This is a post-peer-review, pre-copyedit version of this article. The final authenticated version is available online at: https://doi.org/10.1109/RSSE.2012.6233401

Actionable Identification of Emergent Teams in Software Development Virtual Organizations

Giuseppe Valetto, Kelly Blincoe
Dept. of Computer Science
Drexel University
Philadelphia, Pa. USA
valetto@cs.drexel.edu , kelly.blincoe@drexel.edu

Sean P. Goggins
I-School
Drexel University
Philadelphia, Pa. USA
sgoggins@drexel.edu

***Abstract*— We present a method for identifying emergent teams of developers who need to work together and coordinate, within larger software development organizations. Our goal is to identify these socio-technical constructs as they emerge, so that we can provide timely awareness and actionable recommendations to managers, technical leads and members of the emergent team alike. Our technique is rooted in the analysis of Social Networks, which are constructed from real-time traces of the activity of each individual developer within her development environment, contextualized with respect to her assigned tasks and the corresponding artifact working set.**

***Keywords*-Emergent teams; Social Network Analysis; developers' coordination; task contexts; IDE interactions.**

I. Introduction

Large-scale or geographically dispersed software development organizations, ranging from open source communities, to global industrial software houses, to multi-enterprise joint ventures that practice off-shoring and outsourcing, operate as Virtual Organizations (VOs). In VOs, co-located work and in-person communication and coordination are the exception, and work and communication occur mainly through the mediation of a set of technologies and tools that make up the production infrastructure of the organization, and the work environment for the project.

In software VOs, collaboration and teamwork opportunities are not limited by the boundaries of organizational units, and coordination needs that may arise between participants must be followed opportunistically. In these environments, developers are keenly interested in information that helps coordinate with their colleagues [14], but have increasing difficulty obtaining it, as the scale and distribution of the software project increase [15].

We present a method that addresses a facet of this problem, that is, enables the identification of emergent teams within a larger software development VO, based on the coordination needs exhibited by the developers through their interactions with the work environment and the software artifacts the manipulate during their assigned tasks. Our method provides timely awareness of these emerging relationships, and can lead to actionable recommendations for project managers, technical leads and developers alike.

II. Related Work

Several approaches and tools that facilitate the recognition of emergent coordination relationships in software development organizations have been proposed. Some promote awareness of possible work conflicts [4][19], while others use some conceptualization of work dependencies to compute likely coordination needs [17][18].

One important observation is that all of these approaches, even those that explicitly attempt to focus on "teams", in fact draw upon dyadic relationships, and hence can only highlight and recommend pairs of developers that should, or are likely to, work together [21]; or alternatively, a set of colleagues from the point of view of an individual developer [20].

Our work, instead, aims at identifying one or more cohesive groups within a software VO at large, by understanding and analyzing the content and context of their individual work, and how that may pull them together to work as a team on a common coordination need or collaboration opportunity. The fulcrum of our analytical method is Social Network Analysis (SNA), which is increasingly employed in empirical software engineering, to tackle questions that are somewhat germane to the identification of emergent teams; for instance, stakeholder identification [23], and discovery of collaboration patterns [22]. We apply SNA techniques on a particular kind of bi-partite, socio-technical network that represents the collaboration opportunity within the organization.

III. Method Description

A. Contextualized Analysis of Work Traces

Our method derives from a framework for the analysis of group emergence in virtual organizations, on the basis of contextualized traces of work, called the Group Informatics Model, which we have developed from a number of empirical studies in different domains [13].

In all cases, we start from traces that we mine from either records of inter-personal exchanges, or records of work within the online environment of the VO. We call these contextualized interactions, since they must include metadata that situate them with respect to the work being carried out in that environment. For example, attributes that associate each interaction to recognizable tasks undertaken by the VO; or,

timestamps that situate the interactions in a specific epoch of the VO collaborative endeavors.

Our method has then a number of steps, which transform, weigh, distil and aggregate those mined contextualized interactions, to finally produce social network representations of emergent VO sub-groups (or teams). These teams are identified – in a quantifiable way – as currently operating in a tighter cooperative fashion than the rest of the VO, or in need of such tight cooperation.

Our method also derives information about the team context, from the merging of the contexts of the individual members of an emergent team. This may include *intentional* information about the reason (common interest or requirement) for the team to emerge; or *descriptive* information, about the kind and content of activities the group as a whole carries out. This data enhances the computed social network, and helps in shedding light upon the nature of the underlying multi-party relation.

The Groups Informatics Model thus enables us to fill a gap in the extensive literature of work that apply network analysis to raw electronic trace data, that is, they do not typically reveal groups that exist in the data [1], nor offer information on intent and content of group work. As we discuss in the remainder, our analytic framework also enables us to obtain these insights in a way that is timely and actionable for the purpose of recommendation.

B. Identifying Emergent Teams in Software VOs

Contextualized interactions in software development VOs can be readily extracted from a number of minable software repositories. Often, though, data in those repositories are traces that regard work already completed in the past, and thus are not actionable. We leverage instead fine-grained records of the interactions each developer has with software artifacts throughout the course of her work. These "live" traces can be collected by several IDE instrumentation tools, like Team Weaver [16], or Mylyn [2]. We have experience with Mylyn, one of the most widely used plugins for the Eclipse IDE. It records the developer's interactions with the Eclipse GUI, and captures actions with side effects on the work environment (such as invocation of tools, or modification of artifacts), as well as other actions that are not visible from *post hoc* traces kept in software repositories, like artifact consultation and navigation.

The original purpose of Mylyn is to ease the cognitive load of the individual developer when she needs to switch among multiple concurrent tasks, by presenting in a prominent way the most pertinent information for her current task, that is, the *task context*. Accumulating and maintaining context traces of the developer's work has recently commanded more and more attention in the domain of software engineering support tools, for instance because it can yield enhanced tool integration strategies [24].

Our work uses task context traces in a different way, that is, to analyze coordination needs. We have recently shown how that accurate and actionable evidence of the needs to coordinate between pairs of developers can be computed from those traces, as work progresses [3]. Here, we leverage and extend those findings and extend them to the identification of common needs and opportunities to coordinate in developers' groups.

To apply the analytical framework of the Group Informatics Model, we first consider the Mylyn context traces associated to all tasks carried out within a time window that is significant for the project, say, a release, and produce a set of *intersection records* in the form:

$$I_x = (Task_a, Task_b, Task_c, \ldots) \rightarrow (Art_1, Art_2, \ldots, Art_n); \quad (1)$$

Those intersection records indicate the intersections between the working sets of the various tasks in that release.

From there, we construct a valued bi-partite network: the developers involved in the tasks listed in each intersection represent a mode of the network, and the intersections themselves are the other mode. The weight of the arc between a developer D_a and intersection I_x is the number of artifacts manipulated by D_a in each task she has worked on that is included in I_x. Notice that, since developer D_a may be responsible for multiple tasks within the same intersection Ix, the weight of the edge between D_a and I_x can be a multiple of the artifact cardinality of the intersection.

We call this a *collaboration opportunities network*. The intersections represent the subject and the content of the potential collaboration between the developers that have operated on those artifacts in one or more of their assigned tasks within a software development release. The weighted incident arcs from a developer D_a to an intersection I_x must be intended as a representation of the interest - or need - of D_a to collaborate with other developers on the set of artifacts included in Ix. Moving from a local to a global view of the network, the set of incident arcs to various intersections departing from the D_a node show all the collaboration opportunities for D_a.

With the following step, we want to identify groups of developers in the network who gravitate towards common work (i.e. overlapping working sets) across several different tasks, and who have significant amount of overlap in those working sets. For this purpose, we look at the network construct of bi-cliques [5]:

1. we dichotomize the collaboration opportunities network at a level above the median of the weight of all edges;
2. we compute bi-cliques, which capture what subset of the developers' community tend to co-participate in the same intersections;
3. we compute a structural correlation matrix, based on the relationship between the set of developers and the set of bi-cliques they are part of.

As a result, we obtain a new person-by-person network, in which the weight of the arcs is a Pearson correlation coefficient, which signifies how similar two developers are in terms of the bi-cliques they are part of. We want to filter out the weaker correlations, therefore, we use a cutoff point of 0.4 for dichotomizing the arcs in the network. This new network distils the relations between developers, and any cohesive groups that appear within that network is a

candidate emergent team in the software development organization.

Notice that, throughout the process described above it is possible to "carry over" information that describes the collective context of those teams, derived from the original contextualized traces provided by Mylyn. The most important source of information comes from the intersection records, from which we can easily derive the artifact sets that are involved in each bi-clique, and underlie the bilateral developer-to-developer correlation arcs in the final network. The union of those artifact sets provides a look at the common "field of work" [6] of each emergent team.

IV. Empirical Results

We have experimented with the method described above using the Mylyn open source community itself as our case study, since its developers consistently abide to the convention of including Mylyn context records together with the other software artifacts, when they submit a patch or commit for their assigned development tasks.

We examined eight releases of the Mylyn software (v2.0 through v3.3), from December 2006 until October 2009. Each release lasted between three and four months, and saw the contribution of 13 to 18 developers. We consider all the tasks by developers in the same release as potentially concurrent. From the Bugzilla repository of the project we then extracted 1,970 software development tasks in that period. Each of those tasks is associated to one or more context records, for a total of 588,796 context events. We filtered those contextualized interactions, and considered only actions upon source code artifacts (450,747 events), since other types of artifacts are by-products – as opposed to subjects – of development work.

We also collected data from other project repositories; in particular, we mined the communication archived in the project mailing lists and discussion threads about the project tasks in the same release periods. The population contributing these comments is much larger than the set of actual developers. Over the 3 years considered, more than 400 distinct user IDs posted comments in these forums. This information represents triangulation data that we used to validate the results of our method. Another source of triangulation is the history of the Mylyn project during the period chosen for this study, which is freely available, as is the custom of open source projects by consulting the project repositories and archives on the Web.

We collected data and applied our method to all of the eight releases mentioned above. Given the limited space, we offer hereby only a sample of the results, and a discussion of their accuracy and significance, for release v.2.0, i.e., the first we analyzed and the one with the smallest data set. Figure 1 shows the network output by our analytical framework, which includes 13 anonymized developers.

We limit our analysis to constructs more complex than a triad. Two instances of a 2-clique are visible, composed by {304, 143, 399, 463, 373} and {304, 35, 22, 312, 319), which we consider as our "candidate" work groups, and refer to as WG1 and WG2, respectively. WG1 and WG2 have developer #304 in common; also, WG1 is densely connected, while WG2, is a "vulnerable" 2-clique [7]. As discussed – among others - by Erickson [8], a network construct that has more redundant arcs is more likely to truly represent an actual distinct group within the organization depicted by the network; therefore, our primary candidate for an emergent work group is WG1, while WG2 is a marginal candidate.

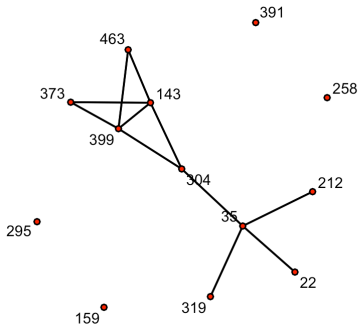

Figure 1: Mylyn v2.0 - cohesive groups based on task context

Triangulation with our knowledge of the project history helps to interpret these results, We know that the central figure of developer #304 is the most active in release 2.0, working on many tasks, and participating in the most intersection records. Moreover, from developers' profiles and communications records, we know the nature of the work relationship made evident by the arc between developers #304 and 35. The involvement of #35 in Mylyn v.2.0 amounts to a volunteer sub-project (the *Google summer of code*[1]), organized as an individual but intense task, in which #35 worked on hundreds of code artifacts, and was mentored by #304. The summer work of developer #35 concerned a component of the Mylyn product outside of the critical project path: as such, it was kept well-separated from the main stream of work and only intersected the activities of a few other contributors. The history of Mylyn v.2.0 thus corroborates the structure and groups visible in Figure 1.

We then compared our results with the communication traces from the archived developers' discussions. From those traces we have constructed the "talk" social network for Mylyn v.2.0, and compared it with the "work" network distilled by our analysis of contextualized interactions. Although the talk network and the communication records do not represent ground truth, we maintain that a triangulation between our method and network-analytic procedures carried out on an independent set of inter-

[1] http://code.google.com/soc/

developer relationships may be used to corroborate (or disprove) our results.

The talk network includes many more actors and relations (64 nodes and 131 arcs) than the work network. Therefore, we focused on our candidate work groups, and looked at whether they are also recognizable in the talk network. As a first step, we looked at properties of network connectivity, and seeked structurally cohesive subgroups [9], by means of the `cohesive.blocks()` algorithm implemented in the R package *igraph*. [10]. That procedure segments the overall graph in a hierarchy of increasingly more cohesive groups, each of which is a subset of the group that precedes it. Figure 2 shows the results, by drawing and color-coding the talk network according to the computed cohesive blocks it identifies. The most cohesive "talk" group is the set of seven red nodes TG1 = (304, 143, 399, 373, 159, 457, 391).

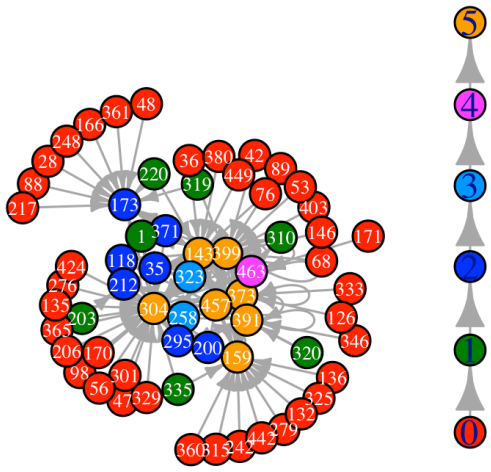

Figure 2. Mylyn v.2.0 – cohesive groups based on talk data.

We can immediately observe that - in line with [8] - our marginal emergent team candidate WG2 does not find confirmation in the topology of the talk network: among the five members of WG2 only #304, who is in common with WG1, is included in any of the most cohesive groups. We can also see that four of the five WG1 members also appear in TG1. The exception, developer #463 happens to be the lone orange node in Figure 2. That means that WG1 is fully represented in TG2 = (304, 143, 399, 373, 159, 457, 391, 463), which is the immediate superset of TG1 and the second most cohesive group. Therefore, the role of WG1 as an emergent distinct group in the Mylyn development organization finds further corroboration, from a structural standpoint.

Finally, we can carry out further quantitative analysis, and evaluate whether candidate group WG1 can be seen as "tighter" than other possible choices, e.g., highly cohesive structures like TG1 or TG2, on the basis of the communication data. Boch and Husain [11], as well as many other SNA scholars, have maintained that a distinct subgroup in a network should be characterized by more, or more intense, ties between group members, in comparison to the set of ties of the group members with the rest of the network In the case of a weighted, directed graph like our talk network, that contrast can be expressed with the formula in (2), where: GS is the ratio that denotes the group strength, a_{ij} is the weight of a tie between two nodes; $\overline{G}$ is the vertex cardinality of a candidate group G; and $\overline{N}$ is the vertex cardinality of the whole network.

$$GS = \frac{\dfrac{\sum_{i\in G}\sum_{j\in G} K_{i,j} * a_{i,j}}{\overline{G}*\left(\overline{G}-1\right)}}{\dfrac{\sum_{i\in G}\sum_{j\notin G} a_{i,j} + \sum_{i\notin G}\sum_{j\in G} a_{i,j}}{2*\overline{G}*\left(\overline{N}-\overline{G}\right)}} \qquad K_{i,j}=0 \text{ if } i=j \;/\; K_{i,j}=1 \text{ if } i\neq j \tag{2}$$

With (2), we can compute the ratio between the average strength of the group-internal arcs and arcs and the arcs towards network nodes external to the group (also known as centripetal vs. centrifugal strength [12]): the higher the ratio, the tighter the group, especially relatively to the average strength of arcs in the whole network.

We computed the group strength ratio for WG1 vs. TG1 and TG2, as displayed in Table 1. Although all groups are significantly tight, WG1 maintains a slight edge. We interpret this data to mean that WG1, which has been identified through our analytical framework, is at least as good a candidate for an emergent group as TG1 and TG2, which are derived instead from the topology of the communication network.

TABLE I. STRENGTH OF GROUPS IN COMMUNICATION NETWORK

v.2.0 – Avg. tie strength =5.87	WG1	TG1	TG2
Centripetal ties strength (Total)	408	507	531
Centrifugal ties strength (Total)	295	245	222
Centripetal strength (Avg.)	20.40	12.07	9.48
Centrifugal strength (Avg.)	0.50	0.31	0.25
GS ratio	**40.80**	**39.32**	**37.92**

When taken all together, the various elements we have derived from our triangulation confirm and strengthen results from our analysis of contextualized work traces.

V. DISCUSSION AND CONCLUSIONS

The analytic process that we have described has several important characteristics. First of all, it can be fully automated, since its steps are algorithmic in nature, with only a couple of parameters, whose values can be chosen either automatically or *a priori*.

Also, the process aims at distilling a large amount of project information (the bi-partite collaboration opportunities

network) into a much simpler network – with low node and edge cardinality – showing only significantly strong relations that make the fabric of cohesive work groups. From such a network, identifying emergent teams is often visually straightforward. Even in case the network produced by our method were larger and less sparse, patterns and structures revealing cohesive subgroups, like cliques, etc. could be easily extracted by running standard SNA algorithms.

Finally, and most importantly for the construction of recommender systems on top of our analytical framework, although the case study we have presented here is retrospective, the nature of the contextual traces that represent the starting point for our method is not. In fact, they can be incrementally accreted as work progress, and as shown in [3], leveraging IDE-recorded interactions such as those produced by Mylyn leads to the timely recognition of coordination needs between developer pairs. Although we have not yet experimentally validated it in a "live" user study, it stands to reason that the group detection procedure we have described can occur in an equally timely fashion. We envision its implementation within a tool that continuously collects the contextualized traces from the socio-technical systems and all of its participants, and periodically executes our analysis in a centralized server. Such a tool can follow in quasi-real time how teams in a VO form and evolve, based on the coordination concerns, task content and work context of its members. Major stakeholders for such a tool will be project managers and technical leads, for project governance, or the developers themselves, to enhance their coordination and team awareness.

ACKNOWLEDGMENT

This work was partially supported by the NSF through grant no. CCF-0916891.

REFERENCES

[1] J. Howison, A. Wiggins, and K. Crowston, K. " Validity Issues in the Use of Social Network Analysis with Digital Trace Data". Journal of the Association of Information Systems, 12(2) , 2012.

[2] M. Kersten, and G.C. Murphy. "Using task context to improve programmer productivity". Proc. of SIGSOFT FSE 2006.

[3] K. Blincoe, G. Valetto, and S.P. Goggins. "Proximity: a Measure to Quantify the Need for Developers' Coordination". Proc. of CSCW 2012, February 2012.

[4] M. Cataldo, P. Wagstrom, J. Herbsleb, and K. Carley. "Identification of Coordination Requirements: Implications for the Design of Collaboration and Awareness Tools". Proc. of CSCW'06, 2006.

[5] S.P. Borgatti, and M.G. Everett. "Network Analysis of Two Mode Data". Social Networks, 19(3), 243-269, 1997.

[6] K. Schmidt, and C. Simone. "Coordination mechanisms: Towards a conceptual foundation of CSCW systems design". Computer Supported Cooperative Work, 5(2-3):155–200, 1996

[7] S.B. Seidman, and B.L. Foster. "A graph-theoretic generalization of the clique concept". Journal of Mathematical sociology, 6(1), 139-154, 1978.

[8] B.H. Erickson. "The relational basis of attitudes." Social structures: A network approach, 99, 121, 1988.

[9] J. Moody, and D.R. White. "Structural cohesion and embeddedness: A hierarchical concept of social groups". American Sociological Review, 103-127, 2003.

[10] G. Csardi, and T. Nepusz. "The igraph software package for complex network research". InterJournal Complex Systems, 1695, 2006.

[11] Bock, and Husain. "An Adaptation of Holzinger's B-Coefficients for the Analysis of Sociometric Data". Sociometry, 13, 146-153, 1950.

[12] R.D. Alba. "A graph-theoretic definition of a sociometric clique". Journal of Mathematical Sociology, 3(1), 113-126, 1973.

[13] S.P. Goggins, C. Mascaro, and G. Valetto. "Group Informatics: A Methodological Approach and Ontology for Understanding Socio-Technical Groups". JASIS&T, 2012 (Under Review).

[14] T. Fritz, and G.C. Murphy. "Using information fragments to answer the questions developers ask". Proc. of ICSE 2010.

[15] J.D. Herbsleb, and A. Mockus. :An Empirical Study of Speed and Communication in Globally Distributed Software Development". IEEE Transactions on Software Engineering, 29(3):1-14, 2003.

[16] W. Maalej and H. Happel. A lightweight approach for knowledge sharing in distributed software teams. Proc. of the 7th International Conference on Practical Aspects of Knowledge Management. Springer, Jan 2008

[17] A. Sarma, A., Z. Noroozi, and A. van der Hoek. "Palantír: raising awareness among configuration management workspaces". Proc. ICSE 2003

[18] C.R. de Souza, , S. Quirk, E. Trainer, and D.F. Redmiles. "Supporting collaborative software development through the visualization of socio-technical dependencies". Proc. of the International ACM Conference on Supporting group work. 2007.

[19] J.T. Biehl, M. Czerwinski, G. Smith, G.G. Robertson, and B. Bailey."FASTDash: A Visual Dashboard for Fostering Awareness in Software Teams". Proc. of CHI, 2007, 1313-1322

[20] S. Minto, S. and G.C. Murphy, " Recommending Emergent Teams". Proc. of the International Workshop on Mining Software Repositories, 2007.

[21] F. Xiang, A. T. T. Ying, P. Cheng, Y. B. Dang, K. Ehrlich, M. E. Helander, P. M. Matchen, A. Empere, P. L. Tarr, C. Williams, and S. X. Yang. "Ensemble: a recommendation tool for promoting communication in software teams" Proc. of RSSE 2008.

[22] D. Damian, S. Marczak, and I. Kwan. "Collaboration patterns and the impact of distance on awareness in requirements-centered social networks". Proc.. of the 15th Int. Req. Eng. Conf., pages 59–68, 2007.

[23] S.L. Lim, D. Quercia, and A. Finkelstein. "StakeNet: using social networks to analyse the stakeholders of large-scale software projects". Proc. of the 32nd International Conference on Software Engineering (ICSE 2010), May 2010.

[24] W. Maalej. "Task-First or Context-First? Tool Integration Revisited". Proc.of the 2009 IEEE/ACM International Conference on Automated Software Engineering (ASE '09).

.