Software Development Roles: A Multi-Project Empirical Investigation

Murat Yilmaz
Department of Computer Engineering,
Çankaya University, Turkey
e-mail: myilmaz@cankaya.edu.tr

Rory V. O'Connor
School of Computing
Dublin City University, Ireland
e-mail: roconnor@computing.dcu.ie

Paul Clarke
Regulated Software Research Center
Dundalk Institute of Technology, Ireland
e-mail: paul.clarke@dkit.ie

ABSTRACT

In our earlier research work, we developed a conceptual framework that identifies the different types of roles that can exist in contemporary software development projects (including both agile and traditional software development approaches). The purpose of the framework is to assist software project managers when tuning software development project roles to the demands of individual projects, with our previous research indicating that there is a need to tailor software development roles to individual projects. In this work, we extend the earlier research through the use of a series of semi-structured interviews within seven Turkish software companies. The results, which are consistent with previous findings, offer further evidence of the usefulness of the framework, while also identifying possible areas for future work in this space.

Categories and Subject Descriptors

D.2.9 [Software Engineering]: Management : Productivity, Software Process, Software Teams

General Terms

Software Management, Human Factors

Keywords

Software Process Improvement, Team Roles, Roles Wheel, Social Aspects of Development

1. INTRODUCTION

Software development is a task-oriented social activity. It can be considered as a collective team effort where teamwork requires cooperation and therefore social interactions. Such interactions should be defined in terms of roles, teams and resources. Therefore, the relations between the teams and resources are usually allocated and controlled by the roles that are defined in a software development methodology. Although there are common roles for software development activities, several distinctive different roles have been emerged within the software have emerged.

In contemporary software development, assigning the appropriate roles to software practitioners has become a central issue for improving the quality of software processes [7]. Research has shown that software development teams have an important influence on organizational performance [8]. Consequently, researchers have shown an increased interest in the methods for tailoring practitioners' roles, especially while selecting a software methodology for a specific development project [1].

In our previous work [11], we comprehensively reviewed the roles in the software development processes that were selected by industry experts based on their impact on software development productivity. To help software practitioners tailor the roles in their software development organizations, we constructed a role-based conceptual framework. Preliminary evidence suggests that the proposed framework is useful for software development organizations that are planing to customize their development processes. Based on the initial feedback, the present study assesses the Roles Wheel (see Figure 1). To this end, a survey was conducted with software practitioners and its results were evaluated. Furthermore, we revisited the framework, refining it based on study findings.

The remainder of this paper is divided as follows: In section two, we briefly introduce the related work regarding roles in software development research. The following section explains the research process. The fourth section presents the findings of the research, focusing on the roles that are used in a group of software development organizations. Finally, the conclusion gives a brief summary and critique of the results. It further suggests future directions.

2. RELATED WORK

Although assigning the appropriate roles to software practitioners is a primary concern in the field of software development, only a few studies have focused on the management of roles in the software engineering process. Acuna et al. [1] built a role assignment technique regarding the capabilities of individuals involved in software development. The goal was to match the practitioners' capabilities with the necessary requirements to perform that role.

To enhance the involvement of software team members, Dubinsky and Hazzan [6] developed a role scheme (based on 12 different roles that are grouped into 4 categories) where each team member has a distinctive role to support the management of the development process. Downey [5] proposed an artifact-oriented approach for building improved job descriptions for teams based on roles in software development. To bridge the skill gaps in software teams, it has been suggested that meaningful job specifications for individuals could be defined based on the planned software artifacts. Andre et al. [2] developed a tool to support role assignments for suitable human assets by using multidimensional aspects (i.e. psychometric-test and data-mining tools) to assign practitioners to project-based roles and as well as for the team

Figure 1: A Summary of Roles Contained in the Different Approaches [10]

formations. Borges et al. [3] simplified Rational Unified Process (RUP) roles for the needs of small software development teams (with 13 basic roles) by using a set of mapping rules between RUP and roles for SMEs (small and medium sized companies).

We illustrated the first form of our role-based schema, which can be considered useful for tailoring practitioners' roles to the software development project needs [11]. In other words, we utilize such a wheel to customize a set of suitable roles for a software development process (see Figure 1).

Through the development and evaluation of the Roles Wheel, we demonstrated; the roles in earlier software development approaches have sometimes been re-branded for use in more recent software development methodologies with little change. Perhaps more importantly, we found that the scope of individual roles can vary from project to project and company to company, even where the role (or title associated with the role) is the same. A further important finding from earlier work suggested that the social structure of organisations and teams should be taken into account when tailoring software development roles [10].

3. THE RESEARCH PROCESS

To further investigate the usability of the Roles Wheel (see Figure 1), we conducted a series of semi-structured interviews with practitioners with titles including but not limited to: software developer, technical lead, database administrator, software tester, business and system analyst, quality manager, and user interface designer from seven software organizations with various sizes. The objective was to explore the usability of the roles summary chart based on the opinions of different software practitioners. The survey used in our approach has both multiple-choice and open-ended questions. In the first part, we asked questions such as (1) *How many employees work in your organization/company?* (2) *How many people work in your project team?* (3) *What is the sector/the general area of implementation for the software product(s) being developed in your project?* (4) *What is your role/title in your project team?*

In the second part, we showed the Roles Wheel to the participants and requested them to answer questions based on the wheel. This section of the questionnaire required respondents to give their opinion on roles shown in the wheel. The questions were *(5) Which approach does organization/company use for your organization/company? (6) Which is the closest category of the chart to the role scheme in your project?* Finally, a set of open-ended

questions were asked to explore the usability of the Roles Wheel. The questions were *(7) Comparing your project roles with the roles in the chart category of your answer to Q5/Q6, what are the additional roles in your project? (8) Comparing your project roles with the roles in the chart category of your answer to Q5/Q6, what are the missing roles in your project? (9) Comparing your project roles with the roles in the chart category of your answer to Q5/Q6, what are the overlapping roles (responsibilities of more than one role on same person) in your project? (10) With the help of the information on this chart and due to your organizational conditions, which roles do you think can be added to your project to improve the development process?*

4. STUDY FINDINGS

We conducted twelve semi-structured interviews with a variety of software practitioners from Turkish software industry. The interviews took an average of 45 minutes to complete, and were conducted by the first author. As shown in Figure 2 the majority of participating companies operate as public institutions or within the defense industry. Additionally, other participants were from the domains of energy distributions, IT and information services, and Internet businesses.
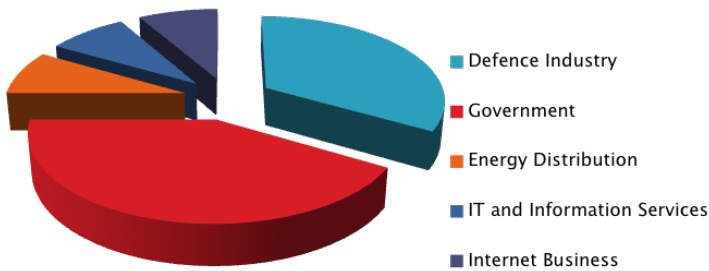
Figure 2: Domain of Participants

The roles participants identified from the semi-structural interviews are shown in Figure 3. It can be seen from the Figure 3 that participants were developers, which was followed by the quality assurance and configuration managers. The other roles of the participants were software testers, architects, team leaders.
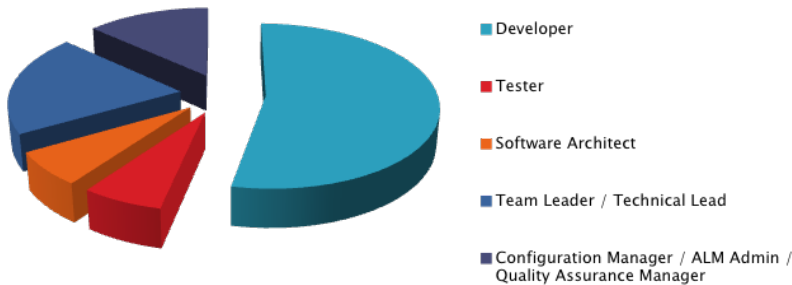
Figure 3: Participant Role Profile

Interview quotation:"The [government-based] projects

we are working on should meet with the entire standards that were defined by the clients, and therefore the roles in the project are automatically assigned to practitioners based on the software requirements. However, an issue is usually assigned to a project lead role. The project lead then assigns the issue to an appropriate employee. Unfortunately, this can be problematic sometimes. I believe a chart stating the roles with the participants name might be helpful for the project lead in the process. "

A small number of those interviewed suggested that more roles should be added to the Roles Wheel. As one interviewee said:

Interview quotation:"There is more than one answer again. To make clear here are the details. Project has: - pre-defined schedule for the delivery of the end product - pre-defined milestones (similar to the waterfall e.g. PDR, CDR, TRR etc.) - more than 1500 requirements - more than 80 team members - AQAP 2110, ISO 9001 and IEEE 12207 imposed standards integrated with CMMI - DoD templates, Guidelines, checklists, scripts, review cycles, documentation, reporting etc. - System Engineering activities (process planning and defining, Requirement analysis, Use Case, Feature and HMI derivation) - Design (SwDD, class, sequence and activity diagrams in addition to system level design, design decisions mapped to requirements) - Development (OO programming, started with Scrum Agile then changed as incremental [with mini waterfall cycles], unit integration tests, gated check-ins, nightly builds) - Test (component, component integration, dryruns, system, acceptance etc.) - Deployment (automated installation) - Maintenance "

The majority of the participants supported the concept of using role tailoring for software projects, reporting that there were benefits to incorporating this technique - even though the issues related to tailoring a process based on the roles were not particularly prominent in the interview data. Additionally, one interviewee argued that chart had some missing roles (e.g. see quotation below) that can be found in a large-scale defense project. Talking about this issue the interviewee said:

Interview quotation:"Here is the general list which includes the different combinations of the roles in the aforementioned chart; Project Manager (with PMO, Program Manager, Contract Manager, Delivery Manager), Quality Assurance Manager, Quality Assurance Engineer (Auditor), Configuration Manager, Configuration Management Specialist, Engineering CM, Technical Lead, Integration and Test Manager, Integration Manager, Test Manager, Design Manager, Development Manager, Training Manager, Roll-out and Sustainment Manager, Technical Architect , Component Lead(s) (Scrum Master), System Engineer, Software Engineer (Developer, Integrator), Technical Writer, Subject Matter Expert(s), HMI Coordinator, System Admin, Database Admin, etc."

After carefully examining the chart, one interviewee said:

Interview quotation: *"We have some other roles that are specific to our project. For example, ALM (Application Lifecycle Management) Admin, Analysis Specialist, Field Experts, Team Leader, etc. I think our project requires more testers, and also the number of analyst should be increased. The chart also reminded me of our problem that the role of software architect should be separated. [In addition], the roles SM and STM are overlapping. SM is a member of STM in the project."*

And another interviewee said:

Interview quotation: *"Scrum is very popular in Turkish software industry at the moment. A majority of software companies are interested in adopting this methodology into their working project. However, only a limited number of individuals in projects have hands-on experience who really understand the philosophy behind it. Not surprisingly perhaps, the roles selected based on limited manuscripts, usually not from the industrial perspective."*

Furthermore, there was a sense of agreement on the benefits of the roles chart amongst interviewees. One interviewee claimed that there are overlapping roles in small teams, e.g. programmers and analysts are also doing testing. Furthermore, it was suggested that "the roles with some overlapping characteristics such as database designer and software architect should be separated to improve the tailoring process."

Interview quotation: *"I think the chart would add value to the role selection process of a software development organization; it seems like an innovative approach that facilitates the roles management in a workplace. However, it should likely to be improved by removing overlapping roles and further it should extended by having more roles."*

Figure 4 provides an overview of the methodologies used in the domains where the qualitative data was obtained. It is apparent from this figure that many projects in the defense industry prefer plan-driven approaches. According to participants, a reason for this is that most of the requirements were fixed early, and documented based on international standards.

Interview quotation: *"Many large scale government projects are now moving from traditional to agile approaches. This means that the managers in public institutions are starting to realize that agile team can work very effectively without the role of a single leader. In agile software teams, information and role sharing is vitally important otherwise teams cannot be autonomous. Therefore, roles should certainly be tailored to meet a team's needs."*
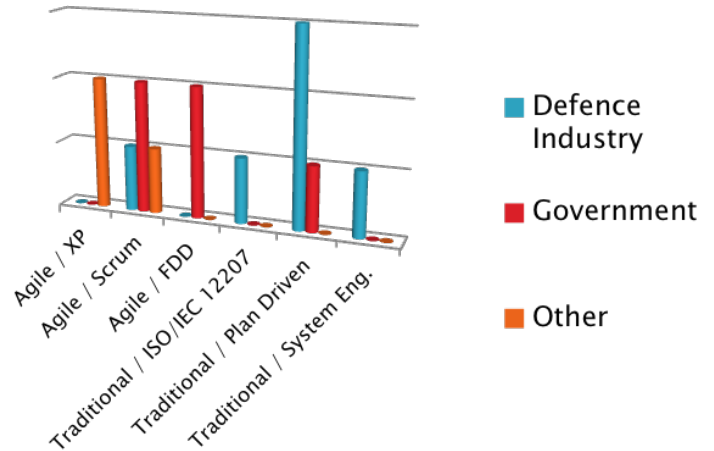

Figure 4: Software Development Methodologies Versus Industrial Domains

Taken together, these findings suggest that there is a need to take a holistic view of information regarding the roles of different software methodologies. In fact, many practitioners prefer to process the roles in different methodologies to tailor their software processes more efficiently.

5. CONCLUSIONS AND FUTURE WORK

The study material presented in this paper reports on the application of the Roles Wheel, and in particular the tailoring of roles to individual software projects. An evaluation of the feedback elicited from seven Turkish software development companies suggests that traditional roles are more heavily utilized in the defense sector, while agile roles were more preferred in other identified domains, such as internet-related software development. We further find that many agile teams report using Scrum, with a preference for its associated roles. However, in general, we have also found that larger software development organizations tend to have even greater numbers of roles than those identified in the Roles Wheel, with smaller companies tending to merge roles in a manner best suited to their operating constraints. Therefore, the Roles Wheel would appear to be most applicable to medium-sized organizations. Nonetheless, the findings from the study indicate that the Roles Wheel in its present incarnation is helpful when selecting the most appropriate roles for software development projects. Future work should seek to elaborate the role-based schema, to better address the needs of the smaller and larger companies. In this respect, a more thorough investigation of the material on software development from both literature and industry could be required. And an associated analysis could adopt a Delphi approach [9] to compare the opinions that exist regarding roles so as to develop a consensus among software professionals.

Perhaps the most interesting and lasting finding of the study reported upon herein is the confirmation from industrial engagement of the gap between theory and practice in our field of software engineering. Theory is clear and prescriptive for most of the widely accepted software development approaches - be they contemporary agile development methods or more traditional software development frameworks. Practice on the other hand is less clear and much less predictable. Just as the methods for software development really only provide the scaffolding to enable a

generic release, so too do the prescribed roles provide just a useful working reference to be moulded and shaped to the needs of individual projects and the groups of people that comprise the software development team. There is in effect no boilerplate software development methodology or related roles to fit all software development projects and environments.

In this study, we have seen that the practice of identifying the scope and remit of roles in software development is complex and depends on the mode and culture of the people implementing the software solution. This finding is congruent with some of our earlier work in which the importance of individual software development contexts has also been found to be a complex process (perhaps more complex than we are readily aware of or willing to accept) [4]. So there is a gap between the theory and the practice, an important gap that is bridged through the application of the all of the fuzziness of human thinking.

This raises an at once interesting question: how far down the road of harnessing human interaction and fuzzy thinking do we in the software engineering field wish to go? Such an investigation could be something of a rabbit hole, twisting and turning in unforeseeable ways. Such exploration would be taking us down the road of incorporating some wisdom from the social sciences and business worlds, where issues related to human potential and interaction have long since been a subject of concern. We could as a field of inquiry decide to simply limit ourselves to the domain of generalized roles and processes, and choose to rely on the ability of individuals to resolve the harmonization of theory to individual projects. However, given the intensive usage of human capital in software development [12], it would appear to be at least worth taking some cursory steps towards assessing the rabbit hole - and that could be an interesting journey for us all.

6. REFERENCES

[1] S. T. Acuña and N. Juristo. Assigning people to roles in software projects. *Software: Practice and Experience*, 34(7):675–696, 2004.

[2] M. André, M. G. Baldoquín, and S. T. Acuña. Formal model for assigning human resources to teams in software projects. *Information and Software Technology*, 53(3):259–275, 2011.

[3] P. Borges, P. Monteiro, and R. J. Machado. Mapping rup roles to small software development teams. In *Software Quality. Process Automation in Software Development*, pages 59–70. Springer, 2012.

[4] P. Clarke and R. V. O'Connor. The situational factors that affect the software development process: Towards a comprehensive reference framework. *Information and Software Technology*, 54(5):433–447, 2012.

[5] J. Downey. An artifact-centric method for creating software job descriptions. In *Proceedings of the 2008 ACM SIGMIS CPR conference on Computer personnel doctoral consortium and research*, pages 12–21. ACM, 2008.

[6] Y. Dubinsky and O. Hazzan. Using a role scheme to derive software project metrics. *Journal of Systems Architecture*, 52(11):693–699, 2006.

[7] N. Pal and D. Pantaleo. *The agile enterprise: Reinventing your organization for success in an on-demand world*. Springer, 2005.

[8] E. Salas, S. Tannenbaum, D. Cohen, and G. Latham. *Developing and Enhancing Teamwork in Organizations: Evidence-based Best Practices and Guidelines*, volume 33. John Wiley & Sons, 2013.

[9] R. Thorpe and R. Holt. *The Sage dictionary of qualitative management research*. Sage, 2007.

[10] M. Yilmaz. *A software process engineering approach to understanding software productivity and team personality characteristics: an empirical investigation*. PhD thesis, Dublin City University, 2013.

[11] M. Yilmaz, R. O'Connor, and P. Clarke. A systematic approach to the comparison of roles in the software development processes. In *Software Process Improvement and Capability Determination*, volume 290, pages 198–209. Springer-Verlag, 2012.

[12] M. Yilmaz and R. V. O'Connor. Social capital as a determinant factor of software development productivity: An empirical study using structural equation modeling. *International Journal of Human Capital and Information Technology Professionals (IJHCITP)*, 3(2):40–62, 2012.