Telephone Service" (POTS), Voice over Internet Protocol (VoIP)	Direct calls, conference calls	Familiar to everyone, provides instant interaction
Instant messaging and chat, video chat	Yahoo Messenger, MSN Messenger, AOL Instant Messenger, Internet Relay Chat, Skype	Instant interaction, less intrusive than a phone call	Provides instant interaction, can be used in asynchronous mode if all parties remain connected, with video chat conveying facial expressions for richer communication
Web casts	NetMeeting, WebEx, Citrix, GoToMeeting, ATT ConnectMeeting	Live audio, dynamic video, whiteboard, application sharing	Real-time interaction, augments speech with images and live action displays
Online translators	Google, Yahoo, various other free and commercial varieties available	Instant translation of words, paragraphs, and entire documents	Real-time translation

Table 2. Communication tools: synchronous

rely on the Internet connectivity. Global distributed teams are heavily dependent on the Internet; in fact, the entire growth of globalization for software development has been made possible by the existence of the Internet.

Many teams still rely on telephone communication for direct immediate conversations. The Internet has brought about advances in how distributed teams can benefit from a different type of phone—the Voice over Internet Protocol (VoIP) phone. This technology makes it possible for team members in any location to have a number in the same area code as other team members, reducing long distance or international calling costs. VoIP can also be used to put team members on the same phone network, meaning that a team member in a different location could still be reached by typing an extension versus entering in the entire sequence of area code, and so on. Simple things like being on the same phone network can make teams feel more cohesive.

Instant messaging (IM) can be a quick real-time way to get an answer from a teammate without the disruption of a phone call. Most IM software allows each person to set his or her status, indicating availability. If the person is not available, then IM could be considered asynchronous and function similar to sending an e-mail. If he or she is available, communication can be quick with little disruption. One-on-one communication is referred to as IM; when there are two or more participants, it is referred to as "chat." Some of the IM software allows for the use of web cams and voice along with typed conversation.

"Webcast or webinar, what's the difference?" The difference varies by who is using the term, but most often, a webcast refers to an online presentation, where the audio is only one way. Webinars generally allow two-way or multiway communication. They are similar to seminars and are like a short class or discussion session, but instead of all of the participants looking at a classroom's blackboard, each participant is in front of a computer screen. Some webinars include voice capability through VoIP, while others provide a telephone dial-in number for vocal communication. Webinars are useful for discussions or show-and-tell where all of the team can look at the same information at the same time. Webcasts are useful for one-way sharing, such as updates of information or status reports.

Online automated translation services and locally installed translation software can provide some timely information to a distributed team because the services are readily available and accessible when needed. Global teams sometimes need to work out differences between languages. If there are simple item labels in a software tool being developed, such as a command button with simple commands that needs to be rendered in different languages, these translation tools could provide the basic translation immediately. They could also be used to translate pieces of a message or a document shared by the team for brief or informal communications. If team members are not fluent in a language, these tools could clarify the meaning. Although these tools can help out "in a pinch" (i.e., temporarily or in an emergency), language translation can be fairly complicated because meaning sometimes depends on context and ambiguities may not be resolved correctly. The message may lose coherence in the translation.

Many distributed teams rely on synchronous tools, but there are situations within teams where real-time communication just does not work well. Global or distributed teams have to deal with varying time zones and different work schedules, and so not all of the team can be available at the same time. Even collocated teams find situations where synchronous tools are not right for the situation. Deadlines can make disruptions such as phone calls and e-mail messages undesirable. These types of situations create an opening for asynchronous tools to fill the gap.

4.3 Asynchronous communication

In the globalized workforce, one significant problem is that of time zone differences because it can be difficult for team members to find a common time of day at which all are available for meetings or other forms of communication. And yet, time zone differences can benefit a software development team's productivity because they let the combined team work around the clock. Asynchronous tools are therefore an asset to distributed teams because they facilitate communications without respect to time zone.

Many of the current tools that support asynchronous communication recognize the fact that they may be used by people whose schedules are not well aligned (see Table 3). One team member may be available while the next one is busy. Some of the tools provide indicators as to the user's current state, such as online, busy, unavailable, away from desk, and on the telephone. These software features can be used to advantage by team members to communicate without interrupting or disturbing their coworkers.

Tool	Examples	Features	Primary advantage
E-mail	Numerous vendors and free applications	Send messages or files	Simple way to share information that team members can read when time permits
Groupware/ shared services	Lotus Notes, Microsoft Exchange, Novell Groupwise	Calendars, contact lists, arrange meetings	Takes into account different time zones when scheduling a meeting
Issue trackers	Bugzilla, Mantis, Trac, Redmine, Request Tracker, OTRIS, EventNum, Fossil, Bug Genie, Issues, numerous others	Varies by package: Project Wiki, time tracking, Lightweight Directory Access Protocol (LDAP) authentication, reports	Clear ownership of issues, great for any type of team
Recordings	Recorded meetings, VoIP tools, Skype add-on software, AT&T Connect Meeting option	Audio and/or video recording	Archived record of communication can be made available to nonparticipants

Table 3. Communication tools: asynchronous

Familiar tools of traditional teams, such as e-mail and online calendars, help to fill the need for managing basic team communications across these differences in time zone and schedules. File transfer, issue trackers, and recordings put information within reach of the team members when it is needed. Whether the team is distributed or traditionally collocated, communication and sharing of information need to happen for the team to be

successful. Asynchronous communication tools allow the team to choose the right time to access the information, minimizing disruption to any individual's schedule.

E-mail has been around long enough now that almost everyone is aware of what it is. In 2002, the number of e-mail users worldwide was around 890 million (Gwizdka, 2002) and by 2009 approximately 1.4 billion (Radicati Group, Inc., 2009). It has become a standard form of communication within teams and is hard to replace. With the flexibility of e-mail, a team member or leader can send messages to a single individual or to the entire group and can include file attachments if needed. Beyond the basics of sending and receiving, e-mail software can be used to organize and archive information for ready access. Many e-mail systems have calendars that allow users to set reminders, keep track of tasks, and take advantage of a variety of other features. Messages can be sent without perfect knowledge of the language or perfect pronunciation, aiding teams whose members have differing native languages. For many distributed teams, e-mail is the communication channel of choice.

Groupware or shared calendars are useful tools for keeping up with the availability and commitments of team members. Each member keeps up with his or her own calendar and sets limits on who has access to entries. Other team members can see who is available, who is on vacation, or when they might be available to meet. With shared calendars, each can see this information at a glance for individuals or a whole team. When working with distributed teams, a view of the team's shared calendar is as close as one might get to a traditional team's option of walking down the hall to see which programmers are at their desks.

One of the complexities of writing software as a team is determining who is currently responsible for a specific task, class, build, or package. For programmers, the question may be who is working on a particular bug. This issue crosses into both global and collocated teams, so issue trackers can be a great answer. An issue tracker can be a simple tracking system for one project, a single system for multiple projects, or a part of an integrated project management system. These systems allow issues or bugs to be assigned to a specific team member, with others on the team notified as needed. Viewing bugs by who has been assigned, by team, by category, and by project are desirable features with this type of software.

Often, a team meeting via web or conference call may occur at a time that is inconvenient for one or more staff. When that happens or when critical and complex information is being exchanged, the team manager may choose to record the call, creating an archival record that can be referenced later. Recordings could be viewed in the same way as webcasts, but webcasts most often provide live-streaming information. By recording the webcast or call, the information can be grouped with online training materials or text-based minutes for asynchronous review.

4.4 Knowledge transfer

Knowledge transfer is a specialized form of communication, and it can be difficult to do at a distance. For global software development teams, whose members need to coordinate their activities closely and ensure that their products integrate well, knowledge transfer can present a challenge. In response to that challenge, several products have been introduced and adopted in the marketplace, particularly by distributed teams.

What is knowledge transfer? In any team, some people have greater expertise in one domain than in another. For example, in a software development team, there may be roles for a specialist in gathering requirements, designing a database, determining the best user interface, or evaluating the appropriate platform. These roles may be held by one individual

or shared by several people. It can be very important for each specialist to explain some of the concepts, constraints, or implications of his or her knowledge and understanding. What kinds of tools exist to support the communication of that expertise?

Table 4 shows examples of many tools that can be used to share knowledge in a routine informal way or for more controlled and planned training processes. The table gives a sampling only, and should not be seen as endorsement by the authors nor their organization. Each category of tool is discussed in the paragraphs following the table, based on the authors' experience.

Tool	Examples	Features	Primary advantage
Wiki	WikiPedia, PBWiki, TikiWiki, DocuWiki, MediaWiki	Information in many languages on many topics, constantly updated by contributors, easy to create a custom wiki for a team	Open sharing of information, anytime
Electronic library, institutional repositories	BRICKS, Fedora, Greenstone, Invenio, Refbase, and numerous others	Immediate access to library-type information, books, journals, graphics	Open sharing of information, anytime
Search tools	Google, Yahoo, Bing, Ask.com	Specialized search libraries, such as Google Scholar; access to worldwide extensive information sources	Open sharing of information, anytime
e-learning	Articulate Online, Adobe Elearning Suite, Moodle, SnagIt	Audio/video capture, develop tutorials, learning management systems, image capture (classroom)	Create once, and reuse any time, web-based presentation for easy access

Table 4. Communication tools: knowledge sharing and training

File transfer is commonly linked with data exchange, but when software teams need to communicate, large documents or files can be an important part of that communication. Some programmers may use a File Transfer Protocol (FTP) site as a drop-box location. When items are complete or ready for the next step, they are dropped to the FTP site. The communication is clearly visible as the presence of a file, indicating that this item is ready for the next step. Another team member can pick it up and continue work on it.

What is a wiki? A wiki is a website that anyone can contribute to by using built-in editing tools that post directly back to the web pages. Wikipedia, the free encyclopedia, is a well-known example offered in many languages (http://www.wikipedia.org/). Distributed

teams can create and use a wiki site to share information on a new tool, on a package they are developing, or anything they want to collaborate on—the list could be endless. A wiki is a place to keep information organized in a central location where everyone can add to it or learn from it. Ward Cunningham, one of the creators of the wiki software, calls it, "The simplest online database that could possibly work" (Leuf & Cunningham, 2001).

How often do software developers search online for some piece of information they need to know? Programmers make extensive use of online search tools when they need an answer or an example. Electronic search engines, such as Google and Yahoo, are frequently a programmer's greatest resource because of the immediacy of information and access to technical posts from around the world. For a distributed team, who cannot ask questions to the person at the next desk, online search tools are a common resource to turn to. This ability to search by topic and keyword is a form of knowledge sharing with the rest of the world. Team members can share the knowledge with each other without having to copy or store it; they simply share or pass along the web location.

There many software packages for groups to use when creating their own training materials. They range from simple capture of voice and computer screen input, all the way to learning-management systems that track training progress and full classroom-type image capture. The simple desktop tools, such as SnagIt or Captiva, allow teams to record, save, and share information easily and informally. The first person to learn a new tool could record a quick how to or beginner guide, and the other members could review it when they need it. More complex tools, such as Moodle, offer the ability to create a more formal presentation, quizzes for certification, and shared training materials. For a large and highly organized international team, such as an organizational division that spans the globe, it may be worthwhile to develop training materials in multiple languages and post them on a centralized site.

Electronic libraries have many names, including digital library, virtual information services, institutional repositories, and many other variations, but they all serve the same purpose. These libraries are electronic repositories for reference materials, documents in preparation, regulatory materials, specifications, software lifecycle documentation, and other types of written information. They store most of this information in digital formats or scanned images rather than printed versions. Having access to search through these types of data can open doors for global team members enabling them to find information and share it. For a distributed software team, a library can provide support at each stage of development, from conceptual architecture through design and finally as a host for final product documentation. The ability to search across a wide range of articles, search within an article, and interact with multiple levels of information objects are significant features of electronic journals (Liew et al., 2000).

In the formative stages of the team, or in the design phase of software development, team members often need to expand their specialized skills, understanding of systems, awareness of confidentiality rules, and other topics that may not be easily learned as an individual. E-learning tools offer a way to provide customized, standardized training to the team, without requiring travel. E-learning is any type of learning that is done using some combination of a computer and/or Internet access. It generally includes web-based learning, computer-based learning, or a virtual classroom arrangement. Information is most often delivered via the Internet via recorded package or streaming video, or from a CD or DVD. These can be self-paced or led in real-time by an instructor. E-learning has the major advantage of being accessible from any location and having a variety of formats available from which to choose.

Social network software, such as Facebook, MySpace, LinkedIn, and other public-access websites, now play a growing role in society. To our knowledge, their penetration into software development teams is slight from a business perspective. Although team members may be active participants for their private connections, these sites tend to be frequented by less-focused groups. For business purposes and day-to-day work, members of a distributed software team typically use less open means of communication and transfer, such as those listed in Table 4. As security controls improve and the population becomes accustomed to these cloud-based systems, they may become more prevalent in the future.

5. Software development tools

Software development is a wide and diverse field. A simple breakdown of the types of software reveals three groups: (a) *platform- or device-specific tools*, such as software drivers, operating systems, or hardware-specific products; (b) *commercial or open-source application software*, such as word processors, development languages and environments, research applications, software for creating installation packages, or other independent applications; and (c) *custom user-developed add-on software*, such as document templates, macros, scripts, or command files.

Distributed teams may have different equipment and configurations depending on what type of development they are doing. It may or may not be critical to have the same software, perhaps even the same release, on each team member's workstation. As mentioned in the discussion on team management, the team leader or manager may need to provide and coordinate the development environment according to the needs of the specific project. Coordinating the development environment may be straightforward for small traditional teams, but the effort becomes more complex as the team size grows, the geographic distribution grows, and the number of home countries increases. In one study of computational science, "58% of scientists reported that they do development on their own; 17% work with one other person, and 18% work in teams of 3 to 5 people, while only 9% work in larger groups" (Wilson, 2009).

If development is geared toward a specific platform, and not all team members have the same platform, the team may have to reorganize who is responsible for doing what. If the development is not platform-specific, having development occur on multiple platforms could improve the robustness of the software being developed. The more platforms and equipment the software is tested on, the more reliable the product becomes.

Once development is under way, another type of tool is needed for tracking the versions of the work products and documentation. Most software teams follow a standardized life cycle for development, and many of the formalized approaches, such as the Capability Maturity Model–Integrated process levels (Chrissis et al., 2004), require a significant amount of version-controlled documentation. That voluminous work, in addition to monitoring configuration and build versions, can best be handled by a commercial or open-source revision-control package. There are two basic methods of version control—centralized and distributed. Version control using a centralized model places the code in a shared location, and allows only one person at a time to check out a piece of code. This is the most common approach, and a variety of tools are available to help manage it, such as Microsoft's Visual Source Safe. The other method is a distributed version control system, which allows a programmer to get a full copy of the source and only pass in updates as needed or when an

Internet connection is available, such as the open-source SVK. Some integrated development environments have a version control system integrated within them.

6. Collaborative software for exchanging materials

As Herbsleb pointed out in his paper for *The Future of Software Engineering*, the key phenomenon of global software development is coordination over distance (Herbsleb, 2007). Collaborative software allows people to work together on the same documents and projects over local and remote networks. One of the fundamental needs of any project is the ability to store and share documents, data, test reports, user manuals, and other electronic files among team members, including the software itself. At the most basic level, every team needs to have a centralized location for software, data and documentation, accessible to all. For some teams, this function is served by revision-control software described in Sections 4.3 and 5 while for other teams it may be as simple as a network-level shared disk or folder. Others may use more specialized tools, as described here.

Traditional and distributed teams have different dependencies on the collaborative exchange of materials because of the distance among team members. We mention here some of the available tools and indicate which ones may be more effective in various scenarios. At the simplest level, a traditional team from a single organization may or may not have all of the tools listed in Table 5, depending on the size and complexity of the institution. However, a traditional team is likely to share intranet disk space and therefore may function well without additional tools for exchanging materials. In distributed teams, the Information Technology (IT) infrastructure of organizations may not be compatible among all of the team members' sites and may need a cloud or Internet site with exchange tools, such as web forums, or other universal-access tools. We explore here some options for these teams and also touch upon the project requirements that drive the decision process.

The premise of "cloud" computing has reached most of the world with access to the Internet. It is a viable solution for many collaborative efforts, but there are differing opinions on what constitutes a cloud. For this chapter, we define it as a configuration of scalable IT services that allows for centralized document storage and management. In other words, the cloud offers a way for anyone to use an Internet-enabled computer from any location to access files stored there. Because the concept of cloud storage can underlie many of the tools presented here for discussion, we are not referring to any specific tools as being in the cloud. As discussed earlier, teams in a globalized environment need to communicate and also to exchange materials among themselves. Although informal written and oral communications are important, it is also critical that teams have the tools to facilitate transfer and collaborate on formal documentation, source code, presentations, and so on.

Many types of collaborative software are used by software development teams to exchange materials. We categorize them according to their general purpose and potential use by teams. Both commercial and open-source products are presented, but we do not differentiate them in our discussion. Collaborative software is a rapidly growing market, and new products are introduced at an amazing rate. The list presented in Table 5 is representative, not comprehensive, and it is not an endorsement by the authors or their organization. Some categories and products appear here and in the communications section because the same software may be used for multiple purposes. We talk about each category of tool in the following paragraphs.

Tool category	Examples of commercial or open-source products	Selected features	Primary advantage
E-mail	Outlook, Gmail, Yahoo Mail	Attached files, shared folders	Ability to converse between e-mail servers
File transfer	FTP, Filezilla, CrossFTP, WebDrive, WINSCP, SSH	Secure, access permissions	Security
File transfer	SecureFX, CuteFTP, WSFTPPro, PSFTP, WebDrive, WINSCP, FileZilla, FIRE FTP	Varies by package: directory compare, syncing directories, SSL encryption, search/filtering, integrity checks, remote editing, drag & drop	Very useful when more than a few files or very large files need to be downloaded or transferred
Collaboration platforms	eRoom, WorkZone, Central Desktop, HyperOffice	Alerts, discussion, version control, access permissions	Centralized
Document sharing and management	MS SharePoint, Drop Box, GoogleDocs, CloudPointe, Network Servers, Visual SourceSafe, custom solutions	Cloud storage of files and materials, version control, access permissions	Easy deployment, templates
Instant messaging, chat	Google Chat, Skype, AIM	Available anywhere, attached files	Informal
Shared desktops	Skype, Netmeeting	File transfer, electronic whiteboard, virtual conferencing	Real-time collaboration, compatible across Windows operating systems
Content management	Alterian, MediaValet, Liferay	Librarianship for the web, workflow approval process	User friendly

Table 5. Selected collaboration tools for material exchange and their capabilities

As noted before, e-mail has been at the center of project communications for decades. It can serve as a messaging system, work as a file transfer system, and provide shared folders for archival purposes. It provides a basic form of data dissemination; information may be embedded in the e-mail text, provided as a link to a shared location, or included as an attachment. Although e-mail serves its purpose well, it has many drawbacks. Duplication of documents and version control both present problems that can affect projects greatly; who has not received large attachments on each message in an e-mail chain, overwhelming the mailbox and making it nearly impossible to reconcile the versions? Any savings in using e-

mail over a dedicated tool for document exchange may be diverted to storage costs incurred for multiple copies filed by each e-mail recipient. Although e-mail has its drawbacks, it does remain a very useful and worthwhile tool for project teams. Even in the most widely distributed team, members may send documents via e-mail attachments across e-mail platforms. The attachments may be in any format and if encrypted can allay security concerns. One major drawback is the limitations in some systems for the size of attached files, and teams need to be aware of the impact that large attachments have on e-mail servers or controlled-size mailboxes. Another growing concern is the use of e-mail to spread viruses and other malware, resulting in restrictions and filters on many e-mail systems that prohibit executables and other types of attachments. Regardless of the concerns and limitations, e-mail remains extremely popular as a means of exchanging electronic materials.

A classic method of Internet-based file movement is FTP (mentioned earlier). Several forms are used by institutions, and these packages have varying levels of security. For simple transfers of materials with no security concerns, Anonymous FTP may be employed, while more sensitive material may need to be transferred with or without encryption via products, such as FileZilla. Another feature offered within FTP systems is the ability to add a structure and access levels to the structure. Not all products support all operating systems, so teams need to choose a system that will handle the needs of the team. CrossFTP is one of the most flexible systems and is compatible with Windows, Mac OS X, Linux, BSD, Unix, and AmigaOS. FileZilla can handle all of these except AmigaOS. Collaborative teams may or may not share operating systems, so some forms of FTP may not be a viable tool for exchanging documents.

SSH File Transfer Protocol, otherwise known as Secure File Transfer Protocol (SFTP), is another file transfer protocol that offers file transfer capabilities. Although it is similar to FTP systems, it is based on a different underlying protocol and is more platform independent. In addition to file transfer, SFTP also provides file access and management.

In the past decade, collaboration platform products have emerged as important tools to enable successful teaming across any boundary. Although collaborative software has been available since the mid- to late 1990s, in recent discussions "cloud computing" references have become commonplace. For distributed teams, an option has arisen of storing shared files "in the cloud," that is, at a location accessible through the Internet, but not explicitly known to the user. The cloud provides a solution for a centralized location of project documents and discussions. Some of the commercial cloud platforms offer a wide array of tools. For example, eRoom has discussions, version control on documents, search features, and security access. Another of its key features is the ability to send alerts to team members. Distribution lists are maintained, and the user may send a notice when documents are updated or action is requested. The notices are via e-mail and sent from the server. Although packages like this are very useful and convenient for multiorganization teams, they do come at a cost. The software for some of them must be installed locally and can also be quite expensive.

Document sharing and management is a need as long as a project consists of more than one person, and cloud computing services have had a remarkable effect on software development teams. As long as members have access to the web, materials may be disseminated and updated from any locale. Services such as Google Docs are great choices when teams have access to the Internet and need to collaborate with a document format available to all members. Because the services are housed in the cloud and access to the Internet is required, work will be slowed when connections are interrupted. Unexpected

problems, such as natural disasters, power outages, and political uprisings, can result in Internet service interruption. These situations serve as a reminder that teams that depend on cloud-based services can be adversely affected by outside factors. Other systems, such as Lotus Notes or Microsoft (MS) Sharepoint, may be a more reasonable choice in some environments. For example, teams that are users of MS Office may select MS Sharepoint. Licenses are required, but integration with MS Office is an advantage in many situations. Other features, such as wikis, searches, and bulletin boards, make it a product to be considered. For highly confidential exchange, or for teams with very specific needs, a custom website with document-exchange features may be more cost-effective than a commercial choice (Sattaluri & Thissen, 2005).

If the team is proprietary in affiliation, its document sharing and management needs may be met by the use of the organization's own network servers or intranet. The use of software packages that track changes and materials stored on network servers allow multiple users to work on a central file; changes are marked for team members to accept or reject. This is a simple solution and may be very secure depending on the architecture of the server database. Further security may be achieved by controlling access within the server. Another advantage of network servers is that no access to the public Internet is needed, a very valuable feature at times when Internet service is interrupted.

IM was mentioned earlier as a communications tool. At a glance, IM services may not seem like a reasonable choice for an exchange of materials. However, they do serve a purpose and are used by many. They may be viewed as abbreviated forms of e-mail because files may be sent as attachments. With IM services or chat rooms, team members can have a spontaneous conversation about shared documents, software, or data files. A downside is that with some messaging software, there is no record kept of the conversation unless the participants actively save it.

There are several considerations when selecting tools for materials exchange, including team type, team size, team location, operating systems, security, budget, time constraints and project timelines, and the size and type of files.

Software development teams often need to view prototypes of products and systems as modifications are made. For collocated proprietary teams, this may result in an impromptu meeting in an office to do the viewing. For distributed teams, this may not be an option, and services that enable desktop sharing may fit the need. Desktop sharing provides an effective and impromptu way to demonstrate software systems as development progresses. Although products such as Netmeeting are commonly used for video conferencing and chatting over the Internet, they are also used for file transfer. By simply dragging a file icon to a Netmeeting window, the file may be globally transferred to all conference members. Similar to IM, there is no record of the meeting, but it is a quick and inexpensive way to collaborate on project materials.

For very large teams, or for teams that prepare a large volume of documentation, there may be value to designating a specific person as the librarian, someone who keeps track of the versions and content of each document. Typically, the documents will reside in a common location, and often the librarian may employ an organizing tool, such as content management software. Products such as Liferay offer a platform for web application development and content management. A key feature of Liferay for software developers is the Application Programming Interface (API) with a built-in rules engine, and enhanced Workflow APIs support integration with external workflow engines. This type of tool is particularly valuable for distributed teams, but may also aid traditional teams.

How does one determine what a specific software development team needs? As indicated above, there is a wide variety of choices and approaches. Careful planning and review of project-specific characteristics must be the driving force in choosing tools for successful teams. Determine how much exchange of materials will take place, and evaluate the barriers to sharing project files and documents. Define security and confidentiality requirements for storage, then evaluate software and storage options. Budget constraints may influence a choice between commercial and open-source software. Finally, select products and ensure that team members know how to use them.

7. Conclusion

In this chapter, we have examined the nature and functioning of software development teams that include members who work at locations distant from one another, a type of team that is becoming more common as globalization reaches into all parts of the world economy. These distributed teams may include telecommuters, staff from regional offices of a company, collaborators from various organizations, or even workers at worldwide locations. Regardless of the reasons for team delocalization, the practices of long-distance work take on a different flavor from those of teams who work together at a single site.

What is the impact of this transformation of the software development team structure? What are the driving forces behind team formation, and what are the factors that lead to their success? What are the implications for management and productivity? The topic is an important one for observation, discussion, and analysis in the coming years as globalization rises.

We have attempted to address these questions from a practical perspective with realistic examples to inform the broader community of software development professionals. We believe that distributed teams, already commonplace, may soon become the norm, and those who enter the globalized market unaware will be at a disadvantage. We hope to have increased the readers' level of knowledge of the contributing pressures, benefits, challenges, and obstacles and to have provided some common ground by way of examples to aid others in the software development domain.

8. References

Andres, H.P. (2002). A Comparison of Face to Face and Virtual Software Development Teams. *Team Performance Management*, Vol. 8, pp. 39-48, ISSN 1352-7592

Baruch, Y., & Nicholson, N. (1997). Home, Sweet Work: Requirements for Effective Home Working. *Journal of General Management*, Vol. 23, No. 2, pp. 15-30, ISSN 0306-3070

Bell, R.M., Koren, Y., & Volinsky, C. (2010). All Together Now: A Perspective on the Netflix Prize. *Chance*, Vol. 23, No. 1, pp. 24-29, ISSN 0933-2480

Brooks, F.P. (1975). *The Mythical Man Month*, Addison-Wesley, ISBN 0201835959, New York

Chrissis, M. B., Konrad, M., & Shrum, S. (2004). *CMM: Guidelines for Process Integration and Product Improvement*, Addison-Wesley, ISBN 0321279670, New York

Chui, K. (2005). Managing Projects in Asia: The Challenge of Cultural Diversity, *PMI Global Congress Proceedings*, Singapore, Republic of Singapore

Covey, S.R. (1989). *The 7 Habits of Highly Effective People*, Simon & Schuster, ISBN 0671663984, New York

Denning, P.J., Flores, F., & Luzmore, P. (1989). The Profession of IT: Orchestrating Coordination in Pluralistic Networks, *Communications of the ACM*, Vol. 53, No. 3, pp. 30-32, ISSN 0001-0782

DePree, M. (1989). *Leadership Is an Art*, Dell Publishing, ISBN 0385512465, New York

El Guindi, A., & Kamel, S. (2003). The Role of Virtual Multicultural Teams in Corporate Culture, In: *Advanced Topics in Global Information Management*, F.B. Tan (Ed.), 62-86, IGI Publishing, ISBN 1591400643, Hershey, Pennsylvania

Glaeser, E.L. (2010). *Making Sense of Bangalore*, Legatum Institute, ISBN 978-1-907409-05-9, London, England

Guaspari, J. (1991). *The Customer Connection*, American Management Association, ISBN 0-8144-5837-8, New York

Guimaraes, T., & Dallow, P. (1999). Empirically Testing the Benefits, Problems and Success Factors for Telecommuting Programmes. *European Journal of Information Systems*, Vol. 8, No. 1 (March 1999), pp. 40-54, ISSN 0960-085X

Gwizdka, J. (2002). Reinventing the Inbox: Supporting the Management of Pending Tasks in Email, In: *Proceedings of CHI EA '02, Human Factors in Computing Systems*, pp. 550-551, ISBN 1-58113-454-1, ACM, New York

Harragan, B.L. (1977). *Games Mother Never Taught You*, Warner Books, ISBN 0892560193, New York

Herbsleb, J.D. (2007). Global Software Engineering: The Future of Socio-technical Coordination. In: *Future of Software Engineering, 2007 (FOSE '07)*, L.C. Briand & A. Wolf (Eds.), pp. 188-198, ISBN 0-7695-2829-5, IEEE Computer Society, Washington, DC

Hill, L.A. (1992). *Becoming a Manager: Mastery of a New Identity*, Penguin Books, ISBN 0875843026, New York

Hinds, P.J., & Bailey, D.E. (2003). Out of Sight, Out of Sync: Understanding Conflict in Distributed Teams. *Organization Science*, Vol. 14, No. 6, (November–December 2003), pp. 615-632, ISSN 1047-7039

Howe, J. (2006). The Rise of Crowdsourcing. *Wired Magazine*, Vol. 14, No. 6, p. 06, ISSN 1059-1028

Humphrey, W.S. (2000). *A Discipline for Software Engineering*, Addison Wesley, ISBN 0201546108, New York

Humphrey, W.S. (2010). *Reflections on Management: How to Manage Your Software Projects, Your Teams, Your Boss, and Yourself*, Addison Wesley, ISBN 032171153X, New York

Krames, J.A. (2005). *Jack Welch and the 4 E's of Leadership: How to Put GE's Leadership Formula to Work in Your Organization*, McGraw-Hill, ISBN 0071457801, New York

Kraut, R., Maher, M.L., Olson, J., Malone, T.W., Pirolli, P., & Thomas, J.C. (2010). Scientific Foundations: A Case for Technology-Mediated Social-Participation Theory. *Computer*, Vol. 43, No. 11, pp. 22-28, ISSN 0018-9162

Krishna, S., Sahay, S., & Walsham, G. (2004). Managing Cross-Cultural Issues in Global Software Outsourcing. *Communications of the ACM - Human-Computer Etiquette*, Vol. 47, No. 4 (April 2004), pp. 62-66, ISSN 0001-0782

Leuf, B., & Cunningham, W. (2001). *The Wiki Way: Quick Collaboration on the Web*, Addison Wesley, ISBN 0-201-71499-X, New York

Liew, C.L., Foo, S., & Chennupati, K. R. (2000). A Study of Graduate Student End-Users' Use and Perception of Electronic Journals. *Online Information Review*, Vol. 24, No. 4, pp. 302-315, ISSN 1468-4527

McConnell, S. (1998). *Software Project Survival Guide*, Microsoft Press, ISBN 1-57231-621-7, Redmond, Washington

Noll, J., Beecham, S., & Richardson, I. (2010). Global Software Development and Collaboration: Barriers and Solutions. *ACM Inroads*, Vol. 1, No. 3, ISSN 2153-2184

Quinones, P.-A., Fussell, S.R., Soibelman, L., & Akinci, B. (2009). Bridging the Gap: Discovering Mental Models in Globally Collaborative Contexts, *Proceeding of the 2009 International Workshop on Intercultural Collaboration*, pp. 101-110, ISBN 978-1-60558-502-4, Association of Computing Machinery, Palo Alto, California, February 20-21, 2009

Radicati Group, Inc. (2009, May 6). *Email Statistics Report, 2009-2013: Including Statistics for Email, Instant Messaging, and Wireless Email* [press release]. Available from: http://www.radicati.com/?p=3237

Salacuse, J.W. (1998). Ten Ways That Culture Affects Negotiating Style: Some Survey Results. *Negotiation Journal*, Vol. 14, No. 3, (July 1998), pp. 221–240, ISSN 1571-9979

Sattaluri, S., & Thissen, M.R. (2005). Cross-Organizational Collaboration Using Web Based Technologies. Presented at the 2005 International Field Directors and Technology Conference, Miami Beach, Florida. Listed at http://www.ifdtc.org/pc2005.html

Shachaf, P. (2008). Cultural Diversity and information and communication technology impacts on global virtual teams: An exploratory study, *Information & Management*, Vol. 45, pp. 131-142, ISSN 0378-7206

Teasley, S.D., Covi, L.A., Krishnan, M.S., & Olson, J.S. (2002). Rapid Software Development Through Team Collocation, *IEEE Transactions on Software Engineering*, Vol. 28, No. 7, (July 2002), pp. 671-683, ISSN 0098-5589

Thissen, M.R., Page, J.M., Bharathi, M.C., & Austin, T.L. (2007). Communication Tools for Distributed Software Development Teams, *Proceedings of the ACM-SIGMIS CPR '07 Conference: The Global Information Technology Workforce*, pp. 28–35, ISBN 978-1-59593-641-7, St Louis, Missouri, USA, April 19-21, 2007

Vardi, M.Y. (2010). Globalization and Offshoring of Software Revisited. *Communications of the ACM*, Vol. 53, No. 5, p. 5, ISSN 0001-0782

Wiegers, K.E. (1999). *Software Requirements*, Microsoft Press, ISBN 0-7356-0631-5, Redmond, Washington

Wilson, G. (2009). How Do Scientists Really Use Computers? *American Scientist*, Vol. 97, pp. 360-362, ISSN 0003-0996
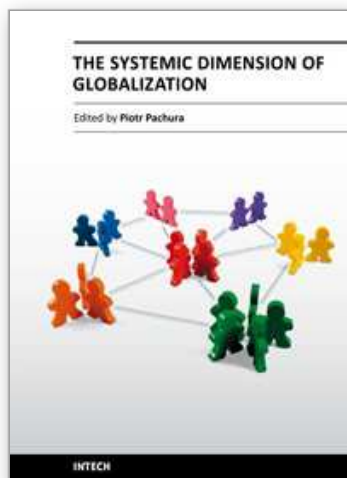
The Systemic Dimension of Globalization

Edited by Prof. Piotr Pachura

Today science is moving in the direction of synthesis of the achievements of various academic disciplines. The idea to prepare and present to the international academic milieu, a multidimensional approach to globalization phenomenon is an ambitious undertaking. The book The Systemic Dimension of Globalization consists of 14 chapters divided into three sections: Globalization and Complex Systems; Globalization and Social Systems; Globalization and Natural Systems. The Authors of respective chapters represent a great diversity of disciplines and methodological approaches as well as a variety of academic culture. This is the value of this book and this merit will be appreciated by a global community of scholars.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

M. Rita Thissen, Susan K. Myers, R. Nathan Sikes, Jean P. Robinson and Valentina Grouverman (2011). Globalization of Software Development Teams, The Systemic Dimension of Globalization, Prof. Piotr Pachura (Ed.), ISBN: 978-953-307-384-2, InTech, Available from: http://www.intechopen.com/books/the-systemic-dimension-of-globalization/globalization-of-software-development-teams

INTECH
open science | open minds