# Online tools for virtual pair programming

E. Mnkandla
Department of Business Information Technology
University of Johannesburg
Johannesburg, South Africa
emnkandla@uj.ac.za

**Abstract**:

As agile methodologies advance in process maturity we find that most of their practices such as Test-Driven Development, refactoring, and pair programming (specifically for Extreme Programming) are becoming the order of the day in a number of organizations that were traditionally sceptical about agile development. Without implying that Microsoft has ever been against agile development it is interesting to note that they now have a very comprehensive set of tools for agile development under the MSF for Agile set. Published literature has a lot of empirical evidence on the gains of using pair programming for development teams and even for teaching programming. However, the remaining challenges relate to the use of pair programming in distributed development environments. Hence, a gap still exists in determining the feasibility (especially with regards to data security) of doing pair programming for virtual teams and also developing appropriate tools for such practices. I suppose the lack of appropriate tools for such activities could have delayed the comprehensive adoption of this kind of practice. In today's globally connected world where the world network can be traversed on a hand-held device in a split of a second, it is worth investigating what sort of tools could be securely used for virtual pair programming. Such an investigation becomes more valuable especially if we consider the ever growing complexities of all sorts of viral attacks on our data and the apparent growth of other evils such as terrorism. This paper investigates the prevalence, effectiveness and security issues of using online collaboration tools such as NetBeans, VNC, Google docs, and some open source tools such as RSS Dashboard and many others to implement virtual pair programming.

**Keywords**: Virtual pair programming, online collaboration tools, virtual teams.

## 1. Introduction

One of the major claims of agile development is simplicity in both design and implementation. It is therefore a virtue in agile development to employ simple tools during design and implementation. The result of such efforts is seen in the vast number of tools used by agile developers at different stages of development. The more tools we use the easier it is for any developers to find their comfort zone. Paradoxically however, to novice developers the abundance of tools can appear to be a complexity and an impediment to the learning process. It is not the intention of this paper to investigate the pros and cons of virtual pair programming but rather to reveal the abundance of online tools that can be used for virtual pair programming though initially intended for other kinds of online collaborative work. The common tools that come to mind are online collaboration tools such NetBeans, VNC, Google docs, and some open source tools such as RSS Dashboard and many others that can be used to implement virtual pair programming. The major challenge of doing virtual pair programming is that while the code may be shared and jointly developed and edited the means of communication will not be face-to-face which happens to be a value in agile development. You may argue that skyping and the use of webcams could alleviate the problem but the benefits of face-to-face communication are

hard to replace or even emulate. Collaborative online tools have been used mainly for more social activities where data security may not be of as much value as in the development of private applications for clients.

A survey of the use of online collaborative tools for the purposes of virtual pair programming was carried out and the results are discussed in the sections that follow. The rest of the paper presents in the next section a brief background of virtual pair programming and online collaboration tools. Then the results of the survey are discussed. A discussion follows and a conclusion closes the presentation.

## 2. Background

This section reviews the backgrounds of virtual pair programming, and online collaboration tools.

### 2.1 Virtual pair programming

Pair programming is an agile practice where two programmers take turns to write the same code on one computer. While one programmer is writing the other programmer reviews the code. This practice has for many years been used by Extreme Programming (XP) practitioners. In the early days of XP this practice was surrounded with a lot of controversy as to its effectiveness and productivity. After almost a decade of research in XP practices there is published empirical evidence of benefits resulting from the use of this practice. The following references have details on the value of pair programming (Bryant et al, 2006; Canfora et al, 2005: 92-99; Cockburn and Williams, 2001: 223- 248; Williams et al, 2000: 19- 25). The practice of pair programming becomes a challenge in environments where the teams are not collocated and this brings us to the concept of virtual pair programming.

Virtual pair programming is applicable in distributed project environments. In order for teams to practice pair programming in distributed environments the use of collaboration tools is inevitable. Gould (2000) defines virtual teams by combining the common aspects of different definitions and these mutual characteristics are: they should be a team, they should be physically separated by time and /or space, and they usually interact electronically. If virtual teams apply pair programming it means that besides taking turns to work on the same piece of code these team members who are separated geographically and by time need to communicate 'face-to-face'. You may ask why face-to-face? Well, because pair programming is an XP practice and XP is an agile methodology and agile methodologies value face-to-face communication more than other forms of communication. It is in fact very hard to replace face-to-face communication. In agile development teams are supposed to be small i.e. less than ten developers per team in order for face-to-face communication to be effective. Of course a methodology is only a guideline or at least it should be. So if you are faced with distributed teams or large teams that have more than ten members (which is very common) then you may need to tailor the methodology to your project environment, and any reasonable methodology would allow such a practice. Hence in order tailor the face-to-face communication practice there is need to go back to the principle from which this practice is derived. Face-to-face communication brings with it accountability and physical presence. So the organisation should find a way of communicating such that accountability and physical presence are not compromised.

The benefits of virtual pair programming include not only applying agile practices despite the distributed nature of the teams, but also some benefits of virtual collaboration such as: telecommuting, the ability to work with people in spite of their physical location, and no need to deal with other operational logistics that are normally associated with physical presence of employees.

## 2.2 Online collaboration tools

A number of online collaboration tools exist today but the ones discussed here are the most commonly used in IT whether for programming or other use. The focus of this research though is to consider the use for programming purposes. The benefits of pair programming as shown in (Baheti et al, 2002; Foley, 2000; Hanks, 2004; Last, 1999; Schummer and Schummer, 2001) build a case for use of collaboration tools beyond levels that can be ignored.

**RSS Dashboard**: is an open source online collaboration tool that is mainly used to monitor online conversations across the board from RSS-based to Yahoo pipes. There are several online discussions that have raised concerns about the security issues around RSS Dashboard as people especially publishers find more use of this tool the hackers are also taking opportunity to see what they can prey upon. When RSS was still used by technical people security issues were not necessarily a problem, but now that the general public has expanded RSS to include all sort of communication and file attachments security dangers are looming.

**Netbeans**: is basically a platform that was used to develop an IDE (integrated development environment) for Java. Because of the flexibility of this IDE which allows collaborative development, modular development etc it is being used for collaborative team work. The value of Netbeans comes from the ability of modules to be developed independently which means that different developers can extend (collaborate) to the modules. Hence its applicability to virtual pair programming since the developers do not have to be collocated in order for them to collaborate. In terms of security issues Netbeans actually allows the developer to set the security features which dispels most security fears.

**VNC**: stands for virtual network computing. VNC is a protocol that allows a view of any computer's desktop through the use of a VNC server which is platform independent. It is therefore possible for VNC users to share any kind of data including code because the system cross-communicates all keyboard and mouse events. Security is generally a weakness of VNC due to a lack of password encryption in its protocol. However, security can be beefed up if VNC is sent over SSH (Secure Shell) or VPN (Virtual Private Network)

**Google Docs**: this includes a number of useful collaborative tools such as Google Documents List Data API which allows users to put their documents on Google docs and collaboratively edit them. Some developers have used Google Documents List Data API for coding in different environments. However, the most common use is with word documents and other office applications. The security of Google Documents List Data API like any other Google Docs application is based on the confidentiality of passwords which lies with the user.

**Skype**: is an application that allows users to make voice calls and video calls through the Internet. It also provides integrated communication with cellular phones and fixed lines. Hence it

serves as a telephone communication system and does not amount to a code sharing environment. This means that it has to be used with other tools.

**VOIP**: is a protocol for sending voice calls using the IP protocol. It makes it cheaper to call distant numbers than on ordinary fixed lines for all users connected to the Internet. Like Skype it can only be used in conjunction with another tool that allows code to be shared. Its advantage is that it uses the computer which is where the developer could be working.

## 3. A comprehensive evaluation of online collaboration tools

As the adoption of agile methodologies continues to grow their application has expanded to all types and sizes of organisations from small to large and from single to distributed teams. The challenges of applying agile practices have therefore exceeded the initial intentions of the authors of agile methodologies. This research work involved carrying out a survey to determine the extent of virtual team usage with agile development. A questionnaire was sent out mainly to agile practitioners to find out among other things the variables shown in table 1 and table 2.

### 3.1 Overview of collaboration tools

Table 1 shows the general opinions of practitioners who use the tools by considering issues such as the frequency of use, the effectiveness and security of use.

**Table 1: Analysing online collaboration tools**

| Tool | Prevalence | Effectiveness | Security |
|---|---|---|---|
| RSS Dashboard | 2% | 80% | 40% |
| Netbeans | 70% | 90% | 95% |
| VNC | 2% | 85% | 5% |
| Google docs | 10% | 95% | Not known |
| Skype | 2% | 20% | 50% |
| VOIP | 7% | 85% | 50% |
| Telephone | 2% | 81% | Not known |
| Email | 5% | 87% | Not known |

The results shown on table 1 are based on a survey of ten developer organizations that use some form of virtual pair programming for some projects.

- Prevalence: measures the percentage of the people surveyed who use/used the tool.
- Effectiveness: measures the success (achieving what was intended) of using the tool.
- Security: reveals the user's opinion or understanding of the security of their data when using the tool.
- Not known: means that people were not sure of the security of the tool.

The data reveals that:

- Prevalence: the most frequently used tool is Netbeans

- Effectiveness: most between (80% and 90%) found their tools effective. But the results do not mention that some users actually combined their tool with other tools e.g. Netbeans used together with VOIP communication.
- Security: serious security concerns are shown for tools like VNC.

## 3.2 Choice of collaboration tools

Table 2 shows the results of online tool choice and usage. The percentages shown here indicate the level of usage as shown by which tools people prefer to use. The results also show what users think about the security of their data when using the tools. The issue of the need for training also appears. There are fewer people who however failed to use the tools effectively.

**Table 2: Online tool choice and usage**

| Topic | | Strongly Agree | Agree | Disagree | Strongly Disagree |
|---|---|---|---|---|---|
| I prefer to use: | RSS Dashboard | 2% | | | |
| | Netbeans | 60% | 10% | 0.5% | |
| | VNC | 1% | 1% | | |
| | Google docs | 10% | | | |
| | Skype | 20% | | | |
| | VOIP | 25% | | | |
| | Telephone | 40% | | | |
| | Email | 35% | | | |
| | Other tools | 20% | | | |
| Use of these tools may pose security problems | | 20% | | 30% | |
| More information or training on the tools is needed | | 1% | | 50% | |
| Have tried using one of the tools without success | | 0.5% | 1% | | 20% |

The percentages will not add to a hundred for each row because other respondents used the columns to indicate their partial agreements or disagreements with the use of certain tools.

## 4. Discussion

The results of the survey show that most organisation who actually need virtual pair programming by nature of their business do not necessarily use virtual pair programming due to the hesitancy of using online collaboration tools. Some do not really know what tools are out there and others need some education as to the effectiveness of the tools. Some have failed to manage the use of the tools. Analysing the security issues for most of these tools reveals that the developers of tools need to improve security of the tools in view of the generally extended use. Most of the tools are not being used for what they were originally designed to do. In most cases it is these innovative applications of the tools that have resulted in security holes.

## 5. Conclusion

This paper gave an overview of online collaboration tools that are used or can be used in virtual programming environments. Particular focus was put on virtual pair programming which is an XP practice and would stand to benefit from online collaboration since it was originally intended for collocated teams. The general conclusion is that there are many tools out there but people do not use them due to among other reasons la ack of knowledge about the tools, a lack of security confidence on the tools, and bad experience from previous failure to effectively use some of the tools. Those who are involved in online collaborative work are finding more and more innovative ways of adding value to their business. Virtual collaboration tools are therefore worth trying rather than avoiding.

## 6. References

Baheti, P., Gehringer, E. and Stotts, D. 2002. Exploring the efficacy of distributed pair programming. In: *Proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods - XP/Agile Universe 2002*, p.208-220, August 04-07, 2002

Bryant, S., Romero, P and du Boulay, B. 2006. The Collaborative Nature of Pair Programming. In: *Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering (XP2006).*

Canfora, G., Cimitile, A. and Visagio, C. 2005. Empirical study on the productivity of the pair programming. In: *Proceedings of the 6th International Conference on Extreme Programming and Agile Processes in Software Engineering*, Sheffield, UK. Baumeister, Machesi, Holcombe (eds). Springer-Verlag, Berlin, 2005: 92-99.

Cockburn, A and Williams, L. 2001. The costs and benefits of pair programming. In: *Extreme Programming Examined*, Succi, G., Marchesi, M. (eds). pp. 223–248, Boston, MA: Addison Wesley, 2001.

Foley, S.P. 2000. The Boundless Team: Virtual Teaming. A Report for MST 660, Seminar in Industrial and Engineering Systems, Master of Science in Technology (MST) Graduate Program, Northern Kentucky University, July 24, 2000. Available http://esecuritylib.virtualave.net/virtualteams.pdf (accessed 10 May 2009).

Gould, D. 2000. Leading Virtual Teams. Leader Values, Available http://www.seanet.com/~daveg/vrteams.htm (accessed 05 May 2009).

Hanks, B.F. 2004. Distributed pair programming: An empirical study. *XP/Agile Universe 2004*, Salt Lake City, USA.

Last, M.Z. 1999. Virtual Teams in Computing Education. *SIGCSE 1999: The Thirtieth SIGCSE Technical Symposium on Computer Science Education*, LA, New Orleans, 1999, Doctoral consortium.

Schummer, T and Schummer, J. 2001. Support for Distributed Teams in Extreme Programming. In: *Extreme Programming Examined*, Succi, G., Marchesi, M. (eds). 355–377, Boston, MA: Addison Wesley, 2001.

Williams, L., Cunningham, W., Jeffries, R. and Kessler, R.R. 2000. Strengthening the case for pair programming. *IEEE Software,* 17(4):19-25.