Mining the Technical Skills of Open Source Developers

João Eduardo Montandon[1], **Marco Túlio Valente**[1]

[1] Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)

{joao.montandon,mtov}@dcc.ufmg.br

Abstract. *Software has "eaten the world" as we witness the rise of companies whose business model is totally centered on software. The successful implementation of these systems heavily depends on the quality and expertise of their software development teams. However, software-based companies are facing an increasing software developers shortage issue. On the one hand, technical recruiters are increasingly relying on the information provided by Social Coding Platforms (SCPs)—e.g., GitHub, Stack Overflow, etc—to prospect new talent. On the other hand, the large volume of data available force job recruiters to only assess superficial information of their candidates. In order to tackle this problem, we described in the thesis an extensive investigation of methods and techniques to identify the technical skills of software developers based on their activity in SCPs. We organized the thesis in three major working units, where we first investigated the most demanded technical and soft skills under the eyes of IT companies, and then assessed developers' technical skills from deep and broad prespectives. These studies resulted in contributions to both research and industrial communities.*

1. Problem and Motivation

Software development is a human-centric activity, which makes developers the most important asset of software companies [DeMarco and Lister 1999]. Moreover, after 25 years of the invention of modern Internet technologies, software has "eaten the world" and we everyday observe the rise of companies totally centered on software. Among the examples, we can mention companies that recently relied on software to disrupt traditional markets, such as Uber, AirBnB, Zoom, Google, Facebook, and many others. In fact, the recent COVID-19 pandemic accelerated the transition of a variety of businesses to the digital world.

Specifically, software systems have become more complex artifacts, which increasingly demands new levels of specialization of their development teams. For example, current software teams include experts in different areas, such as databases, security, human-computer interface (or front-end design), core features (or back-end design), mobile development, etc. Besides, software companies require their developers to master several specific technologies so they can perform their daily work.

This scenario is leading to a worldwide shortage of skilled software engineers. A recent study conducted at BrassComm[1] estimated that more than 420,000 professionals

[1] https://brasscom.org.br/pdfs/estudo-brasscom-formacao-educacional-e-empregabilidade-em-tic/, accessed in March 2021.

will be demanded by the IT industry in Brazil until 2024, whereas only 160,000 will be available in the same period, i.e., a deficit of 260,000 professionals. As a second example, the Bureau of Labor Statistics[2]—a US government agency that provides statistics about the labor market—reported that "the employment of software developers is projected to grow 24% from 2016 to 2026, much faster than the average for all occupations". Consequently, IT-based companies are giving high importance when it comes to hiring new professionals. For instance, Mark Zuckerberg has publicly stated that "our [Facebook's] policy is literally to hire as many talented engineers as we can find".[3]

In the context of industrial software development, the ideal large-scale hiring process should enable technical recruiters to visualize a large number of developers' profiles containing different levels of technical expertise, such as their principal hard skills[4] and professional roles. Put differently, companies are interested in candidates who have deep knowledge in their main area but also have a broad understanding of the software development cycle as a whole. People carrying such a profile are known as T-shaped professionals, and pick up the characteristics of both generalists (broad) and specialists (deep). These professionals are highly-valued due to their capacity of solving problems in a multidisciplinary environment, and to interact with other fields in order to build an innovative product [Susskind and Susskind 2015, Epstein 2019].

2. Goals and Contributions

Thereby, **in this thesis we study techniques to identify developers' technical skills in both broad and deep perspectives given their activity in Social Coding Platforms**. To make this research possible, we conducted three major studies. First, we study in more detail the side view of the skills required by IT companies when looking for new professionals. In the second study, we rely on data-driven methods to mine developers' expertise in a deep perspective. More specifically, we assess the expertise level of software developers in third-party libraries. The third study also relied on data-driven techniques but to evaluate the expertise of software developers in a broad perspective, where we proposed to automatically identify their technical roles.

These studies resulted in the following contributions:

- We conducted a large-scale analysis of 20,000 job opportunities, and revealed which kind of hard and soft skills are demanded in 14 IT professional roles, including Backend, Frontend, Mobile, DevOps, etc. We observed that programming languages are largely required even in management-based positions. Furthermore, experience in third-party components—i.e., libraries and frameworks—is frequently mentioned in developer-based ones. Our findings also reinforced the importance of communication, collaboration, and problem-solving skills to software developers. Section 3 covers this investigation in more detail.

- We build a public dataset containing activity-based information of 575 developers experts in three well-known JavaScript libraries: FACEBOOK/REACT,

[2] `https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm`, accessed in November 2020

[3] `https://code.org/quotes`, accessed in November 2020

[4] In this thesis, the terms *hard skills* and *technical skills* are used interchangeably.

MONGODB/NODE-MONGODB, and SOCKETIO/SOCKET.IO. We then analyzed the performance of supervised and unsupervised approaches to predict developers' expertise level. In summary, we were able to produce clusters where the number of experts ranges from 65% to 75%. We also triangulated the results of such models with information available on LinkedIn profiles. Indeed, 72% of the experts in FACEBOOK/REACT explicitly cite this framework in their LinkedIn profiles. Section 4 describes the methodoloy adopted to perform this study, as well as its results.

- We build another public dataset containing information of 2,284 developers, and evaluated the effectiveness of three machine learning strategies in classifying the competence of software developers in six technical roles: *Backend*, *Frontend*, *FullStack*, *Mobile*, *DataScience*, and *DevOps*. These models presented competitive results with respect to precision (0.88) and AUC (0.89) when identifying all six roles. We describe this work in more detail in Section 5.

3. What Skills do IT Companies look for in New Developers?

There is a growing demand for information on how companies deal with the skills they need when looking for new developers. To find out what are these skills, we reported in this initial study an analysis of more than 20,000 job opportunities available in Stack Overflow Jobs portal, a platform that allows companies to publish new opportunities for IT professionals.

We decided to analyze job opportunities because they represent a declaration of expectations between employers and employees. A job opportunity describes what a company expects from its candidates, as well as what the candidates should expect from the company. For this, a job opportunity includes important company details, such as mission, culture, benefits, etc. At the same time, the company should provide enough details so candidates can decide whether they are qualified or not for the position. In other words, job opportunities describe hard and soft skills required by the disclosed position. Therefore, in this investigation, we analyze both hard and soft skills required by IT job opportunities.

In particular, we leveraged the main characteristics of 14 IT roles, such as Backend, Frontend, and Mobile developers. We also analyzed which soft skills are mostly requested by IT companies when selecting new candidates. Lastly, we discussed the implications of this analysis for both recruiters and developers. Due to space constraints, we report in this manuscript only the results from our hard and soft skills analysis. In case the reader is interested in the data collection process adopted in this work, please refer to the Chapter 3 of the thesis.

Analyzing Hard Skills. As a key result of this first study, we were able to reveal the hard skills demanded for 14 developers roles, as illustrated in Figure 1. The figure shows a heatmap with the distribution of high-level hard skills (rows) per developer roles (columns). With this heatmap, we intend to provide a technology-agnostic analysis, i.e., one that focuses on high-level hard skills (e.g., *Languages*) instead of current technologies (e.g., *Java*). In this way, we also intend to increase the validity of our results against technological changes.

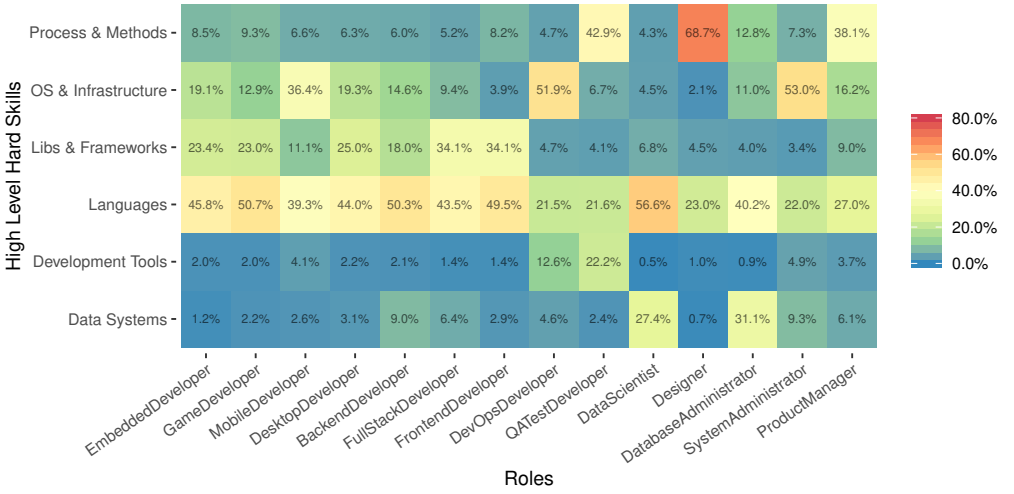

Figura 1. Hard skills required for developers roles.

Figura 2. Soft skills word cloud.

Primarily, *Languages* is the most required hard skill for 9 out of 14 technical roles analyzed. Furthermore, the concentration of *Languages* is higher for development-based roles, e.g., MOBILEDEVELOPER (39.3%), GAMEDEVELOPER (50.7%), FULLSTACKDEVELOPER (43.5%), and FRONTENDDEVELOPER (49.5%). These characteristics make *Languages* the only skills that are significantly mentioned in all roles. Likewise, development-based roles also demand skills on *Libs & Frameworks*, which generally have more than 20% of participation in this group.

Analyzing Soft Skills. Figure 2 depicts the word cloud for the top-100 most frequent terms. We can observe that *communication* plays a central role among the most required soft skills. Indeed, $32\% \pm 5\%$ of the posts in our population (i.e., about one in three posts) mention this skill. *Collaboration*-based skills also feature a special position in this ranking. For instance, the word "team" appears in at least 22% of the jobs posts (i.e., $27\% \pm 5\%$). Lastly, some companies also require experience in *problem-solving* skills, as the words "analytical", "problem solving", or "deliver" are present in at least one out of ten posts (i.e., $15\% \pm 5\%$).

> **Takeaways:** *Languages* skills are largely required for every IT position. Furthermore, experience in *Libs & Frameworks* are frequently mentioned in development-

based ones. As for soft skills, communication, collaboration, and problem-solving skills are considered the most important ones.

4. Identifying Experts in Software Libraries and Frameworks

Modern software development heavily depends on libraries and frameworks to increase productivity and reduce time-to-market [Ruiz et al. 2014, Sawant and Bacchelli 2017]. In this context, identifying experts in popular libraries and frameworks—for example, among the members of global open-source software development platforms, like GitHub—has a practical value. For example, open-source project managers can use this information to search for potential new contributors to their systems. Private companies can also benefit from this information before hiring developers for their projects, as observed in Section 3.

Previous work on software expertise focused on identifying experts for internal parts of a software project, but not on external components, such as libraries and frameworks [Mockus and Herbsleb 2002, Fritz et al. 2014, Schuler and Zimmermann 2008, Da Silva et al. 2015]. By contrast, in this thesis we decided to extend existing expertise identification approaches to the context of third-party software components. Our key hypothesis was that when maintaining a piece of code, developers also gain expertise on the frameworks and libraries used by its implementation. Based on this conjecture, we answered two research questions: (RQ.1) How accurate are machine learning classifiers in identifying library experts? and (RQ.2) Which features best distinguish experts in the studied libraries?

Methodology: To conduct this work, we focused on three popular JavaScript libraries: FACEBOOK/REACT (for building enriched Web interfaces), MONGODB/NODE-MONGODB (for accessing MongoDB databases), and SOCKETIO/SOCKET.IO (for real-time communication). Once these libraries were selected, we collected low-level activity data from developers who have contributed to their popular clients in GitHub.[5] We then processed this data and extracted 13 features covering three dimensions of *changes* performed by them: *Volume*, *Frequency*, and *Breadth*. These features include the number of commits on files that import each library, the period they started working with the aforementioned libraries, and the number of client projects a developer has contributed to. Finally, we built the ground-truth for this work by surveying a sample of 575 GitHub developers, where we asked them to declare their expertise level in the studied libraries.

In this manuscript, we focus at reporting the results obtained in this work. We kindly ask the reader to check out Chapter 4 of the thesis for more details about the methodology and data collections adopted.

(RQ.1) How accurate are machine learning classifiers in identifying library experts? First we executed the machine learning classifiers exclusively for REACT, considering five levels of expertise: Novice 1, Novice 2, Intermediate, Expert 1, and Expert 2.[6] We then executed the same classifiers for three levels of expertise (Novice, Intermediate, and Expert), this time considering all three libraries.

[5] A project is popular if it was among the top-10K most popular ones when the data was retrieved.

[6] SOCKET.IO and NODE-MONGODB were not included since their dataset are not sufficient for this setup.

Tabela 1. Clustering results (cluster 1 has the highest % of experts)

Cluster	% Novices	% Intermediate	% Experts	# Devs
FACEBOOK/REACT				
C1	0.03	0.23	0.74	97
C2	0.12	0.28	0.60	129
C3	0.18	0.27	0.55	192
MONGODB/NODE-MONGODB				
C1	0.12	0.24	0.65	17
C2	0.21	0.43	0.36	14
C3	0.35	0.35	0.30	37
SOCKETIO/SOCKET.IO				
C1	0.00	0.25	0.75	4
C2	0.29	0.36	0.36	28
C3	0.33	0.33	0.33	15
C4	0.50	0.40	0.10	30
C5	0.67	0.33	0.00	12

In the five-classes scenario, the results are not good for almost all performance metrics and classifiers. For example, the machine learning classifiers scored a maximal F-measure of 0.24 for REACT. The results improved in a three-classes scenario, though. For instance, precision results were greater for experts than for novices, both for REACT (0.65 vs 0.14) and NODE-MONGODB (0.61 vs 0.60), while SOCKET.IO has had the highest precision for novices (0.52). F-measure was 0.36 (REACT), 0.56 (NODE-MONGODB), and 0.42 (SOCKET.IO). By contrast, the baseline results for F-measure were 0.25 (REACT) and 0.19 (NODE-MONGODB and SOCKET.IO).

(RQ.2) Which features best distinguish experts in the studied libraries? Table 1 shows the percentage of novices (scores 1–2), intermediate (score 3), and experts (scores 4–5) in the clusters of each library. In FACEBOOK/REACT, we found a cluster where 74% of the developers are experts in the framework; in MONGODB/NODE-MONGODB and SOCKETIO/SOCKET.IO we found clusters with 65% and 75% of experts, respectively. More importantly, such clusters also presented the lowest rate of novice developers, with 3%, 12% and 0% for FACEBOOK/REACT, MONGODB/NODE-MONGODB, and SOCKETIO/SOCKET.IO.

Triangulation with LinkedIn Profiles: To provide further evidence on the value of the clustering procedure described previously, we triangulated the results obtained for REACT developers with the expertise information available on LinkedIn. First, we mapped each REACT developer who did not answer our survey—and therefore was not considered at all in the proposed method—to one of the clusters produced for REACT, as discussed before. We also conducted this study for the other libraries, but they were omitted due to space constraints. After that, we manually searched for the Linkedin page of these developers, looking for their names and possibly e-mails on Linkedin. We were able to find the Linkedin profile of 160 developers. Finally, we manually examined these profiles, searching for clues of expertise on REACT. As a result, 115 developers (72%) explicitly

refer to REACT on their Linkedin profile. Altogether, this triangulation with Linkedin shows that the proposed clustering-based method was able in most cases to find several GitHub developers with evidence of having experience in REACT.

> **Takeaways:** When it comes to identify developers expertise level, we achieve promising results by adopting unsupervised analysis. For all three libraries, we were able to find clusters that are both dominated by experts and have the lowest novices rate.

5. Identifying Technical Roles of Software Developers

As observed in Section 3, IT companies organize their development teams accordingly to the technologies the developers master, e.g., *frontend*, *backend*, *mobile*, and others. For instance, *frontend* developers are specialized on the application's interface, as opposed to *backend* who are responsible for core features. In this context, we currently lack approaches for inferring developers' technical roles. Therefore, our key goal in this study is to identify the technical roles played by developers using information available in open source platforms. To perform this investigation, we assume the following hypothesis: *the technologies that developers master define their technical roles*. For instance, data scientists might have a deeper understanding of data analysis tools. Likewise, frontend developers should master Web APIs concepts, such as REST, AJAX, etc.

Hence, we elaborated four research questions to be answered. In this manuscript, we focus in two of them: (RQ.1) How accurate are machine learning classifiers on identifying developers' technical roles? and (RQ.2) What are the most relevant features to distinguish technical roles?

Methodology: To answer these questions, we built a ground-truth which relied exclusively on Stack Overflow's data, since this data would not be used further in our study. Specifically, we selected from Stack Overflow API users who provided a link to their GitHub pages, and then used the information of their Stack Overflow profiles to label them in at least one of the five technical roles: BACKEND, FRONTEND, DEVOPS, DATASCIENCE, and MOBILE. We ended up with a ground truth with 1,662 developers containing 2,022 role assignments. Afterwards, we collected the GitHub data for each developer. To provide a code agnostic solution (and therefore analyze repositories in multiple programming languages), we collected mostly textual data about the developers' profiles and the projects they own. More specifically, we selected 1,471 features belonging to five high-level categories: *Projects' Dependencies* (798), *Programming Languages* (217), *Projects' Descriptions* (169), *Projects' Names* (155), *Short Bio* (69), and *Projects' Topics* (63). As usual, the reader can check all the details of this work in the Chapter 5 of the thesis.

(RQ.1) How accurate are machine learning classifiers on identifying developers' technical roles? The Random Forest classifier presented the best results overall, scoring 0.77 for precision and 0.71 for AUC. Considering the technical roles individually, the classifier presented high precision rates—i.e., above 0.7—for 4 out of 5 roles: DATASCIENCE (0.86), MOBILE (0.78), FRONTEND (0.77) and DEVOPS (0.70). On the other hand, it scored poor results especially for BACKEND role (recall and F1 equal to 0.12 and 0.18, respectively).

(RQ.2) What are the most relevant features to distinguish technical roles? Figure 3 shows the top-10 most relevant features by technical role. Features associated with

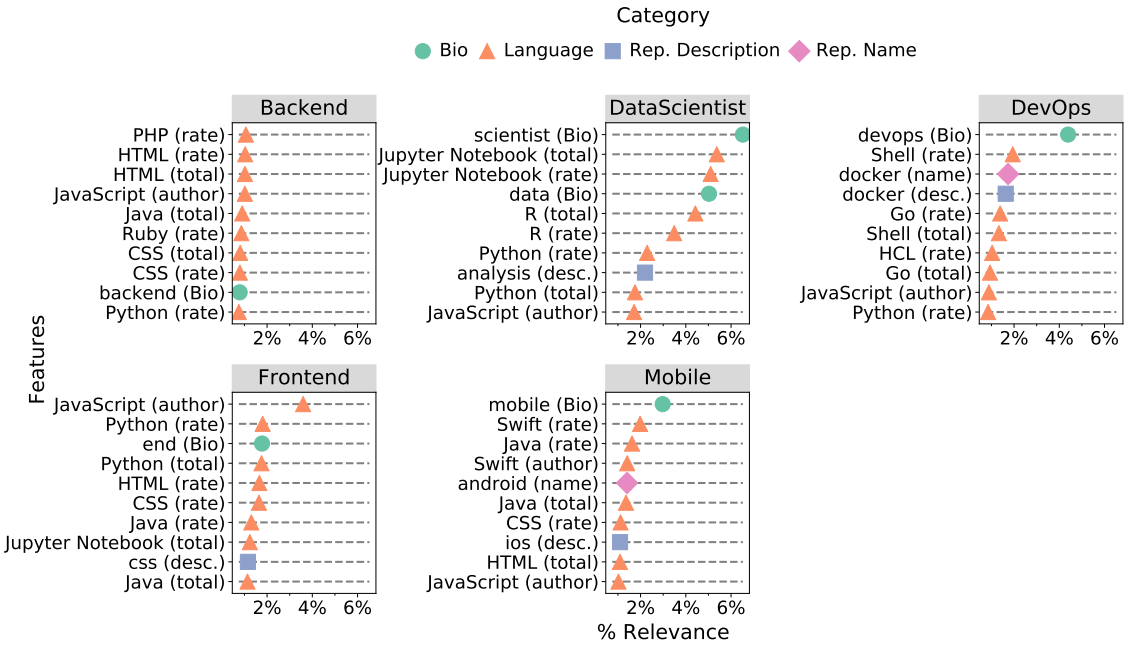

Figura 3. Most relevant features for each technical role.

programming languages are largely predominant for all five roles, representing 38 out of 50 features. When we analyze the results for each role, we see that DATASCIENCE has the highest relevant features, as six features stand out: *scientist (Bio)*, *Jupyter Notebook (total)*, *Jupyter Notebook (rate)*, *data (Bio)*, *R (total)*, and *R (rate)*. In other roles, fewer features presented higher importance: *devops (Bio)* for DEVOPS, *mobile (Bio)* for MOBILE, and *JavaScript (author)* for FRONTEND. Particularly, no feature stands out from the others for BACKEND.

> **Takeaways:** We showed that it is possible to identify major technical roles from developers given their publicly avaiable information in SCPs. For instance, we achieved remarkable precision scores when identifying DATASCIENCE (0.86), MOBILE (0.78), and FRONTEND (0.77) professionals. *Programming languages*-based features demonstrated to be the most relevant ones when identifying developers in all roles.

6. Related Work

When it comes to the state-of-the-art concerning expertise in software development, the existing literature unveils developers' abilities in several scenarios. Some studies focus on discerning developers' abilities according to a particular domain they are involved. Da Silva et al. [Da Silva et al. 2015] and Honsel et al. [Honsel et al. 2016] proposed to identify experts for source code elements—such as methods, classes, and packages—of specific projects. Constantinou and Kapitsaki [Constantinou and Kapitsaki 2017] overcame with a model to detect the contribution roles of developers in open source projects, e.g., bug fixer, triager, core developer, etc. Avelino et al. [Avelino et al. 2019] used the truck-factor metric to identify core developers—those who have more knowledge of the system's structure—in popular projects. However, these approaches are intrinsically li-

mited to the scope of a specific project, i.e., they are not centered on the general expertise of developers.

Other works aimed at leveraging developers' skills independently of the context they are working on. Teyton et al. [Teyton et al. 2013] performed a syntactical analysis over developers' commits to measure their expertise in third-party libraries. Likewise, Saxena and Pedanekar [Saxena and Pedanekar 2017] instrumented the profile of GitHub users by annotating them with Stack Overflow tags. In both cases, relying on abstract syntax trees to obtain expertise information may reduce the performance of the solution in a larger dataset. Oliveira et al. [Oliveira et al. 2019] relied on more simplified information—activity-based features such as the number of commits—to detect experts in Java libraries, but restricted the expertise granularity level and the universe of developers considered in their analysis.

Notwithstanding the acknowledged effort in identifying the abilities of software developers, the aforementioned studies do not completely meet the demands of industry players., i.e., automatically leverage skills profiles that—based on developers' general activity in SCPs like GitHub—can assist recruiters and employers during the hiring process.

7. Final Remarks

In recent years, developers have been playing an increasingly preponderant role in the software development process. Indeed, IT-based companies are giving importance to hiring new professionals at unprecedented levels. When it comes to the state-of-art in this field, most of them investigate developers' expertise in particular open-source projects. By contrast, the industry is more interested in obtaining software developers' information from a more general perspective. We summarized in this manuscript the efforts to understand which abilities do these companies look for in their collaborators, as well as investigated methods to automatically identify such professionals. For this, we performed a set of three major studies where we extensively analyzed data-driven methods and techniques to leverage software developers' profiles, considering their activity in SCPs. In our view, the results of these studies may impact software engineering researchers and practitioners, either by better understanding the most valued abilities in the eyes of IT industry, or by laying out the strengths and limitations of data-oriented techniques in this context.

The research of this thesis generated three publications in Qualis A1 venues, two of them in full format. Since 2019—when the first paper was publicly released—our publications have been referred in more than 40 works. Besides, one of these works have organically featured in Reddit's JavaScript community, gathering attention of its users; so far this paper have been read by more than 1,7K people. We also made publicly available the datasets we used in all three studies as a self-contained replication package, which have been used as baseline for studies in this area. Altogether, these datasets have been downloaded more than 11K times. The reader can find the URL to each one in the thesis. Lastly, it is worth mention that part of this research was conducted in a partnership with researchers from Concordia University (Canada), which is still in progress and resulting in new joint works.

Referências

[Avelino et al. 2019] Avelino, G., Passos, L., Hora, A., and Valente, M. T. (2019). Measuring and analyzing code authorship in 1+118 open source projects. *Science of Computer Programming*, 1(1):1–34.

[Constantinou and Kapitsaki 2017] Constantinou, E. and Kapitsaki, G. M. (2017). Developers expertise and roles on software technologies. In *Asia-Pacific Software Engineering Conference (APSEC)*, pages 365–368.

[Da Silva et al. 2015] Da Silva, J. R., Clua, E., Murta, L., and Sarma, A. (2015). Niche vs. Breadth: Calculating Expertise Over Time Through a Fine-grained Analysis. In *International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pages 409–418.

[DeMarco and Lister 1999] DeMarco, T. and Lister, T. R. (1999). *Peopleware: Productive Projects and Teams*. Addison-Wesley Professional.

[Epstein 2019] Epstein, D. (2019). *Range: Why Generalists Triumph in a Specialized World*. New York.

[Fritz et al. 2014] Fritz, T., Murphy, G. C., Murphy-Hill, E., Ou, J., and Hill, E. (2014). Degree-of-knowledge: modeling a developer's knowledge of code. *ACM Transactions on Software Engineering and Methodology*, 23(2):14:1–14:42.

[Honsel et al. 2016] Honsel, V., Herbold, S., and Grabowski, J. (2016). Hidden Markov Models for the Prediction of Developer Involvement Dynamics and Workload. In *International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE)*, pages 1–10.

[Mockus and Herbsleb 2002] Mockus, A. and Herbsleb, J. D. (2002). Expertise browser: a quantitative approach to identifying expertise. In *International Conference on Software Engineering (ICSE)*, pages 503–512.

[Oliveira et al. 2019] Oliveira, J., Viggiato, M., and Figueiredo, E. (2019). How well do you know this library? Mining experts from source code analysis. In *Brazilian Symposium on Software Quality (SBQS)*, pages 1–10.

[Ruiz et al. 2014] Ruiz, I. J. M., Adams, B., Nagappan, M., Dienst, S., Berger, T., and Hassan, A. E. (2014). A large-scale empirical study on software reuse in mobile apps. *IEEE Software*, 31(2):78–86.

[Sawant and Bacchelli 2017] Sawant, A. A. and Bacchelli, A. (2017). fine-GRAPE: fine-grained API usage extractor - an approach and dataset to investigate API usage. *Empirical Software Engineering*, 22(3):1348–1371.

[Saxena and Pedanekar 2017] Saxena, R. and Pedanekar, N. (2017). I Know What You Coded Last Summer: Mining Candidate Expertise from GitHub Repositories. In *ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW)*, pages 299–302.

[Schuler and Zimmermann 2008] Schuler, D. and Zimmermann, T. (2008). Mining usage expertise from version archives. In *International Working Conference on Mining Software Repositories (MSR)*, pages 121–124.

[Susskind and Susskind 2015] Susskind, R. and Susskind, D. (2015). *The Future of the Professions: How Technology Will Transform the Work of Human Experts*. OUP Oxford, reprint edição edition.

[Teyton et al. 2013] Teyton, C., Falleri, J.-R., Morandat, F., and Blanc, X. (2013). Find your Library Experts. In *Working Conference on Reverse Engineering (WCRE)*, pages 202–211.