What Questions Do Programmers Ask About Configuration as Code?

Akond Rahman, Asif Partho*, Patrick Morrison, and Laurie Williams
North Carolina State University, Nested Apps*
aarahman@ncsu.edu,asif@nestedapps.com*,pjmorris@ncsu.edu,williams@csc.ncsu.edu

ABSTRACT

Configuration as code (CaC) tools, such as Ansible and Puppet, help software teams to implement continuous deployment and deploy software changes rapidly. CaC tools are growing in popularity, yet what challenges programmers encounter about CaC tools, have not been characterized. A systematic investigation on what questions are asked by programmers, can help us identify potential technical challenges about CaC, and can aid in successful use of CaC tools. *The goal of this paper is to help current and potential configuration as code (CaC) adoptees in identifying the challenges related to CaC through an analysis of questions asked by programmers on a major question and answer website.* We extract 2,758 Puppet-related questions asked by programmers from January 2010 to December 2016, posted on Stack Overflow. We apply qualitative analysis to identify the questions programmers ask about Puppet. We also investigate the trends in questions with unsatisfactory answers, and changes in question categories over time. From our empirical study, we synthesize 16 major categories of questions. The three most common question categories are: (i) syntax errors, (ii) provisioning instances; and (iii) assessing Puppet's feasibility to accomplish certain tasks. Three categories of questions that yield the most unsatisfactory answers are (i) installation, (ii) security, and (iii) data separation. We also observe 8 of the 16 question categories to have an increasing trend, whereas, one identified question category has a decreasing trend.

KEYWORDS

challenge, configuration as code, continuous deployment, devops, infrastructure as code, programming, puppet, question, stack overflow

ACM Reference Format:
Akond Rahman, Asif Partho*, Patrick Morrison, and Laurie Williams. 2017. What Questions Do Programmers Ask About Configuration as Code?. In *Proceedings of International Workshop on Rapid Continuous Software Engineering (RCOSE'18).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Continuous deployment (CD) is a software engineering process where software changes are automatically tested in a continuous manner, and frequently deployed to production environments [17] [7]. Using CD, information technology (IT) organizations deploy software changes rapidly, as often as twice a day, as is done by Facebook [17] [11]. Practitioners have observed use of configuration as code (CaC) tools such as Ansible [1] and Puppet [2], as an essential practice to implement CD [11]. CaC is the technique of defining computing and network configurations through source code, which can be stored in version control systems to allow reproducibility and testing [7]. CaC is used to create and manage an automated deployment pipeline that enables IT organizations to deploy their software changes rapidly [7] [11]. As with all new technologies, users of CaC tools experience challenges and ask questions on how to overcome the challenges. For example, in an online forum one programmer described how resolving Puppet discrepancies could be challenging [3]. Systematically identifying what questions programmers ask about CaC can help current and potential CaC adoptees. Such investigation can also provide insights on development platforms that programmers use to manage infrastructure, and functionalities that they want from CaC tool vendors.

The goal of this paper is to help current and potential configuration as code (CaC) adoptees in identifying the challenges related to CaC through an analysis of questions asked by programmers on a major question and answer website.

In prior work researchers have mined questions posted on programming-related question and answer (Q&A) websites such as Stack Overflow (SO), to identify questions asked by mobile programmers [19] [10] and web programmers [1]. Researchers [22] [19] also quantified which categories of questions receive more unsatisfactory answers than others to gain insights about the identified challenges. On Q&A websites, programmers describe the technical challenges they face using a wide range of questions to seek help and advice. These posted questions give researchers the opportunity to highlight the knowledge needs of programmers [2]. Researchers [2] have also used the posted SO questions to identify how programmers interest in a topic evolves over time. In our research study, we systematically investigate the categories of questions asked by programmers who work with CaC tools. We answer the following research questions:

- **RQ1: What questions do programmers ask about configuration as code?**
- **RQ2: How many answers do configuration as code (CaC)-related question categories yield? How many questions with unsatisfactory answers do CaC-related question categories yield?**
- **RQ3: How many views do question categories related to configuration as code yield?**
- **RQ4: What temporal trends do the question categories related to configuration as code exhibit?**

We focus on the questions related to programming with Puppet, a tool to implement CaC. We select Puppet because it is one of the most popular CaC tools [8], and has been used by practitioners since 2005 [9]. We extract 2,758 Puppet-related questions from SO, a popular programming-related question and answer (Q&A)

[1] https://www.ansible.com/
[2] https://puppet.com/

[3] https://redd.it/3yo7lf

RCOSE'18, RCOSE 2018, Gothenburg, Sweden
2016. ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

website for the period January 2010 to December 2016. We used card sorting [23], a qualitative analysis technique, to identify the question categories from the extracted questions. We use the timestamps of Puppet-related questions to examine temporal trends of the question categories.

We list our contributions as following:

- A category of questions asked by programmers who work with CaC;
- A ranking of the CaC-related question categories, sorted based on amount of questions that have unsatisfactory answers; and
- A summary of temporal trends extracted for the identified question categories

We organize the rest of the paper as following: Section 2 describes our methodology conducted for this paper. We present our findings in Section 3. We list the limitations of our paper in Section 4. Section 5 provides prior research work relevant to our paper. Finally, we conclude the paper in Section 6.

2 METHODOLOGY

In this section, we first provide background information necessary to conduct our methodology.

2.1 Background

CaC is a practice to specify the configurations of an environment automatically in which software will be installed or tested [7]. CaC helps in managing environment configurations on one or multiple machines starting from installation, upgrading, and maintenance. Practitioners attribute the concept of infrastructure as code to Chad Fowler, in his blog published in 2013 [4]. The phrase 'as code' in IaC corresponds to applying traditional software engineering practices, such as code review and version control systems (VCS), for IaC scripts [11][7]. To automatically provision infrastructure, programmers follow specific syntax, and write configurations in a similar manner as software source code. IaC scripts use domain specific language (DSL) [20].

Practitioners use automated tools such as Chef, and Puppet, which are dedicated for implementing CaC. CaC scripts are also known as 'infrastructure as code' (IaC) scripts [21] [7]. We chose to focus on Puppet because it was first released in 2005 [9], and since then Puppet is considered as one of the most popular tools to implement CaC [8]. Compared to the release of Puppet, Chef is relatively new, and our assumption is that the Puppet-related questions on SO will give us the opportunity to look at a representative sample of questions needed for our research study.

For collecting CaC-related technical questions, we selected the programming-related Q&A website SO. We selected SO because this Q&A website is the most popular programming Q&A website and has been extensively used in software engineering research [22] [19] [2].

In SO, users can post questions that describe a specific problem that they want to seek advice on [5]. Each question has a title that provides a concise summary of what the question is about [5]. The details of the question is presented in the body, where users can describe the problem in detail with additional references, such as web links, code snippets, and screenshots [5]. Each question can have multiple views, multiple answers, and one accepted answer [5]. Views represent the number of total visits for a question by both non-authenticated and authenticated users of the website [5]. An answer is a solution for the provided question of interest [5]. An accepted answer is an answer that was verified as satisfactory by the user who posted the question [5]. Each question has one or many tags. A tag is used to connect with advanced website users so that the question has a higher probability of getting a quick answer [5]. In our paper we used a dataset collected from the Stack Exchange Data Explorer, on Dec 30, 2016. This dataset includes all information related to SO, which includes questions, tags, answers to questions etc.

2.2 Step One: Determining Tags

Tags provide a means to extract SO questions that belong to a certain technology. Our assumption is by identifying the tags that are related to Puppet, we will be able to collect a representable set of Puppet-related questions posted on SO. We used the SO dataset to determine the tags needed to extract Puppet-related questions. We used the following two steps:

- First, we identify the tags that included the string 'puppet'. We extract these tags along with their description.
- Second, by manually reading the description of each tags we determine which tag could be used for relevant content collection. From the description, if we observe the tag to be related to Puppet, we consider that tag for our paper. For example, from the tag description "Puppet Enterprise is the commercially supported and packaged release of Puppet." for tag 'puppet-enterprise', we observe tag 'puppet-enterprise' to be related to Puppet.

Upon completion of the above-mentioned steps, we obtain a list of tags that are related to Puppet. We use these tags in Step Two.

2.3 Step Two: Content Collection

Using the tags identified from Step One, we obtain a set of questions posted on SO. For further analysis, we identify questions that include at least one tag identified from Step One. For each of the identified question we obtain the ID, title, body, creation date, the count of views, and a flag that indicates whether or not the question has an accepted answer. We used the collected dataset to extract necessary content.

2.4 Step Three: Identifying Question Categories

We use card sorting [23], a qualitative analysis to identify the question categories from the collected Puppet-related questions. Card sorting is a qualitative technique to identify categories from textual artifacts [23]. We use card sorting because this method helps in achieving insightful, non-overlapping categories, and is used widely in software engineering research [23] [8]. We followed Zimmerman et al. [23]'s recommendations and implemented card sorting in the following three phases:

- Preparation: We collect the ID, title, and body of each Puppet-related question for card sorting analysis.

[4] https://www.oreilly.com/ideas/an-introduction-to-immutable-infrastructure

- Execution: The first and second author separately conduct card sorting on the collected questions. For deriving the question categories, each used the body and title of the collected questions.
- Analysis: Once the first and second author complete their card sorting analysis, the derived question categories are cross-checked by both authors. We apply negotiated agreement [3], where the first and second author discussed to which question categories they disagreed upon. For each of the question category, for which both first and second author agreed upon, is identified as a question category.

2.5 Step Four: Analysis

We use the list of CaC-related question categories from Step Three to answer the four research questions as following:

2.5.1 RQ1: What questions do programmers ask about configuration as code (CaC)?. We answer RQ1 by first, providing the list of question categories that we identified from Step Three. Next, we calculate the proportion of questions that belong to each question category using 'proportion of questions for category x, (Qx)'. We compute the proportion of questions that belong to question category x using Equation 1

$$\mathrm{Q}(x) = \frac{\text{total count of questions included in category } x}{\text{total count of questions related to Puppet}} \quad (1)$$

2.5.2 RQ2: How many answers do configuration as code (CaC)-related question categories yield? How many questions with unsatisfactory answers do CaC-related question categories yield? We answer the first part of RQ2 by calculating the answer to question ratio. We use the following metric in this regard: 'Answer to Question Ratio for category x, (AQ(x))': We use Equation 2 to measure AQ:

$$\mathrm{AQ}(x) = \frac{\text{total count of answers for questions included in category } x}{\text{total count of questions included in category } x} \quad (2)$$

A question with no accepted answer may imply that the programmer who posted the question was not satisfied with the posted answers. A question might not have an accepted answer due to emerging technologies and limited support from Q&A community. Questions related to CaC can have unaccepted answers that might indicate a key category and needs support. Analysis of the second part of RQ2 involves quantifying which of the CaC-related question categories yield more questions with unsatisfactory answers than other categories. Similar to prior research [19] that has used the concept of satisfactory and unsatisfactory answers, we use the metric 'unsatisfactory answers for category x, UNS(x)'. We calculate UNS(x) using Equation 3:

$$\mathrm{UNS}(x) = \frac{\text{questions with unsatisfactory answers included in category } x}{\text{questions included in category } x} \quad (3)$$

Answer to RQ2 includes the UNS scores of each identified question category, and a ranked order of the identified question categories sorted based on their UNS scores.

2.5.3 RQ3: How many views do configuration as code (CaC)-related question categories yield? Programmers can view a question and its answers without becoming a registered user on SO. By capturing the view count of each identified category we can capture both, registered and non-registered users' interest in a particular question category. We use the metric 'view count per question', to answer RQ3. For category x, view count per question presents the count of views per questions included in category x. We use Equation 4 to calculate VQ:

$$\mathrm{VQ}(x) = \frac{\text{summation of view count for all questions included in category } x}{\text{total count of questions included in category } x} \quad (4)$$

2.5.4 RQ4: What temporal trends do the configuration as code (CaC)-related question categories exhibit? Similar to prior research [1] [2], we investigate temporal trends to identify how the amount of questions related to the identified question categories evolves over time. We determine the trends of category x using the following step: for each month m when one or more CaC-related questions are created, we calculate temporal_trend of category x using Equation 5

$$\text{temporal\_trend}(x, m) = \frac{\text{count of questions posted in month } m \text{ that belong to category } x}{\text{count of questions posted in month } m} \quad (5)$$

We apply the Cox-Stuart test [4] to determine if the exhibited trend is significantly increasing or decreasing. The Cox Stuart test is a statistical technique that compares the earlier data points to the later data points in a time series to determine whether or not the trend observant from the time series data is increasing or decreasing with statistical significance. We use a 95% statistical confidence level to determine which topics exhibit increasing or decreasing trends. To determine temporal trends of each category we apply the following strategy:

- if Cox-Stuart test output states the temporal frequency values are 'increasing' with a p-value < 0.05, we determine the temporal trend to be 'increasing'.
- if Cox-Stuart test output states the temporal frequency values are 'decreasing' with a p-value < 0.05, we determine the temporal trend to be 'decreasing'.
- if we cannot determine if the temporal trend is 'increasing' or 'decreasing', then we determine the temporal trend to be 'consistent'.

3 RESULTS

We use this section to provide our findings and answer the four research questions:

3.1 Step One: Determining Tags

Altogether we identify nine tags that are related to Puppet: 'librarian-puppet', 'puppet-3', 'puppetlabs-aws', 'puppetlabs-mysql', 'puppet-provider', 'puppet', 'puppet-enterprise', 'puppetlabs-apache', and 'rspec-puppet'. We used these tags to extract necessary Puppet-related questions.

3.2 Step Two: Content Collection

From the collected nine tags we identify 2,758 Puppet-related questions that spanned from January 2010 to December 2016.

3.3 Step Three: Identifying Question Categories

The first and second author respectively, identify 18 and 22 unique question categories from the collected 2,758 Puppet-related questions. The question categories identified by the two authors are susceptible to bias. We account for this bias by cross-checking the identified question categories, and including question categories to which both authors agreed upon. Altogether the first and second author agreed upon 16 technical question categories.

3.4 Step Four: Analysis

We answer the four research questions in the following subsections:

3.4.1 Answer to RQ1: What questions do programmers ask about configuration as code (CaC)?. We identify 16 unique question categories related to Puppet presented in Table 1. In Table 1 each category, a brief description of the question categories, and a representative example is presented respectively, in the columns 'Category', 'Description', and 'Example'. In the 'Category' column the count of each question for each category is included parenthesis.

In Table 2, we report the values of four metrics for each identified question category. Table 2 is sorted based on the proportion of questions for category (Q) metric. The 'Category' column presents each of the 16 unique question categories. The proportion of questions (Q), answer to question ratio (AQ), view count per question (VQ), and unsatisfactory answers (UNS) scores of each category is respectively presented in columns 'Q (%)', 'AQ', 'VQ', and 'UNS (%)'. According to Table 2, the top three question categories based on the Q metric are Syntax Error, Provisioning, and Feasibility. Together these three categories account for 46.2% of the total questions.

3.4.2 Answer to RQ2: How many answers do configuration as code (CaC)-related question categories yield? How many questions with unsatisfactory answers do CaC-related question categories yield? According to Table 2, the answer to question ratio (AQ) scores of all 16 question categories varied between 1.0 and 1.4. The unsatisfactory answer for each category (UNS) score varied between 61.8% and 34.2%. We also observe that 11 of the 16 identified question categories to have UNS score of more than 50.0%. On average, our findings indicate that a Puppet-related question has one answer, and the posted answer might not be satisfactory for majority of the question categories.

3.4.3 Answer to RQ3: How many views do configuration as code (CaC)-related question categories yield? From Table 2, we observe 10 of the 16 question categories to have a view count per question (VQ) score of at least 1000. We notice that even though Puppet-related questions on average receive one answer, and the posted answers do not always yield satisfactory answers, the posted questions have relevance amongst programmers, as observed from VQ. Table 3 presents an ordered list of the question categories sorted based on VQ, in a descending order. We also present the question categories in a descending order, sorted based on AQ and UNS.

The ranking of the 16 question categories help us to make further observations:

- Template ranks first based on answer count (AQ) and last based on unsatisfactory answers (UNS). Overall, we observe Template-related questions to be well addressed, than other categories.
- Security ranks second in terms of views and unsatisfactory answers, but ranks 15^{th} based on answer count. This observation suggests that even though security-related question categories are common amongst programmers, they often do not get satisfactory answers from the SO community.
- Data Separation ranks as third lowest based on answer count, and third highest with respect to unsatisfactory answers. This observation implies that in general, the SO community has provided unsatisfactory answers in resolving questions related to Data Separation.
- Testing ranks 15^{th} based on views and unsatisfactory answers, implying even though Testing draws lesser interest amongst programmer, the questions are well addressed compared to that of majority of the question categories.

3.4.4 Answer to RQ4: What temporal trends do the configuration as code (CaC)-related question categories exhibit? We have presented the temporal_trend values for each question category in Figure 1. Figure 1 presents a scatterplot with smoothing plot with the trends highlighted. Whether or not these trends are significantly decreasing or increasing can be understood from Table 4. Categories for which we observe $p - value < 0.05$, are highlighted in grey. From Table 4 we observe eight question categories with 'increasing' trends, and one category with 'decreasing' trend.

Tables 4 and 3 help us to make the following observations:

- Three of the top five question categories with unanswered questions have 'increasing trends'. Our findings suggest that the three categories Installation, Data Separation and Network, is increasingly getting attention amongst programmers, but their questions are not well-answered.
- Posting of questions related to Feasibility is decreasing amongst programmers, according to Cox-Stuart test. We also observe that Feasibility is ranked third highest based on answer count, and fifth highest based on view count. We suspect that even though programmers are interested in feasibility-related tasks, they may be getting their needed information from other sources such as existing SO posts, blog posts, and video tutorials.

4 THREATS TO VALIDITY

We discuss the threats of the paper as following:

- *Selection of Q&A website*: We only have used questions posted on SO, and did not consider other Q&A websites. In the future, we plan to add other Q&A websites.
- *Selection of CaC tools*: We have not considered questions related to other CaC tools, such as Ansible and Chef. In the future, we will apply our analysis on questions related to Ansible and Chef.
- *Bias in identifying question categories*: The identified 16 question categories are susceptible to bias. We have accounted for this bias by cross-checking the derived question categories, and including question categories to which both individuals agreed.

Table 1: Answer to RQ1. We identify 16 question categories. References to all the examples are available online [14]

Category (Count)	Description	Example
Syntax Error (560)	Programmers ask questions about resolving syntax errors related to Puppet utilities, such as attributes, and data types . Even though the Puppet documentation claims that using this language "does not require much formal programming experience", we observe the most frequently occurring question category to be related to syntax.	*How to fix 'can't convert String into Integer' after calling the split function in Puppet?*
Provisioning (386)	Programmers ask about how to use Puppet to provision containers, virtual machines and cloud instances. We observe programmers to use Puppet to provision instances on a wide range of platforms, such as Microsoft Azure [5].	*Provisioning with Puppet using Vagrant, correct approach*
Feasibility (329)	We observe programmers to ask about in assessing the feasibility of Puppet to accomplish a certain task. For feasibility-related questions, we observe programmers describing a use case or a hypothetical scenario, and asking SO users if Puppet can be used for the given scenario or use case.	*How to create hierarchy in Puppet*
Installation (262)	Programmers ask about installing the Puppet tool, installing open source Puppet modules, and installing software packages using Puppet. On SO, programmers seek information about pre-built modules that can be installed and inherited to resolve specific tasks. While installing these modules, programmers encounter errors and ask about how these errors can be resolved.	*Trouble Using Puppet Forge Module example42/splunk*
Filesystem (229)	Programmers' questions about file and file system includes mounting logical volumes, and performing file operations . The Puppet documentation provides instructions on how Puppet components related to file operations, such as the 'file' resource type, could be used. Programmers might be using these components to fit their own needs and finding existing documentation insufficient, as one programmer stated: "I can't believe how difficult Puppet is being with Windows - particularly Windows permissions!"	*How to remove all /etc/*.txt files with Puppet?*
Security (160)	Puppet provides utilities for software security. While using these utilities, programmers face difficulties. We observe programmers facing difficulties in creating certificate authority (CA) using Puppet , managing built-in SSL certificates of Puppet , and automated management of user credentials). Even though Puppet is advertised as an easy-to-use tool to implement security and maintain compliance, we observe programmers facing challenges while using Puppet for security.	*External CA configuration with Puppet agent*
Data Separation (132)	Programmers ask about data separation from scripts while programming with CaC. CaC tools such as Puppet, provide utilities to separate data from scripts. Hiera is an example utility provided by Puppet, using a key-value lookup system [6]. Puppet users face challenges when using Hiera that includes solution to Hiera-related errors, and usage of parameters in Hiera.	*Evaluation Error while using the Hiera hash in Puppet*
Troubleshooting (127)	Programmers ask questions when they encounter unexpected behavior . They also query about tools to troubleshoot Puppet files. As one programmer stated: "I'm learning Puppet and the biggest frustration I have with the entire paradigm is the try/run/fix development process I'm using to build functional Puppet code...I know writing a debugger for Puppet would require some creativity given it's nature but I think this is something the community could really benefit from."	*Why Puppet beaker fails to resolve role "agent", whereas master/database/dashboard works just fine?*
Network (105)	We observe programmers to ask questions about when they perform network-related tasks. Examples of such questions include configuring network interfaces, and extracting IP addresses.	*How do I use Puppet manifest to configure network settings?*
Environment Variables (95)	Environment variables, for example, kernel version, host name, and type of operating system are managed using CaC tools such as Puppet [13]. Facter is a Puppet utility that discovers and manage environment variables in local/remote hosts [13]. We observe programmers facing challenges while working with Puppet and environment variables.	*How to schedule Puppet custom facts to execute every X hours?*
Testing (83)	Testing-related questions are also common amongst programmers. Examples of testing-related questions include mocking, code coverage of Puppet code, and testing Puppet code that has dependencies.	*How to Measure Code Coverage: Rspec-Puppet?*
Template (73)	Configurations can be rendered using Embedded Ruby (ERB)-based templates in Puppet [13]. Programmers ask questions about while working with templates, for example, criteria-based template selection, and analyzing output logs generated from ERB templates.	*How do I use the output of a command in a template in Puppet?*
Dependencies (70)	In Puppet, a module is a collection of Puppet files grouped together based on a common functionality. One or more Puppet files can be dependent on one or more modules. We observe programmers to ask about dependencies, for example installing Puppet module dependencies, and resolving dependency errors . In case of dependency problems, error messages can be confusing, as one programmer pointed out "I'm getting a warning from Puppet about an unmet dependency, and yet it appears the dependency is met. I'm more than a bit confused about this error message."	*dependency error while running windows feature*
Database (58)	Programmers ask questions related to database operations for vendors such as MySQL, and MongoDB. Common database operations include creating database backups, creating tables, and managing SQL scripts. We also observe programmers using Puppet to install databases on multiple platforms such as Ubuntu and containers.	*How to update MySQL Table with Puppet?*
Resource Ordering (49)	Puppet resources such as files, classes, and manifests are executed randomly. Programmers ask questions about what mechanism can be used to control the execution order of Puppet manifests , and analyzing the errors while applying resource ordering mechanisms.	*Puppet Resource Ordering not working*
Daemon Services (40)	In CaC, daemon services such as Puppet Agent retrieves configuration information and applies it to local or remote hosts [13]. Programmers work with these agents to automate infrastructure, and face challenges with Puppet agents. Examples of such challenges include seeking mechanisms on implementing a specific task with Puppet agent, and resolving Puppet agent discrepancies.	*Puppet agent: provider Git is not functional on this host*

5 RELATED WORK

Our paper is related to empirical studies that have focused on CaC technologies, such as Puppet. Sharma et al. [21] investigated smells in CaC scripts and proposed 13 implementation and 11 design configuration smells. Hanappi et al. [6] investigated how convergence of Puppet scripts can be automatically tested, and proposed an automated model-based test framework. Jiang and Adams [8] investigated the co-evolution of CaC scripts and other software artifacts, such as build files and source code. They reported CaC scripts to experience frequent churn. Rahman et el. [15] investigated which factors influence usage of CaC tools. In a recent work, Rahman and Williams [16] characterized defective CaC scripts using text mining, and created prediction models using text feature metrics. The above-mentioned studies motivate us to explore the area of CaC by taking a different stance: we analyze CaC-related questions posted on Stack Overflow (SO) to identify the question categories related to CaC programming.

Our paper is also related to prior research studies that have used data extracted from Q&A websites. Barua et al. [2] studied the topics and trends in SO questions and answers spanning from July 2008 to September 2010. They observed web and mobile application development to gain interest over time. They also observed developers to ask about a wide range of questions, such as career advice, C# syntax, and version control systems. Bajaj et al. [1] studied SO data

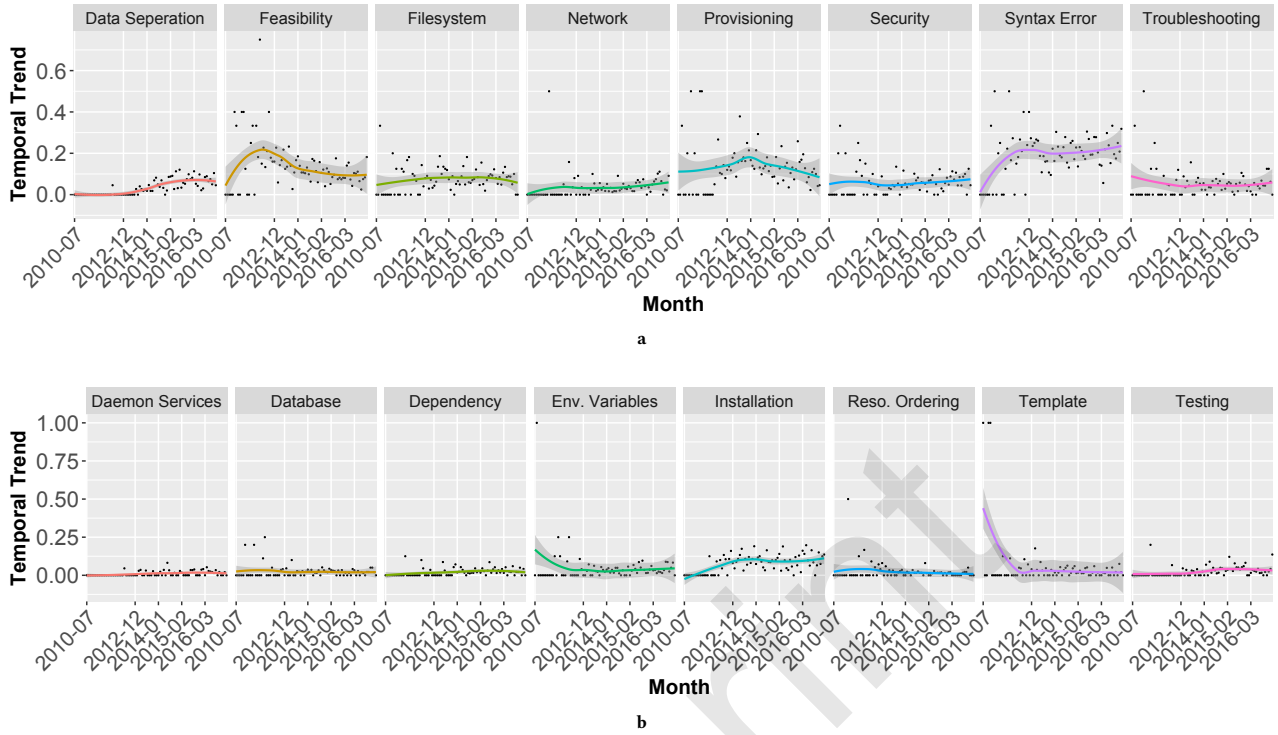

Figure 1: Temporal trend of each identified question category. Figures 1a, and 1b present the temporal_trend values for the 16 question categories.

Table 2: Answer to RQ1, RQ2, AND RQ3: Summary of Identified Question Categories

Category	Q(%)	AQ	UNS(%)	VQ
Syntax Error	20.3	1.1	48.9	1141.8
Provisioning	14.0	1.3	55.6	1356.1
Feasibility	11.9	1.3	52.5	1294.2
Installation	9.5	1.1	61.8	1177.7
Filesystem	8.3	1.2	52.8	1295.0
Security	5.8	1.0	59.3	1546.9
Data Separation	4.7	1.1	56.0	614.7
Troubleshooting	4.6	1.1	47.2	1222.8
Network	3.8	1.0	55.2	673.1
Environment Variables	3.4	1.2	51.5	1138.3
Testing	3.0	1.2	44.5	575.2
Template	2.6	1.4	34.2	2558.0
Dependencies	2.5	1.1	51.4	944.5
Database	2.1	1.2	53.4	1021.0
Resource Ordering	1.7	1.3	46.9	799.3
Daemon Services	1.4	1.2	52.5	569.3

Table 3: Answer to RQ3: Ranked Order of Identified Question Categories

Metric	Category (Sorted in a Decreasing Order by Metric, from left to right)
AQ	Template, Resource Ordering, Feasibility, Provisioning, Daemon Services, Filesystem, Database, Testing, Environment Variables, Installation, Syntax Error, Troubleshooting, Dependencies, Data Separation, Security, Network
UNS	Installation, Security, Data Separation, Provisioning, Network, Database, Filesystem, Feasibility, Daemon Services, Environment Variables, Dependencies, Syntax Error, Troubleshooting, Resource Ordering, Testing, Template
VQ	Template, Security, Provisioning, Filesystem, Feasibility, Troubleshooting, Installation, Syntax Error, Environment Variables, Database, Dependencies, Resource Ordering, Network, Data Separation, Testing, Daemon Services

to study common challenges in web development. They observed interest in web development is drawing increasing interest amongst SO participants. Bajaj et al. [1] also observed that form validation issues and new HTML features to have more difficulty than other challenges. Rosen and Shihab [19] performed an empirical study on SO data to identify the discussion topics related to mobile application development. They found that mobile developers most often asked about app distribution, mobile development libraries, and user interface (UI) development. Pinto et al. [12] analyzed SO posts to identify questions related to energy consumption. They identified five main themes of questions: 'measurement', 'general knowledge', 'code design', 'context specific', and 'noise'. Reboucas et al. [18] studied the use of Swift programming language on SO, and observed that even though programmers find Swift easy to understand, 17.5% of the posted questions are related to the basic elements of Swift.

The above-mentioned studies indicate that Q&A websites provide researchers opportunity to analyze the question categories of programmers that are related to energy issues of software, mobile development, and web development. We take inspiration from

Table 4: Answer to RQ4: Temporal Trends

Category	Cox Stuart, p-value	Trend
Syntax Error	⇑, 0.02	Increasing
Provisioning	⇑, 0.5	Consistent
Feasibility	⇓, 0.02	Decreasing
Installation	⇑, 0.01	Increasing
Filesystem	⇑, 0.16	Consistent
Security	⇑, 0.09	Consistent
Data Separation	⇑, < 0.001	Increasing
Troubleshooting	⇑, 0.06	Consistent
Network	⇑, < 0.001	Increasing
Environment Variables	⇑, 0.01	Increasing
Testing	⇑, < 0.001	Increasing
Template	⇑, 0.43	Consistent
Dependencies	⇑, < 0.001	Increasing
Database	⇑, 0.50	Consistent
Resource Ordering	⇑, 0.24	Consistent
Daemon Services	⇑, 0.008	Increasing

these studies and focus our research efforts in identifying question categories related to CaC.

6 CONCLUSION

Similar to any new technology that is slowly getting popular, programmers ask questions about CaC and seek solutions to answer the questions. CaC is a fundamental pillar to implement continuous deployment, and by identifying the questions programmers ask about, we can understand in which areas programmers are facing challenges with. In this paper, we focus on a popular CaC tool, Puppet, and have analyzed 2,758 questions related to Puppet. We have identified 16 high-level questions using card sorting. We observe that questions related to Syntax Errors to appear most frequently, but questions related to Installation yield the most unsatisfactory answers. We observed Template-related questions to have the most views, on average. Based on our findings we recommend:

- **development of specialized integrated development environments (IDEs) for CaC** so that programmers can get assistance on questions they want answers from, for example, syntax, dependencies, and clarification on tutorials;
- **future research in CaC** that will investigate why programmers ask the identified question categories, and which of the categories are conceptual or implementation-related; and
- **future research in CaC** that will investigate if the identified questions are related with quality of CaC scripts, such as defects, anti-patterns, and security issues.

REFERENCES

[1] Kartik Bajaj, Karthik Pattabiraman, and Ali Mesbah. 2014. Mining Questions Asked by Web Developers. In *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*. ACM, New York, NY, USA, 112–121. https://doi.org/10.1145/2597073.2597083

[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 Softw. Engg.* 19, 3 (June 2014), 619–654. https://doi.org/10.1007/s10664-012-9231-y

[3] John L Campbell, Charles Quincy, Jordan Osserman, and Ove K Pedersen. 2013. Coding in-depth semistructured interviews: Problems of unitization and intercoder reliability and agreement. *Sociological Methods & Research* 42, 3 (2013), 294–320.

[4] D. R. Cox and A. Stuart. 1955. Some Quick Sign Tests for Trend in Location and Dispersion. *Biometrika* 42, 1/2 (1955), 80–95. http://www.jstor.org/stable/2333424

[5] Stack Exchange. 2017. Stack Exchange. https://data.stackexchange.com/. (2017). [Online; accessed 04-12-2017].

[6] Oliver Hanappi, Waldemar Hummer, and Schahram Dustdar. 2016. Asserting Reliable Convergence for Configuration Management Scripts. *SIGPLAN Not.* 51, 10 (Oct. 2016), 328–343. https://doi.org/10.1145/3022671.2984000

[7] Jez Humble and David Farley. 2010. *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation* (1st ed.). Addison-Wesley Professional.

[8] Yujuan Jiang and Bram Adams. 2015. Co-evolution of Infrastructure and Source Code: An Empirical Study. In *Proceedings of the 12th Working Conference on Mining Software Repositories (MSR '15)*. IEEE Press, Piscataway, NJ, USA, 45–55. http://dl.acm.org/citation.cfm?id=2820518.2820527

[9] Spencer Krum, William Van Hevelingen, Ben Kero, James Turnbull, and Jeffrey McCune. 2013. *Pro Puppet* (2nd ed.). Apress, Berkely, CA, USA.

[10] M. Linares-Vasquez, B. Dit, and D. Poshyvanyk. 2013. An exploratory analysis of mobile development issues using stack overflow. In *2013 10th Working Conference on Mining Software Repositories (MSR)*. 93–96. https://doi.org/10.1109/MSR.2013.6624014

[11] C. Parnin, E. Helms, C. Atlee, H. Boughton, M. Ghattas, A. Glover, J. Holman, J. Micco, B. Murphy, T. Savor, M. Stumm, S. Whitaker, and L. Williams. 2017. The Top 10 Adages in Continuous Deployment. *IEEE Software* 34, 3 (May 2017), 86–95. https://doi.org/10.1109/MS.2017.86

[12] Gustavo Pinto, Fernando Castor, and Yu David Liu. 2014. Mining Questions About Software Energy Consumption. In *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*. ACM, New York, NY, USA, 22–31. https://doi.org/10.1145/2597073.2597110

[13] Puppet. 2017. Puppet Documentation. https://docs.puppet.com/. (2017). [Online; accessed 04-01-2018].

[14] Akond Rahman. 2018. Reference to Examples. http://tiny.cc/example-reffs-so. (2018). [Online; accessed 30-01-2018].

[15] Akond Rahman, Asif Partho, David Meder, and Laurie Williams. 2017. Which Factors Influence Practitioners' Usage of Build Automation Tools?. In *Proceedings of the 3rd International Workshop on Rapid Continuous Software Engineering (RCoSE '17)*. IEEE Press, Piscataway, NJ, USA, 20–26. https://doi.org/10.1109/RCoSE.2017..8

[16] Akond Rahman and Laurie Williams. 2018. Characterizing Defective Configuration Scripts Used for Continuous Deployment. In *2018 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. To appear. Pre-print: http://akondrahman.github.io/papers/icst2018_tm.pdf.

[17] A. A. U. Rahman, E. Helms, L. Williams, and C. Parnin. 2015. Synthesizing Continuous Deployment Practices Used in Software Development. In *2015 Agile Conference*. 1–10. https://doi.org/10.1109/Agile.2015.12

[18] M. Reboucas, G. Pinto, F. Ebert, W. Torres, A. Serebrenik, and F. Castor. 2016. An Empirical Study on the Usage of the Swift Programming Language. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, Vol. 1. 634–638. https://doi.org/10.1109/SANER.2016.66

[19] Christoffer Rosen and Emad Shihab. 2016. What are mobile developers asking about? A large scale study using stack overflow. *Empirical Software Engineering* 21, 3 (01 Jun 2016), 1192–1223. https://doi.org/10.1007/s10664-015-9379-3

[20] Rian Shambaugh, Aaron Weiss, and Arjun Guha. 2016. Rehearsal: A Configuration Verification Tool for Puppet. *SIGPLAN Not.* 51, 6 (June 2016), 416–430. https://doi.org/10.1145/2980983.2908083

[21] Tushar Sharma, Marios Fragkoulis, and Diomidis Spinellis. 2016. Does Your Configuration Code Smell?. In *Proceedings of the 13th International Conference on Mining Software Repositories (MSR '16)*. ACM, New York, NY, USA, 189–200. https://doi.org/10.1145/2901739.2901761

[22] Xin-Li Yang, David Lo, Xin Xia, Zhi-Yuan Wan, and Jian-Ling Sun. 2016. What Security Questions Do Developers Ask? A Large-Scale Study of Stack Overflow Posts. *Journal of Computer Science and Technology* 31, 5 (01 Sep 2016), 910–924. https://doi.org/10.1007/s11390-016-1672-0

[23] T. Zimmermann. 2016. Card-sorting: From text to themes. In *Perspectives on Data Science for Software Engineering*, Tim Menzies, Laurie Williams, and Thomas Zimmermann (Eds.). Morgan Kaufmann, Boston, 137 – 141. https://doi.org/10.1016/B978-0-12-804206-9.00027-1