Are You Talking about Software Product Lines? An Analysis of Developer Communities

Jacob Krüger
Otto-von-Guericke University Magdeburg, Germany
jkrueger@ovgu.de

ABSTRACT

Community-question-answering systems, such as Stack Overflow, provide a platform for various communities to ask questions, discuss topics, and find knowledge. Especially software developers are heavily relying on such systems to identify solutions for their problems. While the content of community-question-answering systems may be less scientific, it usually represents practical knowledge from various perspectives and backgrounds. Thus, analyzing this content can be valuable for the scientific community to understand previous and current (i.e., open questions) needs of practitioners. In this paper, we report a systematic analysis of two websites that comprise communities with a focus on software development: Stack Exchange and Quora. We extract questions, answers, comments, and discussions on software product lines in general and feature modeling in particular. The results provide a historical perspective, an overview on commonly addressed scopes, and a classification of discussed topics and problems. Moreover, our findings are interesting to understand the practical impact of software-product-line techniques outside of well-analyzed case studies, to support lectures by identifying regularly asked questions, and to scope tool development based on reported technical problems.

CCS CONCEPTS

• **Software and its engineering → Software product lines**.

KEYWORDS

Software product line, variability modeling, feature modeling, Stack Overflow, Quora, community question answering

ACM Reference Format:
Jacob Krüger. 2019. Are You Talking about Software Product Lines? An Analysis of Developer Communities. In *Proceedings of 13th International Workshop on Variability Modelling of Software-Intensive Systems (VAMOS '19).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3302333.3302348

1 INTRODUCTION

Software product lines are an established technique to systematically reuse and customize software variants [1, 26]. To this end, developers create a set of reusable artifacts that serve a specific function, referred to as *features.* Features can have different dependencies, such as mandatory, optional, or requires, that define how a valid variant can be configured. These feature dependencies are usually managed in a variability model, of which *feature models* are the most common ones in practice [3, 8, 28].

Several case studies and surveys show the success of software-product-line techniques in industrial and open-source projects, such as Linux [3, 11, 29, 34]. Still, these reports are usually based on developers that are familiar with these techniques, even if they may call them differently. Moreover, many studies include the same subject systems, for instance, the Linux Kernel is commonly used due to its size [21, 22, 25, 29], and organizations that report on ongoing and new experiences—with constantly improved systems and increasing knowledge of the developers. Thus, the problems and needs that arise and are reported may be limited to a certain subset of developers that are already familiar with most of the concepts and apply the same techniques as well as processes, biasing the results towards well-experienced users.

In this paper, we investigate the practical impact of software product lines from another perspective: Is there a broader awareness and interest in software product lines besides the well-known and established cases? To answer this question, we utilize community-question-answering systems [30], such as Stack Overflow, to perform an empirical analysis. These communities comprise larger numbers of users with diverse backgrounds (e.g., developers, students, researchers) that post questions and answer those of others, resulting in millions of potentially interesting entries.

For our study, we analyzed a total of 174 systems from two websites, namely the Stack Exchange network[1] and Quora.[2] From these, we identified a total of eight communities and 73 questions in which software product lines and especially feature modeling have been a question, answer, or argument. We analyzed these questions, their answers, and comments to identify what scopes of software product lines are of interest to the communities and what topics they discuss. Our results show that especially software product lines as a concept and feature models are discussed in these communities. In most cases, conceptional questions (e.g., definitions) arise and software-product-line techniques are proposed as a solution, particularly as an alternative to cloning variants, for example, in version control systems.

In detail, we contribute the following:

- We report the results of analyzing different communities on their awareness for software product lines in general and feature modeling in particular.
- We categorize the identified questions and put them into the scope of software product lines.

[1]https://stackexchange.com
[2]https://quora.com/

VAMOS '19, February 6–8, 2019, Leuven, Belgium
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6648-9/19/02…$15.00
https://doi.org/10.1145/3302333.3302348

- We discuss the raised questions and their implications for practitioners, researchers, and lecturers.

Overall, the results can be helpful for multiple purposes: They show that developers are familiar with software-product-line techniques, also outside of organizations and systems that are established in the research community. The identified questions can be helpful for lecturers to cope with students' questions, for practitioners to find solutions, for tool developers to identify problems or opportunities, and for researchers to find and motive potentially new research directions. We highly encourage experts of the different topics to answer open questions in such systems to increase the visibility of software product lines outside of the research community—with community-question-answering systems arguably being a more practice-oriented communication channel with a broader audience than research papers [36].

2 BACKGROUND

In the following, we provide a brief overview of *software product lines* and *community-question-answering systems*.

Software Product Lines. The idea of software product lines is to implement and systematically manage a set of features that can be reused to customize software variants [1, 26, 34]. Features implement a specific, user-visible functionality and may comprise additional artifacts, such as models and tests. For this purpose, developers have to define the features' dependencies (e.g., optional, requires, alternative). These dependencies are managed in a variability model [3, 8, 28]—based on which valid configurations that fulfill all defined dependencies can be derived. In an automated step, tools build the configured variant by selecting the defined features and instantiating the corresponding variant. Researchers have proposed numerous techniques to implement the variability mechanism that allows to configure the source code [1, 10], such as preprocessors, components, aspect-oriented programming [13], or version-control systems. We remark that we will refer to any kind of variability model as feature model in the remaining paper, as these are best-known in practice [3], which matches our results.

Community-Question-Answering Systems. Community-question-answering systems allow their users to ask and answer questions on a website [30, 37]. Consequently, such systems build a knowledge base to which a community of users with the same interests contributes to by sharing experiences [2]. Depending on their users, community-question-answering systems have varying scopes and are restricted to certain domains. For example, Stack Overflow is concerned with software development, while the Stack Exchange network—of which Stack Overflow is part of—comprises over 170 additional communities. Similarly, Quora is not limited to any topic, but allows its users to define their preferences and set up groups that focus on a specific topic.

Community-question-answering systems aim to identify the best answer for any question, facilitating the search of users with similar problems [30]. The advantages of such systems are the broad background of the users that may be practitioners, researchers, and students, as well as their global background, as these systems are not limited to a specific region [20]. Thus, they can provide a good basis for an empirical analysis to address some of the known biases of locality and sampling in empirical studies. Still, community-question-answering systems also face several problems, for instance, managing the large number of data, integrating the right users, as well as ensuring the quality of questions and answers [20, 30, 31].

3 STUDY DESIGN

In this section, we report the details of our study design, namely our *research questions*, *subject systems*, *search strategy*, and *inclusion criteria*. Overall, we roughly follow guidelines for systematic literature reviews by applying a similar protocol [14]. By following such a protocol, we aim to ensure that our study is comprehensible, verifiable, and can be repeated.

3.1 Research Questions

The goal of this study was to identify whether software product lines and especially feature modeling are topics that developers are aware of and that they discuss. Numerous case studies have been reported and indicate the importance of software-product-line techniques for industrial and open-source systems alike [3, 11, 34]. However, industrial case studies also report that the notion and mindset of software product lines is still unfamiliar to many developers [6, 9]. Thus, while they apply such techniques, the corresponding variability mechanisms are usually introduced without planning a software product line in advance. Moreover, while the concepts may be familiar, many case studies report from the same organizations and domains, indicating a potentially limited scope.

We aimed to provide another perspective on this issue, independent of well-known organizations: Instead of relying on developers that are evidently familiar with software product lines, we conducted an empirical analysis on community-question-answering systems to identify whether the same techniques are a topic in those. Such systems are used by a variety of users with different backgrounds, knowledge, and interests. Thus, it can be challenging to find questions for a specific topic, but these questions can result in a broader knowledge base that is used to answer them. Moreover, the questions may indicate all kinds of problems, for example, on the conceptual understanding or technical problems, as any user can initially ask any question. Moderators close or remove questions only in exceptional cases.

Overall, our goal was to provide additional insights into the practical relevance of software product lines and feature modeling. To achieve this goal, we defined three research questions that we answer within this paper:

RQ$_1$ To what extent are software product lines and their modeling topics in community-question-answering systems?
RQ$_2$ What scope (e.g., feature modeling, tools) do the identified questions have in the context of software product lines?
RQ$_3$ What topics/problems are raised in the identified questions and how are software product lines connected to those?

With the first research question, we aim to provide a chronological overview to identify trends and the overall interest into software product lines. In contrast, for the other two research questions, we analyzed the questions, answers, and comments in detail to identify the actual issues and problems that are discussed. Especially, open questions may pose new research opportunities and improve the awareness for technical or conceptual problems—and answering those can help researchers and practitioners alike.

3.2 Subject Systems

We selected our subject systems from two community-question-answering websites: The Stack Exchange network and Quora. Stack Exchange comprises a total of 173 different systems (last checked November 13th 2018), with the most prominent one being Stack Overflow. We selected this website, due to this large number of systems that we can crawl at once and the close relation of several of these systems to software development (e.g., Stack Overflow, Software Engineering). Consequently, we expected that software product lines should appear in some questions of these systems. Similar to Stack Exchange, Quora allows its users to create sub-communities and has an even broader scope than the Stack Exchange network. Thus, Quora provided an additional system to enrich our analysis with a different point of view.

3.3 Search Strategy

We used the web-searches of Stack Exchange and Quora to identify relevant questions in each of our subject systems. To this end, we applied the following search strings independently and as exact searches (variations in brackets):

- Variability model(s)
- Variability model(l)ing
- Variability management
- Feature model(s)
- Feature model(l)ing
- Product line engineering
- Software product line(s)

Moreover, we checked the tags *software-product-line* and *software-product-lines* in all of our subject systems. We remark that some technical issues with the Stack Exchange web-interface made it impossible to retrieve all results, wherefore a small set of potentially relevant questions may be missing. Namely, Stack Exchange indicated a total of 100 results for *feature model*, but we were only able to retrieve 50 of them. Still, this seems to be a minor issue, as we identified numerous duplicates among the search strings. In addition, especially feature model appeared in various unrelated contexts, as we explain in the next section.

3.4 Selection Criteria

In order to identify relevant questions, we manually analyzed all retrieved results. To this end, we only applied a single inclusion criterion to consider all relevant questions: Any part of a question, including its answers, comments, and tags, has to directly refer to software product lines or feature modeling. Depending on the subject system and search string, we found that several questions did not fulfill this criterion. For example, many questions included formulations that some products or events feature models or were connected to feature models in data mining.

4 CONDUCT

We last repeated and updated our search on October 25th 2018. At that point in time, we received a total of 183 results (without the 50 that were not accessible) for Stack Exchange and 52 results for Quora. We display the 73 results after applying our inclusion criterion and removing duplicates in Table 1 in Appendix A, ordered by analyzed community and the date a question has been posted. Quora provides fewer information and mechanisms to the community, for example, it does not show the date a question has been posted. Thus, the date in these cases is the one of the oldest answer, or we left it empty if there has not been an answer yet. The identifiers are derived from the Stack Exchange links, which comprise this key, and they are artificial for Quora. All of the identifiers we use in this document are hyper-links to ensure attribution of the users that posted and answered questions. To allow replication and verification, even if the links in Table 1 and the remaining document may be dead at some point, we contribute all of the displayed questions in a public repository.[3]

In Table 1, we further show the rating and, thus, the relevance of a question for Stack Exchange users (Quora has currently no rating) and the status of each question:

A *Answered*: The question has an accepted answer for Stack Exchange or any answer at all for Quora (which does not have an accept mechanism).

C *Closed*: Questions in Stack Exchange communities can be closed, for example, if they are too broad or out of scope. However, they may still have an accepted answer, which we always preferred over closed.

O *Open*: The question has no accepted answer for Stack Exchange or no relevant answer at all for Quora (in one case, we found an answer asking for more information).

We still analyzed all content of a question, independent of its status.

Finally, the last two columns in Table 1 represent the *scope* within software-product-line engineering and the abstracted *topic* of each question. To identify scope and topic, we analyzed each question. For the scope, we differentiated whether a question is mainly related to *configurations*, *feature modeling*, *implementation*, *tools*, or *software product lines* as a whole. If multiple scopes were applicable, we decided for the one that seemed to be best fitting.

To define the topic, we analyzed the actual question with all of its remaining content, derived keywords (e.g., clones, valid configuration, example), and compared all results to identify commonalities. For instance, multiple questions asked about managing variants in version-control systems. In most cases, software product lines have been suggested as a better solution than using forks in such systems, wherefore we derived the topic clone alternative. We discuss these two columns in more details within the next section.

Overall, we identified 73 questions in which software product lines appeared. Unsurprisingly, over half of the questions are part of Stack Overflow (39), which is the largest of our analyzed systems with several millions of questions and users, followed by Quora (17), and Software Engineering (11). The questions have been asked at different points in time and have varying ratings, as we would expect from a community-question-answering system. In a similar study, Barua et al. [2] identified the 40 most prominent topics in almost 3.5 million Stack Overflow posts, for example, coding styles (4.5%), object-oriented programming (3.2%), and SQL (2.2%). Thus,

[3]https://bitbucket.org/Jacob_Krueger/vamos-2019-data

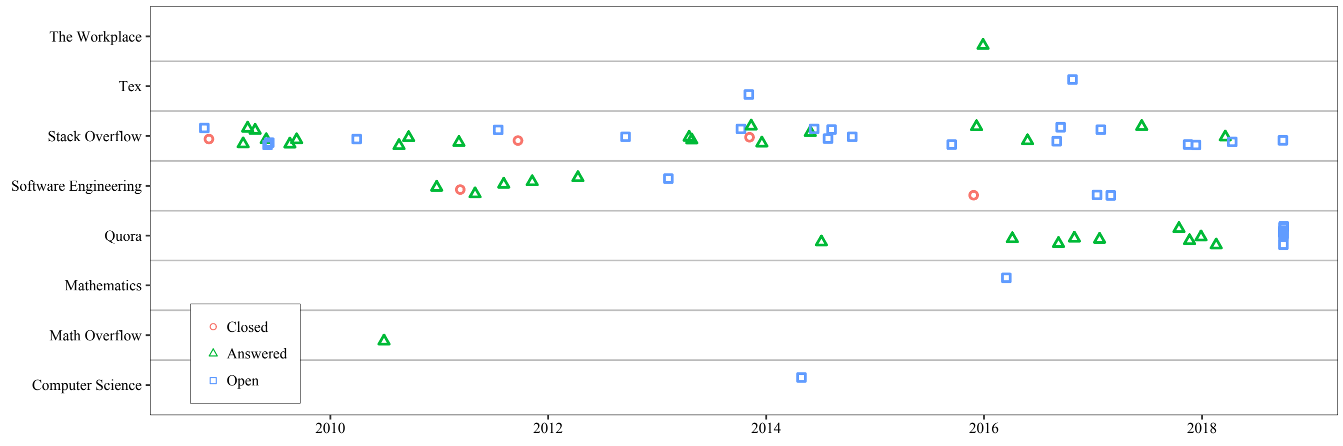

Figure 1: Chronological overview of questions included in this study separated for each community-question-answering system. The vertical jitter shall only reduce overlapping.

software product lines (as explicit term) contribute only to a small portion of the content in our subject systems.

5 RESULTS & DISCUSSION

In this section, we report and discuss our findings. For this purpose, we provide a separate subsection for each research question. These subsections are separated in a results presentation and discussion, closing with a short summary to answer each research question.

5.1 RQ$_1$: Relevance

For **RQ$_1$**, we are concerned with the evolution and distribution of questions to show the relevance of software product lines.

Results. We display the chronological development of the questions we present in Table 1 in Figure 1. Note that the last seven questions in Quora have not been answered yet and, thus, have no date to which we can refer (as aforementioned, we otherwise use the oldest answer's date). We represent these questions as the furthest right entries, as they are still open. Furthermore, for Stack Exchange systems, all dates we display refer to the asked question for consistency, but the actual reference to software product lines may be in a later answer or comment.

As we can see, in five of our subject systems, namely The Workplace, Tex, Mathematics, Math Overflow, and Computer Science, questions on software product lines appear only occasionally. Still, this is hardly surprising, as these systems are less related to programming. In Stack Overflow, Software Engineering, and Quora, we identified more constant interest in this topic. However, only since 2016 software product lines are an actual topic in Quora, with only a single question (i.e., `Quora-01` in 2014) being posted before. Moreover, we can see that the interest of the Software Engineering community, at least to post new issues, seems to decrease, comprising only three new questions since the middle of 2013. For Stack Overflow, the distribution of questions is rather constant, with only two outlier patterns: At two points in time questions are slightly clustered, namely in 2009 and around 2013/14. In contrast, we also see two longer periods without any question in 2012 and 2015.

Considering the ratings within the Stack Exchange network, we found that only four questions have a comparably high number of votes (more than 10). All of these are rather old, as no question with five or more votes has been asked after 2013. This is rather unsurprising, as older questions can collect more ratings. We will investigate these four questions in more detail for **RQ$_3$**, as these ratings indicate that the questions have been more interesting to the communities. In contrast, we found only a single question that has currently been down-voted (i.e., `44498966`).

Discussion. Our results indicate that there is a constant interest in software product lines within the analyzed systems. While not all questions are directly concerned with them, they have often been proposed as solution to reuse and variability problems. Thus, software product lines seem to have neither a decreasing nor increasing trend. They only comprise a small set of results within the millions of questions in our subject systems. Consequently, they are not among the main topics, for example, in the analysis of Barua et al. [2], and statistical explorations seem inappropriate at this points in time. The patterns we see could be interesting for further analyses, however, at least the first one around 2009 can be easily explained, as Stack Overflow was initiated only the year before. Arguably, several domains that have been established then have similar clusters and distributions [2].

For **RQ$_1$**, we summarize:

> There is a small, but constant interest into software product lines within the analyzed community-question-answering systems. This indicates that they have become an established concept in practice that is also known and referred to outside of organizations that are already using them.

5.2 RQ$_2$: Scopes

For **RQ$_2$**, we are concerned with the scope of each question to understand the users' interests—potentially indicating lacking knowledge, room for improvements, or opportunities for research.

Results. In Figure 2, we present a summarized overview of the software-product-line scopes that the analyzed questions are mainly

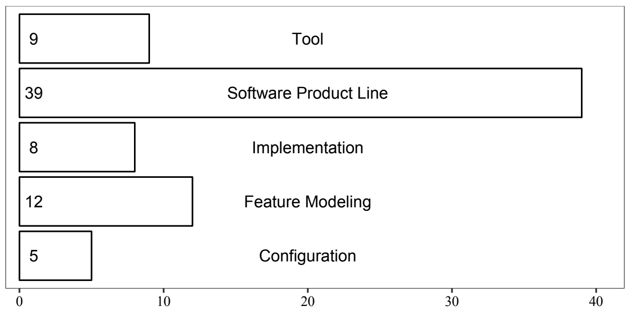

Figure 2: Number of questions related to a specific scope.

concerned with. We see that most questions are referring to software product lines as a whole. Mostly, we selected this category if the question asks for definitions or concepts (e.g., `Quora-05`, `343110`), if a user recommends software product lines as a solution (e.g., `25184295`), or if a user searches for complete examples (e.g., `24258`). Thus, it is not surprising that this scope appears the most.

The second largest scope that we identified are actually feature models. As other authors found in their works [3, 8, 28], feature models (over 60 initial search results) seem to be an established technique in practice and the term seems to be far more common than variability model (only 4 initial search results). In this scope, we clustered questions that are concerned, for instance, with deriving valid configurations from a model (e.g., `39249293`), formalisms of feature models (e.g., `1698508`), and LaTeX exports (e.g., `335708`).

We identified three other scopes, which are tools (9), implementation details for reusable or variable code (8), and configurations and their management (5). Most questions in the tooling scope are about technical problems of integrating plug-ins or using a development environment (e.g. `16077355`). Implementation questions are concerned with the variability mechanisms on code level (e.g., `19246598`). Finally, questions in the configuration scope aim to find answers on how to define and validate that the features are correctly configured and matched, for example:

> "Detect configuration errors with FeatureIDE" `24985049`

Discussion. The distribution of questions in the scopes of software product lines is not surprising: In many cases, the users ask about general concepts and definitions, arguably aiming to understand what a software product line actually is. A concrete example is the following, open question:

> "What is the difference between code reuse and software product line engineering?" `343110`

In the context of industrial adoption and also teaching, we remark that we found several identical and similar questions, indicating that a clear distinction—and its communication—between "simple" and systematic reuse seems necessary. More precisely, comparing and explaining reuse practices and processes in a comprehensible overview may be helpful.

While the other scopes appeared less often in our analysis, they are all connected and should not be disregarded. In particular, feature modeling and tooling are of major concerns. Unsurprisingly, questions in these scopes are often about problems in using

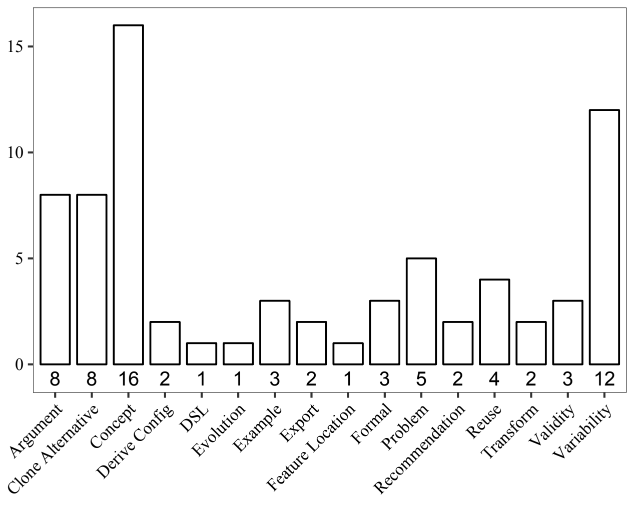

Figure 3: Number of questions related to a specific topic.

tools, for example, installing plug-ins (e.g., `12455731`) or converting feature models into other formats (e.g., `47302532`). Consequently, we can only recommend tool vendors to search for their tools in community-question-answering systems to support their users more flexible and potentially faster. For instance, we found questions on FeatureIDE [33] (e.g., example in the results: `24985049`), Pure::Variants [4] (e.g., `647830`), and Eclipse EMF (e.g., `26399257`).

Some questions are interesting for researchers, as they report potentially new problems and ideas. In particular, open questions on modeling or implementation details and their formalisms can be challenging to address (e.g., `30088`). Moreover, for researchers, it may be interesting to analyze questions about specific contexts that the communities are also aware of, for example, on dynamic product lines (e.g., `24258`) or software product lines for domain specific languages, which we discuss further in the next section:

> "Does it make sense to focus on Domain Specifics Languages (DSL) development following a Software Product Line approach?" `3753417`

For **RQ$_2$**, we summarize:

> The identified questions are concerned with various scopes of software product lines, mostly with general concepts, but also feature modeling, tools, implementation, and configuration. Partly, they go into great detail and ask about specialized concepts that require expert knowledge. Thus, we argue that community-question-answering systems can provide valuable insights on software product lines and highlight the importance of different research areas for practice.

5.3 RQ$_3$: Topics and Problems

For **RQ$_3$**, we are concerned with the actually addressed topics and problems, independent of the scope.

Results. In Figure 3, we show the number of questions we assigned to what topic. Overall, we identified 16 topics based on what has been asked and discussed within each question. We can see that we assigned considerably more questions to four topics compared to the remaining 12 topics, namely concept (16), variability (12), argument (8), and clone alternative (8). Some topics are rather specialized and occur only sparsely, for example, evolution, domain-specific language (DSL), and feature location. In the following, we will separately discuss different topics based on concrete examples. To this end, we discuss *frequent topics*, *highly rated questions*, and *rare topics*. We exemplify concrete problems (*P*) that users raised and that may guide future research to show the potential of analyzing community-question-answering systems. However, identifying such problems is not our focus and, thus, incomplete.

Frequent Topics. We discuss two of the most frequent topics only briefly: **Concepts** includes questions that ask for definitions or general explanations, such as the quoted example in the previous section (i.e., 343110). Similarly, in **argument** questions, software-product-line techniques are only mentioned in some comment or in an answer, but do not contribute to an actual solution. Both topics are of limited interest, as the corresponding questions are more related to definitions and provide starting points for explorations.

The main topic that we could identify and that is connected to software product lines is **managing variability**. We found 12 questions in which the user focused on implementing customizable software, while reuse was not explicitly emphasized. For instance:

> "[...] Soon there are like hundreds of configurable options, and even naming them becomes a nightmare, let alone managing them, changing them, tracking them in code, etc. [...]" 303737

This question is concerned about managing configuration options and one user proposes to rely on software-product-line techniques. Such techniques can facilitate the management, for instance, by supporting automatic tracking of features in the code and providing configurators. Still, the problem *(P_1) of having too many configurable options* remains—organizations' usual work-around is to reduce the complexity by removing features or making them mandatory.

The last topic that we regularly found are **software product lines as an alternative to clone-based software development** (a.k.a. clone-and-own [32]). Apparently, the migration from clones to a software product line (or building one with version-control systems) is not only a recent research area, but an issue that developers struggle with. Interestingly, the questions are structured quite similar, involving a set of variants that exists and for which new variability is intended. Only in such situations, developers seem to start thinking about more systematic designs that allow for customizations and reuse. Still, the concept *(P_2) of thinking differently of versions and variants* seems problematic for some developers. Such questions emphasize the importance of research on extractive software-product-line adoption [15] and can also provide hints on the scope of projects—helping researchers to design new techniques. The basic solution to the problem is highlighted by an answer:

> "Use the modularity of your language to organize a software product line, rather than the version control system. Quite simply, the merge process is not something you want to be a regular part of building your products. Rather, the design of your product should already take this into account." 29514

Highly Rated Questions. As described, we identified four questions that are comparably highly rated. In the following, we will analyze these in more detail.

> "What is Component-Driven Development?" 933723

This question is concerned with a specific programming paradigm: Component-driven software development. However, components are also a technique to implement software product lines, which one user points out in his answer and provides additional background. As our previous results already indicate, such **concept** questions are the most prominent ones and are more an issue of *(P_3) clear definitions and communication.*

> "Why aren't we all doing model driven development yet?" 55679

The user who asked this question is interested in another, more advanced technique to implement reusable software: Model-driven software development [5]. Still, one user points out that most successful applications of this technique are in the domain of software product lines—there may be a *(P_4) missing awareness for the relation between different software development techniques.* As there is no further description or answer based on software product lines, we categorized this question into the **argument** topic.

> "How do I create and maintain a code reuse library?" 1302141

This is one of the few questions in which software product lines are explicitly mentioned in the context of software **reuse**. The user aims to develop a set of reusable modules, indicating features, and argues about structuring them hierarchically, as a feature model would do. Consequently, a user points out that software product lines aim to achieve the desired reuse. While this question is arguably exactly where software product lines can be useful, it also supports the argument that *(P_5) software reuse concepts seem less established* than they potentially should be.

> "Branching model suggestion for same project multiple clients" 72685

The last question is again concerned with **clones** of a system. Again, the accepted answer proposes to use software-product-line techniques. This emphasizes that cloning is a common principle applied by many developers, especially if they are *(P_6) unaware of more systematic techniques*, such as software product lines.

Overall, we can see that none of these questions is explicitly asking about software product lines. Still, they are closely related, for example, to software reuse and extractive adoption of software product lines from clones. Moreover, for each of these questions, a user suggested software product lines as a solution. This indicates that the investigated communities comprise developers that are aware of and see the benefits of using the corresponding techniques. In contrast, it also seems that a lot of the users are rather unaware of the same techniques.

Rare Topics. There are several topics that comprise only one or two questions, but are related to software product lines. While some of them are rather standard, for example, deriving configurations from feature models (e.g., 39249293) or asking for tool recommendations (e.g., 647830), others are quite interesting and highlight the relevance of various research areas, for example:

- One question is concerned with the **evolution** and versioning of software product lines (i.e., 41898931). The question highlights that variability is not only needed along features, but also along time (versions) [16].

- Another user is concerned with understanding how **feature location** is performed for software product lines (i.e., `Quora-12`). This clearly highlights the importance of investigating this research area to support practitioners and to improve existing techniques [18, 27].
- It was rather surprising that we found two questions that are related to **transforming** artifacts. While one is only concerned with loading feature model formats, the other question is actually interested in transforming or updating the variability mechanism (i.e., `37486472`). This supports research that has been conducted in this direction [12, 19].
- Finally, we found one question (i.e., `3753417`) that is concerned with developing similar domain-specific languages (**DSL**s) as software product lines, already in 2010. This research is gaining new attention, referred to as language product lines and seems to be of practical interest [23].

These questions are rather specific, wherefore we expected only few of them. Still, they may be more interesting to researchers, highlighting potential for new research directions.

For $\mathbf{RQ_3}$, we summarize:

The users ask some questions about software product lines multiple times. Most prominently, developers aim to understand concepts of software product lines, variability, and alternatives for cloned software systems. This partly shows a lack of awareness for systematic software reuse, but also indicates that the communities comprise quite knowledgeable users. Some questions highlight the importance of certain research areas, such as software-product-line evolution and transformation, feature location, and language families.

5.4 Threats to Validity

We are aware of some threats to the validity [38] of our study. Concerning *internal validity*, the major threat is our manual analysis performed by a single author. Potentially, we have misinterpreted some questions or answers and other researchers may derive a different categorization than we did. However, we have carefully analyzed the content and limited our interpretation as far as possible, focusing on keywords to abstract scopes and topics. Moreover, we asked a colleague to check and verify our classification to ensure agreement on our decisions. Finally, we ensure that anyone can replicate and verify the results by linking to the corresponding websites and providing downloads in our repository.[3]

Concerning *external validity*, potential threats are: We could not retrieve all results from Stack Exchange, we considered a subset of all available community-question-answering systems, and our results may differ for experienced software-product-line practitioners. Still, our results are valuable, provide interesting insights, and indicate research directions. We mitigated the mentioned threats by using different search strings, aiming to identify all relevant questions even if one search did not work properly. Also, we included 173 subsystems in our study, which comprise the most prominent one for our domain: Stack Overflow. Finally, it was not our goal to necessarily identify questions of experienced practitioners, but show the relevance of and awareness for software product lines for a broader audience of software developers.

6 RELATED WORK

Srba and Bielikova [30] surveyed and classified numerous studies on community-question-answering systems. However, only few perform an empirical analysis of the actual content and its meaning, for example, Coleman and Lieberman [7], Okon and Hanenberg [24], and Venkatesh et al. [35]. Moreover, they are concerned with other topics than software product lines. Based on such works, we described methods to perform empirical analyses of the contents and discussed their flaws and limitations [20]. The closest to this paper is our own previous work, in which we presented some initial findings on aspect-oriented programming [17]. Still, besides the topic, our goal was also different: In the previous paper, we aimed to show the suitability of our methods, while we focused on results in this paper. Similar to us, Barua et al. [2] proposed a method to mine the actual content of community-question-answering systems and performed statistical analyses of Stack Overflow. They focused on the most common topics and their trends, which is out of our scope and did not include software product lines.

7 CONCLUSION

In this paper, we reported an empirical analysis of two community-question-answering websites: Stack Exchange and Quora. Overall, we searched in 174 systems and identified 73 questions that are related to software product lines and feature modeling. Based on these, we analyzed the relevance of this topic for a broader audience than the usual case study subjects. Our findings are:

- There seems to be a small, but constant interest for software product lines in Stack Overflow and Quora.
- The users of the investigated systems are mainly interested in the general concepts of software product lines. However, other topics are also present, namely feature modeling, tools, variability implementation, and configuration.
- Our results indicate a missing awareness of software product lines and reuse in general.
- We found several questions that motivate further research and communication on, for example, managing variability (P_1), developing versions and variants (P_2), definitions (P_3), linking related research areas (P_4), reuse concepts (P_5), and the awareness for systematic reuse (P_6). Moreover, some questions raise specific research opportunities, for instance, on extractive adoption, evolution, feature location, and transformations of software product lines as well as using these for domain-language variants.

The results show that software product lines are a relevant problem in practice and that community-question-answering systems are a valuable source of information that can also be helpful to increase visibility. Moreover, the raised issues motivate different research scopes, areas, and even specific questions. Thus, the results can help researchers to perform and compare similar analyses as well as to motivate and design research directions or tools.

In future work, we aim to extent our analysis significantly. To this end, we want to include other topics, automate our analysis, dig into more details, and perform interview studies. Also, we plan to compare our findings to closely related works in the future, including surveys and reviews on software product line research and communities. While we focused on questions that are explicitly connected to software product lines in this paper, another interesting perspective is to search for implicit connections. Thus, we will extent our search strings and keyword analysis, for example, to include the terms configuration, feature, or preprocessor.

Acknowledgments. We thank Sebastian Krieter for his feedback and for verifying our classifications.

A APPENDIX: INCLUDED QUESTIONS

In Table 1, we show an overview of all questions that are included in this paper. The identifiers serve as hyper-links to the corresponding website, which can also be found in our repository.[3]

Table 1: Overview of the 73 questions included in this study. Each ID links to the corresponding question to attribute the users, independently of whether we identified it through the question, an answer, or a comment. All data is available in a separate repository[3] to allow replications. We also show the rating (only Stack Exchange), status (St.), scope, and topic of each question.

ID/Link	Date	Rat.	St.	Context	Topic
Computer Science					
24258	30.04.2014	2	O	SPL	Example
Mathematics					
1698508	15.03.2016	1	O	FM	Formal
Math Overflow					
30088	30.06.2010	1	A	IMPL	Formal
Software Engineering					
57547	13.03.2011	7	C	SPL	Reuse
72685	30.04.2011	11	A	SPL	Clone Alternative
98451	04.08.2011	6	A	SPL	Clone Alternative
7498823	21.09.2011	0	C	SPL	Variability
55679	09.11.2011	17	A	SPL	Argument
29514	23.12.2010	5	A	SPL	Clone Alternative
143734	10.04.2012	1	A	IMPL	Argument
186368	06.02.2013	2	O	SPL	Clone Alternative
303737	27.11.2015	1	C	SPL	Variability
340127	13.01.2017	5	O	FM	Variability
343110	28.02.2017	2	O	SPL	Concept
Stack Overflow					
261829	04.11.2008	6	O	IMPL	Variability
305423	20.11.2008	3	C	SPL	Concept
647830	15.03.2009	0	A	TO	Recommendation
696021	30.03.2009	9	A	SPL	Argument
785441	24.04.2009	3	A	SPL	Argument
933723	01.06.2009	18	A	SPL	Concept
950516	04.06.2009	4	O	IMPL	Concept
983390	11.06.2009	4	O	SPL	Concept
1302141	19.08.2009	11	A	SPL	Reuse
1400199	09.09.2009	4	A	SPL	Clone Alternative
2551608	31.03.2010	1	O	SPL	Reuse
3519852	19.08.2010	1	A	SPL	Variability
3753417	20.09.2010	2	A	SPL	DSL
5214935	07.03.2011	2	A	FM	Variability
6722770	17.07.2011	2	O	SPL	Example
12455731	17.09.2012	0	O	FM	Derive Config
16077355	18.04.2013	0	A	TO	Problem
16239895	26.04.2013	1	A	CONF	Validity
19246598	08.10.2013	5	A	IMPL	Variability
19842986	07.11.2013	2	C	FM	Concept
19936776	12.11.2013	2	A	SPL	Clone Alternative
20640444	17.12.2013	2	A	SPL	Clone Alternative
Stack Overflow (continued)					
23934768	29.05.2014	2	A	CONF	Variability
24157842	11.06.2014	0	O	CONF	Variability
24985049	27.07.2014	1	O	CONF	Validity
25184295	07.08.0214	0	O	SPL	Clone Alternative
26399257	16.10.2014	1	O	TO	Problem
32584976	15.09.2015	0	O	IMPL	Variability
34132032	07.12.2015	1	A	CONF	Validity
37486472	27.05.2016	0	A	IMPL	Transform
39249293	31.08.2016	1	O	FM	Derive Config
39500746	14.09.2016	0	O	TO	Problem
41898931	27.01.2017	0	O	SPL	Evolution
44498966	12.06.2017	-1	A	SPL	Reuse
47302532	15.11.2017	1	O	FM	Transform
47764139	12.12.2017	1	O	TO	Problem
49412028	21.03.2018	0	A	SPL	Variability
49800115	12.04.2018	0	O	TO	Problem
52547796	28.09.2018	0	O	FM	Formal
TeX					
142176	04.11.2013	4	O	FM	Export
335708	24.10.2016	1	O	FM	Export
The Workplace					
43464	30.12.2015	1	A	SPL	Argument
Quora					
Quora-01	06.07.2014	-	A	SPL	Example
Quora-02	05.04.2016	-	A	SPL	Concept
Quora-03	06.09.2016	-	A	FM	Argument
Quora-04	31.10.2016	-	A	TO	Argument
Quora-05	31.01.2017	-	A	SPL	Concept
Quora-06	15.10.2017	-	A	SPL	Concept
Quora-07	19.11.2017	-	A	TO	Argument
Quora-08	29.12.2017	-	O	SPL	Concept
Quora-09	18.02.2018	-	A	SPL	Concept
Quora-10	-	-	O	SPL	Concept
Quora-11	-	-	O	TO	Recommendation
Quora-12	-	-	O	IMPL	Feature Location
Quora-13	-	-	O	SPL	Variability
Quora-14	-	-	O	SPL	Concept
Quora-15	-	-	O	SPL	Concept
Quora-16	-	-	O	FM	Concept
Quora-17	-	-	O	SPL	Concept

A: Answered, C: Closed, O: Open
CONF: Configuration, FM: Feature Modeling, IMPL: Implementation, SPL: Software Product Line, TO: Tooling

REFERENCES

[1] Sven Apel, Don Batory, Christian Kästner, and Gunter Saake. 2013. *Feature-Oriented Software Product Lines.* Springer.

[2] Anton Barua, Stephen W Thomas, and Ahmed E Hassan. 2014. What are Developers Talking about? An Analysis of Topics and Trends in Stack Overflow. *Empirical Software Engineering* 19, 3 (2014), 619–654.

[3] Thorsten Berger, Ralf Rublack, Divya Nair, Joanne M. Atlee, Martin Becker, Krzysztof Czarnecki, and Andrzej Wąsowski. 2013. A Survey of Variability Modeling in Industrial Practice. In *International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS).* ACM, 7:1–7:8.

[4] Danilo Beuche. 2008. Modeling and Building Software Product Lines with Pure:: Variants. In *International Software Product Line Conference (SPLC).* IEEE, 358–358.

[5] Sami Beydeda, Matthias Book, and Volker Gruhn. 2005. *Model-Driven Software Development.* Springer.

[6] Lianping Chen and Muhammad Ali Babar. 2010. Variability Management in Software Product Lines: An Investigation of Contemporary Industrial Challenges. In *International Conference on Software Product Lines (SPLC).* Springer, 166–180.

[7] Emma Coleman and Zach Lieberman. 2015. Contributor Motivation in Online Knowledge Sharing Communities with Reputation Management Systems. In *Annual Research Conference on South African Institute of Computer Scientists and Information Technologists (SAICSIT).* ACM, 1–12.

[8] K. Czarnecki, P. Grünbacher, R. Rabiser, K. Schmid, and A. Wąsowski. 2012. Cool Features and Tough Decisions: A Comparison of Variability Modeling Approaches. In *International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS).* ACM, 173–182.

[9] Thomas Fogdal, Helene Scherrebeck, Juha Kuusela, Martin Becker, and Bo Zhang. 2016. Ten Years of Product Line Engineering at Danfoss: Lessons Learned and Way Ahead. In *International Systems and Software Product Line Conference (SPLC).* ACM, 252–261.

[10] Cristina Gacek and Michalis Anastasopoules. 2001. Implementing Product Line Variabilities. *ACM SIGSOFT Software Engineering Notes* 26, 3 (2001), 109–117.

[11] Arnaud Hubaux, Andreas Classen, Marcilio Mendonça, and Patrick Heymans. 2010. A Preliminary Review on the Application of Feature Diagrams in Practice. In *International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS).* 53–59.

[12] Christian Kästner, Sven Apel, and Martin Kuhlemann. 2009. A Model of Refactoring Physically and Virtually Separated Features. *ACM Sigplan Notices* 45, 2 (2009), 157–166.

[13] Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier, and John Irwin. 1997. Aspect-Oriented Programming. In *European Conference on Object-Oriented Programming (ECOOP).* Springer, 220–242.

[14] Barbara A. Kitchenham and Stuart Charters. 2007. *Guidelines for Performing Systematic Literature Reviews in Software Engineering.* Technical Report EBSE-2007-01. Keele University and University of Durham.

[15] Charles W. Krueger. 2001. Easing the Transition to Software Mass Customization. In *International Workshop on Software Product-Family Engineering (PFE).* Springer, 282–293.

[16] Charles W Krueger. 2002. Variation Management for Software Production Lines. In *International Conference on Software Product Lines (SPLC).* Springer, 37–48.

[17] Jacob Krüger. 2018. Separation of Concerns: Experiences of the Crowd. In *Symposium on Applied Computing (SAC).* ACM, 2076–2077.

[18] Jacob Krüger, Thorsten Berger, and Thomas Leich. 2018. *Software Engineering for Variability Intensive Systems.* CRC Press, Chapter Features and How to Find Them: A Survey of Manual Feature Location.

[19] Jacob Krüger, Marcus Pinnecke, Andy Kenner, Christopher Kruczek, Fabian Benduhn, Thomas Leich, and Gunter Saake. 2018. Composing Annotations Without Regret? Practical Experiences Using FeatureC. *Software: Practice and Experience* 48, 3 (2018), 402–427.

[20] Jacob Krüger, Ivonne Schröter, Andy Kenner, and Thomas Leich. 2017. Empirical Studies in Question-Answering Systems: A Discussion. In *International Workshop on Conducting Empirical Studies in Industry (CESI).* IEEE, 23–26.

[21] Jörg Liebig, Sven Apel, Christian Lengauer, Christian Kästner, and Michael Schulze. 2010. An Analysis of the Variability in Forty Preprocessor-Based Software Product Lines. In *International Conference on Software Engineering (ICSE).* ACM, 105–114.

[22] Rafael Lotufo, Steven She, Thorsten Berger, Krzysztof Czarnecki, and Andrzej Wąsowski. 2010. Evolution of the Linux Kernel Variability Model. In *International Conference on Software Product Lines (SPLC).* Springer, 136–150.

[23] David Méndez-Acuña, José A Galindo, Benoit Combemale, Arnaud Blouin, and Benoit Baudry. 2017. Reverse Engineering Language Product Lines from Existing DSL Variants. *Journal of Systems and Software* 133 (2017), 145–158.

[24] Sebastian Okon and Stefan Hanenberg. 2016. Can We Enforce a Benefit for Dynamically Typed Languages in Comparison to Statically Typed Ones? A Controlled Experiment. In *International Conference on Program Comprehension (ICPC).* IEEE, 1–10.

[25] Leonardo Passos, Jianmei Guo, Leopoldo Teixeira, Krzysztof Czarnecki, Andrzej Wąsowski, and Paulo Borba. 2013. Coevolution of Variability Models and Related Artifacts: A Case Study from the Linux Kernel. In *International Software Product Line Conference (SPLC).* ACM, 91–100.

[26] Klaus Pohl, Günter Böckle, and Frank van der Linden. 2005. *Software Product Line Engineering: Foundations, Principles, and Techniques.* Springer.

[27] Julia Rubin and Marsha Chechik. 2013. *Domain Engineering.* Springer, Chapter A Survey of Feature Location Techniques, 29–58.

[28] Pierre-Yves Schobbens, Patrick Heymans, and Jean-Christophe Trigaux. 2006. Feature Diagrams: A Survey and a Formal Semantics. In *International Requirements Engineering Conference (RE).* IEEE, 136–145.

[29] Julio Sincero, Horst Schirmeier, Wolfgang Schröder-Preikschat, and Olaf Spinczyk. 2007. Is the Linux Kernel a Software Product Line?. In *International Workshop on Open Source Software and Product Lines (OSSPL).*

[30] Ivan Srba and Maria Bielikova. 2016. A Comprehensive Survey and Classification of Approaches for Community Question Answering. *ACM Transactions on the Web* 10, 3 (2016), 1–63.

[31] Ivan Srba and Maria Bielikova. 2016. Why is Stack Overflow Failing? Preserving Sustainability in Community Question Answering. *IEEE Software* 33, 4 (2016), 80–89.

[32] Ştefan Stănciulescu, Sandro Schulze, and Andrzej Wąsowski. 2015. Forked and Integrated Variants in an Open-Source Firmware Project. In *International Conference on Software Maintenance and Evolution (ICSME).* IEEE, 151–160.

[33] Thomas Thüm, Christian Kästner, Fabian Benduhn, Jens Meinicke, Gunter Saake, and Thomas Leich. 2014. FeatureIDE: An Extensible Framework for Feature-Oriented Software Development. *Science of Computer Programming* 79 (2014), 70–85.

[34] Frank J. van der Linden, Klaus Schmid, and Eelco Rommes. 2007. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering.* Springer.

[35] Pradeep K. Venkatesh, Shaohua Wang, Feng Zhang, Ying Zou, and Ahmed E. Hassan. 2016. What Do Client Developers Concern When Using Web APIs? An Empirical Study on Developer Forums and Stack Overflow. In *International Conference on Web Services (ICWS).* IEEE, 131–138.

[36] Ivonne von Nostitz-Wallwitz, Jacob Krüger, Janet Siegmund, and Thomas Leich. 2018. Knowledge Transfer from Research to Industry: A Survey on Program Comprehension. In *International Conference on Software Engineering: Companion Proceeedings (ICSE-C).* ACM, 300–301.

[37] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2013. Wisdom in the Social Crowd: An Analysis of Quora. In *International Conference on World Wide Web (WWW).* ACM, 1341–1352.

[38] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering.* Springer.