E-Mentoring for Software Engineering: A Socio-technical Perspective

Erik H. Trainer, Arun Kalyanasundaram, James D. Herbsleb
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA
{etrainer, arunkaly, jdh}@cs.cmu.edu

***Abstract*—Mentoring is one of the most effective pedagogical tools, holding great promise for software engineering education. When done badly, however, it can lead to dysfunctional interpersonal relationships and may turn off mentees from careers in software engineering. In this qualitative interview-based study we examine how socio-technical dimensions of software impact the formation of social ties important for satisfying two goals of mentorship, building technical skill and interpersonal development. We find that mentees working on user facing, interdependent software form a balance of ties that facilitate both goals, while mentees working on non-user facing software mostly form ties important for building technical skill. Work practices that create opportunities for unstructured contact between mentees and community members, such as code review in a mentee cohort, can help to overcome this imbalance. Our findings have important implications for task definition in software engineering e-mentoring program schemes.**

***Keywords*-E-mentoring; free and open-source software; tie content; instrumental ties; expressive ties; qualitative methods.**

I. Introduction

Mentoring is widely viewed as one of the most effective measures in pedagogy [1]–[3], having become an international priority [4] as evidenced by the thousands of institutional programs and practices that include a mentoring component [5]. Software engineering as an application area holds great promise, given the mounting body of evidence of the effectiveness of mentoring in science, technology, engineering, and mathematics (STEM) fields [6], [7]. When situated within the context of a complex real-world task, mentoring is one of the most successful ways for learners to develop mastery [8], [9]. Exposure to such tasks helps mentees to develop more mature mental models and problem solving strategies similar to those used by experts [10], [11], and build confidence to participate in the disciplines they wish to pursue.

There are other benefits to mentoring that are social in nature. By bringing in new contacts and making personal introductions, mentors can help mentees gain access to individuals in careers they aspire to join [6], [12], [13]. Mentoring also helps learners develop important interpersonal skills [14], [15]. A focus on real-world tasks facilitates the development of complex communication skills such as explaining, persuading, negotiating, and building understanding [11].

E-mentoring, a computer-mediated variation on traditional face-to-face mentoring, scales up mentoring, allowing mentees to access more information and connect with more people than traditional face-to-face mentoring allows [6], [13]. E-mentoring is of particular relevance for open and collaborative communities such as free and open-source software (FOSS) because the work is conducted in a distributed way. Google Summer of Code (GSoC), for instance, is an e-mentoring program that aims to connect university students with established members of FOSS communities, and give them real-world software development experience that employers value [16].

The potential impact of e-mentoring has made it a subject of much research. Research on e-mentoring thus far has looked at phases constituting e-mentoring programs and their importance [13], [17], as well as the positive outcomes of such programs [6], [18]. Less research, however, has looked at task definition for e-mentoring, that is defining the technical work itself and the work practices that should be followed to complete it. It would be very helpful to know how task definition impacts the goals of mentoring, so that program organizers and mentors can create a good mentoring experience.

This is an important area to address because mentoring experiences that do not meet the goals of mentees [19] have the potential to alienate mentees from communities of practice [20] they hope to join. Such experiences can result in the erosion of trust between mentor and mentee, low perceptions of the effectiveness of their relationship, and reduced personal satisfaction with the work [21]. They may also contribute to greater intentions to withdraw from one's current career path, a concern of particular importance in STEM [22].

With free and open-source software development as its setting, this qualitative, interview-based study reports on how software project design (i.e., visibility to end users and interdependence) and work practices (i.e., cohort code review, virtual group meetings, and face-to-face meetings) influence key goals of mentoring. We add to literature on e-mentoring that these factors can be practically used to structure work to provide a desirable mentoring experience. At the same time, we contribute to literature on software as a socio-technical system, which had previously only examined factors impacting the *structure* of developers' social networks. We offer rich, qualitative evidence of the impact of technical work and social practices on the *content* flowing through these networks. Accessing this content is important for building technical skill and making personal connections.

In the next section, we review existing literature on e-mentoring and introduce theory on tie content, our analytical lens into a mentee's interactions. We then describe our research methodology, and present and discuss our findings in the subsequent sections.

II. Background

Mentorship, a relationship in which a more senior individual (the mentor) provides guidance and support to a more junior member (the mentee), is widely viewed as one of the most effective measures in education [1]–[3]. When conducted using electronic communications it is referred to as *e-mentoring* [23]. It is widely held that e-mentoring retains the benefits of traditional face-to-face mentoring [13]. At the same time e-mentoring scales traditional mentoring up to provide mentees access to more information and more people through increased interactions with others [13], [24]. E-mentoring is particularly relevant for large online communities like FOSS where software is developed in an open and collaborative way. Successful participation in FOSS requires not only understanding technical aspects of the code, but the ability to communicate effectively when engaged in such things as submitting code, responding to code submitted by others, documenting code, and managing conflict.

Research on e-mentoring thus far has largely looked at the significance of key phases constituting e-mentoring programs [13], [25], as well as the positive outcomes of such programs [19], [26]. We will argue that understanding how to define the tasks a mentee takes on is a neglected yet important design consideration for e-mentoring programs.

A. e-Mentoring Program Components

Several studies thus far have identified the components of an e-mentoring scheme [3], [13], [25]. Salmon [25] proposed a five-stage model of e-mentoring consisting of *access and motivation*, *online socialization*, *information exchange*, *knowledge construction*, and *development*. Single and Single's [13] three-phase model includes *planning*, *program structure*, and *assessment*. Both models highlight the importance of mentors and mentees connecting and familiarizing themselves with each other early on, focused training provided by the mentor to the mentee, the inclusion of an e-moderator who can act as a resource to both mentor and mentee, and reflecting on and evaluating the learning process. An aspect highlighted by Single and Single [13] not included in Salmon's model [25] is group e-mentoring, where participants listen in and learn from other mentors and mentees by critiquing and offering feedback on their work. In addition to obtaining help from people other than their mentor, group e-mentoring can allow participants to develop a greater affiliation with the e-mentoring program as a whole [23].

B. e-Mentoring Program Outcomes

A significant benefit of such programs to the mentee is the transfer of information and subject-matter [13], [19], [26]. When based around a complex, real-world task, mentoring is one of the most effective ways for learners to develop technical mastery. By practicing authentic tasks with expert mentors, mentees cultivate cognitive processes beneficial for problem solving, and demonstrate the legitimacy of their understanding [8]. They develop more mature mental models of problem solving strategies, and build confidence as "culturally relevant members of a community" [10], [11].

In addition to learning about how to solve complex tasks, e-mentoring helps mentees to develop interpersonal skills important for teamwork and networking. When combined with real-world tasks, e-mentoring facilitates the development of communication skills such as explaining, persuading, negotiating, and building understanding [11]. E-mentoring can also facilitate the development of strong and meaningful personal connections. For instance, mentees may come to trust and appreciate their mentors in the context of working with them on goal-oriented tasks [5], [21]. Through communication, mentors may help mentees better understand, express, and regulate their emotions [27]. Positive experiences including the expression of social support can enable mentees to interact with others more effectively [5]. These benefits can be particularly important for marginalized populations. Mentees with disabilities, for instance, may be more willing than in face-to-face mentoring to cultivate relationships since the Internet makes their disabilities less visible [18].

It would be helpful to know more about how task definition impacts these two important goals. The interactions mentees have lead to technical mastery and the development of interpersonal relationships, and these interactions flow from the real-world tasks that mentees take on with their mentors. A spontaneous tutorial, or help offered on a particular topic, may help the mentee simultaneously build up knowledge and create personal connections. These activities are not identical, however. Social activities that help build familiarity may not immediately enhance the mentee's technical knowledge. Answering a technical question will facilitate learning but not necessarily build familiarity. Can both goals of mentoring be satisfied simultaneously, or is there a tradeoff between them?

Done poorly, task definition may result in unfavorable attitudes, strain on the mentoring relationship, and dissatisfaction with the experience. Young and Perrewe [21] investigated levels of met expectations in a mentoring relationship, showing that exhibiting career and social support behaviors that meet the expectations of a partner result in positive perceptions of the relationship effectiveness and trust. Eby and Lockwood [19] further substantiated the importance of examining met expectations in a qualitative study, documenting instances where mentors failed to meet mentees' expectations (e.g., "opening doors" by introducing them to key people in their field) and linking these unmet expectations to dissatisfaction with the experience. Since people tend to respond more strongly to bad experiences [28], badly structured e-mentoring may lead to greater intentions to withdraw from one's career path.

A mentee's social network has important implications for learning and interpersonal development, since social networks facilitate and constrain the flow of resources between and

within groups [29]. Two concepts central to understanding this flow are its *structure* and the *content* of the connections.

C. Tie Structure

Tie structure refers to the pattern of connections, often called the network topology, among people in the network [30]. One can also look at the ties associated with a specific node in a social network. A node's centrality [31], for which there are several distinct measures, refers to its positioning relative to other nodes in the network. For instance, someone who has high degree centrality is tied to numerous people on a team and is therefore said to be central in the social network.

D. Tie Content

The nature of the resources that flow through a social network's tie structure, called tie content, is equally important. Tie content can be *instrumental* or *expressive* [30], [32]. Instrumental ties are pathways of work-related information, examples, advice, and introductions, and are therefore important for task performance [30], [33]. Expressive ties reflect friendships, and contain emotional content. They are considered important conduits of social support [30], [32], [33].

Much of what we currently know about attaining technical knowledge and interpersonal development is based on tie structure. For instance, having network contacts with high positions in a status hierarchy allows the learner to benefit from their experience and greater understanding about how to perform tasks [34]. A central person in the social network has greater access to, and a larger amount of, information or social support [35]. But understanding how to form ties with both types of content is important as well, since this content shapes the benefits mentees gain from their mentoring experience. We need to understand, therefore, what factors contribute to forming both expressive and instrumental ties.

E. Ties in Socio-technical Software Systems

Based on what we know about tie structure, research into software as a socio-technical system provides evidence of some factors that may be relevant. In a study of a geographically distributed software project, Cataldo and Herbsleb [36] found that developers who worked on software tasks that cut across numerous subsystems became more central in the project's communication network. Studying newcomers to offshored legacy projects, Zhou et al. [37] found that working on self-contained tasks not interdependent with the rest of the system required few interactions with other developers. But as newcomers learned more and worked on more central tasks, they moved to the center of the communication network.

These studies have contributed to our understanding of the impact of socio-technical factors on tie structure, but we still know relatively little about their influence on tie content. What project characteristics impact tie content? What social processes impact tie content? How do project characteristics and social processes interact? These gaps suggest that more research is needed to uncover the relationships between socio-technical factors and tie content with respect to FOSS e-mentoring. We therefore aim to advance this line of research by asking the following questions:

RQ1. How do technical task characteristics impact the formation of instrumental and expressive ties by mentees?

RQ2. How do social task characteristics impact the formation of instrumental and expressive ties by mentees?

III. Method

Our setting is Google Summer of Code (GSoC), a FOSS e-mentoring program. GSoC aims to get university students involved in real-world software development and enhance their employment opportunities in areas related to their academic pursuits [16]. Each year umbrella organizations apply to Google to be eligible to participate. Umbrella organizations are groups of FOSS organizations that work very closely, have similar goals or communities, or produce similar products. For instance, in 2015 the Python Software Foundation served as an umbrella organization for SciPy and SymPy, Python-based software for mathematics, science, and engineering. Google gives accepted umbrella organizations slots for one or more coding projects that university students apply to work on. Umbrella organizations select students and assign mentors to the projects. Google then pays accepted students stipends to work on the projects over the summer.

A. Data Collection

We chose to focus our investigation on FOSS *scientific software*, FOSS written by and for scientists. FOSS as a whole has a varying mix of extrinsic and intrinsic incentives for participation [38], [39]. In contrast, the primary incentive to build scientific software is inherent in its purpose. To produce scientific results, scientists need tools, and will invest time and resources to create them [38], [39]. Restricting our sample to scientific software helps us avoid the combined variance in incentives of scientific software and all other FOSS.

Moreover, much effort has been devoted to educating a diverse workforce in science, technology, engineering, and mathematics (STEM). The advantage of looking at scientific software is that it combines several of these high priority areas: science, technology, and to an extent, mathematics.

To start, we chose GSoC umbrella organizations from year 2015. Of the 137 umbrella organizations, we identified 26 (19%) as scientific software-oriented by manually inspecting entries in the GSoC project database (Google, accepted projects 2015). To identify these organizations, we looked at the "Name" field in the database for mentions of academic labs developing software. We also examined the "Tags" field for mentions of the domain in which the software was meant to be used (e.g., "genomics", "chemistry"). Finally, we read the list of proposed projects on the web sites provided in the "Ideas" field for descriptions of who would be using the software and in what contexts.

We interviewed mentors as well as mentees. We expected mentors to have additional insights into factors influencing

tie formation, for instance the GSoC project's relationship to ongoing work in the community and norms around communication. To cover as many organizations as possible, we randomly selected projects within each of the 26 organizations and sent e-mail invitations to interview the associated mentees and mentors. In total we received responses from 15 mentees and 9 mentors coming from 16 out of the 26 organizations (62%) and 20 projects. In four projects we spoke with both mentee and mentor. Table I shows demographic information for our participants. We assigned each participant an ID, shown in the first column. The first character of that ID indicates the participant's role: "S" for student or "M" for mentor.

We conducted semi-structured interviews with participants. Our goal was to identify instrumental and expressive ties that existed by understanding the interactions that happened to form those particular ties. Our general strategy was to identify everyone with whom participants interacted across all phases of GSoC, including proposal planning, the coding period, and follow-up work. We then probed further to understand the nature of their interactions. For instance, was the content of these exchanges about work-related issues and problems or were they more about encouragement and emotional support? We gathered additional detail on such things as who initiated the interactions, the frequency of the interactions, and their significance to participants. Interviews lasted approximately one hour, and were recorded and transcribed verbatim.

We encouraged participants to share additional materials with us after the interviews to support the analysis process. These materials included public communications with other community members, self-published reports of their progress throughout the summer, and project-related documentation.

B. Data Analysis

We applied standard qualitative analysis techniques [40] to our interview data. We started with the conceptual categories *instrumental ties* and *expressive ties*. Two members of our research team defined a tie as a mention of having interacted with another student, mentor, or someone else within or outside the FOSS umbrella organization. We read the interview transcripts to identify ties, and then open coded the specific interactions constituting those ties. We also open coded the transcripts to identify social and technical factors influencing the creation of those ties. In parallel we wrote, shared, and discussed descriptive memos about emerging themes. In the next part of our analysis, we compared each code to other examples and identified additional conceptual categories using themes from our memos as support. We met weekly to discuss our progress, and resolve occasional coding disagreements by joint consensus. We continued this process in an iterative manner until no new factors or tie content were being captured by our emerging categories.

Our final codebook had 16 codes organized in a parent-child hierarchy, with 4 parent and 12 child codes respectively (Table II). For instance, the *instrumental ties* parent code included child codes *technical assistance* and *task clarifications*. The *project design* code included the child codes *front end* and *interdependent*. We used the archival data participants shared with us to triangulate and confirm the relationships identified in our analysis. Our set of archival data included 423 online forum posts, 367 comments from software issue trackers, 42 blog posts, 18 chat excerpts, and 37 project documents (e.g., proposals, design-mock ups, and so on).

TABLE I
PARTICIPANT DEMOGRAPHICS. G = GENDER; C = COUNTRY; D = EDUCATIONAL DEGREE PURSUED: UNDERGRADUATE, MASTERS, OR DOCTORATE.

ID	G	C	D	ID	G	C	D
S1	M	IN	U	S13	M	RU	M
S2	M	US	D	S14	M	IN	U
S3	M	US	M	S15	M	US	U
S4	F	IN	U	M1	M	US	-
S5	M	TR	M	M2	M	AM	-
S6	M	IN	U	M3	M	US	-
S7	M	RO	M	M4	M	US	-
S8	M	US	D	M5	M	BE	-
S9	M	US	M	M6	M	CZ	-
S10	M	IN	U	M7	M	RO	-
S11	M	PT	M	M8	M	US	-
S12	M	CA	U	M9	M	US	-

IV. RESULTS

A. RQ1. How do technical task characteristics impact the formation of instrumental ties and expressive ties by mentees?

The overarching technical factor we identified as impacting instrumental and expressive ties was *project design*. We identified two dimensions of importance: *front end vs. back end*, and *modular vs. interdependent*.

Front end projects are projects where the student's code changes are directly or indirectly expressed in a user facing interface. Under this definition, both a standalone application that a user interacts with directly and a performance optimization that makes a user interface more responsive are both front end projects. We consider projects not meeting this criterion to be back end. Of the 20 projects, we classified 11 as front end and 9 as back end.

Modular projects are projects where students did not reuse existing libraries (10 projects). Interdependent projects are projects where students did reuse existing libraries (10 projects). Sometimes, students and mentors did not decide to reuse code until their GSoC projects were well underway. As such we relied on students' descriptions of their projects and accounts of their interactions with others, in addition to GSoC project summaries on their organization's website to classify the code's structure.

As summarized in Table III, students working on front end, interdependent projects form a balance of instrumental ties via *technical assistance* and *task clarifications* (because others dependent on their work can provide details of their use) and expressive ties via *appreciation* from users (because the product of their work is visual). In contrast, students working on back end, modular projects form mostly instrumental ties via

TABLE II
FULL CODEBOOK WITH DESCRIPTIONS AND EXAMPLE QUOTATIONS.

Code	Description	Example Quotation
+Instrumental ties	Social connections based on exchange of information resources and knowledge to complete tasks	
Technical assistance	Mentee receiving technical help from project developers	*"You should build symengine (C++ library) as a shared library: `cmake -BUILD_SHARED_LIBS=on`. Otherwise if you build a static library you will need to link each of the dependencies of symengine (teuchos, gmp, etc.)"*
Task clarifications	Project developers, users giving advice on what should be implemented, who will use it, other software it should work with	*"We do not plan to rely very much on this standalone because it based an implementation which oracle abandoned :) So a GSoC project based on our current standalone is not a good idea."*
+Expressive ties	Social connections based on positive emotion, encouragement, and support	
Appreciation	Explicit acknowledgment and recognition of the mentee's contribution	*"Very nice work! [. . .] Bravo to your GSOC project!"*
Positive emotion	Expression of pleasure or enjoyment	*"It really feels good to see the green check mark beside the commit. Thank you [mentor] [other student] [S13] for helping me get this working :blush:"*
Personal comments	Informal conversations not directly related to the GSoC project	*"I do hope the working late gets a bit better in the next weeks [. . .] if the release I'm working on is tested and rolled out to production, things hopefully get back to normal."*
+Project design	Visibility of code to end users and relationship to other projects	
Front end	Code directly or indirectly expressed in a visual user interface	*"So my project uses web graphics to plot the CALIPSO satellite LiDAR profiles over the earth."*
Back end	Code not expressed in a visual user interface	*"[My project] calculates the properties of fluids using Taylor expansion [. . .]"*
Interdependent	Project reuses existing libraries under development elsewhere	*"We actually e-mailed some of the Siphon developers and said, 'Hey, we have some bugs we need to fix. Can you give us some pointers?'"*
Modular	Project does not reuse existing libraries under development elsewhere	*"These visualizations were done from scratch. [The user] doesn't have to download anything else."*
+Organization practices	Work practices, patterns of behavior expected of participants	
Cohort code review	Examinations of source-code performed by mentee pairs, used to find and correct mistakes	*"We made students review—cross review their code and make suggestions and sort of—if somebody has answers, encourage them to talk [. . .]"*
Virtual group meetings	Video conferences where current mentors and mentees give updates on progress, announce issues, and state next steps	*"Weekly we have an IRC meeting. You have to summarize what you have done in the last week, or what you are going to do in the next week and what are the problems."*
F2F meetings	Collocated events where mentees meet their mentors and other community members, present their projects, and get feedback	*"I was in a meeting of the organization this year and I talked with some users to know what they need and I showed them my project to see what—how can I help the final user with this project."*

technical assistance but lack opportunities to form expressive ties via appreciation.

1) Front End vs. Back End: Students who worked on front end projects tended to form expressive ties. They received appreciation from potential users of their projects, even though students working on both types of projects provided weekly updates in blog posts and on their organizations' mailing lists. For instance, a student who worked on a front end project in 2014 and a back end project in 2015 noted a stark difference in the amount of comments reflecting appreciation for her work: *"And like last year, I was getting more [comments] from even the people who are less technical, so because you can see the toolbar. You can see the editor. . . But this project, I am not expecting so much. . . because you won't notice it much." (S4)*, For example, an appreciative user left the following comment on S4's blog: *"How can we get this new version? This is exactly what I was finding missing in [name of umbrella organization's software]."*

Students working on front end projects also formed instrumental ties with developers via technical assistance. The visual form of front end projects facilitated technical help from developers not only within the FOSS organization (S3, M3, M4), but also outside it (S6). For instance, after a student visualizing NASA satellite data using a third party visualization library uploaded his application for testing, a developer of that library was able to identify an issue in his library because of the poor responsiveness of the interface: *"For some reason, the stars took a bit longer than usual to load, which exposed issue #1829. The fix for this will be in [third party library] 1.10, which comes out on June 1."*

Students working on back end projects also formed instrumental ties with developers via technical assistance. Sometimes, however, the help had to do with the scientific domain to which the software would be applied (S5, M4, S8). Besides their own mentors, students turned to and received assistance from domain scientists and professors at their universities. In one case, a student referencing mathematical models of brain cells described in a research paper asked a professor at his university for help because important details in the paper were missing and the authors were unresponsive (S8).

TABLE III
FOR POSSIBLE VARIATIONS IN PROJECT DESIGN, THE TYPES OF CONTENT THAT FLOW THROUGH MENTEES' SOCIAL TIES.

	Modular	**Interdependent**
Front End	Instrumental ties via: *Technical assistance* (developers) Expressive ties via: *Appreciation* (users)	Instrumental ties via: *Technical assistance* (developers) *Task clarifications* (users, developers) Expressive ties via: *Appreciation* (users)
Back End	Instrumental ties via: *Technical assistance* (developers)	Instrumental ties via: *Technical assistance* (developers) *Task clarifications* (developers)

In sum, students working on front end projects and students working on back end projects formed instrumental ties primarily through technical assistance. Because of the visual form of their projects, students working in front end projects frequently formed expressive ties via appreciation; students working on back end projects did so only rarely.

2) Modular vs. Interdependent: We did not find evidence that the absence or presence of interdependency in a GSoC project was related to students forming expressive ties. As we explained in the previous sub-section, the extent to which the project is expressed in a user facing interface seemed to be much more important.

Students working on modular projects formed instrumental ties via technical assistance. These ties tended to be about helping students understand and solve errors in the code they had written, or helping them understand parts of the organization's codebase (S4, S10). As one student put it: *"If I asked something related to code, they'd [other developers] give me the GitHub link that the file or the part of the code in there, so like direct me to that part of the code." (S4)*

Students working on interdependent projects also formed instrumental ties via technical assistance. Compared with students working on modular projects, students working on interdependent projects were able to learn more in depth about their organization's codebase as a result of their interactions with developers (S8, S9). For instance, a student (S9) writing an application programming interface (API) described how a big part of his task was understanding buggy code written by others: *"We knew about some of these bugs, we didn't know their scope quite so much, because you think handling exceptions for arithmetic operators; that's an easy fix in some file somewhere. It's not." (S9)*

In response, he asked the developers of that code for technical assistance in identifying the code causing the bugs. Though this took substantial effort and time, he got familiar with the codebase in more depth: *"I understand the Cython code a lot better" (S9)*, and felt that he learned about making good software design decisions: *"Better design principles is something that I've been trying to get a handle on a lot lately... this helped provide an avenue that would let me work on that a bit more... knowing how to work with a buggy codebase and choosing which battles to fight and which battles to leave for later." (S9)*

Although our focus is on tie content, we note that the network structures formed in interdependent projects were different than those formed in modular projects. Students working with interdependencies mentioned going to more people for technical help, indicating that they formed more instrumental ties with other developers compared with students working on modular projects.

Unlike students working on modular projects, students working on projects with interdependencies formed instrumental ties via task clarifications. This was useful for learning about managing design decisions. One student working on a broad range of issues in preparation for his organization's next software release constructed a list of all the downstream functions that would be affected by his changes. Presenting this list of functions on the mailing list led to *"a lot of discussion with the community"* wherein the student was an active participant (S12).

Working on projects with interdependencies also helped students learn more about how their projects would be used by others (e.g., complementary software), and what features would be most useful (S11, S12, S13). For instance, a mentor recalled that he and his student were torn between two design approaches. It wasn't until a student using their code posted a bug report that they discovered the best solution: *"I remember we said, 'okay, we don't actually know what we're gonna do'... a significant amount of time later a different student from the [organization] came up and said, 'Hey, I need to do this,' and it was actually this exact same thing we did. So he had an idea how we could do it and we did do it that way because he was one of the people using that program. So he was one of the users and also one of the students of the other students. His suggestion made sense."* (M7)

In sum, students working on modular projects and students working on interdependent projects each formed instrumental ties via technical assistance. The students working on interdependent projects, however, were able to learn more about codebases in more depth and form instrumental ties via task clarifications, which helped them gain insight into how to make the software more generally useful. The absence or presence of interdependency was not related to the formation of expressive ties.

B. RQ2. How do social task characteristics impact the formation of instrumental and expressive ties by mentees?

The overall social factor influencing formation of instrumental and expressive tie was *organization practices*. Although

there are many practices that organizations used, participants mentioned the following practices as having the most impact on the personal connections they formed with others: cohort code review, virtual group meetings, and face-to-face-meetings. Table IV summarizes these findings.

1) Cohort Code Review: Some umbrella organizations required all of their current students to participate in two or three code reviews. In this cohort code review, students and mentors from all current GSoC projects in the organization examined one or two files of code written by other students, ran associated tests to assess code correctness, and commented on specific lines of problematic code. These reviews resemble features of group-e-mentoring [13], which allow participants to listen in and learn from other mentors and mentees.

Cohort code review was uncommon (2 of the 16 organizations used it), yet it was surprisingly helpful for building expressive ties, going beyond simply helping to advance technical work as the name suggests. Students benefitted from the wisdom and encouragement of other mentees. A student we talked to told us that even though GSoC designates a period at the beginning of summer as the "Community Bonding Period," much of the real bonding happened only after he started participating in code reviews (S14). Of the students working on back end projects, only those participating in cohort code review formed expressive ties with other students and mentors. In the example below one such student (S14) exhibits *positive emotion* toward his reviewers after receiving suggestions about how to resolve bugs in his code: *"It really feels good to see the green check mark beside the [code] commit. Thank you [mentor] [other student] [S13] for helping me get this working :blush: " (GitHub issue comment)*

Students and mentors also talked about how this flavor of code review was a *"fun exercise"* that *"helped break psychological barriers"* (M7) resulting from perceptions in differences between the skill levels of students and other students, and students and mentors. Code review improved interactions between students in the same organization, making it easier to ask them for help in the future (S13, M8, S14).

Cohort code review, combined with the reviewability [41] and transparency [42] of GitHub issues facilitated forming instrumental ties in which even students and mentors working on other projects were able to offer technical assistance: *"[Other student] reviewed my code and found a bug that related to outputting updated data…His review was very thorough and helpful. He both pointed me on missing and wrong docs and fixed some of them in his pull request. And he also gave me idea about a notebook with an example that should be created." (S13's blog entry)*

2) Virtual Group Meetings: Shorter than code reviews in duration, and taking place synchronously via chat rooms and video conference, virtual group meetings provided occasions for students to announce what they were working on, problems they faced, and what they were planning to do next (S3, S5, M3, S10, S13, M7, S14). Participants included students, mentors, and developers, all of whom belonged to the same umbrella organization.

TABLE IV
FOR EACH KIND OF ORGANIZATION PRACTICE, THE TYPES OF CONTENT THAT FLOW THROUGH MENTEES' TIES.

Practice	Instrumental Ties via	Expressive Ties via
Cohort Code Review	*Technical assistance* (mentees)	*Positive emotion* (mentees)
Virtual Group Meetings	*Technical assistance* (developers)	*Personal comments* (developers)
F2F Meetings	*Task clarifications* (developers)	*Appreciation* (users)

These meetings facilitated the formation of expressive and instrumental ties. The informal nature of the practice provided occasions for students to form expressive ties that acted as conduits for *personal comments*, communications that were off-topic or not directly related to project tasks (M3, S10). These communications occurred in preparation for and following meetings. For instance, students and developers exchanged small talk about holidays:

Developer: happy heroes day :)
Student: thanks man. you too :D

They also shared personal information (S4, M7). For instance, after a group meeting, a developer offered time management advice to a student, and opened up about his personal work situation:

Developer: sleep is important too - and timezones are spoiling the fun a bit ;) I do hope the working late gets a bit better in the next weeks
Student: How? Ohh, working late in office
Developer: yep, if the release I'm working on if tested and rolled out to production, things hopefully get back to normal :)
Student: hmm, it's ok, All the best :)

Students participating in group meetings also formed instrumental ties with others via technical assistance. These meetings had a lower start-up cost [41] compared with code review or sending e-mail. As a mentor told us: *"Like email just takes too long to write, craft and there's a lot of back and forth that's not necessary." (M1)*

Often, students included a brief description and snippets of code needed to illustrate problems. The brevity and contextual information may have helped to lower the understanding cost [41] for recipients, who often were able to quickly provide help. For instance, when it was their turn to speak in the meeting, one student said: *"I get [link] this error now to restart the service."*

A few moments after seeing this, a developer who was present in the chat room offered a potential solution: *"I think you might also be having issues because you're using the Tomcat Monitor as well as the Windows server…it's probably better to just use one. Anyway, the problem is that your service isn't starting, so you need look at the event log to find out why."*

3) Face-to-Face Organization-Wide Meetings: In some cases, e-mentoring is augmented with brief face-to-face meetings because it can change the interactions mentors and

mentees have, and deepen the bonds among them [18]. Five students (S4, S5, S8, S9, S11) met their mentors and other developers and users in their organizations at community face-to-face meetings. These events tended to happen in the months prior to GSoC. Organizations invited accepted students to meet community members, get feedback on their project ideas, and present prototypes of their projects.

Students and mentors did not use the meetings to rapidly advance development as studies of the use of face-to-face meetings in distributed work might suggest [43], [44]. Rather the meetings bookended students' projects, giving them extra preparation to hit the ground running during coding, and appreciation for completed projects. These meetings facilitated the formation of expressive and instrumental ties.

Participating in these meetings gave students working on back end projects (S4, S5, S8, S9) opportunities to form expressive ties via appreciation that were otherwise lacking (see Table III). For instance, one student we spoke with met her organization's administrator, the person who oversees the overall progress of a mentoring organization and its students, at a reunion event: *"I met our community head, so that was a really nice experience... he works in U.S. Army and he shared a bit on like how people react to our software and how they appreciate it while using it..." (S4)*

Students participating in face-to-face meetings also formed instrumental ties. An interesting difference in face-to-face meetings, compared with cohort code reviews and group meetings, was that students formed instrumental ties with users via task clarifications. In particular, face-to-face meetings were good opportunities to meet users and understand their needs. As one student told us, he was able to find out about specific problems users faced, and worked with developers to formulate promising solutions: *"For instance sometimes they [users] have layers and maps that takes a lot of time to process and when it's processing the user can't do anything in the program, so it's very good to have the opportunity to put some algorithm to run and keep using the program." (S11)*

Although these meetings were not primarily focused on advancing technical work, they provided occasions for students to get core developers' time, and dedicate that time to technical assistance and problem solving (S5, S6, S8, S9). These meetings also had low formulation and start-up costs [41] that students used to their advantage. Students cited the speed of interactions as beneficial compared with e-mail or IRC (S5, S9, S11). As one student said: *"I got to sit down with [the lead developer] and just do a lot of back and forth questioning on why is this this way, what are we doing here, what does this mean for the project as a whole, a lot of these different things. Very, very distinctly we had one of the days for - it was around the lunch hour we basically sat down for probably a good 45 minutes and I would just fire question after question and then he explained a bunch of stuff like that." (S9)*

V. Discussion

In this study we aimed to understand how characteristics of a FOSS development task influence the kinds of interactions mentees completing that task have, and thus the technical and social benefits they receive. We found that front end tasks with interdependent code structures involve the creation of balance of instrumental ties and expressive ties. In contrast, back end tasks with modular code structures involve formation of instrumental ties but relatively few expressive ties; practices like cohort code review can boost the formation of expressive ties under these conditions.

The present study makes two primary contributions. Firstly, by showing how technical and social task characteristics impact the content that flows through interactions mentees have, we establish a link between the structure of an e-mentoring task and the benefits mentees can expect to receive. We therefore extend literature on e-mentoring that had not previously considered task definition as an important design consideration for e-mentoring program schemes. Second, by considering effects of task definition on tie content, we extend literature on software as a socio-technical system that previously had only looked at the effects of technical characteristics on tie structure. We bring attention to other key technical factors like the visibility of the project to end users, as well as social processes, namely cohort code review, virtual group meetings, and face-to-face meetings.

A. Implications for Practice

Our results have several implications for designing formal FOSS e-mentoring programs. Firstly, our results bring attention to the importance of structuring the mentee's task to bring important learning and interpersonal benefits. With respect to learning, we found evidence that mentees may want to tailor their experience (S8, S9, S11, S12), e.g., *"I wanted to have a community that was big enough so that I could go to them for questions and have the community itself provide some feedback on my work." (S12)* There may be other students who want to tailor their experience, but are not aware that they can do so.

The mentee's desire to customize their experience should be viewed in light of the goals and needs of the FOSS community in which they intend to participate. Much of the time, mentee tasks have a relationship to the codebase, be the part of ongoing work (S12) or on the community's roadmap to be completed at a later date. This may make it impractical, for instance, to modify a task so that it is front end rather than back end, interdependent rather than modular to fit the desires of the mentee. For example, if a mentee wants to work on a back end task because of the knowledge they will gain about the inner workings of the codebase but is concerned about not being able to develop useful connections to the community, it may be helpful for the mentee and mentor to think about how they can provide social interactions around the task. One of many possibilities would be scheduling streaming video presentations whenever mentees reach a milestone, where they report on what they have done and what they plan to do next. Their mentors, other mentors and their students, and eventual users of the software could watch the stream live and comment as the presentation unfolds. The presentations could be archived on a community forum so that anyone unable to

watch live could still see the progress being made and leave comments. Hopefully events like this provide occasions for information exchange, feedback, and praise.

Our findings also point to implications for technology support that should be in place for assisting e-mentoring. Social coding environments and synchronous chat performed many of the needed functions, supporting mentees in asking questions, getting feedback directly on their code, sharing updates with their mentors and other community members, and building relationships through informal interactions. An additional way that technology could support the experience is to assist mentors in recognizing when mentees become stuck and need help. We observed that in some cases, lapses in mentee-mentor communication would occur, and the mentor would not find out about it until much later (M7, M9, S15). One possibility might be to augment the interface of social coding tools with a threaded "question window view" where questions are primary entities. When a mentee gets stuck, they can create a question which then appears in the mentor's interface as a notification. Tools like Ateleier [45] provide features that resemble this. Another possibility that might be worth exploring is instrumenting code editors to detect inactivity or high churn, and alert mentees to the area of code that the mentee is struggling with. In this way, questions might be more actionable and urgent, with issues solved more swiftly.

B. Limitations and Future Work

One limitation of this study is its generalizability, as is common with qualitative studies of this kind. GSoC is unusual in that mentoring occurs around a single primary task, while other forms of mentoring may involve a series of interrelated or loosely related tasks, with the mentoring relationship lasting for a longer period of time. The benefit of investigating a single task, however, was that it allowed us to isolate effects of different types of tasks on the formation of instrumental and expressive ties, since each mentee had just one over the summer. If we had considered multiple tasks being performed by mentees in our analysis, it would have been difficult to tease apart the impacts of one task from another. Focusing on how to construct a portfolio of tasks that can better achieve mentorship goals is an important next step.

GSoC mentoring does not occur in an organizational context, where very often the goal is to retain skilled employees and develop strong leaders within an organization. There is also no expectation that mentees will continue to remain in a long-term relationship with their mentors. GSoC e-mentoring has more of a standalone flavor, where mentees are outsiders looking for opportunities to prepare themselves for careers elsewhere. As such, the GSoC experience is much more typical of other e-mentoring programs, which tout the benefits of overcoming time and distance constraints, access to a large numbers of mentors and mentees, and informal, spontaneous discussions [6]. We suggest, therefore, restricting generalizations made from this study to online mentoring programs.

We chose to study mentees working on scientific software, which may be distinct from other kinds of FOSS. For instance, scientific software developers have a different set of incentives from many other kinds of communities [38], [39]. Members of these communities may instead focus on establishing their reputation rather than guiding mentees [46]. It would be interesting for future research to explore the kinds of ties developed in a broader survey of the FOSS community. Other research methods that do not rely on self-reports of interactions, such as archival analysis of rich mailing list archives, could further serve to validate our findings.

VI. Conclusion

This study aimed to explore how e-mentoring task selection impacts the interactions that mentees have, and the technical and social benefits they can expect to receive. Using research methods that offer rich, qualitative data, we found that front end, interdependent projects facilitate the simultaneous formation of social ties important for technical mastery and interpersonal connections, while back end, modular projects primarily facilitate the formation of ties important for technical mastery. Work practices that create the opportunity for unstructured contact between mentees and community members facilitate the formation of ties important for interpersonal connections. Examples are cohort code reviews, virtual group meetings, and organization-wide face-to-face meetings. Our work contributes both a practical dimension in advice for structuring software engineering e-mentoring program schemes, as well as in expanding our understanding of how software as a socio-technical system impacts the content flowing through a social network, not just the structure.

Acknowledgments

This research was supported in part by National Science Foundation awards 1064209, 1111750, 0943168, 1322278, and 1546393, the Alfred P. Sloan Foundation, and the Google Open Source Programs Office. Thanks also to our participants, and to our anonymous reviewers for their comments on an earlier draft of this paper.

References

[1] M. W. Lipsey and D. B. Wilson, "The Efficacy of Psychological, Educational, and Behavioral Treatment: Confirmation From Meta-Analysis," *American Psychologist*, vol. 48, no. 12, pp. 1181–1209, 1993.

[2] B. S. Bloom, "The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring," *Educational Researcher*, vol. 13, no. 6, pp. 4–16, 1984.

[3] L. Akin and J. Hilbun, "E-Mentoring in Three Voices," *Online Journal of Distance Learning Administration*, vol. 10, no. 1, 2007.

[4] J. E. Girves, Y. Zepeda, and J. K. Gwathmey, "Mentoring in a Post-Affirmative Action World," *Journal of Social Issues*, vol. 61, no. 3, pp. 449–479, 2005.

[5] D. L. DuBois, N. Portillo, J. E. Rhodes, N. Silverthorn, and J. C. Valentine, "How Effective Are Mentoring Programs for Youth? A Systematic Assessment of the Evidence," *Psychological Science in the Public Interest*, vol. 12, no. 2, pp. 57–91, 2011. [Online]. Available: http://psi.sagepub.com/lookup/doi/10.1177/1529100611414806

[6] H. Stoeger, X. Duan, S. Schirner, T. Greindl, and A. Ziegler, "The effectiveness of a one-year online mentoring program for girls in STEM," *Computers and Education*, vol. 69, pp. 408–418, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.compedu.2013.07.032

[7] J. McCarthy, "International design collaboration and mentoring for tertiary students through Facebook," *Australasian Journal of Educational Technology*, vol. 28, no. 5, pp. 755–775, 2012.

[8] A. Collins, J. S. Brown, and S. E. Newman, "Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics," *Knowing, learning, and instruction: Eassays in honor of Robert Glaser*, vol. 18, pp. 32–42, 1989.

[9] S. A. Ambrose, M. W. Bridges, M. Dipietro, M. C. Lovett, and M. K. Norman, *How learning works: Seven research-based principles for smart teaching*. John Wiley & Sons, 2010.

[10] Boz, "Facebook Engineering Bootcamp," 2009. [Online]. Available: https://www.facebook.com/notes/facebook-engineering/facebook-engineering-bootcamp/177577963919/

[11] M. M. Lombardi, "Authentic Learning for the 21st Century: An Overview," *Educause*, 2007.

[12] A. A. Friedman, M. Zibit, and M. Coote, "Telementoring as a Collaborative Agent for Change," *The Journal of Technology, Learning and Assessment*, vol. 3, no. 1, 2004. [Online]. Available: http://ejournals.bc.edu/ojs/index.php/jtla/article/view/1654

[13] P. B. Single and R. M. Single, "E-mentoring for social equity: review of research to inform program development," *Mentoring & Tutoring*, vol. 13, no. 2, pp. 301–320, 2005.

[14] A. Haas, C. Tulley, and K. Blair, "Mentors versus masters: Women's and girls' narratives of (re)negotiation in web-based writing spaces," *Computers and Composition*, vol. 19, no. 3, pp. 231–249, 2002.

[15] R. Brown and S. Dexter, "E-Mentors: Connecting Caring Adults and Kids Through E-mail," vol. 46, no. 6, pp. 60–63, 2002.

[16] Google, "What is Google Summer of Code?" [Online]. Available: http://write.flossmanuals.net/gsocstudentguide/what-is-google-summer-of-code/

[17] J. Headlam-Wells, J. Gosland, and J. Craig, "There's magic in the web: e-mentoring for women's career development," *Career Development International*, vol. 10, no. 6/7, pp. 444–459, 2005.

[18] C.-N. Shpigelman, P. L. (Tamar) Weiss, and S. Reiter, "E-Mentoring for All," *Computers in Human Behavior*, vol. 25, no. 4, pp. 919–928, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747563209000405

[19] L. T. Eby and A. Lockwood, "Protégés' and mentors' reactions to participating in formal mentoring programs: A qualitative investigation," *Journal of Vocational Behavior*, vol. 67, no. 3, pp. 441–458, 2005.

[20] J. Lave and E. Wenger, *Situated learning: Legitimate peripheral participation*. Cambridge University Press, 1991.

[21] A. M. Young and P. L. Perrewe, "What Did You Expect? An Examination of Career-Related Support and Social Support Among Mentors and Proteges," *Journal of Management*, vol. 26, no. 4, pp. 611–632, 2000. [Online]. Available: http://jom.sagepub.com/content/26/4/611.short

[22] NSF, "Shaping the future: New expectations for undergraduate education in science, mathematics, engineering, and technology," National Science Foundation, Tech. Rep., 1996.

[23] P. B. Single and C. B. Muller, "When Email and Mentoring Unite: The Implementation of a Nationwide Electronic Mentoring Program," in *Creating Mentoring and Coaching Programs*, L. Stromei, Ed. Alexandria, VA: American Society for Training and Development, 2001, no. 1, pp. 107–122.

[24] L. L. Bierema and S. B. Merriam, "E-mentoring: Using Computer Mediated Communication To Enhance the Mentoring Process." *Innovative Higher Education*, vol. 26, no. 3, pp. 211–227, 2002.

[25] G. Salmon, A. Things, G. Salmon, B. Distance, M. Zoos, L. Futures, and S. Queensland, *E-moderating: The Key to Teaching and Learning Online*, 2nd ed. London: RoutledgeFalmer, 2004.

[26] L. L. Bierema and J. R. Hill, "Virtual Mentoring and HRD," *Advances in Developing Human Resources*, vol. 7, no. 4, pp. 556–568, 2005.

[27] D. J. McDowell, M. Kim, R. O'Neil, and R. D. Parke, "Children's emotional regulation and social competence in middle childhood: The role of maternal and paternal interactive style," *Marriage & Family Review*, vol. 34, no. 3-4, pp. 345–364, 2002.

[28] R. F. Baumeister, E. Bratslavsky, C. Finkenauer, and K. D. Vohs, "Bad Is Stronger Than Good," *Review of General Psychology*, vol. 5, no. 4, pp. 323–370, 2001.

[29] D. J. Brass, "Being in the Right Place: A Structural Analysis of Individual Influence in an Organization," *Administrative Science Quarterly*, vol. 29, no. 4, pp. 518–539, 1984.

[30] P. Balkundi and D. A. Harrison, "Ties, Leaders, and Time in Teams: Strong Inference About the Effects of Network Structure on Team Viability," *Academy of Management Journal*, vol. 49, no. 1, pp. 49–68, 2006.

[31] J. Scott, *Social Network Analysis*, 3rd ed., K. Metzler, Ed. Thousand Oaks, CA: SAGE Publications, Inc., 2012.

[32] J. R. Lincoln and J. Miller, "Work and Friendship Ties in Organizations: A Comparative Analysis of Relation Networks," *Administrative Science Quarterly*, vol. 24, no. 2, pp. 181–199, 1979.

[33] H. Ibarra, "Personal Networks of Women and Minorities in Management: A Conceptual Framework," *Academy of Management Review*, vol. 18, no. 1, pp. 56–87, 1993. [Online]. Available: http://amr.aom.org/cgi/doi/10.5465/AMR.1993.3997507

[34] E. W. Morrison, "Newcomers' Relationships: The Role of Social Network Ties During Socialization," *Academy of Management Journal*, vol. 45, no. 6, pp. 1149–1160, 2002.

[35] P. S. Adler and S.-w. Kwon, "Social Capital: Prospects for a New Concept," *Academy of Management Review*, vol. 27, no. 1, pp. 17–40, 2002.

[36] M. Cataldo and J. D. Herbsleb, "Communication Networks in Geographically Distributed Software Development," in *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, vol. 18, no. 4. ACM, 2008, pp. 579–588. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1460563.1460654

[37] M. Zhou, A. Mockus, and D. Weiss, "Learning in offshored and legacy software projects: How product structure shapes organization," in *ICSE Workshop on Socio-Technical Congrunce*, 2009. [Online]. Available: http://mockus.org/papers/invconway.pdf

[38] J. Howison and J. D. Herbsleb, "Scientific Software Production: Incentives and Collaboration," in *Proceedings of the ACM 2011 conference on Computer supported cooperative work*. ACM, 2011, pp. 513–522.

[39] ——, "Incentives and Integration in Scientific Software Production," in *Proceedings of the 2013 conference on Computer supported cooperative work*. New York, New York, USA: ACM, 2013, pp. 459–470. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2441776.2441828

[40] J. Corbin and A. Strauss, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 4th ed. Thousand Oaks, CA: SAGE Publications, Inc., 2014.

[41] H. H. Clark and S. E. Brennan, "Grounding in Communication," in *Perspectives on Socially Shared Cognition*, L. B. Resnick, J. M. Levine, and S. D. Teasley, Eds. Washington, DC: APA, 1991, pp. 127–149.

[42] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository," in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. ACM, 2012, pp. 1277–1286.

[43] S. Teasley, L. Covi, M. Krishnan, and J. S. Olson, "How Does Radical Collocation Help a Team Succeed?" in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 2000, pp. 339–346. [Online]. Available: http://dl.acm.org/citation.cfm?id=359005

[44] E. H. Trainer, A. Kalyanasundaram, C. Chaihirunkarn, and J. D. Herbsleb, "How to Hackathon: Socio-technical Tradeoffs in Brief, Intensive Collocation," in *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. ACM, 2016, pp. 1118–1130.

[45] R. Suzuki, N. Salehi, M. S. Lam, J. C. Marroquin, and M. S. Bernstein, "Atelier: Repurposing Expert Crowdsourcing Tasks as Micro-internships," *Chi 2016*, 2016.

[46] D. R. Musicant, Y. Ren, J. A. Johnson, and J. Riedl, "Mentoring in Wikipedia: A Clash of Cultures," in *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*. ACM, 2011, pp. 173–182.