Position Papers of the Federated Conference on Computer Science and Information Systems pp. 291–298 DOI: 10.15439/2016F362 ACSIS, Vol. 9. ISSN 2300-5963

Competencies outside Agile Teams' Borders: The Extended Scrum Team

Gerard Wagenaar
Avans University of Applied Science, Academy for Engineering & ICT, Lovensdijkstraat 61, 4818 AJ Breda, the Netherlands
Email: g.wagenaar@avans.nl

Sietse Overbeek
Utrecht University, Faculty of Science, Department of Information and Computing Sciences, Princetonplein 5, 3584 CC Utrecht, the Netherlands
Email: s.j.overbeek@uu.nl

Remko Helms
Open University, Faculty of Management, Science and Technology, Valkenburgerweg 177, 6419 AT Heerlen, the Netherlands
Email: remko.helms@ou.nl

***Abstract*—According to the Scrum process framework a Scrum team should have all necessary competencies to accomplish its work. Fragmented and anecdotal evidence hints at Scrum teams still needing additional, external competencies. To contribute to theories on Scrum team composition and practitioner's concerns in staffing a Scrum team we investigated Scrum teams' cross-functionality: To whom do Scrum teams turn for additional competencies, which competencies are involved and how are Scrum teams aware of additional competencies they need? To this extent we analysed the communication in three Scrum teams during one of their Sprints. Our results show that additional competencies are called for, not only on an ad hoc basis, but also on a structural basis. To include those structural competencies the notion of an extended Scrum team is introduced.**

I. Introduction

THE application of an agile software development method, for instance XP or Scrum, is nowadays common in delivering state-of-the-art software [1-2]. Team members with high competence and expertise are a critical success factor in agile software development, especially with regard to timeliness and cost [3].

One of the guidelines accompanying Scrum as a framework for developing and sustaining complex (software) products states on team composition: "*Cross-functional teams have all competencies needed to accomplish the work without depending on others not part of the team*" [4, *p.4*]. Such a cross-functional team should, among others, have skills with regard to software analysis, design and coding [5].

In general, although often implicit, membership of a team is considered to be full time during a Scrum project, certainly for members of the Development Team [6-7]. Emergent team members may support a team during software development [8] or even stronger it is often not clear which members belong to a team and which do not [9] or: "*team boundaries are often permeable*" [10, *p.157*]. Part time membership of a Scrum team could therefore be an option to consider [5], [11].

Fragmented and anecdotal evidence already hints at Scrum teams still needing additional, external competencies [12-14], but this evidence was gathered as a by-product of more general research on communication in agile projects. Unlike this research we only focus on composition of Scrum teams, especially their cross-functionality. We determine its boundaries and look for additional competencies not included in the team. To this extent only we analyse the communication within and outside team boundaries of Scrum teams' members to identify which additional competencies they require and on which basis. In this way our research on the one hand contributes to a better understanding of team composition in Scrum software development, and especially in the competencies included in the team, and those outside its boundaries, and on the other hand allows practitioners to mirror their way of working, and support them in the formation of a Scrum team.

The remainder of this paper is organised as follows. In section 2 we outline the theoretical background relating to our work, based on a literature review. In section 3 we present our research method. The results for three case studies are presented in section 4, followed by a discussion in section 5. Section 6 is the final section in which our conclusions are presented, with their limitations and future work.

II. Theoretical background

In terms of socio-technical congruence [15] the Scrum process framework [4] describes the coordination requirements established by the dependencies among tasks. Upon inspection of the actual coordination activities a match between coordination requirements and activities may be established; such a match has proven to be beneficial to several aspects of software development, for instance reducing the resolution time of modification requests [15] or software build success [16].

We apply socio-technical congruence specifically to the use of competencies in a Scrum team. The coordination requirements defined by the Scrum process framework state cross-functionality. Should a perfect match with the actual activities exist, i.e. they can be carried out without external competencies, then a Scrum team is indeed cross-functional.

Competencies for agile software development have been divided into three major categories, forming a pyramid of agile competencies with engineering practices at the bottom via management practices to agile values at the top [17] or, similarly, technical skills, people or soft skills, and attitudes [18]. Support for the two latter categories is a responsibility of a Scrum master, although an agile coach, operating outside a team's boundaries, is also often involved [19], [20]. Since these are individual rather than team competencies, we use socio-technical congruence to focus on the former category: Engineering skills.

There are already indications that Scrum teams are not entirely cross-functional in this respect. No matter how tightly-knit agile teams are, they need to interact with roles outside the team (e.g. user experience designers, database administrators, system testers), because in practice it is infeasible for all individuals with relevant expertise to be part of the team [14]. This is, for instance, confirmed in the communication between the disciplines of agile development and user experience design [21].

Although internally distributing expertise in agile teams ultimately leads to successful cross-functional teams [22], this appears to be an ideal situation. At least until then, but perhaps even on a more continuous basis, it is an unrealistic goal to aspire to include all competencies in a Scrum development team itself, with its 3 to 9 members. Additional roles, contributing competencies to the Scrum team, have already been hinted at; they could be user experience designers, database administrators, system testers [14], acceptance testers, user interaction designer, or technical writers [13]. Part time members could also be system or database administrators [5].

Four ways to locate expertise in a Scrum team itself have been revealed:

1. Communicating frequently,
2. Working closely together,
3. Declaring self-identified expertise in order to let others know what a member can contribute to the team,
4. Using an expertise directory [23].

It could be the case that these also apply to locating expertise outside a team's border, but this has not been established yet.

Coordinating expertise outside agile teams benefits from five factors:

1. Availability refers to the ability of external specialists to be present in agile teams when their expertise is needed,
2. Agile mind-set concerns dealing with external specialists who might be unfamiliar with agile methods and thus introducing problems for the team, for instance external specialists not being able to align work with the sprints,
3. Stability refers to keeping agile teams stable with a low rate of team members and external specialists turnover,
4. Knowledge retention involves capturing external specialists' knowledge and preserving the knowledge in agile teams,
5. Effective communication is defined as the activity of conveying sufficient information between agile teams and external specialists [24].

These factors were established from responses from individuals, which were not necessarily joined in teams. Although important factors in coordinating expertise outside an agile team were identified, it does not answer the question how this expertise is identified in the first place or what expertise is involved.

Software development in general needs expertise finding to facilitate unplanned collaborative work among software developers, but: "*Who are these additional people, and why are they contributing, or why have they been contacted …?*" [8, *p. 89*]. And these questions are equally applicable to the use of Scrum in software development.

To identify partners outside Scrum team boundaries, it is important to realize that both Product Owner and Scrum Master act as linking pins with the team's environment; the Product Owner in managing the Product Backlog in cooperation with stakeholders, the Scrum Master in serving the (outside) organization, for instance in helping employees and stakeholders understand and enact Scrum and empirical product development. Communication between the Product Owner and the Scrum Master on the one hand and partners like management and stakeholders on the other hand are thus already included in the Scrum process framework. Taking this into account and under the assumption that the Scrum team members attend all regular Scrum events we present an initial sketch of a Scrum team and its external partners (Fig. 1); this sketch can thus be considered as the common composition of a Scrum team (and its partners).

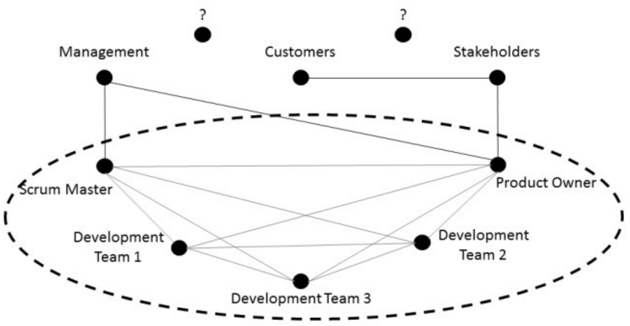

Fig. 1. Communication inside and outside Scrum teams

Question marks represent yet unidentified partners with additional competencies. The members Development Team 1 – 3 represent team members with competencies with regard to software analysis, design and coding [5], where each node represents several team members, depending on the team size. We will use this model both to guide our data analysis as well as to compare its contents with our results.

III. Case Study Design

To investigate the cross-functionality of Scrum teams our research analyses the communication of Scrum teams outside their team boundaries to identify which additional competencies they require and on which basis. To allow for rich evidence we selected an exploratory comparative case study approach as our research method with as unit of analysis one sprint of Scrum software development. This approach is an accustomed way to investigate phenomena in a context where events cannot be controlled and where the focus is on contemporary events [25]. For this case study we drew up the following research questions:

RQ1 To whom do Scrum teams turn to for additional competencies and which competencies are involved?

RQ2 How are Scrum teams aware of what additional competencies are needed?

We used a protocol to guide the case study [26]. This protocol contained:

- Its purpose, guidelines for data and document storage, and publication.
- A brief overview of the case research method.
- Detailed procedures for conducting each case, to ensure uniformity in the data collection process and consequently facilitate both within and cross case analyses.
- Research instruments.
- Guidelines for data analysis.

Given our goal and research questions, organizations were required to use Scrum as software development method with a team of at least 5 members; we chose this lower limit to allow us to find indeed cross-functional teams. We approached three organizations to participate in our research and all three were willing to do so; they all had a team size between 5 and 10 members. We name them Controller, Sunflower and Local for reasons of confidentiality.

A. Data collection

The primary data collection method was semi-structured interviewing of team members. We also inspected available documents and/or the contents of information systems which, in some cases, were used to support the Scrum process. Our questionnaire addressed communication both between team members, exemplified in the use of Scrum practices, as well as communication between team members and non-team members. Some examples of questions of the latter category (from the protocol) are:

- With whom did you communicate (mainly) during the sprint? Please mention all people and include people who might have been outside the scope of the sprint (in first instance).
- What was the communication about?

Interviews lasted, on average, 60 minutes. In total approximately 12 hours with 13 interviewees were available. To achieve a representative sample a team's Scrum Master and Product Owner were always included and, depending on the size of the team, 2 to 4 developers, including designers and testers when these roles were explicitly assigned in a team. Six more additional interviews were held either before the other interviews started to provide general context, or afterwards, to clarify remaining issues.

Thirteen interviews were transcribed and coded, with a combination of open and axial coding [27]. We also used our preliminary sketch (Fig. 1) to guide the coding; it guided the coding in the sense that external partners were either classified as manager, customer or stakeholder or an additional partner not yet included (previously a question mark). Summaries of interviews were consulted with the interviewees. For each organization the results of the interviews and additional material were bundled in a case study report.

B. Validity

Validity of our research method depends on four widely used criteria: construct validity, internal and external validity, and reliability [25].

Construct validity identifies operational measures for the concepts under study. To enhance construct validity (1) key informants should review draft case study reports, (2) multiple sources of evidence should be used, and (3) a chain of evidence should be established [25]. We applied all three: (1) Each interviewee was provided with a summary report of the interview and key interviewees commented on a draft case study report, (2) various team members were involved to complement viewpoints, and (3) interviews (and other materials) were linked to conclusions by using the tool NVivo; NVivo is a software package to aid qualitative data analysis (www.qsrinternational.com).

Internal validity is mainly a concern for explanatory case studies [25], [28]. Our case study is exploratory, but we did apply pattern matching, one of the analytical techniques recommended to enhance internal validity, by consistently coding our base material.

External validity defines the domain to which a case study's findings can be generalized. The use of replication logic is listed as the main guarantee for this [25]. Using a multiple-case study on the basis of a common questionnaire contributes to external validity, and thus to generalizability of results.

Reliability should demonstrate that the study can be repeated. The use of a case study protocol and the development of a case study database [25] were both applied in our study to increase reliability.

IV. RESULTS

In this section we describe the results from our case studies. We first give an impression of the organizations and the composition of their Scrum teams (Table 1). For Controller the role of Scrum Master coincides with one of the developers/testers; for Sunflower the role of Scrum Master coincides with one of the senior developers.

TABLE I. DESCRIPTION OF ORGANIZATIONS

Organization	Domain	Team composition			Scrum experience
		Product Owner	Development Team	Scrum Master	
Controller	Object management	1	2 developers 2 developers/testers	1	2 years
Sunflower	Floral industry	3	3 senior developers 1 junior developer	1	2½ years
Local	Government taxing	1	2 designers 4 developers 2 testers	1	1½ years

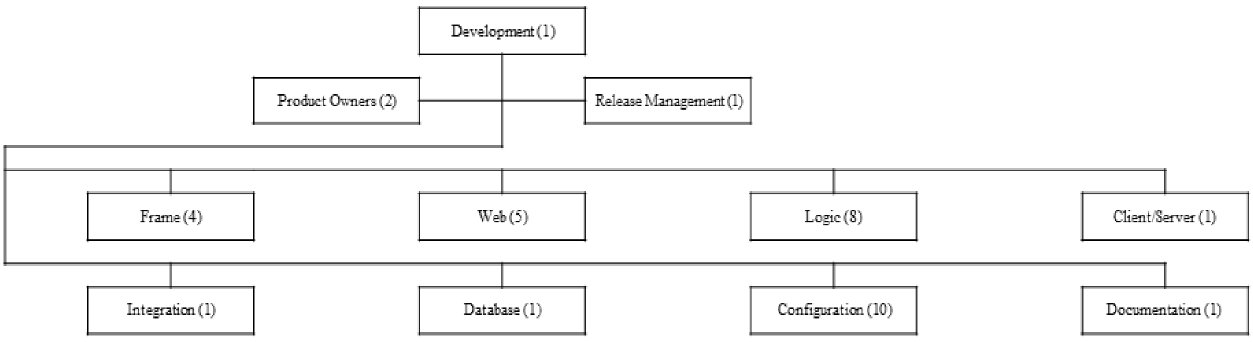

Fig 2 Controller organization chart for Development department

In the next three paragraphs we describe, for each organization in turn, the communication within, but mainly outside their Scrum teams' borders.

A. Controller

The Scrum team of Controller operates in a larger organizational context: the Development department with 35 employees (Fig. 2).The numbers in Fig. 2 refer to the number of employees in a group. The Frame group is responsible for the building blocks (basic components) of the software; Web adds a graphical skin. Together these groups produce a kind of half-fabricate. The groups Logic and Configuration build end products on the basis of these (supporting) components; the Scrum team is staffed from these two groups and complemented with one of the two Product Owners. In this Logic takes care of the process flow in the software, whereas Configuration is concerned with visual aspects, screens, buttons, reports, et cetera. Configuration is also responsible for testing the software. All other groups are small; they support the four groups mentioned before.

The Scrum team communicates in the framework of Scrum events. Sprint Planning Meeting, Daily Scrum, Sprint Review Meeting and Sprint Retrospective are all regularly scheduled 'meetings'. But team members also communicate with non-team members. The figure below (Fig. 3) demonstrates communication within and beyond the team's borders, where the customer is the only partner outside the organization Controller directly communicating with a team member.

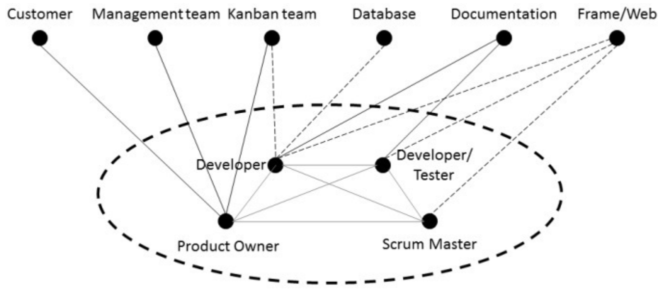

Fig. 3 Internal & external communication for Controller's Scrum team

The members of the Scrum team are shown in the lower half of Fig. 3; there is agreement on the team composition. In Fig. 3 only roles of team members are shown; the team in fact has 2 developers and 2 developers/testers (refer to Table 1). Apart from the Product Owner, all members are full time members.

External partners are shown in the upper half of Fig. 3. Partners are only involved in the team's activities part time, where in vast majority part time means less than a small number of hours per Sprint. Solid lines represent structural communication between the team members, where we defined structural as communication not only in the Sprint under consideration, but also in a majority of Sprints; dotted lines indicate ad hoc communication taking place in this Sprint only, but not necessarily in other Sprints. Furthermore we include only partners who are directly communicating with at least one team member as the team is the focus of our research.

Considering the external partners we focused on the unidentified partners (Fig. 1). Already identified partners for Controller then are Customer and Management team.

The Scrum team is only involved in the development of new parts of the software; maintenance is done by another team with Kanban. The Product Owner is responsible for coordination of the Kanban team; he - every employee is indicated as 'he', whether male or female - coordinates (the prioritization of) maintenance. It is his task to recognize overlapping activities of the Scrum and the Kanban team, with regard to components of the software, and to bring members of the two teams together whenever necessary. Although overlap does not occur every Sprint, communication is frequent whenever the two teams do work on the same piece of code to coordinate activities with regard to new code or adaptations to existing code.

The same applies for communication with the database specialist. He is consulted whenever sprint backlog items have impact on the database structure.

Documentation consists of a technical writer. He is responsible for the construction of a user guide and/or release notes in every Sprint and in cooperation with Development Team members. Communication is mainly on his initiative, where he has basic information available through a registration system.

Frame/Web, as producers of half-fabricates, are often contacted on details of the code they manufactured, although not in every Sprint.

The remaining groups in the department, Client/Server and Integration, were not mentioned to participate in the team's communication.

B. Sunflower

Sunflower belongs to the Small & Medium Enterprises (SME), more specifically a small company with a number of employees around 15; its organization chart is shown in Fig. 4, the numbers referring to the number of employees.

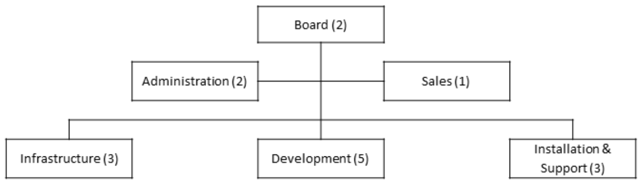

Fig. 4 Sunflower organization chart

Consultants, from Installation and Support, collectively function as Product Owner. The Development team consists of members of the group Development; the role of Scrum Master rests with a senior developer.

Members of the Scrum team communicate in the framework of Scrum. Sprint Planning Meeting and Daily Scrum are both regularly scheduled 'meetings', but they are skipped occasionally. Neither a Sprint Review Meeting nor a Sprint Retrospective is used by the team. Communication within and beyond the team's borders involves a restricted number of partners (Fig. 5), where customer is the only partner outside Sunflower.

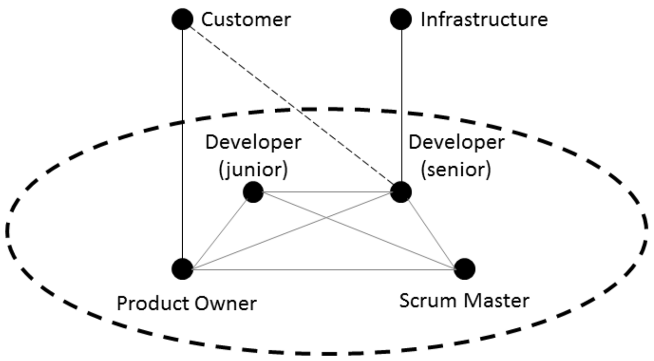

Fig. 5. Internal & external communication for Sunflower's Scrum team

A weekly meeting is scheduled with a representative of Infrastructure to prevent the Development Team from interfering with (scheduled) maintenance. However, whenever necessary, issues may also be taken up with Infrastructure immediately, not awaiting this meeting.

C. Local

The Scrum team of Logic operates in a larger organizational context: The Software Development department, consisting of:

- A Product group that is in charge of the implementation of software with new customers. Its developers convert customers from their previous software to Local's software; its consultants support customers with the set-up of the software and participate in courses for customers.
- Whenever a new customer approves of his software, it is taken into production at the customer and the responsibility within Local is transferred to the Support group. Support is the first point of contact for customers and functions primarily as a helpdesk. Support has some consultants for customer support on site or, again, courses, but does not employ developers. Changes as a result of customer reports are transferred to Development.
- Development deals with all modifications to the software, whether new features or as a result of bug reports. The Scrum team is found within this group.

Analogously to the Controller team Local's Scrum team uses all of the regularly scheduled Scrum 'meetings'. The figure below (Fig. 6) indicates communication within and beyond the team's borders.

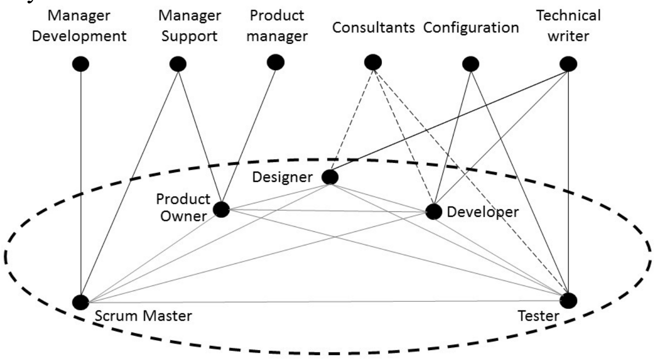

Fig. 6. Internal & external communication for Local's Scrum team

The managers (Development and Support) and the Product manager were already identified as stakeholders for customers. This implies that there is no direct communication between team members and partners outside of Local.

Configuration is in charge of the technical infrastructure. This partner, for instance, transfers software from a development stage to a testing stage to production (or vice versa). Involvement takes place in every Sprint.

The role of the technical writer is equal to the role of the documentarist in the Controller team.

D. Integrating results

We have shown results for the three individual teams with regard to their communication. In integrating these results, and in line with our research questions, we now establish a common vocabulary by mapping the partners from the individual cases to a limited set, enumerate them and indicate whether the communication is structural or ad hoc (Table 2).

V. DISCUSSION

It is important to note that, for none of the teams, there were differences of opinion on their composition. All Scrum teams were crystal clear about their membership. As an example, when interviewing members of Local's Scrum team all members agreed on having ten members on the team and every team member mentioned the same persons in

TABLE II. SUMMARY OF RESULTS

External partner	**Controller**	**Sunflower**	**Local**
Management	*Management team*	-	*Manager Support*
			Manager Development
Consultant	*Consultants*	-	*Product manager*
			▨ *Consultants*
Customer	*Customer*	▨ *Customer*	-
Developer	▨ *Frame/Web*	*Developer*	-
	▨ *Kanban team*		
Documentarist	*Documentation*	-	*Technical writer*
Database specialist	▨ *Database*	-	-
Infrastructure specialist	-	*Infrastructure*	*Configuration*

Structural communication
▨ *Ad hoc communication*

the same roles (Table 1). Whether being involved full time or part time (some Scrum Masters and Product Owners), team membership, and thus also non membership, was incontrovertibly.

In the staffing of the teams it is straightforward that competencies of team members are in one or more of the 'traditional' phases of a software development life cycle: Analysis/design, programming, testing (Table 1, Fig. 3, 5 & 6); these are indeed engineering skills [17-18].

When looking at the partners outside teams' boundaries we distinguish three categories:

1. In a first category we unite those partners who, from the viewpoint of socio-technical congruence, match coordination requirements with actual coordination activities; in fact they are the partners to be expected, as already identified in Fig. 1.
2. In a second category we mention partners who do not match coordination requirements with actual coordination activities from the viewpoint of socio-technical congruence, but join the team through ad hoc communication.
3. The third category concerns partners who also do not match coordination requirements with actual coordination activities, but do so through structural communication.

This first category encompasses consultants, customers and management. None of these partners provides engineering skills. Instead they provide domain knowledge and their communication takes place via the Product Owner (with one small exception where the Scrum Master was also involved). This comes as no surprise, since the Product Owner is the linking pin with stakeholders as he is responsible for managing the Product Backlog, an ordered list of everything that might be needed in the product [4]. Management also communicates on (project) progress with the Product Owner and/or the Scrum Master, because in Scrum there is no role of project manager. In fact none is needed; the traditional responsibilities of a project manager have been divided up and reassigned among the three Scrum roles [29], especially the Product Owner and the Scrum Master [30]. Communication with management is thus to be expected; Scrum teams do not operate in a vacuum. In a previous study we already found that Scrum teams use project plans and progress information for control purposes [31]. And our current results show the structural character of this communication.

The second category consists of developers and database specialists, contributing engineering skills or competencies with regard to code/coding and database (structure) respectively. Often this communication is facilitated by the Scrum Master or arranged by the Product Owner; initiatives originate from these two roles, but often delegated to a developer working on a particular backlog item thereafter. The communication then is from one developer to another. This category is in fact no different from non-Scrum software development: In e-mail discussions in software-engineering communication emergent people were included in a discussion as a result of an explicit request [32].

From the viewpoint of socio-technical congruence our data show a clear mismatch between coordination requirements and actual coordination in the third category. Both documentarist and infrastructure specialist are structurally involved in the Scrum teams' activities. The documentarist is competent with regard to the production of user-related materials, such as a user guide. Communication is supported by data in a registration tool, such as (elaborated) user stories, design documents, test reports. The infrastructure specialist supports the Scrum team with regard to transition of code to a test or production facility; of course this is a part time activity; he also supports other teams. The incorporation of such a specialist in a Scrum team's activities coincides with the DevOps concept; this is an agile operations concept that uses agile techniques to link up departments - Development (Dev) and Operations (Ops) - together, which traditionally operated in silos [33].

No matter to which category partners belong, Scrum teams are well aware of their external partners and their competencies. Differences between the categories refer to whether or not the use of competencies was planned and whether it was structural or ad hoc. Revisiting the five factors for coordinating expertise outside agile teams—availability, agile mind-set, stability, knowledge retention, and effective communication—these factors are in fact already incorporated in the Scrum teams' communication with their external partners.

In fact we introduce the notion of an extended Scrum team, which includes partners structurally involved in a team's activities, because communication occurs Sprint after Sprint (Fig. 7).

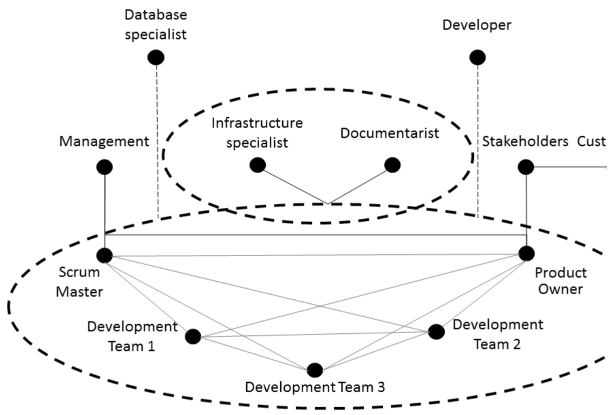

Fig. 7. The extended Scrum team

The extended Scrum team now incorporates an infrastructure specialist and a documentarist, adding technical and writing competencies. They work part time in the Scrum team, but they do not attend Scrum events, or at least not as a habit. The database specialist and the developer are (still) external partners. In other contexts it is conceivable that the database specialist is also a member of the extended Scrum team, because of his additional competencies.

III. Conclusions

Scrum team members agree on the boundaries of their team and are indeed cross-functional as far as 'traditional' phases of software development are concerned. Analysis/design, programming and testing competencies are represented in the teams.

To whom then do Scrum teams turn for additional competencies, which competencies are involved and how are Scrum teams aware of additional competencies they need?

- When additional competencies are required with regard to engineering skills, already represented in the team, the team turns to partners within the own organization. The teams are already aware of their partners and consult them on an ad hoc basis. They include other developers and database specialists.
- Additional competencies are also found in partners who structurally contribute their competencies to the Scrum teams in every Sprint. These partners include management, documentarists and infrastructure specialists. Here management was already expected to do so with regard to domain expertise and progress information; this is indeed a socio-technical match between coordination requirements and actual coordination activities. Others were not; especially documentarists and infrastructure specialists structurally contributed, although not full time, to the Scrum teams; they could be considered to be members of an extended Scrum team.

A. Limitations

In our three case studies documentarists and infrastructure specialists are found to be members of an extended Scrum team; a database specialist was not. This particular result cannot be generalized to Scrum teams in general; depending on, for instance, domain area, software type, or specific features of a team, partners could be distributed in another way, with the infrastructure specialist outside a team's border and, for instance, a database specialist inside. This argument also applies to partners who were not (even) identified in our case studies. What is generalizable, though, is the notion of the extended Scrum team, including partners not considered to be member of the team, but still structurally contributing to a team's activities. This allows practitioners to staff a Scrum team without pursuing overall cross-functionality in the team itself.

B. Future work

Of course we would like to see our results confirmed with other organizations. We believe the extended Scrum team to be an important extension of the notion of a cross-functional Scrum team and we would like to observe more and/or other external partners to elaborate this notion. We are also eager to pursue situational factors, whether organizational or personal, that determine the inclusion of external competencies.

Acknowledgement

The results from this research would not have been possible without the generous cooperation from the three case study organizations. The first author also wants to express his gratitude to Avans University of Applied Sciences for facilitating and supporting this research.

References

[1] D. Bustard, G. Wilkie, D. Greer, "The diffusion of agile software development: Insights from a regional survey", in Information Systems Development: Reflections, Challenges and New Directions, R. Pooley, J. Coady, C. Schneider, H. Linger, C. Barry, M. Lang, Eds., New York: Springer, 2013, pp. 219–230, http://dx.doi.org/10.1007/978-1-4614-4951-5_18.

[2] P. Rodríguez, J. Markkula, M. Oivo, K. Turula, "Survey on agile and lean usage in Finnish software industry", in Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '12), New York, 2012, pp. 139–148, http://dx.doi.org/10.1145/2372251.2372275.

[3] T. Chow, D.-B. Cao, "A survey study of critical success factors in agile software projects", Journal of Systems and Software, vol. 81, no. 6, pp. 961–971, June 2008, http://dx.doi.org/10.1016/j.jss.2007.08.020.

[4] K. Schwaber, J. Sutherland, "The Scrum guide – The definitive guide to Scrum: The rules of the game", 2013, retrieved from https://www.scrum.org/Portals/0/Documents/Scrum Guides/2013/Scrum-Guide.pdf.

[5] K. Schwaber, M. Beedle, "Agile software development with Scrum", 1st ed., Upper Saddle River (NJ): Prentice Hall, 2002.

[6] H.F. Cervone, "Understanding agile project management methods using Scrum", OCLC Systems & Services: International Digital Library Perspectives, vol. 27, no. 1, pp.18–22, 2011, http://dx.doi.org/10.1108/10650751111106528.

[7] K.B. Hass, "The Blending of traditional and agile project management", PM World Today, vol. IX, no. V, pp. 1–8, May 2007.

[8] D. Damian, S. Marczak, I. Kwan, "Collaboration patterns and the impact of distance on awareness in requirements-centred social networks", in Proceedings of the 15th IEEE International Requirements Engineering Conference (RE 2007), Delhi, 2007, pp. 59–68, http://dx.doi.org/10.1109/RE.2007.51.

[9] M. Mortensen, P. Hinds, "Fuzzy teams: Boundary disagreement in distributed and collocated teams", in Distributed Work, P. J. Hinds, S. Kiesler, Eds., Cambridge/London: MIT Press, 2010, pp. 283–308.

[10] K. Ehrlich, K. Chang, "Leveraging expertise in global software teams: Going outside boundaries", in Proceedings of the International Conference on Global Software Engineering (ICGSE '06), Florianopolis, 2006, pp. 149–158, http://dx.doi.org/10.1109/ICGSE.2006.261228.

[11] T. Chau, F. Maurer, "Knowledge sharing in agile software teams", in Logic versus Approximation - Essays Dedicated to Michael M. Richter on the Occasion of his 65th Birthday, W. Lenski, Ed., Berlin Heidelberg:Springer, 2004, pp. 173-183, http://dx.doi.org/10.1007/978-3-540-25967-1_12.

[12] J. Ferreira, H. Sharp, H. Robinson, "User experience design and agile development: managing cooperation through articulation work", Software: Practice and Experience, vol. 41, no. 9, pp. 963–974, August 2011, http://dx.doi.org/10.1002/spe.1012.

[13] A. Martin, R. Biddle, J. Noble, "An ideal customer: A grounded theory of requirements elicitation, communication and acceptance on agile projects", in Agile Software Development: Current Research and Future Directions, T. Dingsøyr, T. Dybå, N. B. Moe, Eds., Berlin Heidelberg: Springer, 2010, pp. 111–141, http://dx.doi.org/10.1007/978-3-642-12575-1_6.

[14] H. Sharp, H. Robinson, "Three "C"s of agile practice: Collaboration, Co-ordination and Communication", in Agile Software Development: Current Research and Future Directions, T. Dingsøyr, T. Dybå, N. B. Moe, Eds., Berlin Heidelberg: Springer, 2010, pp. 61–85, http://dx.doi.org/10.1007/978-3-642-12575-1_4.

[15] M. Cataldo, J.D. Herbsleb, K.M. Carley, "Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity", in Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM 2008), Kaiserslautern, 2008, pp. 2–11, http://dx.doi.org/10.1145/1414004.1414008.

[16] I. Kwan, A. Schröter, D. Damian, "Does socio-technical congruence have an effect on software build success? A study of coordination in a software project", IEEE Transactions on Software Engineering, vol. 37, no. 3, pp. 307–324, May-June 2011, http://dx.doi.org/10.1109/TSE.2011.29.

[17] M. Kropp, A. Meier, "Teaching agile software development at university level: Values, management, and craftsmanship", in Proceedings of the IEEE 26th Conference on Software Engineering Education and Training (CSEE&T), San Francisco (CA), 2013, pp. 179–188, http://dx.doi.org/10.1109/CSEET.2013.6595249.

[18] P. Bootla, O. Rojanapornpun, P. Mongkolnam, "Necessary skills and attitudes for development team members in Scrum: Thai experts' and practitioners's perspectives", in Proceedings of the 12th International Joint Conference on Computer Science and Software Engineering (JCSSE), Songkhla, 2015, pp. 184–189, http://dx.doi.org/10.1109/JCSSE.2015.7219793.

[19] S. Downey, J. Sutherland, "Scrum metrics for hyperproductive teams: how they fly like fighter aircraft", in Proceedings of the 46th Hawaii International Conference on System Sciences (HICSS), Wailea (HI), 2013, pp. 4870–4878, http://dx.doi.org/10.1109/HICSS.2013.471.

[20] M. Paasivaara, C. Lassenius, "Communities of practice in a large distributed agile software development organization - Case Ericsson", Information and Software Technology, vol. 56, no. 12, pp. 1556–1577, December 2014, http://dx.doi.org/10.1016/j.infsof.2014.06.008.

[21] J. Ferreira, H. Sharp, H. Robinson, "Values and assumptions shaping agile development and user experience design in practice", in Agile Processes in Software Engineering and Extreme Programming - Proceedings of the 11th International Conference on Agile Software Development (XP 2010), A. Martin, X. Wang, & E. Whitworth, Eds., Berlin Heidelberg: Springer, 2010, pp. 178–183, http://dx.doi.org/10.1007/978-3-642-13054-0_15.

[22] M.M. Rejab, J. Noble, G. Allan, "Distributing expertise in agile software development projects", in Proceedings of the Agile Conference (AGILE '14), Kissimmee (FL), 2014, pp. 33–36, http://dx.doi.org/10.1109/AGILE.2014.16.

[23] M.M. Rejab, J. Noble, G. Allan, "Locating expertise in agile software development projects", in Agile Processes in Software Engineering and Extreme Programming - Proceedings of the 15th International Conference (XP 2014), G. Cantone & M. Marchesi, Eds., Cham/ Heidelberg/ New York/ Dordrecht/ London: Springer International Publishing, 2014, pp. 260-268, http://dx.doi.org/10.1007/978-3-319-06862-6_19.

[24] M.M. Rejab, J. Noble, S. Marshall, "Coordinating expertise outside agile teams", in Agile Processes in Software Engineering and Extreme Programming - Proceedings of the 16th International Conference XP 2015, C. Lassenius, T. Dingsøyr, M. Paasivaara, Eds., Cham/ Heidelberg/New York/Dordrecht/London: Springer International Publishing, pp. 141-153, http://dx.doi.org/10.1007/978-3-319-18612-2_12.

[25] R.K. Yin, Case Study Research: Design and Methods (Applied Social Research Methods Series - volume 5), 4th ed., vol. 34, Thousand Oaks (CA): Sage Publications, Inc., 2009,

[26] H. Maimbo, "Designing a case study protocol for application in IS research", in Proceedings of the 9th Asia Conference on Information Systems (PACIS 2005), Bangkok, 2005, pp. 1281–1292.

[27] M. Paasivaara, C. Lassenius, "Collaboration practices in global inter-organizational software development projects", Software Process: Improvement and Practice, vol. 8, no. 4, pp. 183–199, October/December 2003, http://dx.doi.org/10.1002/spip.187.

[28] P. Runeson, M. Höst, "Guidelines for conducting and reporting case study research in software engineering", Empirical Software Engineering, vol. 14, no. 2, pp. 131–164, April 2009, http://dx.doi.org/10.1007/s10664-008-9102-8.

[29] P. Deemer, G. Benefield, C. Larman, B. Vodde, "The Scrum primer - A lightweight guide to the theory and practice of Scrum (Version 2.0), 2010, retrieved from http://www.goodagile.com/scrumprimer/scrumprimer20.pdf.

[30] M. Yilmaz, R.V. O'Connor, P. Clarke, "A systematic approach to the comparison of roles in the software development processes", in Software Process Improvement and Capability Determination - Proceedings of the 12th International Conference on Process Improvement and Capability dEtermination in Software, Systems Engineering and Service Management (SPICE 2012), A. Mas, A. Mesquida, T. Rout, R. V. O'Connor, A. Dorling, Eds., Berlin/Heidelberg: Springer, 2012, pp. 198–209, http://dx.doi.org/10.1007/978-3-642-30439-2_18.

[31] G. Wagenaar, R. Helms, D. Damian, S. Brinkkemper, "Artefacts in agile software development", in Product-Focused Software Process Improvement - Proceedings of the 16th International Conference on Product-Focused Software Process Improvement (PROFES 2015), P. Abrahamsson, L. Corral, M. Oivo, B. Russo, Eds., Springer International Publishing, pp. 133–148, http://dx.doi.org/10.1007/978-3-319-26844-6_10.

[32] I. Kwan,, D. Damian, "The hidden experts in software-engineering communication", in Proceedings of the 33rd International Conference on Software Engineering (ICSE '11), NIER Track, Waikiki, Honolulu (HI), 2011, pp. 800–803, http://dx.doi.org/10.1145/1985793.1985906.

[33] S. Karunakaran, "Impact of cloud adoption on agile software development", in Software Engineering Frameworks for the Cloud Computing Paradigm, Z. Mahmood, S. Saeed, Eds., London: Springer, 2013, pp. 213–234, http://dx.doi.org/10.1007/978-1-4471-5031-2_10.