Mutual Assessment in the Social Programmer Ecosystem: An Empirical Investigation of Developer Profile Aggregators

Leif Singer[1], **Fernando Figueira Filho**[2], **Brendan Cleary**[3], **Christoph Treude**[3], **Margaret-Anne Storey**[3], **Kurt Schneider**[1]
[1]Leibniz Universität Hannover, Hannover, Germany
[2]Universidade Federal do Rio Grande do Norte, Natal, Brazil
[3]University of Victoria, Victoria, Canada
leif.singer@inf.uni-hannover.de, fernando@figueirafilho.com, bcleary@uvic.ca, ctreude@uvic.ca, mstorey@uvic.ca, kurt.schneider@inf.uni-hannover.de

ABSTRACT
The multitude of social media channels that programmers can use to participate in software development has given rise to online developer profiles that aggregate activity across many services. Studying members of such developer profile aggregators, we found an ecosystem that revolves around the *social programmer*. Developers are assessing each other to evaluate whether other developers are interesting, worth following, or worth collaborating with. They are self-conscious about being assessed, and thus manage their public images. They value passion for software development, new technologies, and learning. Some recruiters participate in the ecosystem and use it to find candidates for hiring; other recruiters struggle with the interpretation of signals and issues of trust. This mutual assessment is changing how software engineers collaborate and how they advance their skills.

Author Keywords
Virtual Communities; Social Media; Software Development; Social Code Sharing; Reputation; Motivation; Gamification

ACM Classification Keywords
H.1.2. User/Machine Systems: Human factors

General Terms
Human Factors; Management.

INTRODUCTION
The open source and free software movements have altered the way software is built by extending the development process to large numbers of volunteers, often without explicit design or plans [16]. More recently, the advance of social media has changed the way developers collaborate, how they communicate, how they learn, and how they become informed about new technologies [24]. Social media provides new ways for developers to engage in *communities of practice* [28], to articulate their interests across communities, and to follow other developers' activities online.

Collaboration-oriented websites, such as GitHub and Stack Overflow, integrate social features to expose information about developers and their activities across open source projects [7]. These social features create new ways for developers to culturally identify with their communities, and enable new social relationships between developers and organizations within an interconnected ecosystem.

Masterbranch[1] and Coderwall[2] are examples of websites specifically created for the self-display of software developers, aggregating data from other sites such as GitHub and Stack Overflow[3]. The increased exposure generated by these sites has the potential to change which communities software developers engage in, how they interact with others, and how they prioritize their tasks.

This influences how software developers manage their reputation and how other stakeholders engage with developers. Developers use social media to connect with their communities and to monitor, publicize, and grow their skill sets. Social media are connecting like-minded developers, resulting in new social ties that foster collaboration and can encourage entrepreneurship at international scale. Companies, too, are starting to take into account such public profiles when reviewing candidates.

Understanding this *social programmer ecosystem* can give us insights into how social media mechanisms play a role in software engineering skill development, hiring practices, community formation and how social media may influence the quality of software being developed in these communities. For example, social media can be used to motivate developers to engage in software engineering practices more rigorously [21]. Companies and recruiters would benefit from understanding the behaviors and attitudes of their candidates.

This paper investigates how social media, specifically tailored to software developers, can influence the varied and many

[1]`http://masterbranch.com`
[2]`http://coderwall.com`
[3]`https://masterbranch.com/html/about.html`

stakeholders in software engineering communities. By using Masterbranch and Coderwall as an entry point, we surveyed and interviewed participants and bystanders of the social programmer ecosystem. This allowed us to find the different actors' motivations and strategies for participating in this ecosystem.

In this paper, we use the following terminology. A *profile* is a webpage that contains information about a user of the associated website. Profiles play a part in managing one's *public image*, that is, the way one is perceived publicly. *Social network sites* allow their users to create a profile, to connect with each other, and to inspect each other's connections with other users of the site [5] — examples are LinkedIn, Twitter, and GitHub. Masterbranch and Coderwall are *developer profile aggregators*, as they create profiles out of several existing profiles and activities on other sites for a single user.

This paper is structured as follows. In the following section, we discuss recent trends in social software development and the social media infrastructure underpinning those trends. Then we describe a study we conducted to investigate how software developers and other actors are involved and influenced by their participation in the social programmer ecosystem. Finally, we present our findings and analysis, and draw conclusions about how our findings may impact software development practice and research.

BACKGROUND

This section gives an overview of existing research on the use of social media in software development. Following this, we introduce the two websites that we investigated as our window into the social programmer ecosystem.

Social Media in Software Development

Social media is radically changing how software developers communicate and coordinate software development activities online [2]. While the Internet as a communications platform has long been used by developers to talk about and develop software, the advance of social media has introduced new mechanisms that enable large communities of developers to share knowledge, to share code, and to collaboratively create software in a manner that is more public and traceable than previously possible.

Historically, wikis [8] and blogs [18, 17] were the first social media mechanisms used by software developers, utilized mostly in the areas of requirements engineering and documentation, and to communicate high-level concepts [14, 1, 24]. Microblogs, such as Twitter, play a role in conversation and information sharing between software developers [3]. Tags, which became popular in general social media first, now help software developers communicate their concerns in task management [27] and add semantic information to source code [23]. On a larger scale, Stack Overflow and GitHub are two examples of how this kind of social media infrastructure is having an influence on software development practices and software developers.

Stack Overflow[4] is a website that facilitates the exchange of knowledge between software developers. Users post and answer questions related to development, and may comment and rate both questions and answers. The site uses *gamification* concepts — *"the use of game design elements in non-game contexts"* [9] — to encourage and reward community participation. For example, users receive points for posting questions and providing answers, and win "badges" for specific services or contributions to the community. Since its inception in 2008, more than 3.1 million questions have been asked on Stack Overflow, nearly 6.2 million answers have been provided, and over 12 million[5] comments have been submitted — all contributing to a large repository of software development knowledge. A question asked on Stack Overflow has a median answer time of 11 minutes [15]. Member profiles on Stack Overflow contain not only the number of points and badges received for activity on the website, but also a complete record of every question asked and answer provided along with the topics or tags in which the user was most active.

GitHub[6] is a website that allows developers to host their software project repositories using the popular Git revision control system. Since its launch in April 2008, the site has become one of the most popular source code hosting services with over 2 million repositories and over 1 million users[7]. In addition to providing revision control, GitHub also acts as a social network site that enables developers to connect and collaborate with each other. Through GitHub, developers can search for software projects that they are interested in, easily "fork" those projects to make their own contributions, and follow the work of others. The site organizes software repositories by software developer, rather than by project, showing a list of each developer's repositories and their activity across GitHub in a news feed. For a developer, this effectively makes their GitHub profile an easily accessible public portfolio of their open source development activity.

Social media systems such as Stack Overflow and Github are rapidly changing how developers advertise their skills and how they manage their time learning and programming. These types of websites not only provide them with a means for exchanging information, but also enable a level of transparency never seen before in software development. The online profiles of software developers, containing code, questions, and public communication, are available to the whole world for review.

Profiles in Online Communities

The importance of public user profiles for online communities and how people choose to manage their profiles in those communities is well established. One of the most extensively studied social media communities is Wikipedia [13, 12, 19, 11]. Since its inception in 2001, the online encyclopedia has come to be a reference model for how an online community can organize, collaborate, share, and create. Online communities such as Wikipedia are by definition open collaborative

[4] `http://stackoverflow.com`

[5] `http://data.stackexchange.com`

[6] `http://github.com`

[7] `http://github.com/blog/841-those-are-some-big-numbers`

spaces that allow virtually anybody to contribute. Because of this inherent openness, the quality, accountability, and trustworthiness of contributions is often suspect. Therefore, it is important that a community be able to quickly and adequately evaluate a user's contributions to that community.

Several authors have discussed the issue of trust in online communities and how trust issues can be mitigated through the application of theories of social translucence and social transparency [10, 25, 26]. In their work on social transparency, Stuart et al. identify three social dimensions of information exchange: identity transparency (visibility of the identity of an actor); content transparency (visibility of the origin and history of actions taken on information); and interaction transparency (the awareness of actors of each other's participation in an information exchange) [25].

In online communities, different levels of transparency cause different behaviors. For example, higher identity transparency increases actor accountability: members are more likely to act in an accountable manner if their profile is available for other community members to review. Also, members can better assess the quality of information based on reputational accountability of the source. This effect has been shown by Hess and Stein in a study on Wikipedia's "featured articles" (articles voted by the community to be of very high quality) [22]. They found that articles with contributions from higher reputation authors — as judged from their public profiles — were more likely to become featured articles. However, a negative consequence of higher identity transparency could be that creativity suffers due to members not wishing to contribute information that, while potentially valuable to the community, might negatively affect their reputation.

Dabbish et al. investigate how software developers manage their online profiles by studying GitHub users [7]. They found that while explicit self-promotion was frowned upon, users were actively managing their public image and that users believed visibility to be important for the success of an open source project. Watching and being watched also have benefits and requirements. Users said that having watchers was a motivation to continue making contributions. Also, they were more conscious of the quality of their contributions when a project had more watchers.

Developer Profile Management

A recent trend in social media for software development is the emergence of profile aggregation sites for developers, such as Masterbranch and Coderwall. These websites are specifically tailored to software developers and aggregate developers' activities from across several other websites, providing a combined social programmer meta-profile. Contrary to more established sites, such as ohloh[8], Masterbranch and Coderwall focus on the developer rather than on individual projects.

Both sites provide public developer profiles that are mostly generated from activities on social code sharing sites such as GitHub, BitBucket[9], or SourceForge[10]. Other sites, such as

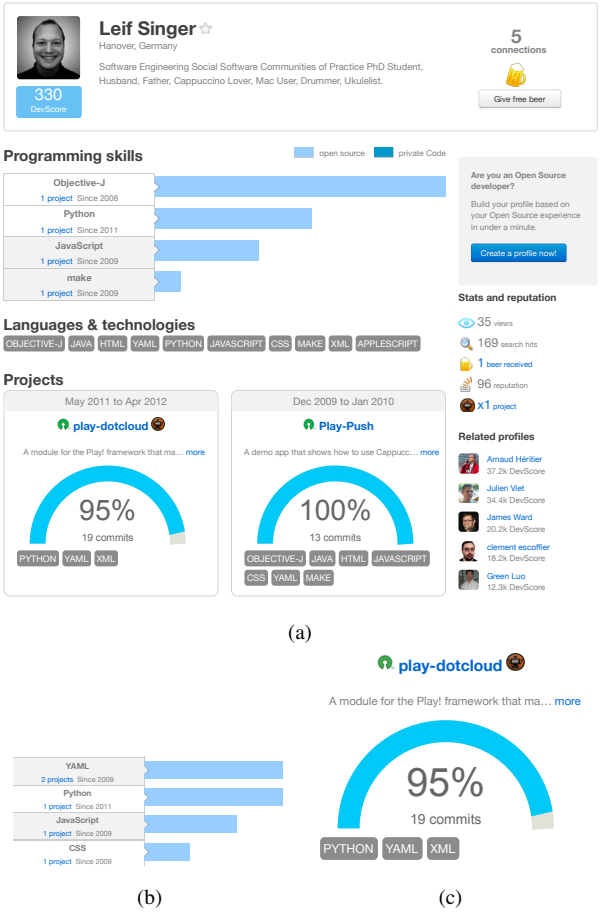


Figure 1. *Elements of a Masterbranch profile: (a) the profile itself; (b) programming skills; (c) details for a project.*

LinkedIn[11] and Stack Overflow, are also supported. The customers of both sites are companies that are looking to hire software developers, for which the sites provide them with special access to their databases of developer profiles. According to their operators, Masterbranch and Coderwall both aim to make it easier for their customers to find candidates suitable for hiring. Customers of Masterbranch so far are mainly web-focused companies of all sizes, most of them younger than 10 years, and some of them very well-known.

Masterbranch

Masterbranch was founded in 2009 by Ignacio Andreu, Juan Luis Belmonte, and Vanessa Ramos. In 2011, the creators started growing it into a community for software developers. As of February 20th 2012, more than 9,000 users had registered with the site.

Fig. 1(a) shows a screenshot of a developer profile[12]. On top, it displays the name, location, and image of the user, as well as the *DevScore* (a value calculated from the developer's activities, such as commits to projects). To the right, a button

[8] `http://ohloh.net`
[9] `http://bitbucket.org/`
[10] `http://sourceforge.net/`
[11] `http://linkedin.com`
[12] Taken from `http://masterbranch.com/lsinger`

allows users to *give free beer* to the developer — a symbolic act of endorsement.

A table generated from the user's repositories displays the distribution of programming languages across these projects (see Fig. 1(b)). Next, the profile lists projects the developer has worked on. For each project, the name, duration, description, and the programming languages used are displayed (Fig. 1(c)). The blue arcs indicate the percentage of commits the user contributed to a project.

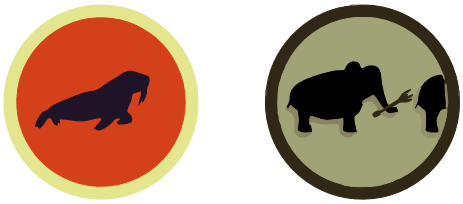
Masterbranch awards *Most Valuable Programmer* (MVP) achievement badges to their users. Each week and for each project known to the site, the most active committer of the project earns the badge.

In addition to developer and project profiles, the site randomly displays some of the most active developers on its front page. An ordered list of the 95 most active developers acts as a simple leaderboard.

Coderwall

Coderwall was founded in 2011 by Matthew Deiters. As of March 1st 2012, more than 15,000 users had registered with the site.

Similar to Masterbranch, Coderwall analyzes the repositories of developers on social code sharing sites. It awards achievement badges to developers when certain conditions are met. For example, if a developer uses at least four different programming languages in the repositories she owns, Coderwall will award her the *Walrus* achievement badge (Fig. 2(a)). The *Forked* achievement is awarded if someone else forked — that is, made their own branch of — a developer's project (Fig. 2(b)).



(a) (b)

Figure 2. *Coderwall's* Walrus *(left) and* Forked *(right) achievement badges.*

Fig. 3(a) shows a screenshot of a developer profile[13]. On top, it displays the name, current company, location, and image of the user. Below this header, there is a timeline that chronologically lists events regarding that developer: when they earned which achievement, when they gave a talk, and others.

The right side of the page displays the developer's skills. A button offers to *endorse* the developer — a low-effort mechanism that, similar to Facebook's *like*, might signal approval (Fig. 3(b)). Next to the button, the number of endorsements the developer has received is displayed. Finally, the profile page lists the achievements earned by the developer

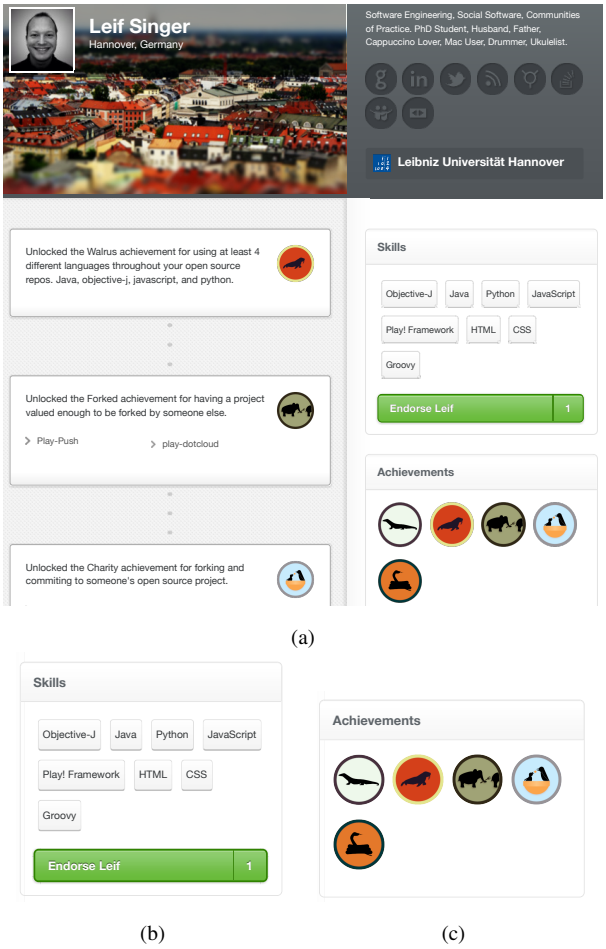


Figure 3. *Elements of a Coderwall profile: (a) the profile itself; (b) programming skills and endorsements; (c) achievements awarded.*

(Fig. 3(c)). During our study, Coderwall added a list of people the developer is connected with on Twitter (not pictured).

Developers registered with Coderwall may join a team, typically named after a company. For each team, the members' contributions are accumulated, resulting in an overall score for the team. This score determines the team's ranking on the Coderwall team leaderboard[14].

STUDY DESIGN

We used a mixed methods approach to better understand how social media, specifically tailored to software developers, can influence the varied stakeholders in software engineering communities.

For the purposes of this paper, we focused on two groups of actors: software developers as well as employers and recruiters who might be using social media to assess potential job candidates and connect with them.

[13]Taken from `http://coderwall.com/lsinger`

[14]`http://coderwall.com/leaderboard`

Research Questions

We designed a set of research questions to help us understand why software developers and other stakeholders participate in the social programmer ecosystem, how they interact, and the impact and challenges they face.

RQ1: Why are software developers and other actors participating in the social programming ecosystem?

Software developers' public display of their development activities is a relatively recent phenomenon enabled by new technology platforms. Our first research question seeks to examine the motives that software developers and other actors have for participating in this emerging ecosystem.

RQ2: How do software developers and other actors interact in the social programmer ecosystem?

Users of developer profile aggregators are constantly signaling information about themselves and their work. This information might be interpreted differently by each group of actors participating in the social programmer ecosystem. We are particularly interested in how software developers might be interacting with each other using profile aggregators, but also in how recruiters and employers might be using information available on those sites to engage and interact with software developers. Research question 2 aims at examining these interactions.

RQ3: What is the impact of participating in the social programmer ecosystem?

Our third question aims to investigate the impact of participation in the ecosystem. We focus on how software developers and other actors might benefit from the environment.

RQ4: What are the risks and challenges faced by participating in the social programmer ecosystem?

Developer profiles might be interpreted differently based on differing organizational and cultural values of the participants of the social programmer ecosystem. Their interpretations might depend on community standards and conventions of practice induced by these standards [4]. Since such differences may give rise to communication problems, we investigate risks and challenges for those people using and participating in the ecosystem.

Procedures

To our knowledge, this paper is the first to report on a study of developer profile aggregators and their role in the social programmer ecosystem. As such, it is of an exploratory nature and is concerned with questions of *why* and *how*. This calls for qualitative methods, which we describe in this section.

In our study, we distributed questionnaires to Masterbranch and Coderwall users and then conducted interviews with some of them. We also interviewed employers and recruiters in order to include their perspectives.

Questionnaires

Our instrument consisted of two web-based questionnaires: one was tailored for Masterbranch users and the other for Coderwall users. The questionnaires were equal in content and order of items, except for the terminology referencing particular features of each site. We pre-tested the questionnaire by distributing it to 10 Coderwall users, of which 5 were sampled at random and 5 were taken from the site's front page. We received responses from 3 users, and 2 of those gave feedback on the survey. After minor changes in both questionnaires, we distributed our survey to a larger sample.

We used several forms of distribution. First, the questionnaires were advertised on Twitter by some of the authors. The operators of Masterbranch and Coderwall supported us by retweeting our invitations. Masterbranch published a blog post, inviting their users to take part in the survey. In this phase, we gathered 28 responses from Codewall users and 9 responses from Masterbranch users. To increase the number of responses, we collected the profile pages and e-mail addresses of 315 random Coderwall members (Masterbranch does not provide random access to member profiles). We then e-mailed those users, inviting them to take our questionnaire. This resulted in another 46 responses.

Each form of distribution contained a URL linking to the respective questionnaire[15]. Both questionnaires were made available online using Google Forms from February 29th to April 19th 2012. Apart from an introductory cover letter, the main content sections were:

Demographics: we asked for information such as age, gender and country. We also measured respondents' professional experience in years and their current employment status, which included their primary responsibility at an organization (if employed), the organization's size and its age.

Site membership: we inquired when the respondent signed up for a profile on the respective site and asked why they did so.

Site features: for each of the features of Coderwall and Masterbranch described earlier, we asked how important they are to the respondent (on a Likert scale). We also asked some specific questions for each feature. For example, we asked respondents whether they care if people look at their profiles and what kind of profiles they are most interested in. We also asked whether they are applying explicit strategies for earning badges such as achievements on Coderwall, or the MVP on Masterbranch. Finally, we inquired about eventual strategies they might be using for earning achievements.

Responders were also asked to optionally submit their e-mail addresses if they agreed to be contacted for an interview with our research team. In total, we received 35 responses with e-mail addresses included, 31 of which came from Coderwall users.

Interviews

To scrutinize the answers from the survey, we invited 35 survey participants to interviews via e-mail. Of these 35, 14 software developers enrolled, 2 of them from the Masterbranch survey. For interviews with recruiters, we used a snowball sampling strategy, which yielded 12 recruiters volunteering.

[15] see **https://bitly.com/yMo22Q** and **https://bitly.com/y85tA0**, respectively

Our strategy involved sending personal messages to our acquaintances and friends working for software companies. Our sample consisted of people we already knew from our personal networks and others referred to by them. Some enrolled after reading a tweet or a blog post we created.

We then conducted semi-structured interviews with software developers and recruiters. We first asked developers about their current job situation — for example, what kind of company they work for or what the team structure is. We then used their answers from the questionnaire as a starting point for deeper inquiry, asking for the reasons and motivations for their behaviors.

In the interviews with recruiters, we asked about their strategies for evaluating candidates for software development positions in industry, especially with regard to the stages in which they would use information from the Web. We presented developer profiles available on Coderwall and Masterbranch to the recruiters, and asked them to think aloud while skimming through the page in order to get their impressions about the various features of the sites.

Interviews were conducted mostly via Skype and were recorded; for 2 of them we used a text chat. In total, we recorded 14 hours and 36 minutes of audio, an average of 33 minutes per voice interview.

Participants

Overall, we received 83 responses to our questionnaires, 74 of which came from Coderwall users. Most respondents were software engineers (68), 14 were team leaders and 1 was a non-technical co-founder of a software company. We interviewed 26 people; among them software engineers, software contractors, and recruiters. Table 1 summarizes our interview participants and lists the identifiers we assigned to them.

Software developers	
D1, D3-D5, D7, D8, D10-D13	employed in a software company
D6, D9, D14	consultant or contractor
D2	unemployed
Recruiting	
R1-R5, R7, R8, R11, R12	for their employer
R6, R9, R10	for customer companies

Table 1. *Summary of interview participants.*

Participants we interviewed as developers start with a *D*, while those we interviewed as recruiters start with an *R*. Note that during the interviews, we noticed that some of the developers were also involved in recruiting practices and we modified our questioning to cover these aspects as well (developers D4, D5, and D10).

Our interviewees had diverse backgrounds. To illustrate this, Table 2 provides a short introduction for some of the participants.

Code	*Background*
D1	Developer at a Web development shop in Norway, mostly Ruby development.
D3	Developer working in an image recognition company from Spain.
D6	Developer and team coordinator in a Linux security and deployment company in the USA.
D10	Developer and team coordinator in a large software company in the USA, managing proprietary and open source efforts.
D11	Java developer for a Polish outsourcing company.
R3	CEO of a local German IT services company.
R5	CTO of a German analytics and research company.
R8	Project manager in a research subsidiary of a large German carmaker.
R9	Employee at a recruiting company based in the UK, focusing on mobile and Web developers.
R12	Hiring manager at a *Big Data* company based in Germany and the USA.

Table 2. *Backgrounds of select interview participants.*

Data Analysis

We used an approach based on grounded theory for data analysis [6]. Questionnaire data was split into two data sets and two of the authors open coded each set independently. Two authors then cooperatively engaged in axial coding our preliminary set of codes and, as a result, 15 categories emerged. We used those categories to code interview data. Then, we transcribed excerpts from the interviews that were related to our research questions. The next phase comprised of selective coding over all extracted quotes (from questionnaire responses and interview transcriptions). During this process, we iterated on our previous code system and identified both the core categories and relationships that would help us answer our research questions.

FINDINGS

This section reports the findings from the surveys and the interviews with developers and recruiters. For brevity, we provide quotes only for some of the findings. The source of each quote is noted in square brackets — *[Dx]* referring to an interview with a developer; *[Rx]* an interview with a recruiter; *[SMx]* and *[SCx]* to survey answers for Masterbranch and Coderwall, respectively.

RQ1: Reasons for Participation

To answer research question 1, we investigated *why* developers and recruiters participate in the social programmer ecosystem.

Developers

Many software developers told us that they are ***curious about technology, passionate to learn, and always trying to improve themselves as developers***. A variation of this was given in 20 of the 83 survey answers as the reason to join one of the developer profile aggregators.

Taking part in the ecosystem allows developers to discover novelty — by joining a developer profile aggregator, publishing code on GitHub, or interacting with others on Twitter. Seeing others do interesting new things with technology ***inspires*** them. A special case are ***high-profile developers*** that are vocal about their technological discoveries, respected in the community, and followed by many.

"On Twitter, I follow a few prominent software developers. For example, Kelly Sommers[16] *from Canada, she's constantly trying new things. I don't think she ever sleeps. She's a great source of inspiration." [D11]*

A reason that was very commonly given for participation in the ecosystem was ***enjoying the interactions*** with other developers: *"this incredible group of fascinating people that get interested in all sorts of things, makes it interesting to talk with them, and work with them" [D1].* This was mentioned in 8 of 14 developer interviews.

Interestingly, very few participants told us that they were looking for ***competition*** with regard to the number of projects, followers, or badges one has on the different social media sites. It was mentioned by 5 developers in the interviews, and only by 3 of 83 survey respondents when asked about achievements. Related to that, there was also the theme of ***pride*** — developers said they liked showing off their achievements and comparing themselves with others. Some specifically mentioned that they were proud of being part of a great team.

The most important reason for enjoying the community was ***recognition*** of others. In the survey, 19 of 83 answers with regard to the *endorsement* features of the websites were about recognition or validation of one's work. Developers like getting respect from their peers — and from random people as well — for the work they are doing. *"I would like to have some recognition from the community [...] [my projects] are fun to me, but if they are only on my hard disk then nobody knows" [D3]*

Similarly, ***helping others*** was also very important for several developers: *"I like to be useful to others, and this is a good way to do that" [D3]* Some explicitly said that ***interacting with, and helping others, motivated them*** to contribute more and to become better developers.

Finally, a few participants mentioned that they were ***pushed into the ecosystem by their peers*** — they had been asked to join a site or a service, and to contribute code of their own. *"I started to publish because my friends told me that [a project] didn't have any [visibility]" [D3]*

[16] `https://twitter.com/kellabyte`

A few developers mentioned that they were striving for visibility because they were ***searching for work***. Taking into account all questions of the survey, 6 of 83 respondents mentioned this. These developers said that they would like to improve their chances of being recognized by recruiters. They believe that the simplifications of experience that the developer profile aggregators provide help non-technical recruiters in assessing developers: *"I love the idea of the badges, it helps communicate to non-technical people (like recruiters) what I know" [SC14]*

As we will see in our findings for research question 4, this is not generally true. Yet, showing off diversity is important for developers, as they believe that recruiters value it as well. The insights for research question 2 will support this finding.

Recruiters

Recruiters have two main reasons for taking interest in the social programmer ecosystem. For one, they see its potential for better assessment of potential candidates. Much information on developers is now publicly available, such as public interaction and engagement. This enables recruiters to find out more about cultural fit, values, and soft skills. For example, they are looking for people who care about their work and those who are ***passionate*** about it. *"I want people who care, and I'm trying to find out whether they do care based on their activity on the Web." [D10]*

The other reason for recruiters to take part is the shortage of software developers to fill open positions. Few are available, and the really good ones are often employed and not looking for a career change. Recruiters want to use the ecosystem to ***actively engage*** with developers they might not be able to sway otherwise.

Related, recruiters are trying to ***speed up finding new candidates***. Many companies desperately need to hire developers quickly — not being able to do so hurts businesses, as it slows down their progress. *"we cannot progress as fast as we would like to. [...] the workload is enormous and that's why we try to find these guys as fast as possible." [R5]*

RQ2: Modes and Terms of Interaction

To answer research question 2, we investigated *how* developers and recruiters interact in the social programmer ecosystem.

Survey Results

Table 3 shows the results regarding the subjective importance of different features of the developer profile aggregators. All 83 respondents were factored in. The questions provided a Likert-type scale ranging from *1 — very important* to *5 — not important at all*. An option reading *"I don't know that feature"* was also available.

The achievements feature, possibly being the most visible, was most important, followed by the endorsement feature. Strikingly, 37 of 83 respondents did not know the "featured developers" on the front pages of the developer profile aggregators. Supported by this additional data, we will now answer research question 2.

	Mean	*Median*	*Don't know*
Achievements	2.5	2.0	1
Endorsements	2.8	3.0	27
Leaderboard	3.3	3.0	15
Featured Developers	3.3	3.0	37

Table 3. *Importance of the features of developer profile aggregators.*

Developers

Many developers participate in the social programmer ecosystem to connect with others — peers, interesting people, or high-profile developers. To be able to do so, they need to ***assess other developers*** first.

As part of this assessment, they are investigating ***what others have created***. Thus, the open source projects of developers mediate relationships in the ecosystem. They allow developers to construct a "coder footprint" of one another. *"[When] I look at repos around this topic [data visualization], I may be interested in seeing the coder footprint of people that work in this area [...] their favorite languages, the topics they write code about, what they work on" [D6]*

Constructing such a footprint not only helps in assessing strangers on the Internet. It is also being used to ***make sense of coworkers*** who might be geographically dispersed. In a similar manner, developers are using ***common interests*** to find interesting people and connect with them. While some developers will interact with those they discover, others choose to ***passively follow*** the activity of developers.

Developers are aware that their peers are assessing them as well. This leads to developers wanting to manage their *"personal brands" [D9]*, that is, consciously constructing a public image of themselves. The social programmer ecosystem is one of ***mutual assessment***.

Developers enjoy and desire ***recognition by peers***, which was mentioned by 11 of 14 interviewed developers: *"there's the social component. So peers — developers, coders — can see that. It gives you a good feeling when others see what I've achieved" [D12]* On the other end of this interaction, developers do recognize and ***acknowledge good work***: *"I knew he worked hard in that area and felt like giving him recognition" [SC58]* This finding may be related to the relative importance of the *endorsement* feature (see Table 3).

Regarding the employment aspect, developers often view recruiters as spammers who send them irrelevant employment offers. Developers have a ***bad perception of recruiters*** and try to avoid them. Instead, they prefer utilizing their personal network and word-of-mouth for finding work. *"I don't like recruiters [...] it's largely who you know" [D5]* Amongst other insights, the following paragraphs will show how recruiters are trying to fight this perception.

Recruiters

From the perspective of recruiters, we found a diverse set of strategies for connecting with interesting software developers. Several of the recruiters we talked to used relatively traditional means: for example, ***generic professional networking sites***, such as LinkedIn or Xing.

Those who were more technical themselves, or even considered themselves or their companies as part of the *in-crowd*, often used their ***personal networks for recruiting***: *"Our lead designer, we hired through referral, basically. Our CEO is friends with [well-known entrepreneur] and [...] our lead designer was the lead designer at [entrepreneur's most successful startup]." [R11]*

Going where the developers are to connect with them — that is, on social code sharing sites or Q&A sites — was considered to be effective by most recruiters we talked to. Some recruiters actually succeeded in doing so, but many were ***struggling with technical culture***, or overcoming the ***bad reputation*** recruiters have with developers. They tried ***using social media*** to appear more authentic and become part of developers' personal networks: *"[...] we use a lot of social media [...] hearing, talking to different developers we already know is another way, recommendations are one of the best ways to find the best developers" [R10]*

When evaluating a developer, recruiters first ***filter by skills***, using skill lists provided by many web-based profiles. *"The most important point here is the tag cloud about his skills" [R5]*. None of the recruiters we talked to told us that they used certifications to assess the skills of developers. When certifications were mentioned by a recruiter, it was to say that they didn't mean much to them.

When these first line filters had narrowed down the candidate pool, recruiters took a more thorough look to find out about the experiences and social skills of a developer. A common baseline check was looking at a developer's ***activity and engagement in open source***. For many recruiters, that would mean profiles on Stack Overflow, GitHub, and Twitter. However, if there was nothing to be found, recruiters said they do not outright reject the candidate — they would assume that this might be a more private person, for example.

If they find something online about developers, recruiters use a diverse set of signals to get a picture of a candidate. Regarding the technical skills and work style of the developers, the more technical recruiters looked at whether they used ***best practices*** adequately, such as testing, version control, or commenting.*"I always look to see if they're writing tests, as a key. If they're not, I don't wanna hire them" [D10]*

Several recruiters used other people's ***endorsements of a developer's code*** as a proxy to assess their technical skills. *"[it's] very helpful if you have a project valued enough to be forked by somebody else, this is quite interesting [...] because it really shows that it has some meaning for other people" [R7]*. They also use these endorsements to assess soft skills, such as communication behavior: *"In the long run, you don't get people to follow your repository by just pushing out code. It means that he's sufficiently good at other things as well,*

such as communication." [R4]

Recruiters try to assess several very intangible attributes of developers using social media and open source activity. 5 of 12 interviewed recruiters mentioned that they were looking for ***diversity in developers***: for example, by looking at the number of different programming languages they are comfortable with. As we questioned this further, two underlying reasons emerged.

For one, they were looking for ***fast learners*** who are adaptable to change. Recruiters mentioned that software development is very fast-paced and technologies come and go. So employees need to be able to react to that with flexibility: *"He's relatively broad. [...] we're looking for people who can get into new concepts quickly. That's what makes a good coder." [R4]*

Secondly, recruiters were often looking for ***passionate developers*** — this was mentioned by 11 of 12 interviewed recruiters. To assess passion, they mostly used social media that were not directly related to code. Many mentioned Twitter as a sign for how interested and engaged a developer was with regard to technology: *"A 9 to 5 developer is not tweeting about the latest stuff that's coming out of the W3C mailing list. A 9 to 5 developer is tweeting a picture of the hamburgers he's frying at 4:30." [D10]* The reason they were looking for passionate developers was one of drive and job satisfaction, both allegedly leading to better results.

Finally, recruiters tried to find candidates who would ***share the company's values***, for example being product-focused: *"we're looking for people who are [...] product-focused, they're gonna talk about things beyond programming. I guess you could say 'well-rounded' [...] they love coding, but it goes deeper than that" [R11]*

RQ3: The Impact of Participation

This section reports what we found with regard to research question 3, which asks about the *impact* of participation in the social programmer ecosystem.

Our interviews revealed that the ***gamification*** employed by features such as Coderwall's achievements is, in fact, at least somewhat effective: *"if I need to make a repository, I'll put it on Github, it'll exist forever, and I can get a badge for it. That's really awesome. [...] It pushed me in the right direction. It forced me to play the game for the right reasons." [D13]* In our interviews, 8 of 14 developers explicitly mentioned that they felt motivated by the developer profile aggregators. This may be related to the high importance given to the achievement feature in Table 3.

A closer look revealed that those features ***lower the barriers for participation*** in developer teams and communities. As a result, opportunities for participation are more visible and tangible: *"I have been trying to get into [the open source] scene for a while, when you see an achievement that is available for a contribution, it is the final nudge to make an actual contribution." [D2]*

Developers also reported other reasons for participation that go beyond ludic motivations and might impact their professional progress. For instance, ***learning new programming languages*** is a consequence of engaging in such an environment: *"I wouldn't think of starting a repository in a specific language just to earn an achievement, but it may push me towards choosing a new language to learn" [D6]*. Also, achievements are seen as a window for ***exploration and experimentation***. When asked about why achievements are important, one respondent said: *"they make me want to achieve something that I didn't know about before or open me up to new ideas." [SC38]*

The impact of such features is not limited to individuals. They ***motivated whole software teams to contribute more***: *"Coderwall has generally upped the game in the office amongst our engineers. It has helped to encourage all of us to publish more code online." [SC51]*

As we saw in the findings for research question 1, developers are interested in novelty. Now we find that their drive for trying new things helps them ***diffuse new ideas***. *"Generally, I sign up for every online service I think I might find interesting. [...] now our entire team is on Coderwall" [SC51]*

Finally, participation also has a concrete impact on recruiting practices. Features on Masterbranch and Coderwall aggregate and quantify developers' activities and artifacts. In doing so, they enable others to ***understand complex attributes*** of developers with less cognitive load. We asked a developer who also turned out to be actively involved in recruiting how fast learning can be identified in developers: *"I used to look for that manually in GitHub repositories, but Coderwall and others make it easier now" [D4]*

RQ4: Risks and Challenges of Participation

Research question 4 is concerned with potential *risks and challenges* that result from participation in the social programmer ecosystem. Backing up the necessity for this inquiry, a developer also greatly involved with recruiting told us: *"It's a new space. I think there's some value to this kind of thing [Coderwall], but I'm not sure where it is yet. Because it's a new idea." [D10]*

Many of the developers we talked to were rather isolated locally, but well-connected with weak ties on a global scale. Even though these were passionate and interested persons, they were not able to gain too much from participation in mutual assessment as implemented by developer profile aggregators: *"[Importance of leaderboard] It's not very important to me now as soon as most of my colleagues are not very interested in open source" [SC24]* They simply ***lack the social connectedness*** that seems to help in adopting such approaches. The relative obscurity of the *leaderboard* feature that can be seen in Table 3 may be related to this finding.

Even though we have found strong preferences for passionate novelty seeking, some developers mentioned that they ***struggle to keep up*** with their fast-paced environment. *"The whole 'what's new today thing' is exhausting" [D1]*

Because developer profile aggregators, social code sharing sites, and social media in general are very public by their

nature, the actual audience of content cannot be determined beforehand. We found two conflicting views regarding this issue.

On the one hand, several developers and recruiters are aware that ***public signals should not receive too much weight*** — they mentioned that the reasons for the existence or absence of such signals can be very different from what people might assume. *"The GitHub repository doesn't show everything though. I wouldn't discard someone with an emptyish GitHub repo. Might just show that most of his/her projects were closed-source. That's why nothing can replace a few e-mails." [D4]*

On the other hand, some participants involved with recruiting said they would be ***irritated by a lack of public activity*** for a developer. One recruiter told us: *"I find it kind of strange if a software developer doesn't use Twitter or GitHub [...] Because git is the factual standard at the moment for version control, yeah? At least for people who try to go with the [latest] technology, the early adopters. [...] we actually want people to be passionate about technology." [R7]* For this recruiter, having adopted Git and being public about it was actually a required sign of passion for technology in a developer.

Non-technical recruiters, however, struggled with the ***interpretation of signals*** from developer profile aggregators and social code sharing sites. *"Well, I don't know all these achievements. They look fun! But they don't really help me." [R12]* As a consequence, these recruiters did not trust the sites and therefore did not use them: *"we don't use those sites [Coderwall, Masterbranch, GitHub, Stack Overflow]. We know too little about them." [R8]*

Of the 12 recruiters we interviewed, 7 had used general social media for candidate assessment before (LinkedIn, Twitter, Google+). 5 had experiences with using developer-specific social media (Stack Overflow, GitHub), and 1 had used a developer profile aggregator before (Masterbranch).

DISCUSSION

In this section, we discuss some of the most important themes that emerged from our study and their potential impact on software engineering practice.

Mutual Assessment

The core category that we found in our study was that of *mutual assessment*. The social programmer ecosystem allows all stakeholders to assess each other using novel reputation signals, or known signals that are being used in novel ways. The aggregation performed by sites such as Masterbranch and Coderwall helps in quantifying the activities of the programmers in the ecosystem, and it also enables the creation of symbols for certain kinds of participation in the ecosystem.

For developers, it appears to be much more important to assess other developers, rather than recruiters or companies. Our findings indicate that developers are in the center of the ecosystem, and that securing employment is only a second priority. For the developers we studied, everything revolves around their passion for technology and learning, as well as being connected to other passionate developers. Thus, they assess each other in terms of the technologies they use and the problems they solve, using each others' "coder footprint" (see below) with the goal of finding new stimuli and collaborators.

Developers use the signals in the ecosystem to assess companies as well: *"I want to gauge the quality of developers in a certain organization, so I can determine if I'm a good fit for a position there."* The presence of a company's developers in the social programmer ecosystem is an indication of that company's philosophy with regard to using the latest technology and participating in open source. While developers do not seem to like recruiters as they are often considered spammers, we found that some recruiters are actively trying to create an authentic identity in the social programmer ecosystem for networking purposes in an attempt to fight this perception.

Passion

The core aim of mutual assessment, we found, was identifying *passion* in others. From the perspective of a company, the active participation of a developer in the ecosystem shows the developer's passion for technology and learning in general.

This passion appears to be the main motivation for programmers to participate in the ecosystem. The participants in our surveys and interviews love programming and creating things. The applications used in the social programmer ecosystem and their features make it easy for them to find interesting potential collaborators as well as interesting new technologies to learn. In that sense, the signals of the ecosystem are a mechanism to reduce uncertainty about technologies as well as other stakeholders.

Developer profile aggregators are a vehicle for programmers to showcase themselves and their achievements. The focus is not on job hunting, but rather on having fun, building something cool, and generally *"being awesome"* — as one of our survey respondents put it. Developers use profile aggregators to show their peers and potential collaborators how passionate they are and that they love what they do. Incidentally, it is this passion that recruiters and companies are looking for as well, in particular passion for learning, using, and mastering new technologies. This passion connects developers with each other, and — to a lesser extent — with recruiters.

Coder Footprint

In several interviews with developers, the interviewees struggled to say exactly what they were looking for in other developers. Some wished to *"get a better picture"*, others wanted to *"get a feel"* for what kind of developer the other person was. They all agreed, however, that multiple factors are important: the developer's technical niche and its popularity in the community; their diversity; their passion for technology; their standing in the community. Quoting one of our interviewees, we call this their *coder footprint*. While sites such as GitHub are also suitable, developer profile aggregators in particular provide a condensed representation of the coder footprint that can be grasped quickly.

Grasping the coder footprint seems to be the primary function of the profile aggregators. Both developers and technical recruiters claimed to be trying to get a feel for what other developers were about. Non-technical recruiters seemingly lacked the technical intuition required and resorted to determining attributes that were more fact-based and easier to interpret without a technical background.

The coder footprint serves an important purpose in the social programmer ecosystem. Developers are driven by a passion for new technologies, new ideas, and learning. The coder footprint allows them to navigate a vast social space much more easily, supporting them in their search for novelty and social connectedness. While sites are still experimenting on how to get the coder footprint right, and several interviewees voiced criticism and concerns regarding the validity of the signals found on developer profile aggregators, the current iteration is an already useful glimpse at what future developers might be using to assess and explore each other.

Diversity

As an essential part of the coder footprint, diversity was considered important by developers and recruiters alike. Developers tried to attain it themselves as part of their personal improvement efforts and also looked for it in other developers. Recruiters used diversity as a signal for fast learners and adaptability, which they deemed very important traits in the fast-paced software development business.

We believe this might be an indicator for a cultural change in software development. Proficiency in a certain niche is no longer enough, as developers are expected to be able to easily adapt to a changing technology landscape. Importantly, this expectation is not only one of companies or recruiters, but also one of developers — they expect it from others and also from themselves.

Several parties might be interested in leveraging this insight. Developers that are aware of it might take it as a motivation to become more diverse themselves. Others might try exploiting the potential for manipulation, for example by creating many repositories containing trivial projects in several programming languages. Recruiters that are not yet proficient in interpreting developer profiles might use it as a guideline in their assessments. Finally, developer training — for example at universities — is already somewhat diverse in that it mostly teaches concepts instead of concrete programming languages. However, these are often restricted to only one or two programming paradigms, such as object-oriented programming. Taking a cue from software development practitioners and companies, universities might want to support their students in attaining a higher level of diversity.

While most participants of our study used diversity in programming languages as a signal for actual diversity and adaptability, there might be other, more appropriate measures for this character trait. A few interviewees stressed the importance of public communication, for example on Twitter, to help in assessing this. Therefore, tag clouds and badges of programming languages might just be a first iteration of a more important idea — simplifying the assessment of diversity in developers.

Impact

The software developers we spoke to viewed participation in social media and mutual developer assessment in a largely positive light. For them, it presents opportunities to learn about new technologies and to connect with other like-minded people who they might not be able to find locally in their own organizations. Several developers also spoke of how participating in the social programmer ecosystem and knowing that other developers were following their activities propelled them to contribute more or better quality code, and to improve themselves as developers. This effect has been shown for GitHub by Dabbish et al. [7].

From a recruiting perspective, the impact is less clear. Some recruiters and companies that are already familiar with the ecosystem are able to parse the activity signals generated by developers to a degree. Others who are outside of the immediate community have more trouble interpreting developer activity in social media and weighing the value of different kinds of activities.

This study explores how a group of early adopter software developers are interacting with new social media platforms. As such, the question arises as to whether and how these trends extrapolate to the rest of the software development community. Software developers outside the early adopters group may find many of the same benefits in using social media as part of their development activities. However, motivations for participation by this group may well differ from those of the early adopters.

There is also the issue of developers feeling compelled to participate in the ecosystem to keep up with expectations of the wider software community. Many developers are not able to participate freely in social media due to their employment circumstances; others may be dissuaded by the potential for public criticism of their work. A future developer landscape that expects a public portfolio of work available for contribution or participation in the community could have the unintended consequence of isolating these developers from the community rather than connecting them to it.

Arguably, recruiters and software companies have the most to gain from increased software developer participation in social media. Coarse-grained quality indicators such as CVs and even professional and academic qualifications have until now been the mainstay of resources for assessing developers' suitability for job opportunities. With more and more developer activity taking place in social media and archived in public portfolios, recruiters and companies will have access to much more active and historical information about candidates than previously available. As interpreting this information might be easier for recruiters with technical backgrounds, this could have an impact on the skills expected from recruiters as well.

LIMITATIONS

As with any research methodology, there are limitations with our choice of research methods.

We conducted surveys and semi-structured interviews, which do not allow us to infer statistical significance of our findings. However, these methods are well-suited to explain how and why the various stakeholders play a role in the ecosystem of social programmers. As this is the first study on the role of developer profile aggregators, we consider these methods and their focus on exploration to be suitable for our purposes.

The responses to our questionnaires and interviews came exclusively from members of the Coderwall and Masterbranch sites, as these were the most visible when we started our research[17]. Therefore, the responses are naturally biased. We found that many participants exhibited traits of the *innovators* adopter category as defined by Rogers [20]: with only few local ties, they were connected globally with many weak ties and used those to bring new ideas and technologies into their own local societies — Rogers calls them *venturesome*. This character trait may have helped with the relatively good response rates for the questionnaires and interviews (e.g., 35 of 83 total respondents offered to do an interview, of which 14 were realized). Even before sending out personal invitations to members of Coderwall, most responses came from members of Coderwall instead of Masterbranch. That might be explained by the relatively high media exposure the site had recently[18].

Unfortunately, we were not able to get in contact with customer companies of Masterbranch or Coderwall that are paying for specialized access to developer profiles for hiring purposes. We believe that these companies would have added another interesting perspective to our research.

Another limitation of this study lies in the relatively low number of interviews conducted. However, we were able to increase the credibility and validity of our findings by triangulating data from interviews and survey responses. Also, the interviewees had a wide range of different backgrounds, allowing us to consider many different perspectives in this work.

The current users of developer profile aggregators are *innovators* and *early adopters*, and thus are not representative of the entire software developmer population. However, they allow us to gain early insights into the role of profile aggregators in the ecosystem of social programmers, and to shed light on the potential impact of these sites for software developers, recruiters, and companies.

Coderwall and Masterbranch are among the first sites to aggregate developer data across various social media developer services. As more individuals start using these sites and other services emerge, additional studies should be conducted to gain further insights into the complex ecosystem of social programmers.

CONCLUSIONS & FUTURE WORK

In this paper, we discussed a group of software developers that are using new social media tools to communicate and share their development activity with fellow developers and other interested actors. These developers, their motivations, and the technology infrastructure they use combine to produce an ecosystem where open collaboration and public sharing of who you are and what you do is ingrained.

Initially, we were motivated to understand why some developers choose to publish and promote their development activity online. In our investigatation, we performed surveys and interviews with software developers and recruiters affiliated with this ecosystem. We identified their motivations for participating, how they interact with other stakeholders, what effect participation has on participants' learning and job prospects, and the risks and challenges facing stakeholders in this ecosystem.

Our findings indicate that the developers that participate in this social programmer ecosystem are motivated by a strong passion for discovery and learning about new software development technologies. To satisfy this compulsion, they explore the technology landscape through the prism of other like-minded developers' activities and coder footprints. They are aware of other developers and actively assess and are assessed by them. This process is facilitated and encouraged by the underlying technology infrastructure of the social programmer ecosystem, which places its emphasis not on software projects, but rather on software developers and their activities across multiple projects and communities. The developers we surveyed and interviewed are at home in this ecosystem, the recruiters less so, but it is becoming an increasingly important aspect of what they do.

This study reports on relatively recent trends in social media and software development and how early adopter software developers are participating and interacting in this ecosystem. As such, our findings have interesting implications for the future of software engineering at large. For companies and recruiters, developer activity archived in social media is a potentially potent resource for assessing developer capability and matching developers to job opportunities. For developers, it represents opportunities for learning and discovery, staying current with trends, and benchmarking against other developers. Yet, it also poses questions about how developers who, by choice or otherwise, choose not to participate in public social media, will be viewed by the rest of the developer community. We plan to investigate these issues in future work.

ACKNOWLEDGMENTS

We thank Vanessa Ramos of Masterbranch and Matthew Deiters of Coderwall for their support in conducting this research. Furthermore, we're grateful to the participants of our survey and interviews.

REFERENCES

1. Ahmadi, N., Jazayeri, M., Lelli, F., and Nesic, S. A survey of social software engineering. In *Automated Software Engineering - Workshops, 2008. ASE Workshops 2008. 23rd IEEE/ACM International Conference on* (sept. 2008), 1 –12.

2. Begel, A., DeLine, R., and Zimmermann, T. Social media for software engineering. In *Proceedings of the FSE/SDP workshop on Future of*

[17] Another recently popular site is **http://geekli.st**.

[18] **http://techcrunch.com/2012/02/27/coderwall-hacker-reputation-system/**

software engineering research, FoSER '10, ACM (New York, NY, USA, 2010), 33–38.

3. Bougie, G., Starke, J., Storey, M.-A., and German, D. M. Towards understanding twitter use in software engineering: preliminary findings, ongoing challenges and future questions. In *Proceedings of the 2nd International Workshop on Web 2.0 for Software Engineering*, Web2SE '11, ACM (New York, NY, USA, 2011), 31–36.
4. Bowker, G., and Star, S. *Sorting things out: classification and its consequences*. The MIT Press, Cambridge, MA, 2000.
5. boyd, d. m., and Ellison, N. B. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication 13*, 1 (2007), 210–230.
6. Corbin, J., and Strauss, A. *Basics of qualitative research: Techniques and procedures for developing grounded theory*, 3rd ed. Sage Publications, 2008.
7. Dabbish, L., Stuart, C., Tsay, J., and Herbsleb, J. 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), 1277–1286.
8. Dagenais, B., and Robillard, M. P. Creating and evolving developer documentation: understanding the decisions of open source contributors. In *Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering*, FSE '10, ACM (New York, NY, USA, 2010), 127–136.
9. Deterding, S., Dixon, D., Khaled, R., and Nacke, L. From game design elements to gamefulness: defining "gamification". In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, MindTrek '11, ACM (New York, NY, USA, 2011), 9–15.
10. Erickson, T., and Kellogg, W. A. Social translucence: an approach to designing systems that support social processes. *ACM Trans. Comput.-Hum. Interact. 7*, 1 (Mar. 2000), 59–83.
11. Gleave, E., Welser, H., Lento, T., and Smith, M. A conceptual and operational definition of 'social role' in online community. In *System Sciences, 2009. HICSS '09. 42nd Hawaii International Conference on* (jan. 2009), 1 –11.
12. Kuznetsov, S. Motivations of contributors to wikipedia. *SIGCAS Comput. Soc. 36*, 2 (June 2006).
13. Lih, A. Wikipedia as participatory journalism: reliable sources? metrics for evaluating collaborative media as a news resource. In *In Proceedings of the 5th International Symposium on Online Journalism* (2004), 16–17.
14. Louridas, P. Using wikis in software development. *Software, IEEE 23*, 2 (march-april 2006), 88 – 91.
15. Mamykina, L., Manoim, B., Mittal, M., Hripcsak, G., and Hartmann, B. Design lessons from the fastest q&a site in the west. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, ACM (2011), 2857–2866.
16. Mockus, A., Fielding, R. T., and Herbsleb, J. A case study of open source software development: the apache server. In *Proceedings of the 22nd international conference on Software engineering*, ICSE '00, ACM (New York, NY, USA, 2000), 263–272.
17. Pagano, D., and Maalej, W. How do developers blog?: an exploratory study. In *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR '11, ACM (New York, NY, USA, 2011), 123–132.
18. Park, S., and Maurer, F. The role of blogging in generating a software product vision. In *Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, CHASE '09, IEEE Computer Society (Washington, DC, USA, 2009), 74–77.
19. Priedhorsky, R., Chen, J., Lam, S. T. K., Panciera, K., Terveen, L., and Riedl, J. Creating, destroying, and restoring value in wikipedia. In *Proceedings of the 2007 international ACM conference on Supporting group work*, GROUP '07, ACM (New York, NY, USA, 2007), 259–268.
20. Rogers, E. M. *Diffusion of Innovations*, 5th ed. Free Press, 2003.
21. Singer, L., and Schneider, K. Influencing the Adoption of Software Engineering Methods using Social Software. In *34th International Conference on Software Engineering (ICSE), NIER Track (in press)* (2012).
22. Stein, K., and Hess, C. Does it matter who contributes: a study on featured articles in the german wikipedia. In *Proceedings of the eighteenth conference on Hypertext and hypermedia*, HT '07, ACM (New York, NY, USA, 2007), 171–174.
23. Storey, M.-A., Ryall, J., Singer, J., Myers, D., Cheng, L.-T., and Muller, M. How software developers use tagging to support reminding and refinding. *IEEE Transactions on Software Engineering 35*, 4 (july-aug. 2009), 470 –483.
24. Storey, M.-A., Treude, C., van Deursen, A., and Cheng, L.-T. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, FoSER '10, ACM (New York, NY, USA, 2010), 359–364.
25. Stuart, H. C., Dabbish, L., Kiesler, S., Kinnaird, P., and Kang, R. Social transparency in networked information exchange: a theoretical framework. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, CSCW '12, ACM (New York, NY, USA, 2012), 451–460.
26. Suh, B., Chi, E. H., Kittur, A., and Pendleton, B. A. Lifting the veil: improving accountability and social transparency in wikipedia with wikidashboard. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, ACM (New York, NY, USA, 2008), 1037–1040.
27. Treude, C., and Storey, M.-A. Work item tagging: Communicating concerns in collaborative software development. *IEEE Transactions on Software Engineering 38*, 1 (jan.-feb. 2012), 19–34.
28. Wenger, E. *Communities of Practice - Learning, Meaning, and Identity*. Cambridge University Press, 1998.