Analyzing Collaboration in Software Development Processes through Social Networks

Andréa M. Magdaleno[1,3], Cláudia Werner[1], Renata Araujo[2,3]

[1]Programa de Engenharia de Sistemas e Computação (PESC) – COPPE/UFRJ
Caixa Postal 68.511 – 21945-970 – Rio de Janeiro – RJ – Brasil

[2]Programa de Pós Graduação em Informática (PPGI) – UNIRIO

[3]Núcleo de Pesquisa e Prática em Tecnologia (NP2Tec) – UNIRIO
22290-240 – Rio de Janeiro – RJ – Brasil

{andrea, werner}@cos.ufrj.br, renata.araujo@uniriotec.br

Abstract. *Plan-driven, agile or free/open source software are models that have been suggested as alternatives for software development processes. Although effective to some extent, they alone cannot fully address all the variability of projects and organizations. In this work, it is argued that two distinct characteristics of these models – collaboration and discipline – can be the drivers to tailor software development processes to meet particular needs of projects and organizations. This article focuses on the aspect of collaboration and argues that it can be analyzed through social networks.*

1. Introduction

Software organizations are continually challenged by the need to improve the quality of software products. In this context, the assumption that the adopted software development process directly influences the quality of the developed product [Cugola and Ghezzi, 1998; Fuggetta, 2000] has motivated many organizations to adopt maturity models, such as CMMI [Chrissis et al., 2006].

This "plan-driven" development model has been used to support the definition of less chaotic, more predictable and managed software development processes. The success of some free/open source software (FOSS) projects, like Linux and Mozilla, also caught the attention of academia, industry and users due to their capability to produce high quality software, quickly and free [Feller and Fitzgerald, 2001]. In addition, agile methods are often presented as an alternative to plan-driven development to cope with changes that occur during a development project through shorter development cycles and with a higher level of involvement and participation of the client [Beck et al., 2001; Cockburn, 2001].

Software organizations engage in a wide variety of projects with different characteristics, where plan-driven, agile and FOSS development models, usually perceived as opponents, complement each other, because each one works better or deals with difficulties in some aspects. None of these development models will fulfill all requirements of a specific project or organization. Thus, approaches that balance the various development processes are necessary. This need can be observed by an increase of development processes tailoring in organizations [Hansson et al., 2006], while the number of organizations that follow a reference model in a completely prescriptive manner decreases [Patel et al., 2006].

Process tailoring is the act of particularizing a general process description to derive a new process applicable to a specific situation [Ginsberg and Quinn, 1995]. In this scenario, the research question investigated in this work is: how to tailor software development processes, according to projects and organizations needs?

However, Pedreira et al. [2007] summarize the negative consequences of performing bad process tailoring in organizations: the project budget, the development time, and the product quality depend directly upon the quality of the software process; a bad software process may involve unnecessary activities that lead to a waste of time and money, or the omission of those activities that are necessary, which may affect the product quality; and inappropriate process tailoring can cause the software process not to comply with the organizational standard process or with international standards such as CMMI [Chrissis et al., 2006].

To avoid these risks, we claim that process tailoring should consider organization, project and team contexts, using collaboration and discipline as main drivers [Magdaleno, 2010a]. Discipline refers to plan and direct process, while collaboration focuses on people interaction. Both are complementary and essential in any project, but in different proportions, depending on the project characteristics. Therefore, they need to be balanced.

This paper focuses particularly on the aspect of collaboration and argues that it can be explained using social networks. In this sense, we identified the requirements necessary to explore collaboration, through social networks, in software development. These requirements led to the beginning of the construction of EvolTrack-SocialNetwork tool.

The remainder of this paper is organized as follows. In Section 2, we present the main characteristics of each software development model. Section 3 details the solution focus on balancing collaboration and discipline. Section 4 is dedicated to social networks. Finally, Section 5 concludes the paper.

2. Software Development Models

A software development model is a set of practices recommended for developing software. These practices are organized into a software process that corresponds to “a coherent set of policies, organizational structures, technologies, procedures and artifacts required to design, develop, deploy and maintain a software product” [Fuggetta, 2000].

Plan-driven, agile and FOSS development models have the same goal: to improve software development, but they adopt different approaches. While the plan-driven development seeks for predictability, stability and reliability [Chrissis et al., 2006], agile development tries to quickly add value to business and adapt to market, technology and environment changes [Cockburn, 2001]. Furthermore, in FOSS development, the main objective is to guarantee users’ freedom [FSF, 2008].

Plan-driven development is typically exemplified by maturity models, such as CMMI [Chrissis et al., 2006], and is characterized by its orientation to planning and emphasis on well-defined processes. The assumptions of agile development, observed in methods such as XP (Extreme Programming) [Beck, 1999] and Scrum [Schwaber, 2004], can be summarized by four values of the Agile Manifesto [Beck et al., 2001].

The FOSS development can be understood by the *bazaar* metaphor [Raymond, 2001], where projects are collaboratively and transparently developed. In this model, developers work on a voluntary basis, geographically distributed around the world, using the Internet as a communication channel.

Each one with its peculiarities, successes and challenges, the three development models have followed separate paths. Due to differences in vocabulary, misinterpretation and misuse of approaches, they are usually perceived as opponents. However, all of them had, in the last decade, an enormous impact and their perspective for future developments is also promising [Ebert, 2007; Theunissen et al., 2008]. As each one represents a universe of development with unique characteristics, research in the area has discussed how to accommodate the characteristics of each model for the definition of development processes that are more effective [Boehm and Turner, 2003; Glass, 2001; Glazer et al., 2008; Warsta and Abrahamsson, 2003].

According to the results obtained through a systematic review, several researchers have investigated the possibility of reconciliation among plan-driven, agile, and FOSS models [Magdaleno et al., 2009]. In general, the existing proposals [Fritzsche and Keil, 2007; Kahkonen and Abrahamsson, 2004; Paulk, 2001] involve the comparison and combination of the practices suggested by different models, aiming to produce a new hybrid one. However, the complexity of software development and the variety of existing methods make the task of comparing them, arduous and inaccurate. This kind of software development models combination limits the potential for synergy among them, possibly resulting in an incomplete method, where it is no longer possible to ensure that the resulting process actually has the desired characteristics.

Boehm and Turner [2003] proposal suggests risk analysis of the project characteristics as a way to select the project adequate method. This proposal has similarities with our research work, since it considers project characterization. However, it only focuses on agile and plan-driven, without considering FOSS development.

This work argues that it is necessary more than the combination of practices of different models. The proposed solution involves software development processes tailoring, by balancing the main conflicting aspects in plan-driven, agile and FOSS models – collaboration and discipline.

3. Collaboration and Discipline

Collaboration can be defined as the group working of two or more people to achieve a common goal. The collaboration is an important factor for software organizations to achieve their goals of productivity, quality and knowledge sharing [Magdaleno et al., 2009]. In particular, software development is a complex process that involves the collaboration of several people over a period of time to achieve a common goal [Cugola and Ghezzi, 1998]. Therefore, software development is a typical example of collaborative work [DeMarco and Lister, 1999; Herbsleb et al., 2005].

Moreover, the discipline is related to the planning level adopted in software process definition and the rigidity of control employed in process execution. Thus, discipline imposes order, systematizing the work, avoiding the chaos and successes dependent on individual talents [Boehm and Turner, 2003].

Both are complementary and essential in any project, but in different proportions, depending on the project characteristics [Magdaleno, 2010a]. For a balanced mix between collaboration and discipline, it is necessary to understand how these aspects vary and distinguish the software development models.

Regarding collaboration, we can consider the different levels of formality in communication, coordination, awareness and memory [Magdaleno et al., 2009]. Regarding discipline, software models vary on emphasis and form of their processes. The plan-driven model is characterized by an emphasis on well-defined and continuously improved processes. Both agile and free/open development use no description or explicit modeling of the process adopted. Instead, they deal with a set of general principles to guide the development.

In order to explicit the collaboration and discipline, some instruments can be considered. To understand the existing **collaboration** among **people** in software projects, **social networks** [Barabasi, 2003] appear as a promising path. A social network consists of a finite set of actors and the relationships among them. We can find several works [Gao et al., 2003; Madey et al., 2002] on social network visualization and analysis, which point to social networks potential to explain how the collaboration occurs within a group.

When we explicit collaboration, its visibility increases, so that members of the organization can achieve greater understanding and motivate themselves. Thus, understanding the social networks involved in development projects can help to understand and monitor the level of collaboration in the project.

Discipline can be measured by regulating the level of control of **processes**. The level of discipline is established through a **measurement** approach. The need of measuring results comes from the premise that you can’t control what you can’t measure. Measurement is an important mechanism for visibility into a project and helps to raise awareness about ongoing processes.

For introducing collaboration and discipline into tailored software development processes, it is important to define how to plan and monitor the needed or desired levels of collaboration and discipline. In this sense, some instruments are being considered, such as the collaborative maturity model (CollabMM) [Magdaleno et al., 2009], social networks [Barabasi, 2003], and measurement [McGarry et al., 2001]. In particular, this paper proposes the use of social networks as a mechanism that helps to explicit and measure the existing collaboration among people in software development projects.

4. Social Networks

A social network consists of a finite set of actors and the defined relationships among them [Wasserman and Faust, 1994]. In a social network, nodes represent actors and edges correspond to possible relationships among them. The semantics of the relationship depends on the analysis that will be made in this network.

Adapting the approach proposed in [Cross et al., 2004], we can summarize four steps of a methodology for studying social networks. The first step is to define the purpose of analysis and provide the semantics of nodes and edges of the network. The next step is to collect data to build the social network. This collection can be done through questionnaires or facilitated by data mining in the repositories. Then, these data can be manipulated for viewing or analysis.

Next, in the social networks visualization step, the visual representation of information is adopted to reduce the cognitive overload of the user and to facilitate understanding and exploration of data through graphs. The social networks visualization allows the observation of facts and knowledge extraction from graphs.

Finally, the last step is the analysis of social networks, which uses the concepts of graph theory to describe, understand and explain the interaction and social organization of a group. This analysis seeks to understand the relationships between people, groups or organizations through its properties. These properties [Wasserman and Faust, 1994] were detailed in a previous study [Santos et al., 2010], which also identified those (i.e., degree centrality, betweenness centrality, closeness centrality and network density) with the greatest potential to explain collaboration. In software development, we intend to use the social network analysis to understand the collaboration among development team members.

4.1. Requirements for social networks tools

Several tools to deal with social networks have been proposed. In a previous study, we analyzed 10 of them, 8 academic, one shareware and one open source tool [Magdaleno et al., 2010]. This study showed that the identified tools already provide an extensive set of generic algorithms that can be readily used to calculate social network properties. However, they do not engage in analysis dedicated specifically to the collaboration. In turn, most visualization tools are not actually available or have significant limitations.

Considering the analysis of contributions and limitations of the tools, the observation of existing proposals for analysis of social networks in software development, and the objectives of this research work, we come to the list of requirements that a social networks tool must meet [Magdaleno et al., 2010]. These requirements were separated into three categories: mining (REQM), visualization (REQV) and analysis (REQA). Few examples of the requirements are presented in Table 1.

Table 1. Examples of social network tools requirements

Name	Description
REQM3	The system must be able to mine data from different sources of information for software development projects: a repository of configuration management, source code, discussion forum and e-mail list.
REQV10	The system must provide the visualization of network evolution over time.
REQA11	The system must calculate the properties of social network analysis.

The studied social networks tools were analyzed in accordance with this list [Magdaleno et al., 2010], as partially presented in Table 2. These requirements, although not forming a complete nor necessarily sufficient list, serve as a guide for developing a tool that intends to be comprehensive and, if possible, bring new contributions in relation to the tools currently available in the technical literature.

Table 2 – Tools x requirements

Requirement	Ariadne	Augur	MiSoN	OssNetwork	Pajek	RaisAware	Sargas	SVNNAT	UCINET	Visone
REQM3	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗

Requirement	Ariadne	Augur	MiSoN	OssNetwork	Pajek	RaisAware	Sargas	SVNNAT	UCINET	Visone
REQV10	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗
REQA11	✗	✗	✗	✓	✓	✓	✓	✗	✓	✓

After this analysis, we concluded that none of the analyzed social networks tools met all requirements. Thus, there is still room to propose other tools that support in a more adequate manner the need for collaboration analysis, through the implementation of these requirements. This motivation has led to the creation of a new tool - called EvolTrack-SocialNetwork.

4.2. EvolTrack-SocialNetwork

EvolTrack-SocialNetwork is an extension of EvolTrack tool [Cepeda et al., 2008], which is one of the tools developed by the Software Reuse Group at COPPE/UFRJ. EvolTrack is a software visualization tool that provides a time based approach to observe the emerging design at different moments during the development life cycle. Basically, it periodically extracts project information from a specific data source and then, after performing some pre-processing and transformation, presents the corresponding software design for that project period of time.

EvolTrack was chosen as the starting point for building EvolTrack-SocialNetwork, because it offers an initial infrastructure for data mining, some visualization features and functionality for analyzing metrics, and has been developed by the same research groups in which this work is being developed.

After this choice, a study to examine EvolTrack feasibility for use in real scenarios was planned and conducted, using seven FOSS projects [Cepeda et al., 2010]. These projects were chosen because they publish their development artifacts, including source code, freely over the Internet. Thus, they represent an opportunity for research due to its diversity, complexity, representativeness and ease of access.

As a result, this study showed the feasibility of using EvolTrack and signaled some scalability limitations in relation to the display of models of very large projects [Cepeda et al., 2010]. Currently, this possibility is already under construction as part of another ongoing research project that aims to expand the capabilities of EvolTrack visualization [Silva, 2010].

After this assessment of the feasibility of using EvolTrack, the design of EvolTrack-SocialNetwork architecture has started. The EvolTrack-SocialNetwork architecture is composed by three modules: mining, visualization and analysis. All of these modules are based on a social network meta-model. Besides, since we reused EvolTrack infrastructure, its components are also used.

Currently, EvolTrack-SocialNetwork tool is under development and we expect that it can contribute to provide information that will help the development team to understand, reflect and interfere in the work being done.

5. Conclusion

Despite being one of the main tasks to be executed by the project manager, process tailoring is not simple. It requires pondering many factors and evaluating a large set of constraints. Due to this complexity, the manager usually is not able to evaluate all available combinations and chooses a process in an ad-hoc manner, based on his/her own experience, possibly selecting one that is not the best alternative for the current project.

In order to facilitate process tailoring, it is possible to support the project manager by automating some of the steps to solve the problem, possibly reducing the effort required to execute this activity and improving the quality and adequacy of the obtained process [Magdaleno, 2010b]. This decision support environment can help in the selection of an appropriate process for a software project according to the best balance between collaboration and discipline.

We claim that social networks, obtained as a result of interactions in software development, can provide useful information for understanding the collaboration among development team members. Therefore, this information can be used in process tailoring. For instance, when a project, which would require a high level of collaboration, is analyzed and a highly centralized coordination network is detected, the support decision environment can review its process tailoring in order to treat this problem and enhance the collaboration.

Acknowledgments

This work is partially funded by CNPq (under process nº. 142006/2008-4) and is part of Programa Institutos Nacionais de Ciência e Tecnologia, supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq/MCT) (under contract nº. 557.128/2009-9) and by Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) (under contract nº. E-26/170028/2008).

Referências

Barabasi, A. L. (2003). "Linked: How Everything Is Connected to Everything Else and What It Means for Business, Science, and Everyday Life". Cambridge: Plume.

Beck, K. (1999). "Extreme Programming Explained: Embrace Change". Boston, MA, USA: Addison-Wesley.

Beck, K., Beedle, M., Bennekum, A. V., et al. (2001). "Manifesto for Agile Software Development". http://agilemanifesto.org/.

Boehm, B., and Turner, R. (2003). "Balancing Agility and Discipline: A Guide for the Perplexed". Boston, MA, USA: Addison-Wesley.

Cepeda, R. D. S. V., Magdaleno, A. M., Murta, L. G. P., et al. (2010). "EvolTrack: Improving Design Evolution Awareness in Software Development". Journal of the Brazilian Computer Society (JBCS), (to appear).

Cepeda, R. D. S. V., Murta, L. G. P., and Werner, C. (2008). "Visualizando a Evolução de Software no Desenvolvimento Distribuído". In: Workshop de Desenvolvimento Distribuído de Software (WDDS) - Simpósio Brasileiro de Engenharia de Software (SBES), Campinas, SP, Brasil: SBC, pp. 41-50.

Chrissis, M. B., Konrad, M., and Shrum, S. (2006). "CMMI: Guidelines for Process

Integration and Product Improvement". Boston, MA, USA: Addison-Wesley.

Cockburn, A. (2001). "Agile Software Development". Boston, MA, USA: Addison-Wesley.

Cross, R., Andrew, P., and Cross, R. (2004). "The Hidden Power of Social Networks: Understanding How Work Really Gets Done in Organizations". Boston, Massachussetts: Harvard Business School Press.

Cugola, G., and Ghezzi, C. (1998). "Software processes: A retrospective and a path to the future". Software Process Improvement and Practice (SPIP) Journal, v. 4, pp. 101-123.

DeMarco, T., and Lister, T. (1999). "Peopleware: Productive Projects and Teams". New York, USA: Dorset House.

Ebert, C. (2007). "Open Source Drives Innovation". IEEE Software, v. 24, n. 3, pp. 105-109.

Feller, J., and Fitzgerald, B. (2001). "Understanding Open Source Software Development". Boston, MA, USA: Addison-Wesley.

Fritzsche, M., and Keil, P. (2007). "Agile Methods and CMMI: Compatibility or Conflict?". e-Informatica Software Engineering Journal, v. 1, n. 1, pp. 9-26.

FSF. (2008). "The Free Software Definition". http://www.gnu.org/philosophy/free-sw.html.

Fuggetta, A. (2000). "Software process: a roadmap". Proceedings of the Conference on The Future of Software Engineering, Limerick, Ireland: ACM, pp. 25-34.

Gao, Y., Freeh, V., and Madey, G. (2003). "Analysis and Modeling of Open Source Software Community". In: North American Association for Computational Social and Organization Sciences Conference (NAACSOS), Pittsburgh, PA, USA: Computational Analysis of Social and Organizational Systems (CASOS), pp. 1-4.

Ginsberg, M., and Quinn, L. (1995). "Process Tailoring and the Software Capability Maturity Model", CMU/SEI-94-TR-024, SEI-CMU, http://www.sei.cmu.edu/publications/documents/94.reports/94.tr.024.html.

Glass, R. L. (2001). "Agile Versus Traditional: Make Love, Not War!". Cutter IT Journal, v. 14, n. 12, pp. 12-18.

Glazer, H., Dalton, J., Anderson, D., et al. (2008). "CMMI or Agile: Why Not Embrace Both!", SEI-CMU, http://www.sei.cmu.edu/publications/documents/08.reports/08tn003.html.

Hansson, C., Dittrich, Y., Gustafsson, B., et al. (2006). "How agile are industrial software development practices?". Journal of Systems and Software (JSS), v. 79, n. 9, pp. 1295-1311.

Herbsleb, J. D., Paulish, D. J., and Bass, M. (2005). "Global software development at siemens: experience from nine projects". Proceedings of the 27th international conference on Software engineering, St. Louis, MO, USA: ACM, pp. 524-533.

Kahkonen, T., and Abrahamsson, P. (2004). "Achieving CMMI Level 2 with Enhanced Extreme Programming Approach". Product Focused Software Process

Improvement, Computer Science.Heidelberg: Springer-Verlag, pp. 378-392.

Madey, G., Freeh, V., and Tynan, R. (2002). "The open source software development phenomenon: An analysis based on social network theory". In: Americas Conference on Information Systems (AMCIS), Dallas, TX, USA, pp. 1806-1813.

Magdaleno, A. M. (2010a). "Balancing Collaboration and Discipline in Software Development Processes". Doctoral Symposium of International Conference on Software Engineering (ICSE), Cape Town, South Africa: ACM/IEEE, pp. 331-332.

Magdaleno, A. M. (2010b). "An optimization-based approach to software development process tailoring". PhD Track - International Symposium on Search Based Software Engineering (SSBSE), Benevento, Italy (to appear).

Magdaleno, A. M., Araujo, R. M. D., and Borges, M. R. S. (2009). "A Maturity Model to Promote Collaboration in Business Processes". International Journal of Business Process Integration and Management (IJBPIM), v. 4, n. 2, pp. 111-123.

Magdaleno, A. M., Werner, C. M. L., and Araujo, R. M. D. (2010). "Estudo de Ferramentas de Mineração, Visualização e Análise de Redes Sociais", Relatório Técnico, Rio de Janeiro, RJ, Brasil: PESC-COPPE, http://www.cos.ufrj.br.

Magdaleno, A. M., Werner, C. M. L., and Araujo, R. M. D. (2009). "Revisão Quasi-Sistemática da Literatura: Conciliação de processos de desenvolvimento de software", Relatório Técnico, ES-730, Rio de Janeiro, Brasil: PESC-COPPE, http://www.cos.ufrj.br.

McGarry, J., Card, D., Jones, C., et al. (2001). "Practical Software Measurement: Objective Information for Decision Makers". Addison-Wesley Professional.

Patel, C., Lycett, M., Macredie, R., et al. (2006). "Perceptions of Agility and Collaboration in Software Development Practice". Hawaii International Conference on System Sciences (HICSS), Kauai, Hawaii, USA, pp. 1-7.

Paulk, M. (2001). "Extreme programming from a CMM perspective". Software, IEEE, v. 18, n. 6, pp. 19-26.

Pedreira, O., Piattini, M., Luaces, M. R., et al. (2007). "A systematic review of software process tailoring". SIGSOFT Software Engineering Notes, v. 32, n. 3, pp. 1-6.

Raymond, E. S. (2001). "The Cathedral & the Bazaar". O'Reilly Media.

Santos, T. A. L., Araujo, R. M. D., and Magdaleno, A. M. (2010). "Identifying Collaboration Patterns in Software Development Social Networks". Infocomp - Journal of Computer Science (to appear).

Schwaber, K. (2004). "Agile Project Management with Scrum". Washington, DC, USA: Microsoft Press.

Silva, M. A. (2010). "IAVEMS : Infraestrutura de Apoio à Visualização na Evolução de Métricas de Software". Projeto Final, Rio de Janeiro, RJ, Brasil (em andamento): UFRJ/IM.

Theunissen, M., Kourie, D., and Boake, A. (2008). "Corporate-, Agile- and Open Source Software Development: A Witch's Brew or An Elixir of Life?".

Balancing Agility and Formalism in Software Engineering: Second IFIP TC 2 Central and East European Conference on Software Engineering Techniques (CEE-SET), Springer-Verlag, pp. 84-95.

Warsta, J., and Abrahamsson, P. (2003). "Is Open Source Software Development Essentially an Agile Method?". Proceedings of the Workshop on Open Source Software Development, Portland, OR, USA, pp. 143-147.

Wasserman, S., and Faust, K. (1994). "Social Network Analysis: Methods and Applications". Cambridge, United Kingdom: Cambridge University Press.