Social Team Characteristics and Architectural Decisions: a Goal-oriented Approach

Johannes Meißner[1] and Frederik Schulz[2]

[1] Research and Development, SK8DLX Services GmbH, Jena, Germany,
johannes.meissner@sk8dlx.de

[2] Research and Development, SK8DLX Services GmbH, Jena, Germany,
frederik.schulz@sk8dlx.de

Abstract. Organizational and social factors like the geographical distribution of team members or their skill levels can have a significant impact on software architectural decisions. In our previous work, we have developed a framework based on the Goal Requirements Language (GRL) to model the interdependencies between project contributors, their realization efforts and the intended project deliverables. Based on our daily work in industrial e-commerce projects, further experiences have shown that the composition of a software development team in terms of social and psychological characteristics of the team members is just as important. Structure-analytical methods like the well established team role model of Belbin are a suitable mean to capture these personal qualities in a systematic way. The obtained information is incorporated into our framework to enable the documentation, the analysis and the discussion of the coherence between social factors and the work-breakdown structure in software development projects.

Keywords: GRL, Organizational context, Social team roles

1 Introduction

Software architectures are interacting with the organizational context of their development. For example, Conways law [6] states that "*organizations which design systems* [...] *are constrained to produce designs which are copies of the communication structures of these organizations*".

In our previous work [13] we introduced the GRL-based framework *Organizational Context Analysis* (OrCA), that captures organizational factors and their interdependencies with the software architecture. The foundation of the framework are three-layered models of the *work-breakdown structure*: Which *contributor* participates in which *realization* effort to achieve which project *deliverable*? Thus OrCA provides a template for the usage of GRL elements to model the relation between a software development organization on the one hand and the intended project outcomes on the other hand. Section 3.2 gives an introduction to the main concepts of the OrCA framework.

Goguen [8] notes, that "*a computer-based system is built for people and by people*". The human and social factors have a strong impact on the resulting system [9]. Our own experiences in ongoing e-commerce projects verified these observations. The software development process in our company features two to four small developer teams. Depending on the current workload, these teams can be assembled in a flexible manner. The composition of the teams and the personal relationships among the team members became focal points concerning the overall team performance. For example, the quality of the communication between team members had a significant impact on the software quality and led either to high efficiency or to exceeded time and cost budgets. Likewise, a team consisting of technically high experienced developers tended to perform better if team members with high communication skills and perseverance were added. We enhanced the OrCA framework by incorporating information about these social factors to consider them in the early project planning phase. This was achieved by employing the well-established *team role* model of *Belbin* [2]. A brief introduction to the concept of team roles is provided in section 3.1.

2 Objectives of the research

The overall goal of our research is the consideration of organizational and social influence factors in architectural decisions. Furthermore, we want to increase the awareness of this issue, as the assignment of tasks to individuals and their personal characteristics and relationships can decide between success and failure of a project. Therefore, we want to achieve the following objectives:

1. The *assignment of individuals to project teams* and its effects on particular project goals have to be incorporated into our framework.
2. The framework has to capture the *assignment of team roles to these individuals.* To this end, the principles of established social role theories should be utilized.
3. This work must provide a *foundation for the further analysis of the team composition* to offer guidance for optimizing the team‘s structure.

3 Scientific Contributions

3.1 Team roles

The assignment of an employee to a project is commonly determined by its *functional roles*: Where is the employee located in the companies hierarchy? What are the technological skills and the level of experience of the employee? In contrast, *non-functional roles* are often neglected. They include personal characteristics like reliability or conscientiousness of the individual employee. However, the consideration of these attributes during a team's forming phase can result in significant benefits [5]. The concept of *team roles* offers an abstract view on this issue. Team roles can be defined as "*a pattern of behavior characteristics of the way in which one team member interacts with another where his performance serves to facilitate the progress of a team as a whole*" [2].

Structure-analytical methods [7, 14, 11] provide the possibility to extract these personal characteristics in a systematic way. A well-established method that is widely distributed in the industry is Belbin's team role model [2]. It distinguishes between nine different roles, that can be assigned to an individual team member. For example the team role *Shaper* characterizes individuals that are challenging and that work best under pressure while a *Finisher* is rather anxious and acts like a perfectionist. These roles enable the documentation and estimation of the quality of a team's composition. Belbin states that a great diversity of team roles leads to a better performance: A team consisting only of highly qualified management staff tends to produce substandard results. In the following, we introduce an approach for the integration of these principles into our GRL-based OrCA framework.

3.2 The OrCA framework

Basically, models in the OrCA framework consist of three layers that capture the *work-breakdown structure* [1]. The bottom *deliverables*-layer contains the achieved project results, e. g. architectural components or software libraries. Each deliverable is depicted by a resource element. To realize these deliverables, tasks like the implementation or the testing have to be performed. These tasks are contained in the intermediate *realizations*-layer. The assignment of realization tasks to certain deliverables is achieved by using dependency links. Finally, the responsible project members are represented in the top *contributors*-layer. A contributor can be an entire development team or a single developer. The assignment of contributors to realizations is done by dependency links, as well.

To enrich a basic-model by additional organizational and technical factors, the OrCA framework introduces *predicates*. These enable the definition of statements about particular elements in the basic-model (the *subjects*) by linking them to other model elements (the *objects*). The particular sort of this relationship is defined by a predicate's type. Predicates can be used to express a contributor's skills, a technological choice for a deliverable or even cost and budget constraints. In the graphical notation, predicates are depicted by triangles that are labeled with the predicate type. The example in Fig. 1 shows only one deliverable `server`. For its realization, the tasks `implement` and `test` have to be performed. The project contributor `team 1` is responsible for the task `implement`, whereas `team 2` is assigned to `test`, respectively. A predicate of the type `is built with` indicates that the deliverable `server` (subject) is based on the `PHP` technology (object). See [13] for detailed examples including multiple contributors and deliverables.

The ternary relationship subject-predicate-object allows for the modeling of the particular effects a statement has on specific goals. Therefore, the predicate is used as a representative of the overall statement and linked to the affected goal by a contribution link. For further details see [13]. In the following, this mechanism is used to refine the work-breakdown structure by assigning individuals as members to the teams in the contribution layer.

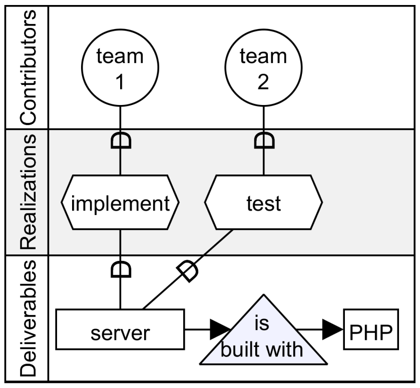


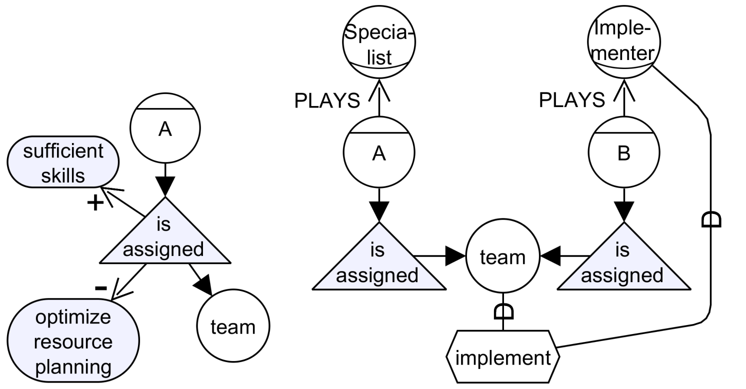
Fig. 1. Simple example for an OrCA basic-model: Contributor `team 1` is responsible for the realization task `implement` to achieve the project deliverable `server` that is based on the technology `PHP`. Likewise, `team 2` has to perform the `test` of the deliverable.

3.3 Extensions to OrCA

Earlier versions of the GRL in [10] adopted the *agents* concept of i*. The meta-model in [3] includes this concept, as well. We utilized these agents to represent individuals. To model their membership in a particular team, we introduced the new predicate `is assigned`. The example in Fig. 2(a) shows an individual `A` assigned to the contributor `team`. This assignment's negative effect on the organizational goal `optimized resource planning` is indicated by a contribution link between the predicate and the goal. A second contribution link reveals a positive effect on `sufficient skills`, which is also an organizational requirement. Thus, we modeled the team assignment *and* its effect concerning the functional role of the individual. This is an advantage over the conventional binary `part`-relationship as defined in the GRL.

Belbin suggests a *questionnaire* and an *observer assessment* to identify the particular team role of an individual. The next step is the incorporation of these roles into our models by utilizing the i* *role* concept. To represent the resulting role assignment, we made use of the *plays*-relationship. Furthermore, a working task can *require* a specific role, as well. For example, the aforementioned *Finisher* is especially qualified for testing, whereas an *Implementer* will usually put a requirement into practice quickly and efficiently. Such a requirement for a role can be modeled by a conventional dependency link between the concerned realization task and the role element. The example in Fig. 2(b) shows a contributor `team` and its two assigned members `A` and `B` with `B` playing the team role `Implementer`. This role is required by the realization effort `implement`, that the team of `B` is responsible for. Thus, `B` is a good choice for the solution of this task.

The work in [2, 4, 12] provides collections of experience and references concerning role combinations and their effect on the team performance. For example, the aforementioned diversity of roles is a prerequisite for a high-quality cooperation. However, this is an overall objective that is not part of the model. Rather, the



(a) Team assignment (b) Team roles

Fig. 2. Example **2(a)**: Agent `A` is assigned to contributor `team` by an `is assigned`-predicate. This assignment has a positive influence on the organizational goal `sufficient skills` and a negative influence on `optimized resource planning`. Example **2(b)**: Contributor `team` has two members `A` and `B`. `A` plays the role `Specialist` and `B` is an `Implementer`. The latter is required by the `implement` task.

team quality has to be examined a posteriori in a comprehensive model analysis. Our concepts for team and role assignment in Fig. 2 enable such an analysis of the team quality according to the theoretical principals of *non-functional roles* as postulated by Belbin. For example, the missing of a team member playing the role *Implementer* can be detected as possible deficiency of the team composition. Furthermore, the models capture information about *functional* roles regarding skills or hierarchy positions and their influence on specific goals. A comprehensive analysis can combine both sides to provide a global examination of the organizational structure.

4 Conclusions

This work provides a foundation for the discussion of the work-breakdown structure in terms of team assignments and social team roles. We attained our goals as defined in section 2:

1. The OrCA predicate concept is utilized to assign individuals to their respective teams. Additional contribution links are used to indicate the impact of this assignment on particular goals.
2. Based on plays-relationships, the individuals are matched to team roles according to the well-established team-role model of Belbin. Furthermore, dependency-links can express the required team roles for a particular task.
3. These contributions are the basis for a further analysis of the model as stipulated by the principles of Belbin: Missing team roles or a mismatch between provided and required team roles reveal possible shortcomings of the team composition.

Thus, we can document the influence of organizational and social factors and the effect of personal characteristics on a project set-up. In an industrial environment this information can be used to communicate and discuss personnel decisions in the context of a software architecture.

5 Ongoing and Future Work

Basing on the concepts provided in this work, we are going to concentrate on the comprehensive analysis of the models. We want to translate Belbin's principles into a set of logical rules, to enable a structured and automated validation. This can be used to provide the user a quick and early feedback on the quality of a model. Additionally, we are going to incorporate other team role theories like Parker [11] or Davis [7] and to formalize their principles about team compositions, as well.

References

1. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Series in Software Engineering, Addison-Wesley, Boston, 2nd edn. (2003)
2. Belbin, R.M.: Management Teams: Why they succeed or fail. Routledge, 3rd edn. (2010)
3. Cares, C., Franch, X., Mayol, E., Quer, C.: A Reference Model for i*. In: Yu, E.S., Giorgini, P., Maiden, N., Mylopoulos, J. (eds.) Social Modeling for Requirements Engineering, pp. 573–606. The MIT Press (2011)
4. Cockburn, A.: The Interaction of Social Issues and Software Architecture. Communications of the ACM 39(10), 40–46 (1996)
5. Constantine, L.L.: Constantine on Peopleware. Yourdon Press Computing Series, Yourdon Press, Englewood Cliffs (1995)
6. Conway, M.E.: How Do Committees Invent? Datamation 14(4), 28–31 (1968)
7. Davis, J., Millburn, P., Murphy, T., Woodhouse, M.: Successful team building: How to create teams that really work. Kogan Page, London (1992)
8. Goguen, J.A.: Social Issues in Requirements Engineering. In: Proceedings of IEEE International Symposium on Requirements Engineering, RE 1993, San Diego, USA, January 4-6, 1993, pp. 194–195. RE'93, IEEE (1993)
9. John, M., Frank, M., Bjørnar, T.: Human and Social Factors of Software Engineering – Workshop Summary. ACM SIGSOFT Software Engineering Notes 30(4), 1–6 (2005)
10. Liu, L., Yu, E.S.: Designing Information Systems in Social Context: A Goal and Scenario Modelling Approach. Information Systems 29(2), 187–203 (2004)
11. Parker, G.M.: Team players and Teamwork: Working with personalities to develop effective teams. Jossey-Bass and John Wiley, San Francisco (2008)
12. Pisani, M.: The Impact of Team Composition and Interpersonal Communication on Perceived Team Performance – A Case Study. European Journal of Social Sciences 35(3), 411–430 (2012)
13. Schulz, F., Meissner, J., Rossak, W.: Tracing the interdependencies between architecture and organization in goal-oriented extensible models. In Proceedings of the Third Eastern European Regional Conference on the Engineering of Computer Based Systems pp. 25–32 (2013)
14. Spencer, J., Pruss, A.: Managing Your Team: How to Organise People for Maximum Results. Piatkus Bks., London (1992)