Distributed eXtreme Programming

Michael Kircher
Siemens AG
Corporate Technology, CT SE 2
Otto-Hahn-Ring 6
81739 Munich, Germany
+49 89 636 33789
Michael.Kircher@mchp.siemens.de

Prashant Jain
Siemens AG
Plot No.6-A, Sector 18
Huda, Gurgaon-122015
Haryana, India
+91 124 639 8491
Prashant.Jain@ggn1.siemens.co.in

Angelo Corsaro, David Levine
Dept. of Computer Science.
Washington University
1 Brookings Drive
St. Louis, MO 63130 USA
+1 314 935 5886
{corsaro,levine}@cs.wust.edu

ABSTRACT
One of the key requirements of eXtreme Programming (XP) is strong and effective communication between the team members. To enable this strong level of communication among team members, XP emphasizes the need to have the team members physically located close to each other. However, for various reasons, that may not be feasible.

To address these situations, we propose a crosscutting idea called "Distributed eXtreme Programming" (DXP), which inherits the merits of XP and applies it in a distributed team environment. Our experiences show that DXP can be both effective and rewarding in projects whose teams are geographically distributed.

Keywords
XP, Distributed Development, Computer Supported Cooperative Work

1 INTRODUCTION

eXtreme Programming (XP) **Error! Reference source not found.**] is a lightweight methodology that has gained increasing acceptance and popularity in the software community. XP promotes a discipline of software development based on principles of simplicity, communication, feedback, and courage. It is designed for use with small teams that need to develop software quickly and in an environment of rapidly changing requirements. XP consists of twelve practices, which are Planning Game, Small Releases, Metaphor, Simple Design, Testing, Refactoring, Pair Programming, Collective Ownership, Continuous Integration, 40-hour Week, On-Site Customer, and Coding Standard. A careful analysis of these XP practices reveals certain key assumptions made by XP:

Close Physical Proximity: XP advocates a strong level of communication among team members. Among all of the XP practices, one of the key practices is pair programming. Pair programming is not just one person programming and the other observing. Instead, it is a dialog between people trying to simultaneously design, program, analyze, test, and understand together how to program better. It is a conversation at many levels, assisted by and focused on a computer **Error! Reference source not found.**]. Therefore, a key assumption made by XP is strong and effective communication between the team members, enabling the diffusion of know-how and expertise throughout the group. To enable this strong level of communication among team members, the literature on XP emphasizes that importance of having the team members physically located close to each other. Ideally, the team members should be all in one room. This enhances the communication among team members through incidental over-hearing of conversations and minimizes any hesitation that the team members might have in communicating to each other.

Close Customer Involvement: Another important practice of XP requires close customer involvement through the entire development cycle. Differing from traditional methodologies, XP stresses the role of an *on-site customer* and thus recommends having a representative of the customer working with the team all the time. The customer or one of its representatives thus becomes an integral part of the team. Therefore, a proper communication channel between the customer and the rest of the team can easily be realized if the customer is physically located on-site.

Thus, close physical proximity of team members along with close customer involvement are key assumptions made by XP. However, if physical proximity of team members or the customer is not feasible or desirable, will XP lose its effectiveness? In this paper, we show how XP can be applied to software development and teamwork in a distributed team environment. Our crosscutting idea is whether it is really necessary for the team members to be physically located next to each other.

Section 2 describes our extension to traditional XP that we call *Distributed eXtreme Programming (DXP)*. We describe the assumptions made by DXP, the rationale behind DXP, the challenges in DXP and finally how to address these challenges. In Section 3 we present our experience report on using DXP and in Section 4 we present our conclusions.

2 DISTRIBUTED EXTREME PROGRAMMING

We define Distributed eXtreme Programming (DXP) as eXtreme Programming with certain relaxations on the

requirements of close physical proximity of the team members. DXP applies XP principles in a distributed and mobile team environment. In DXP, team members can be arbitrarily far apart as well as highly mobile. Some ideas towards DXP have already been mentioned at the "eXtreme Programming and Flexible Processes in Software Engineering XP2000" conference.

DXP addresses all aspects of XP, although to varying degrees. There are in general certain things that are irrelevant to the locality of the team, while others are totally bound to the fact that the team members are co-located. The following table summarizes some of the aspects that are relevant to DXP and some that are not.

XP Practice	Requires co-located team?
• Planning Game • Pair Programming • Continuous Integration • On-Site Customer	Yes. These rely on close interaction among the business people, including the on-site customer, and technical people.
• Small Releases • Metaphor • Simple Design • Testing • Refactoring • Collective Ownership • 40-Hour Week • Coding Standards	No. These can be done independent of the fact that the team is centralized or distributed.

From this table it becomes clear that for effective DXP, we need to address the practices of *Planning Game*, *Pair Programming*, *Continuous Integration*, and *On-Site Customer* in a distributed team environment. In Section 2.6 we will see how these practices are addressed.

Note that we consider Refactoring, by itself, to not require co-location. The actual implementation of a refactor relies on Pair Programming. However, deciding whether it is needed or its high level form is separate. Even more concrete design tasks may best be initiated alone **Error! Reference source not found.**]; we consider this to be part of the Refactoring, while the implementation tasks fall under Pair Programming.

2.2 DXP Assumptions

To be effective, DXP assumes existence of certain conditions as well as availability of several tools and technologies. Beyond the assumptions of XP, like speaking a common language and general openness, DXP assumes:

Connectivity: Some form of connectivity needs to exist between the team members. If communication is performed across long distances, it is assumed that the Internet is used as communication media. For company-local commu -nication an intranet can be used.

E-Mail: The ubiquitous availability of e-mail makes it a key enabling technology for DXP. It can be used as a convenient means to exchange information as well to schedule any DXP sessions.

Configuration Management: Effective management of programming artifacts mandates the use of some kind of configuration management tool. This in turn serves as a key enabler for collective ownership.

Application Sharing: To apply XP practices in a distributed environment, some form of application or desktop sharing software needs to be available to the team.

Video Conferencing: For effective communication using audio and video among distant team members, some kind of video conferencing support is needed.

Familiarity: We expect that DXP can succeed only when team members know each other well, and can view it as an extension of their prior work arrangements.

2.3 Why DXP?

XP stresses the need for close physical proximity of team members. However, circumstances may prevent a team from working in close physical proximity, thus mandating the need for using DXP. A company or a project may therefore be forced to adopt DXP for the following reasons:

- *Constrained by situation*: A company or a project may have little choice due to existing physical distribution of development teams. Many projects are sanctioned with teams residing in different locations, sometimes across the globe.
- *Individual Constraints*: An individual may not be able to work at the main project location, at least temporarily. It thus becomes important that the individual continues to stay part of the development activities even while being physically separated.

Even if a company or a project is not constrained by circumstances, it may still choose to adopt DXP. This is because, in addition to maintaining the benefits of XP practices, DXP offers some additional benefits, including:

- *Cost*: A growing trend in the software industry is to outsource all or part of a software project due to cost. It is often much cheaper to get software developed in some countries such as India or China. As a result, several projects get distributed across two or more countries.
- *Convenient Customer Involvement*: DXP makes it easier to involve the customer, even if he/she is unable to be at the development site. With traditional XP, the customer would have to stay on-site. This may not be desirable to the customer since it may cut the customer off from his/her company. In DXP, however, this problem does not arise because the customer need not be on-site and can simply be available to the development team through videoconferencing.

- *Mobility*: In many organizations, some team members need to travel frequently to, e.g., maintain customer contacts, or to attend conferences. DXP offers a smooth integration of mobile team members. Mobile team members can stay connected with the rest of the team using some form of mobile equipment, such as a notebook with a small camera and an ISDN or a dial-up connection. The team members can then participate in the development activities for part of the day or even for just a few hours.

DXP thus addresses both circumstantial constraints of companies and projects as well as offers tangible benefits beyond those offered by XP.

2.4 Challenges in DXP

In relaxing the requirement of XP of close physical proximity, DXP faces several challenges.

- *Communication*: An important aspect of communication is to know how the other person reacts to what one says. To judge a reaction, typically one would read this information from body gestures, the face, and the voice of the other person. In DXP, because the two people are not physically next to each other, how can one receive this information?

- *Coordination*: When two or more team members working together on a project are in two different physical locations, coordination among them becomes a challenge. This can include synchronizing availability, adjusting for time differences, and coordinating distribution as well as integration of activities. In addition, document/application sharing among the team members can also prove to be a challenge.

- *Infrastructure*: Both communication and coordination among team members in DXP depend heavily on the infrastructure. This includes the available hardware and software as well as the bandwidth of the connecting network. A poor infrastructure can make it very difficult to make up for the close physical proximity that can be missing in DXP.

- *Availability*: Distributed team members may be available at different times. Some of them might be working on multiple projects and hence be restricted by time. Others might be constrained by personal limitations. In addition, the availability of distributed team members can also be affected by different time zones.

- *Management*: The manager of the team needs to have a high degree of trust in his/her subordinates if they are often remote. Direct managerial control over distant subordinates can be difficult to execute and therefore new strategies may need to be defined.

2.5 Addressing the challenges/Solution

DXP offers many challenges. However, each of these challenges can be addressed and in most cases overcome.

- *Communication*: Given a close-knit team, good communication can take place among members without requiring physical collocation. For example, assuming you know the other person pretty well, a video picture of your partner might be sufficient to be able to tell what he/she is thinking and how he/she reacts to your comments. The team members can use many different forms of communication to bridge the physical distance. For example, they could convene a video or a phone conference, or could send each other e-mail. In addition, actual meetings could be convened periodically to enhance inter-personal relationships among team members thus easing remote cooperation. When deciding upon a particular form of communication, many different factors need to be considered. These include cost of equipment and its usage, travel costs, cost of time, available bandwidth and the effectiveness of the particular form of communication with respect to the tasks that need to be performed.

  Remote communication and cooperation can be greatly improved by the ability of sharing documents. With web technologies becoming more and more inexpensive and therefore popular, new ways of communication are now available that allow close involvement among team members across intranet or Internet via video-conferencing and application sharing.

- *Coordination*: Proper coordination of activities among distributed team members requires a good bit of planning. However, making extensive use of different lines of communication can facilitate this. For example, two members in different locations could exchange daily e-mails containing their schedules for the day. They could then assign certain slots within the day for working on a project. In doing so, they would also need to take into account any time differences that may exist.

- *Availability*: The DXP team needs to formulate rules and guidelines in order to ensure availability of the team members. The general XP spirit of not denying help to anyone asking for it should be leveraged to being available for remote communication. A daily or a weekly schedule of availability of each team member should be made available and easily accessible to all the team members. Pair programming sessions or testing sessions should then be scheduled based on the availability of the team members and to allow maximum amount of knowledge diffusion to take place.

- *Management*: Project leaders and the upper management need to learn how to handle distributed teams. In particular, project leaders need to learn how

to manage team members who are at different locations. This can include requiring daily or weekly reports from all the team members, whether local or remote. It can also include giving regular feedback to team members to give them a feel that they are connected and hence an integral part of the team. In addition, regular team events can help build trust and motivation among all the team members.

- *Infrastructure*: The availability of the necessary infrastructure is critical, and not as easy to achieve, as it may seem. Important factors in choosing the infrastructure components are ease of use, interoperability with other tools, and availability on different platforms.

2.6 Addressing XP Practices & Values in DXP

In addressing the challenges of DXP it is important that the practices and values of DXP are not violated. As identified in Section 2.1, only four XP practices get affected in a distributed team environment: Planning Game, Pair Programming, Continuous Integration, and On-site Customer. This section examines each of these practices in the light of DXP and proposes possible solutions that can be applied to keep DXP within the realms of XP practices.

Planning Game - For the planning game with the customer being remote, video conferencing and application sharing software support is needed. For example, application sharing can be used to write the story cards. Ideally more than two participants should be supported. Though this is possible with certain solutions, such as CUseeMe **Error! Reference source not found.**], most video conferencing software support only one pair of participants.

Pair Programming - For pair programming between team members in different locations, Remote Pair Programming (RPP) should be used. This requires video conferencing and application sharing support, to share the Integrated Development Environment (IDE).

Continuous Integration - Because a remote team member cannot move to a separate integration machine, an alternative must be provided. If one team member is working at the central team site, he/she can invite the other remote team member to do common integration at that machine. If both team members are remote, this is not possible and therefore integration needs to be done on the development machine.

On-site Customer - Video conferencing should be used to involve remote customers. In DXP, a remote customer is not really an "on-site customer", but a "virtual on-site customer". The big difference is that the customer needs to conform to a certain set of rules such as coordination and availability.

In order to ensure that we did not modify XP in general, we would like to revisit the four values of - *Communication, Simplicity, Feedback, and Courage* - in the context of DXP.

Communication – The use of available tools makes it possible to communicate effectively regardless of physical location. Therefore, the value of communication in DXP is as much as it is in XP, though it may take different forms.

Simplicity - The philosophy "Make it Simple" doesn't depend on the physical location of the team members, so DXP does not affect this value.

Feedback – The value of Feedback is equally important in DXP as it is in XP. The only difference is that feedback needs to be propagated across distribution boundaries. If there are no hurdles in communication among team members, providing effective feedback should not be an issue in DXP.

Courage - This value is not affected directly by the distribution of the team.

Therefore, DXP does not modify the four XP values.

3 EXPERIENCE REPORT

To put DXP into practice, we set up a distributed team to work on a common project called "Web-Desktop Project". The team consisted of:

Prashant, an Indian, working in Delhi, India

Michael, a German, working in Munich, Germany

Angelo, an Italian, traveling between St. Louis, USA, Catania, Italy and Irvine, USA.

David, an American, working in Pittsburgh, USA

In this section we describe the project and how the team worked together. We then present our experiences doing DXP.

3.1 Project Description

The goal of our project was to develop software called Web-Desktop that will provide the working environment for DXP. The Web-Desktop is a desktop that is accessible via a web page. All applications launched on this desktop will actually run on the machine where the desktop was downloaded. The Web-Desktop provides a set of applications needed for most of the development and management processes. Additional applications are available for on-demand installation. The Web-Desktop is state-full; the state is maintained in the server that provides the Web-Desktop service and its components. Clients are completely stateless. This makes it possible to have real user mobility. Such software would allow a team member to use any PC connected to the Internet to log on and have the same look and feel, and the same working environment. All the team would have to tell the customer to get involved is the address of the web page. The solution would give a lot of flexibility to mobile team members working on a project. A mobile team member could now go to an Internet cafe and plug in his/her web

cam and/or microphone and get connected to the rest of the team. There would be no need to download and install software on every machine that the team member uses.

Within the project, we defined roles for each person. David was the customer while Michael, Angelo, and Prashant were the programmers. As we had very little time available, only approximately 3 weeks, we needed to make sure that we focus on the four XP practices, selected in Section 2.

Planning Game - We ran several videoconference sessions with David, our customer, discussing user stories. We used a regular editor and shared it via an application sharing software. e story cards were then discussed and estimated among programmers. Finally, David assigned priorities and selected the cards for the first iteration. Similar work was done for further iterations.

Pair Programming - We assigned story cards to pairs of programmers and began the development process. We used RPP as described in Section 2.6 thus making extensive use of video conferencing and application sharing. We used email to schedule appointments for our RPP sessions.

Continuous Integration – We used CVS as our configuration management tool. We integrated our changes directly from our development branch into the main branch, without changing computers since no integration computer was available.

On-site Customer - We used videoconferencing to effectively involve our customer throughout the project lifetime. We used e-mail to communicate the time and channel for upcoming videoconference sessions.

3.2 Resources Used

We used tools that were well supported, and easy to use and integrate in our working environment. Whenever possible, we picked tools that were either supported on multiple platforms or could interoperate with analogous software on other platforms or followed some standard. As an example both NetMeeting and CUseeMe support the ITU conferencing standard, and therefore can interoperate.

Every computer, desktop or notebook had a microphone, speakers, and a web cam installed. We used NetMeeting as the videoconferencing and application sharing software. For connectivity, we used a variety of links, ranging from 33Kbps modems, 64Kbps ISDN, to 100Mbps LAN connections.

3.3 Hurdles Encountered

During the project, we experienced the following hurdles:

- Our videoconferencing software, NetMeeting, did not allow more than two participants in a session. An additional conference server would have been needed to enable conferences with more than two participants.
- It was cumbersome to capture story cards in a text file. A better solution might have been to use a custom WikiWikiWeb.
- Sharing of applications across operating system platforms was not possible using the NetMeeting application sharing functionality. Virtual Network Computing might be a solution to this.
- We used a simple text editor for brainstorming, making the process quite cumbersome. A tool like MindMapper could have made discussions about new ideas easier.
- Narrow bandwidth connections, e.g. dial-up, hindered the use of video because of jitter introduced in audio along with reduced responsiveness of application sharing. Our fallback strategy was to use only audio conferencing, or to switch to a chat channel.
- Power outage is at least in India still a problem. A notebook computer with its own battery can be a valuable help, at least for short outages.
- Lack of uniform access to the source code repository is not a major hindrance, but results in inconveniences that can have larger effects in the long run. As Prashant had to work most of the time from behind a firewall, he was not able to connect to the team repository directly. Other team members had to send him snapshots of the code via e-mail. This process was tedious and error-prone.
- Some of the keyboard settings were different among the team members. For example, some characters like braces seemed to work only if the parties involved in the conference used the same keyboard, i.e. both American, or German.

3.4 Lessons Learned

Our project was quite successful in using DXP and in the process we gained some valuable experiences.

We found that using a combination of synchronous communication, such as videoconferencing, and asynchronous communication, such as e-mail, to be the most effective. Even though we used videoconferencing along with application sharing, it could not completely substitute the physical closeness as well as effectiveness offered by XP. A video picture of the partner was sufficient to tell what he was thinking or how he reacted to a comment. However, what was missing was the physical presence of the partner, which usually gives company and can therefore never be completely substituted with any kind of videoconferencing tool.

Parallel development raises the issue of source code integrity. Tools such as CVS and ClearCase address the issue. Even though these tools support distributed development, we have found that making mutually exclusive changes helps reduce merge conflicts.

Therefore, we used an email token to serialize change access when teams were working on common code sections.

4 CONCLUSION

DXP can efficiently integrate remote and mobile team members into the development process and is therefore a valuable extension to traditional XP. In addition, it allows a much more effective involvement of the customer compared to XP, especially in situations where it seems impossible to have an on-site customer.

DXP can therefore actively broaden the acceptance of XP as a lightweight though effective software development process. We are aware that a virtual meeting through a computer-supported interaction can never replace direct human interaction. However, there are situations where such interaction is not feasible, and where a form of XP can still be successfully employed.

As we wrote this paper, we realized that we heavily touched the field of Computer Supported Cooperative Work. Further investigations need to be made how DXP relates to this. We have found, not surprisingly, that for computer-supported interaction to be successful, live pictures and tone, namely video and audio, are elementary.

We plan to document guidelines on how to implement DXP in a project in future papers.

The solutions proposed in this document might just be the first steps to a general revolution in human interaction – the long missed multimedia revolution, which is yet to happen.

REFERENCES

1. K. Beck, Extreme Programming Explained: Embrace Change. Reading, Massachusetts: Addison Wesley Longman, Inc., 1999.

2. CUseeMe Networks, Voice and Visual Communications Over the Internet, http://www.cuseeme.com, 2001.

3. Applied Informatics and Distributed Systems Group, Technical University Munich, Computer Supported Cooperative Work, http://www.telekooperation.de/cscw/, 2001.

4. GNU Project – Free Software Foundation, CVS – Concurrent Versions System, http://www.gnu.org/-software/cvs/, 22 July 2000.

5. M. Kircher, and D. Levine, The XP of TAO – eXtreme Programming of Large, Open-source Frameworks, Extreme Programming Examined, Addison-Wesley, 2001

6. Bosley Group, Mind Mapper, mind mapping software, http://www.mindmapper.com, 2001.

7. Till Schuemmer, and Jan Schuemmer, Support for Distributed Remote Pair Programming, Extreme Programming Examined, Addison-Wesley, 2001

8. R. Steinmetz, and K. Nahrstedt, Multimedia Computing, Communication & Applications, Prentice Hall, NJ, 1996

9. L. A. Williams, and R. R. Kessler, All I Really Need to Know about Pair Programming I Learned In Kindergarten, Communications of the ACM, 2000

10. W. Cunningham, Wiki Wiki Web, Portland Pattern Repository, http://www.c2.com/cgi/wiki?WikiWikiWeb, 5 January 2001.

11. AT&T Laboratories Cambridge, Virtual Network Computing, http://www.uk.research.att.com/vnc/, 2001.

12. Microsoft, NetMeeting Home, http://www.microsoft.com/windows/netmeeting/, 2001.

13. Rational ClearCase, http://www.rational.com/-products/clearcase/index.jsp, 2001.

14. G. S. Cowan, "What kinds of tasks are best performed alone?" in Pair Programming, Portland Pattern Repository, http://www.c2.com/cgi/wiki?PairProgamming, 10 November 2001.