2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR) | 978-1-7281-8710-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/MSR52588.2021.00050

Rollback Edit Inconsistencies in Developer Forum

Saikat Mondal
University of Saskatchewan, Canada
saikat.mondal@usask.ca

Gias Uddin
University of Calgary, Canada
gias.uddin@ucalgary.ca

Chanchal K. Roy
University of Saskatchewan, Canada
chanchal.roy@usask.ca

***Abstract*—The success of developer forums like Stack Overflow (SO) depends on the participation of users and the quality of shared knowledge. SO allows its users to suggest edits to improve the quality of the posts (i.e., questions and answers). Such posts can be rolled back to an earlier version when the current version of the post with the suggested edit does not satisfy the user. However, subjectivity bias in deciding either an edit is satisfactory or not could introduce inconsistencies in the rollback edits. For example, while a user may accept the formatting of a method name (e.g., getActivity()) as a code term, another user may reject it. Such bias in rollback edits could be detrimental and demotivating to the users whose suggested edits were rolled back. This problem is compounded due to the absence of specific guidelines and tools to support consistency across users on their rollback actions. To mitigate this problem, we investigate the inconsistencies in the rollback editing process of SO and make three contributions. First, we identify eight inconsistency types in rollback edits through a qualitative analysis of 777 rollback edits in 382 questions and 395 answers. Second, we determine the impact of the eight rollback inconsistencies by surveying 44 software developers. More than 80% of the study participants find our produced catalogue of rollback inconsistencies to be detrimental to the post quality. Third, we develop a suite of algorithms to detect the eight rollback inconsistencies. The algorithms offer more than 95% accuracy and thus can be used to automatically but reliably inform users in SO of the prevalence of inconsistencies in their suggested edits and rollback actions.**

***Index Terms*—Stack Overflow, rollback edits, inconsistency**

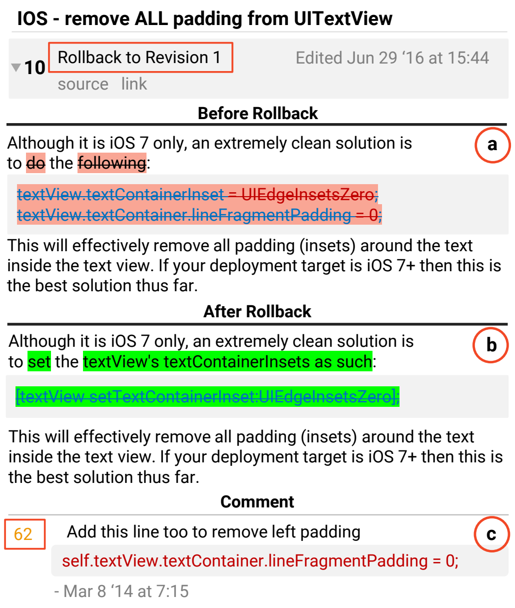
I. Introduction and Motivation

The online developer forum Stack Overflow (SO) is a crowd-shared technical knowledge-sharing platform. It serves as an informal documentation resource for software when the office resources can be lacking [66, 68]. The adoption, growth, and continued success of a crowd-sourced developer forum like SO depend on two major factors: the participation of users and the quality of the shared knowledge [5, 34, 62]. Unlike traditional knowledge sharing venues and stakeholders (e.g., paid events), the developers participating in crowd-sourced Q&A sites are driven by the need to – (1) learn from each other (which is often free), (2) become part of the community, and (3) feel the need to be recognized by peers in the community. Thus, the roles of knowledge seekers and providers in the crowd-sourced Q&A sites are often fluid and interchangeable. This elasticity leads to the design of a semi-decentralized system. However, the lack of authoritativeness can raise the concerns of trust in the quality of the shared contents [69].

To promote quality, SO introduces a collaborative editing system by allowing its users to suggest ways to improve the posts [35, 33, 14]. In particular, collaborative editing helps to keep posts clear, relevant, and up-to-date. For example, users often edit posts to fix grammatical and spelling mistakes, clarify the meaning, and add related resources or hyperlinks. However, such edits can be rejected by *rollbacks* due to undesired changes in posts. Rollback means reverting a post to a previous version in the edit history [19]. The reverted version then appears as the most recent item in the edit history.

In a recent study of SO rollback edits, Wang et al. [70] identified twelve reasons behind the rollback of suggested edits, e.g., undesired text formatting. The initial goal of our study was to develop a suite of machine learning classifiers to automatically detect those twelve rollback edit reasons. However, when we started to analyze the SO rollback edit reasons, we found several *inconsistencies* behind rollback edits. Here, rollback edit inconsistency means similar edits are rejected somewhere but accepted (i.e., brought back) somewhere else by rollback edits. Consider these revisions in [60], where one user made text formatting and grammatical fixes to a post and rejected gratitude (e.g., thanks) from the post. Then, another user rolled back the post to bring the gratitude back since such gratitude was accepted elsewhere. Unfortunately, with the gratitude, grammatical errors were brought back that undoubtedly hurt the quality of the post. Thus, the previous user complained, *"Why did you rollback all of the grammar and formatting fixes I made to your post?"* [61]. To deal with such a confusing situation, one user posted a question at meta [17] and asked either gratitude should be rejected by rollbacks or not. At least 990 users voted up this question (i.e., score = 990), and 284 users marked it as a favourite question. We have also found popular posts where users asked that rejecting status updates (e.g., EDIT: ...) and signature (e.g., Regards, Gustavo) from the body of posts are either right or wrong [24, 25]. It indicates that many users who edit SO posts are being confused by such inconsistencies. Thus, many more questions are posted at meta [16, 45] that discussed such inconsistent rollback edits and asked for suggestions on proper edit, review and rollback guidelines [27, 28, 29, 26, 58, 59, 20, 21, 17, 24, 22, 25].

Most worryingly, inconsistencies in rollback edits may result in incorrect post content. Consider the rollback edit with temporal inconsistency (i.e., multiple accepted edits are rejected by a single rollback) as shown in Fig. 1 that reverts an answer from revision 10 (see Fig. 1(a)) to 1 (see Fig. 1(b)). Thus, it rejects eight (i.e., two through nine) accepted edits at a time. In particular, the second line of the code was excluded from the code segment by this rollback. The question submitter asked for a code that can remove all paddings from UITextView. In Fig. 1(a), the second line of the code was


IOS - remove ALL padding from UITextView

10 Rollback to Revision 1 Edited Jun 29 '16 at 15:44
source link

Before Rollback

a

Although it is iOS 7 only, an extremely clean solution is to ~~do~~ the ~~following~~:

```
~~textView.textContainerInset = UIEdgeInsetsZero;~~
~~textView.textContainer.lineFragmentPadding = 0;~~
```

This will effectively remove all padding (insets) around the text inside the text view. If your deployment target is iOS 7+ then this is the best solution thus far.

After Rollback

b

Although it is iOS 7 only, an extremely clean solution is to set the textView's textContainerInsets as such:

```
~~[textView setTextContainerInset:UIEdgeInsetsZero];~~
```

This will effectively remove all padding (insets) around the text inside the text view. If your deployment target is iOS 7+ then this is the best solution thus far.

Comment

c

62 Add this line too to remove left padding

```
self.textView.textContainer.lineFragmentPadding = 0;
```

- Mar 8 '14 at 7:15

Fig. 1: An example of an inconsistent rollback edit [57]

added to remove left padding. However, the rollback made the solution incomplete as the revised version of the code could not remove the left padding from UITextView. Then, one user commented – *"add this line too to remove left padding self.textView.textContainer.lineFragmentPadding = 0;"* (Fig. 1(c)). A large number of users (60) acknowledged it as a valuable comment. Unfortunately, such a faulty answer was live for about nine months (June 29, 2016 – March 28, 2017). It was edited again and added the required line of code.

The above observations motivated us to shift our focus **from** developing classifiers to detect rollback edit reasons **to** develop solutions for automated analysis and detection of inconsistencies in SO rollback edits. In particular, to reduce inconsistencies in SO rollback edits, we make three contributions in this paper.

1) **A Catalogue of Eight Rollback Edit Inconsistency Types (Section II).** The motivating scenarios above show that inconsistencies exist in SO rollback edits. However, to properly guide the SO editing process, we need a catalogue of all possible inconsistency types in SO rollback edits. To produce the catalogue, we conduct a qualitative study of 777 rollback edits (382 questions + 395 answers). This sample is statistically representative of the entire list of 102K rollback edits in SO data dump of September 2019, which was the start time of the journey of this paper. For each suggested edit that was rolled back, we check whether the same suggested edit also contributed to an accepted edit. If so, we consider the rollback edit as inconsistent. We manually label the reason for the inconsistency. This study produced a catalogue of eight inconsistency types under two categories – (1) user-based inconsistency and (2) system-based inconsistency. *User-based* inconsistencies could arise due to the subjective bias of users rejecting suggested edits. Unlike user-based, *system-based* inconsistencies appear due to the inability of the edit system itself to regulate the rollbacks.

2) **Impact Analysis of the Eight Edit Inconsistency Types (Section III).** The motivating scenarios as we noted above offer exploratory viewpoints on the negative impact of edit inconsistencies. We further sought to capture quantitative evidence and SO user perspectives on the negative impacts of the eight inconsistency types. First, we compute several SO post popularity metrics (e.g., score, favorite count, etc.) of the SO posts in our qualitative study dataset. We divide the posts into two groups: posts that suffered at least one of the eight rollback edit inconsistencies and posts that did not suffer any such inconsistencies. We find that the posts that have undergone inconsistent rollback edits are significantly less popular than the posts that did not go through inconsistent edits. Second, to capture SO user perspectives on the eight inconsistency types, we survey 44 developers from SO. More than 80% of the participants agree that the eight inconsistent types in rollback edits hurt the quality of the post.

3) **Automatic Detection of the Eight Edit Inconsistency Types (Section IV).** Our empirical and user studies show that the eight edit inconsistency types we found are detrimental to the SO post quality. Therefore, we need to warn SO users of the prevalence of any inconsistencies in their suggested edits and during their rollback actions. An automated approach is better as it can be scalable. We develop eight rule-based algorithms to detect the eight inconsistency types in rollback edits. The algorithms offer more than 95% accuracy (on average). We run the algorithms on the entire dataset of around 102K rollback edits that we found in the SO September 2019 data dump. We find that presentation and status inconsistencies are the most prevalent, followed by temporal inconsistency. A temporal inconsistency arises when already accepted multiple edits get rejected by a rollback. Such inconsistency could make multiple valuable contributions to the accepted edits useless.

The findings from our study can guide 1) **Forum Designers** to improve the edit system, 2) **Forum Users** to shape their edit behavior, and 3) **Researchers** to study collaborative editing.

Replication Package that contains all the survey responses has is shared in our online appendix [40].

II. A Catalogue of Rollback Edit Inconsistencies

In this section, we answer the following research question.

RQ1. What are the different types of inconsistencies in SO rollback edits?

As we noted in Section I, inconsistencies exist in SO rollback edits, i.e., the same suggested edit was accepted in one post but rejected in another post. However, we are aware of no previous study that offered insights into the different types of inconsistencies. We aim to produce a catalogue of inconsistency types in SO rollback edits by answering RQ1.

A. Approach

We conduct a qualitative study of 777 SO rollback edits to produce our catalogue of inconsistency types. The study has five steps that are discussed below.

(1) **Data Collection.** We download the September 2019 SO data dump [31] which contains all the essential information about questions and answers. It stores the history of all the events (e.g., edit body, rollback body, post deleted) for each post. It also includes the date of each event, the user who triggered the event.

(2) **Data Preprocessing.** There are 38 types of events that are tracked by SO [30]. In this study, we investigate the revisions where the suggested edits in the body (i.e., *PostHistoryTypeId* = 8) of a post were rolled back. We focus on body edits because the body of a post contains the majority of content. We get 63,071 question revisions and 39,218 answer revisions whose body edits were rejected by rollbacks.

(3) **Random Sampling of Data for Qualitative Analysis.** To achieve a confidence level of 95% with a confidence interval of 5% [7, 70], we randomly sampled 382 from 63,071 rejected question revisions, and 395 (sample size = 381 < 395) from 39,218 rejected answer revisions. To compute the size of our random sample, we use the following formula $\frac{Nz^2p(1-p)}{e^2N+z^2p(1-p)}$, where N is the population size (e.g., 63,071), z is the Z-score corresponding to a particular confidence level (e.g., 1.96 for a confidence level of 95%), e is the confidence interval (e.g., 5%), and p is population proportion (e.g., 0.5).

(4) **Qualitative Analysis to Identify Rollback Inconsistencies.** The first two authors of this paper analyzed the rollback edit reasons that were identified by Wang et al. [70]. While investigating the reasons behind rollback, we find several inconsistencies. That is, similar edits are being rejected somewhere and also brought them back somewhere by rollbacks. We, two investigators, discuss the rollback inconsistencies in multiple interactive sessions. Next, we analyze 200 rollback edits (100 questions + 100 answers) from our selected dataset and label the inconsistencies. For a given rollback edit in our dataset, we meticulously analyze (1) the history of all suggested edits of the post before the rollback edit and (2) our list of common actions that are rejected and accepted (i.e., brought them back) by rollback edits. We consider and categorize a rollback edit as inconsistent if it shows characteristics similar to an accepted edit. We repeated the labeling process to form higher categories. This manual investigation produces a catalogue of eight inconsistency types. We then measure the agreement using Cohen's Kappa [13, 12]. The value of κ was 0.98, which means the strength of the agreement is almost perfect. We resolve the remaining few disagreements by discussion. This means that any coder can do the rest of the labeling without introducing individual bias. The first author of this paper then analyzes the remaining dataset and manually labels the inconsistencies.

B. Results

Our manual analysis reveals eight inconsistency types in the SO rollback edits. Fig. 2 summarizes the inconsistencies into two categories: user-based and system-based. User-based inconsistencies arise due to the subjective bias of users rejecting the suggested edits. System-based inconsistencies focus on the inconsistencies that could appear due to the inability of the edit system itself to regulate the rollbacks. About 41% of the edits in our dataset have one or multiple inconsistencies. A single rollback could exhibit multiple inconsistencies. Among all the inconsistent rollback edits, 79.6% have single inconsistency, whereas 20.4% have more than one inconsistency. The rollback inconsistencies are discussed as follows.

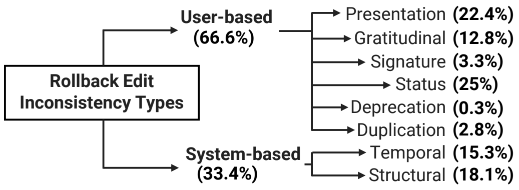
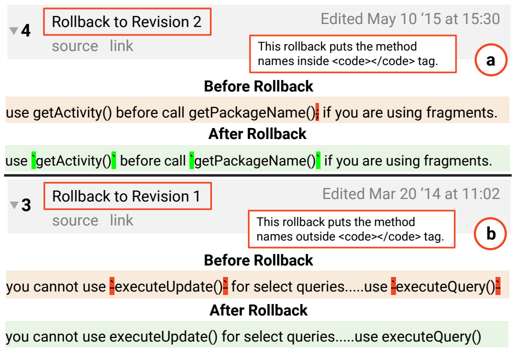


Fig. 2: Rollback inconsistency types and their distribution in our manually analyzed dataset.

• **User-based Inconsistencies** accounted for around 66.6% of all inconsistencies in our manual analysis. We found six types of user-based inconsistencies as discussed below.


4 Rollback to Revision 2 — Edited May 10 '15 at 15:30
source link
This rollback puts the method names inside <code></code> tag. (a)

Before Rollback
use getActivity() before call getPackageName() if you are using fragments.
After Rollback
use getActivity() before call getPackageName() if you are using fragments.

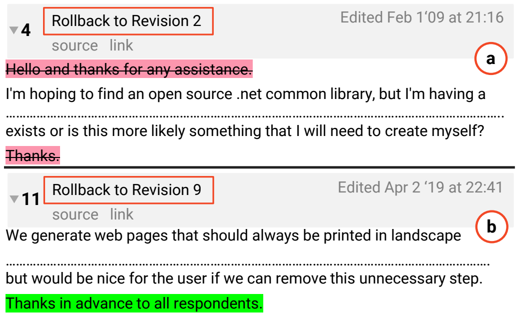
3 Rollback to Revision 1 — Edited Mar 20 '14 at 11:02
source link
This rollback puts the method names outside <code></code> tag. (b)

Before Rollback
you cannot use executeUpdate() for select queries.....use executeQuery()
After Rollback
you cannot use executeUpdate() for select queries.....use executeQuery()

Fig. 3: An example of presentation inconsistency.

(1) ***Presentation Inconsistency.*** Presentation inconsistency refers to the different presentation styles of similar text or code terms. Consider the example shown in Fig. 3, where the first rollback [52] (Fig. 3 (a)) was made to put the method names (e.g., `getActivity()`) inside the code tags (`` in markdown denotes a <code></code> tag). On the contrary, the second rollback [53] (Fig. 3 (b)) was made to reject an suggested edit where method names (e.g., `executeUpdate()`) were put inside the code tags. Such inconsistent rollbacks certainly confuse the users. In another case, one user was frustrated when his suggested code formatting was rejected, and the answer was reverted to a faulty version. Then, he complained, *"Why did you rollback my edit? Your answer is incorrectly formatted, and you spelled 'position' wrong."* [50]. Such presentation inconsistency also includes changing the cases of texts (e.g., uppercase/lowercase), the format of texts (e.g., bold/italic), adding or removing space/newline, creating bullet/number list, replacing acronym by its root words, or bringing the acronyms back, adding links as standard texts/hypertext. Users often reject or bring such styles back by rollback edits.

According to our investigation, formatting and unformatting the inline code terms is the most frequently seen presentation inconsistency in rollback edits. In this study, we thus only consider them for processing convenience. As shown in Fig. 2, we find that 22.4% of rollback edits have presentation inconsistency among all the inconsistent rollback edits.


4 Rollback to Revision 2 Edited Feb 1'09 at 21:16
source link
a
~~Hello and thanks for any assistance.~~
I'm hoping to find an open source .net common library, but I'm having a

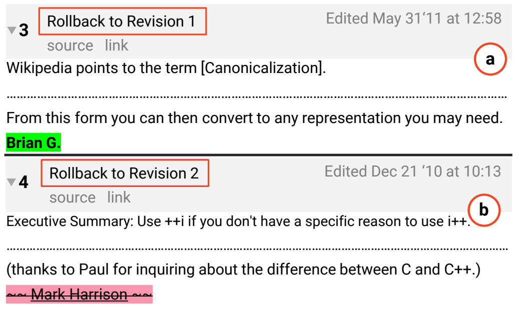
exists or is this more likely something that I will need to create myself?
~~Thanks.~~

11 Rollback to Revision 9 Edited Apr 2 '19 at 22:41
source link
b
We generate web pages that should always be printed in landscape

but would be nice for the user if we can remove this unnecessary step.
Thanks in advance to all respondents.

Fig. 4: An example of gratitudinal inconsistency.

(2) ***Gratitudinal Inconsistency.*** Gratitudinal inconsistency refers to the dual viewpoint of rejecting gratitudes. Consider the example, as shown in Fig. 4(a), where the gratitude (e.g., thanks for any assistance) was rejected by a rollback [43]. Conversely, Fig. 4(b) shows a rollback where the gratitude (e.g., thanks in advance to all respondents) was brought back [44]. Such inconsistency even confuses the users who have been editing many posts over the years. To mitigate confusion, they often ask questions and seek opinions from others [17, 22]. Many users argued that gratitude should be accepted. For example, one user said, *"If the post has nothing else wrong with it and is just book-ended with "Hi/Thanks" then you can probably pass on the edit."* [17]. However, many others express opposite opinions. One user boldly disagreed with accepting gratitudes, *"I've always been against the greetings and salutations."* [17]. Thus, such inconsistency needs to be addressed and resolved. According to our investigation, 12.8% of rollback edits have gratitudinal inconsistency (Fig. 2).


3 Rollback to Revision 1 Edited May 31'11 at 12:58
source link
a
Wikipedia points to the term [Canonicalization].

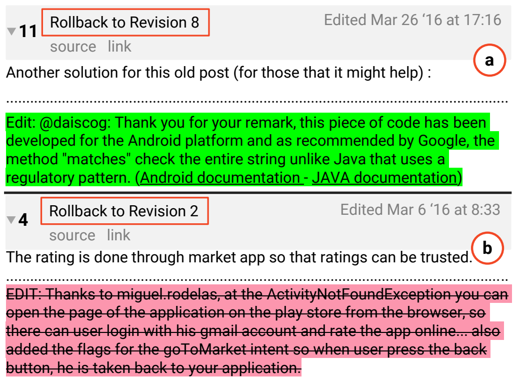
From this form you can then convert to any representation you may need.
Brian G.

4 Rollback to Revision 2 Edited Dec 21 '10 at 10:13
source link
b
Executive Summary: Use ++i if you don't have a specific reason to use i++.

(thanks to Paul for inquiring about the difference between C and C++.)
~~-- Mark Harrison --~~

Fig. 5: An example of signature inconsistency.

(3) ***Signature Inconsistency.*** Users sometimes add their signatures (e.g., name, ID) at the bottom of the posts. In SO, every post/edit is signed with a standard user card, which is linked directly to the user page. Many users thus suggest rejecting the signature. For example, one user said, *"If you use an additional signature or tagline, it will be removed to reduce noise in the questions and answers."* [18]. On the contrary, some users would like to sign their signatures when they significantly contribute to a post. During the manual investigation, we see that rollback edits rejected signatures in some cases (e.g., Fig. 5(b) [46]), while such signatures were brought back in some other cases (e.g., Fig. 5(a) [47]) by rollback edits. As shown in Fig. 2, 3.3% of inconsistent rollback edits have signature inconsistency.

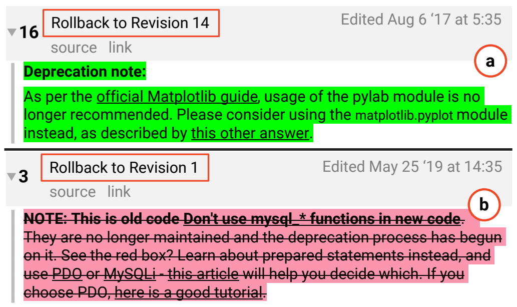

11 Rollback to Revision 8 Edited Mar 26 '16 at 17:16
source link
a
Another solution for this old post (for those that it might help) :

Edit: @daiscog: Thank you for your remark, this piece of code has been developed for the Android platform and as recommended by Google, the method "matches" check the entire string unlike Java that uses a regulatory pattern. (Android documentation - JAVA documentation)

4 Rollback to Revision 2 Edited Mar 6 '16 at 8:33
source link
b
The rating is done through market app so that ratings can be trusted.

~~EDIT: Thanks to miguel.rodelas, at the ActivityNotFoundException you can open the page of the application on the play store from the browser, so there can user login with his gmail account and rate the app online... also added the flags for the goToMarket intent so when user press the back button, he is taken back to your application.~~

Fig. 6: An example of status inconsistency.


16 Rollback to Revision 14 Edited Aug 6 '17 at 5:35
source link
a
Deprecation note:
As per the official Matplotlib guide, usage of the pylab module is no longer recommended. Please consider using the matplotlib.pyplot module instead, as described by this other answer.

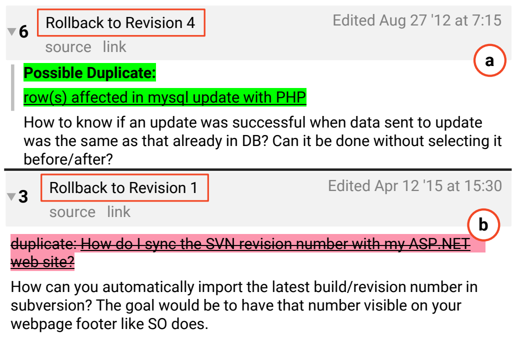
3 Rollback to Revision 1 Edited May 25 '19 at 14:35
source link
b
~~**NOTE: This is old code Don't use mysql_* functions in new code.** They are no longer maintained and the deprecation process has begun on it. See the red box? Learn about prepared statements instead, and use PDO or MySQLi - this article will help you decide which. If you choose PDO, here is a good tutorial.~~

Fig. 7: An example of deprecation inconsistency.

(4) ***Status Inconsistency.*** Users often add status (i.e., personal notes) to clarify others' confusion, append essential messages that were missed during post submission, and acknowledge others' responses. For example, Fig. 6 presents two similar statuses, where users acknowledged the responders and added important notes. One status (e.g., Fig. 6 (a)), was brought back by a rollback [48]. Unfortunately, a rollback rejected another status (e.g., Fig. 6 (b)) [54]. We also find conflicting opinions in many questions posted at meta Stack Exchange ([24, 23]). One user said, *"EDIT and UPDATE are rarely needed, nor helpful. For future readers, posts need to be standalone, without any history"* [23]. However, some other users do not think that a status update is necessarily harmful. It could draw attention to new information. Such a dual viewpoint of accepting the personal status confuses and frustrates users who suggest edits. As you see in Fig. (Fig. 2), 25% of inconsistent rollback edits have status inconsistency.

(5) ***Deprecation Inconsistency.*** Deprecation inconsistency refers to the dual viewpoints of rejecting deprecation notes. As you see in Fig. 7 (a), one user brought the deprecation note

back by a rollback [42]. On the contrary, such a note was rejected by a rollback (e.g., Fig. 7 (b)) [55]. However, such inconsistency is found infrequently in our selected dataset. Less than 1% inconsistent rollback edits of answers have deprecation inconsistency (Fig. 2).


6 Rollback to Revision 4 Edited Aug 27 '12 at 7:15

source link

a

Possible Duplicate:

row(s) affected in mysql update with PHP

How to know if an update was successful when data sent to update was the same as that already in DB? Can it be done without selecting it before/after?

3 Rollback to Revision 1 Edited Apr 12 '15 at 15:30

source link

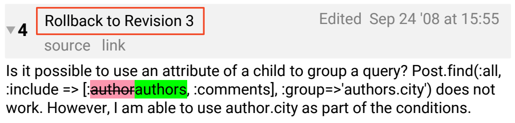
b

~~duplicate: How do I sync the SVN revision number with my ASP.NET web site?~~

How can you automatically import the latest build/revision number in subversion? The goal would be to have that number visible on your webpage footer like SO does.

Fig. 8: An example of duplication inconsistency.

(6) ***Duplication Inconsistency.*** Duplication inconsistency refers to the dual viewpoints of rejecting duplication notes. As you see in Fig. 8 (b), a duplication note was rejected from the body of a question by a rollback [56]). On the contrary, such a note was brought back by rollbacks (e.g., Fig. 8 (a) [51]). About 3% inconsistent rollback edits of questions have duplication inconsistency (Fig. 2).


4 Rollback to Revision 3 Edited Sep 24 '08 at 15:55

source link

Is it possible to use an attribute of a child to group a query? Post.find(:all, :include => [:~~author~~authors, :comments], :group=>'authors.city') does not work. However, I am able to use author.city as part of the conditions.

Fig. 9: An example of structural inconsistency.

• **System-based Inconsistencies.** Among all the rollback inconsistencies, 33.4% of them were system-based inconsistencies (Fig. 2). The system-based inconsistencies are discussed as follows.

(1) ***Temporal Inconsistency.*** Temporal inconsistency arises when already accepted multiple edits get rejected by a single rollback edit. Hence, it makes previously accepted multiple edits useless. According to our investigation, such inconsistent rollback edits not only ignore the quality contents but also could revert the post to an error-prone revision. As shown in Fig. 1, a user rollbacks a revision from 10 to 1 and thus rejects the revisions 2 through 9. Unfortunately, the answer was reverted to a faulty version. Temporal inconsistency contributes to 15.3% of all rollbacks in our dataset.

(2) ***Structural Inconsistency.*** Structural inconsistency reverts a post to the immediate previous revision. Consider an example [49] shown in Fig. 9, where a rollback reverted a post from revision 4 to revision 3. Revision 4 can be revised (i.e., edited) version of 3 since there are no revisions to revert in between 3 and 4. According to our manual investigation, about 18% of inconsistent rollback edits have structural inconsistency (Fig. 2).

III. Impact of Rollback Edit Inconsistencies

In this section, we answer the following two research questions.

RQ2. Is there any correlation between post quality and the eight rollback edit inconsistencies? (Section III-A)
RQ3. How do developers perceive the impact of the eight rollback edit inconsistency types? (Section III-B)

A. Correlation between post quality and rollback edit inconsistencies (RQ2)

We separate the posts in our dataset from RQ1 based on whether their revision history has inconsistent rollback. We then analyze the correlation between posts with inconsistent and consistent rollback edits based on their popularity metrics. In the Software Engineering literature, the three metrics (average view count, favorite count, and scores) are used to measure the popularity/quality, and the two metrics (e.g., number of answers and time to get an accepted answer) are used to estimate the difficulty of a question [4, 2]. The findings from the correlation analysis are discussed as follows.

• **Posts with inconsistent rollback edits have lower popularity scores.** The quality of a post in SO is subjectively evaluated by the users through a voting mechanism. High-quality posts are generally voted up, whereas vague, unclear, or inconsistent posts are likely to be voted down. The net votes (upvotes − downvotes) cast against a given post in SO form an evaluation metric called score, which approximates the posts' quality. Fig. 10 (a) shows the box plots for the score of our selected posts against inconsistent and consistent rollback edits. Here, we normalize the score between 0 and 1. We see that posts with inconsistent rollback edits have significantly lower scores than the counterpart. We use Mann-Whitney-Wilcoxon test, a non-parametric statistical significance test [37] and get statistically significant p-value (i.e., p-value = $0 < 0.05$). We also examine the effect size using Cliff's delta test and find medium effect size, i.e., Cliff's $|d| = 0.37$ (medium) with 95 percent confidence.

We also find similar results for the other two metrics (i.e., favorite count, view count) that measure the popularity of posts. According to our statistical analysis, we see that questions with inconsistent rollback edits have lower favorite count (Fig. 10(c)) and view count (Fig. 10(d)) than the counterpart. We find the differences are statically significant (i.e., p-value < 0.05) with small effect size.

• **Questions with inconsistent rollback edits have less answers.** Fig. 10(b) shows the box plots for the answer count against questions with consistent and inconsistent rollback edits. We clearly see that questions with inconsistent rollback edits receive less answers on average than the questions with consistent rollback edits. We also find a significant difference in the number of answers between two types of questions. Mann-Whitney-Wilcoxon test shows p-value < 0.05, whereas Cliff's $|d| = 0.18$ (small) with 95% confidence.

• **Questions with inconsistent rollback edits take more time to get an accepted answer.** We investigate whether

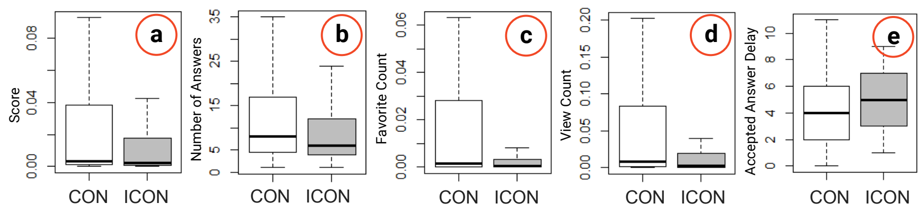


Fig. 10: (a) Score of posts, (b) number of answers, (c) favorite counts, and (d) view counts per question, and (e) time delay between question and accepted answer with consistent (CON) vs inconsistent (ICON) rollback edits

there is a negative correlation between the delay of getting an accepted answer and the presence of inconsistencies. We thus determine the delay (in minutes) between the submission time of a question and receiving an accepted answer. Fig. 10(e) shows the box plots for the delay of getting the accepted answers against questions with consistent and inconsistent rollback edits. We see that the mean and median delay of getting an acceptable answer for the question with inconsistent rollback edits is higher than the counterpart. That is, inconsistency has a negative impact on receiving the appropriate solutions (i.e., accepted answers). However, the difference is not statistically significant.

B. Perceived impact of the eight rollback edit inconsistency types in by developers (RQ3)

We survey 44 SO developers to determine the impact of the eight inconsistency types. We recruited participants who have software development experience (0 – 15+ years) and who are active in editing SO posts. We recruit 24 participants via a snowball approach where we first contacted some participants via personal contact, who then recommended our study to other eligible participants. The rest 20 participants are selected from 33 interested developers whose information we got via advertisements in Facebook and LinkedIn. We ask questions to the survey participants to determine the impact of the inconsistencies on user engagement (Section III-B1) and on the quality of SO posts (Section III-B2).

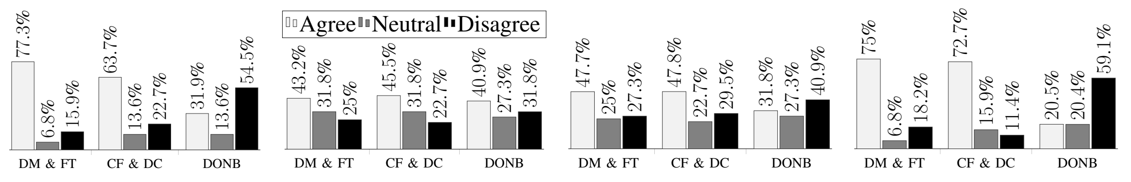
1) Impacts on User Engagement: We presented one example of each of the eight inconsistency types to the participants. Each example was accompanied by a description to ensure the participants understand the underlying context. For each inconsistency type, we asked the following question to the participants: *How does inconsistency in rollback editing impact their participation and contribution to share knowledge in Stack Overflow?* The options were: 1) Demotivate and frustrate you for suggesting edits, 2) Confuse and discourage you for suggesting edits, 3) It does not bother me. The participants were asked to provide their agreement/disagreement under each option in a Likert scale: Agree (Strongly Agree, Agree), Disagree (Strongly Disagree, Disagree), and Neutral. We first analyze the findings for each of the inconsistency types. Then, we analyze the results according to the participants' profession. Finally, we analyze the findings according to the participants' professional experience.

Fig. 11 (11a – 11f) shows the percentage of agreement and disagreement with each of the three given options for user-based inconsistency types. We see that presentation, status, deprecation, and duplication inconsistencies frustrate, confuse and discourage users from suggesting edits in SO. More than 73% (on average) of the participants agree that such inconsistencies demotivate and frustrate them for suggesting edits in SO. Besides, more than 65% (on average) of participants admit that the above inconsistencies confuse and discourage them from suggesting edits. Rollback inconsistencies also highly bother them to contribute to editing to improve posts' quality. Gratitudinal and signature inconsistencies are less likely demotivating than the above mentioned four inconsistencies. However, more than 46% of participants (on average) agree that they demotivate, confuse, and discourage them from suggesting edits. On the contrary, only 26% of participants (on average) disagree that inconsistencies demotivate them for suggesting edits. Overall, 41% more participants agree that user-based inconsistencies demotivate, frustrate, confuse, and discourage them, which is statically significant.

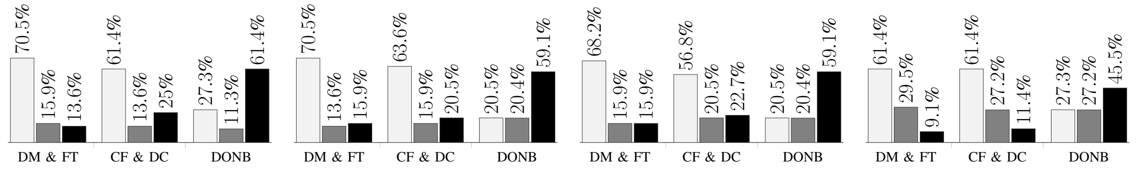
Similar to user-based inconsistencies, we find about 65% of participants agree that system-based inconsistencies (temporal and structural) demotivate and frustrate them (see Fig. 11g & Fig. 11h). In contrast, only about 13% of participants do not agree. About 60% of participants agree that such inconsistencies also confuse and discourage them from suggesting edits, and only 17% of participants disagree. Overall, 47% more participants agree that system-based inconsistencies demotivate, frustrate, confuse, and discourage them, which is statically significant.

We analyze the agreement/disagreement results according to the professions (e.g., software developer, technical lead, academic practitioners) of survey participants. According to our analysis, 62% – 69% participants agree that inconsistent rollback edits demotivate and frustrate, and 55% – 81% participants agree that such inconsistencies confuse and discourage them from suggesting edits. Only 13%–26% of participants disagree that such inconsistencies demotivate, frustrate, confuse, and discourage them from suggesting edits.

We see similar results as the previous section when we analyze by professions. However, we find that the highly experienced participants (e.g., 15+ years) frustrate or confuse lower (44% – 50%) than those of less experienced participants. They tend to take a lot more stress and handle diverse jobs daily in their profession. That's why they are not so concerned about such inconsistencies. However, more than 56% of them agree that such inconsistent rollback edits bother them.



(a) Presentation inconsistency (b) Gratitudinal inconsistency (c) Signature inconsistency (d) Status inconsistency



(e) Deprecation inconsistency (f) Duplication inconsistency (g) Temporal inconsistency (h) Structural inconsistency

Fig. 11: Impact of the inconsistency types in rollback edits in participation and contribution of users to share knowledge (**DM**: Demotivate, **FT**: Frustrate, **CF**: Confuse, **DC**: Discourage, **DONB**: Does Not Bother).

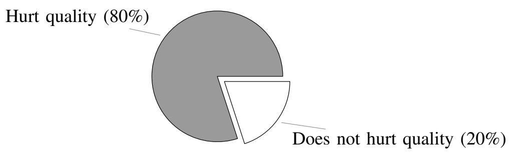
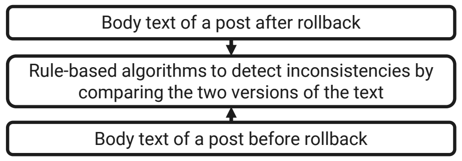


Fig. 12: Impacts of the rollback inconsistency on post quality



Body text of a post after rollback
Rule-based algorithms to detect inconsistencies by comparing the two versions of the text
Body text of a post before rollback

Fig. 13: Overview of inconsistency detection.

2) Impacts on the Quality of the Posts: During our manual analysis, we see that inconsistent rollback edits often discard the required edits that improve the quality of the posts. In this survey, we attempt to validate our findings by seeking the participants' opinions on whether they agree that inconsistent rollbacks hurt the quality of the posts. We asked the following question to the participants: *Do you think such inconsistencies in rollback edits hurt the quality of the shared content?* The options were: 1) Yes, 2) No. According to our analysis, 80% of survey participants agree that inconsistencies in rollback edits hurt the quality of the SO posts (see Fig. 12).

IV. Detection of Rollback Edit Inconsistencies

In this section, we answer the following research question.

RQ4. How effectively can we automatically detect the eight inconsistency types in SO rollback edits?

Our findings from Section III show that that inconsistent rollback edits should be identified and discouraged to – (1) promote the quality of posts, and (2) ensure better user experience. We design a suite of rule-based algorithms to detect the eight inconsistencies in the SO rollback edits (Section IV-A). In Section IV-B, we report the performance on the entire dataset of around 102K rollback edits that we found in the SO September 2019 data dump.

A. Algorithms

We present eight algorithms to detect the eight rollback inconsistencies we presented in Section II. Fig. 13 shows an overview of our inconsistency detection technique. We extract texts before and after the rollback of a post. Then we apply our algorithms that detect inconsistencies by comparing the two versions of texts. We present the algorithms below.

(1) ***Presentation Inconsistency.*** SO users are suggested to put inline code terms (i.e., code terms within the textual description) inside the <*code*>..</*code*> tags and code segments inside the <*code*> under <*pre*> tag. Here, we only consider the presentation inconsistency of inline code terms and thus remove the content under the <pre> tag from the body of a post. Now, consider the following terms. TXT_{br}: texts before rollback, TXT_{ar}: texts after rollback, LCT_{br}: list of inline code terms before rollback, and LCT_{ar}: list of inline code terms after rollback. Then, we identify the presentation inconsistency in the following ways.

IF both LCT_{br} and LCT_{ar} are null OR $LCT_{br} == LCT_{ar}$, THEN there is no presentation inconsistency.

IF any of the code terms of LCT_{br} are not in LCT_{ar} but in TXT_{ar} (i.e., there is at least one element that was formatted as code terms in the text before rollback but formatted as non-code terms in the text after rollback), THEN there is presentation inconsistency.

IF any of the code terms of LCT_{ar} are not in LCT_{br} but in TXT_{br}, THEN there is presentation inconsistency.

(2) ***Gratitudinal Inconsistency.*** We find several keywords during the manual investigation that are related to gratitude such as – *welcome, thanks, sorry, appreciated, thank, ty* (*i.e.,* thank you), *thx, regards,* and *tia* (*i.e.,* thanks in advance). We thus look for those keywords using regular expressions. Then, we attempt to identify the gratitudinal inconsistency in the following ways.

IF we do not find a keyword match to either TXT_{br} or TXT_{ar} OR we find a keyword match to both TXT_{br} and TXT_{ar}, THEN there is no gratitudinal inconsistency.

IF we find a keyword match to TXT_{br} but do not find match to TXT_{br} OR vice versa (i.e., rollback edit either rejects or accepts gratitude), THEN there is gratitudinal inconsistency.

(3) ***Signature Inconsistency.*** We first extract user information, particularly the names of two users: (i) who rollback the edit, (ii) whose edit has been rolled back. We then look for those names using regular expressions.

IF we do not find a match to any of the two names (full or part) to either TXT_{br} or TXT_{ar} OR we find a match to both TXT_{br} and TXT_{ar}, THEN there is no signature inconsistency.

IF we find a match to any of the two names (full or part) to TXT_{br} but do not find a match to TXT_{br} OR vice versa (i.e., rollback edit either reject or accept signature), THEN there is signature inconsistency.

(4) ***Status Inconsistency.*** During the manual investigation, we find that statuses are often updated followed by several keywords such as – *edit, update, note,* and *ps*. We thus look for those keywords using regular expressions. We apply the same detection technique, as discussed in gratitudinal inconsistency, to decide whether there is status inconsistency.

(5) ***Deprecation Inconsistency.*** To mark an answer that discusses deprecated technology (*e.g.,* API), users add a message followed by keywords *deprecation, deprecate*. We thus look for those keywords using regular expressions. We apply the same detection technique, as discussed in gratitudinal inconsistency, to decide whether there is deprecation inconsistency.

(6) ***Duplication Inconsistency.*** To mark a duplicate question, users add a duplicate message followed by keywords such as – *duplicate, duplication, related to*. We thus look for those keywords using regular expressions. We apply the same detection technique, as discussed in gratitudinal inconsistency, to decide whether there is duplication inconsistency.

(7) ***Temporal Inconsistency.*** To detect temporal inconsistency, we find the current revision number (say, R_c) and the revision number where the post has been reverted by a rollback (say, R_p). We then compute their difference, $d = R_c - R_p$.

IF $d > 2$, THEN there is temporal inconsistency.

(8) ***Structural Inconsistency.*** To detect structural inconsistency, we find the current revision number (say, R_c) and the revision number where the post has been reverted by a rollback (say, R_p). We then compute their difference, $d = R_c - R_p$.

IF $d == 1$, THEN there is structural inconsistency.

B. Performance of the Algorithms

We create an evaluation corpus as follows. 1) We collected all SO rollback edits from September 2019 dump. This was the latest dump available during the time of our analysis. The dump contains a total of 102K rollback edits. We used a dump instead of the SO API to ensure reproducibility of our analysis. 2) We ran each of our eight algorithms on the 102K rollback edits to identify the inconsistency each edit may exhibit. 3) We randomly picked 400 rollback edits out of the 102K edits. Out of the 400 edits, we pick 200 such edits where our algorithm did not find any inconsistency and 200 where our algorithm found one or more inconsistencies. This sample size is statistically significant with a 95% confidence level and 5% confidence interval.

To analyze the accuracy of the algorithms, we manually label the sampled 400 rollback edits in a file as follows. 1) **Got**: the inconsistency detected by the algorithm. 2) **Expected**: the actual inconsistency based on our manual analysis. We then create a confusion matrix to analyze the performance of the algorithm as follows. 1) True Positive (TP) = 'got' inconsistency = 'expected' inconsistency 2) False Positive (FP) = ('got' inconsistency $\neq$ 'expected' inconsistency) or ('got' inconsistency but 'expected' no inconsistency) 3) True Negative (TN) = 'got' no inconsistency and 'expected' no inconsistency, and 4) False Negative (FN) = 'got' no inconsistency but 'expected' one or more inconsistency. Using the above matrix, we compute four standard metrics (Precision P, Recall R, F1-score $F1$, and Accuracy A) [38] to compute the performance of each algorithm.

TABLE I: Performance of inconsistency detection algorithms

Inconsistency	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Presentation	1	0.98	0.99	0.99
Structural	1	1	1	1
Temporal	1	1	1	1
Gratitudinal	1	1	1	1
Status	1	1	1	1
Duplication	1	0.94	0.97	0.99
Signature & Deprecation	0.33	1	0.5	0.99
Overall	0.99	0.99	0.99	0.99

Table I shows the performance of our algorithms. Overall, our algorithms show high precision and recall (i.e., 0.99). The precision is 1 for each algorithm except signature & deprecation. For the signature inconsistencies, the misclassifications happened due to some users adding a different name (*e.g.,* nickname) as a signature. That is, the name they added is different from their SO account name. Thus when we attempt to match the name with the body text of a question or answer, we did not find the name.

V. Discussions

In this section, we report the prevalence of the eight inconsistency types in the entire SO dump of 2019 (Section V-A). We then discuss the implications of our study (Section V-B).

A. Prevalence of the Eight Inconsistencies in Entire SO

We apply our eight inconsistency detection algorithms from RQ4 on the entire rollback edits of the SO data dump of September 2019. Fig. 14 shows the distribution of inconsistencies in the dataset. Temporal inconsistency is the most frequent in the rollback edits. We find that about 33% of the inconsistent rollback edits have temporal inconsistency. A high percentage of temporal inconsistency suggests that users often

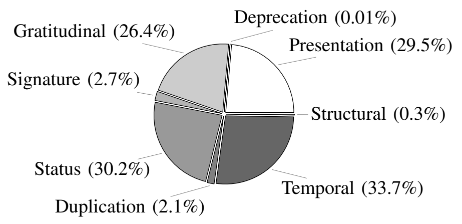


Fig. 14: Inconsistency distribution of rollback edits on the entire SO dump of September 2019.

perform less important or unwanted edits. However, we see that rejecting multiple accepted edits together also ignore the quality content (e.g., Fig. 1). In our dataset, 34,602 accepted edits were rejected by 14,479 rollback edits with temporal inconsistency. That is, on average, 2.4 accepted edits were rejected per rollback with temporal inconsistency.

Status inconsistency is the second most prevalent inconsistency. It was seen that about 30% of the inconsistent rollback edits have status inconsistency. That is, personal messages and acknowledgments are often rejected or brought back by rollback edits. The presentation inconsistency in rollback edits was seen as the third most frequent (i.e., 29.5%) among all the inconsistencies. New programming languages and updated versions of existing programming languages come with a set of new constructs (i.e., code terms). The unfamiliarity of code terms that are discussed at SO posts and personal preference of formatting code terms could increase presentation inconsistency in rollback edits.

Gratitudinal inconsistency is also very frequent (i.e., 26.4%) in SO rollback inconsistencies. SO authority discourages to add such gratitude with posts. Unfortunately, many users do not follow the rules, which increase rollback edits with gratitudinal inconsistencies. The remaining inconsistencies such as signature, duplication, structural, and deprecation were observed infrequently. Interestingly, some users rollback edits only to reject or bring back signatures, duplication, or deprecation notes. However, we did not see any noticeable contribution of them to improve post quality. Thus, such inconsistent rollbacks should be identified and discouraged in order to encourage true contributors.

B. Implications of Findings

The findings from our study can guide the following major stakeholders in crowd-sourced platforms: 1) **Forum Designers** to improve the edit system, 2) **Forum Users** to guide their edit behavior, and 3) **Researchers** to study collaborative editing.

• **Forum Designers.** Given that content quality is paramount to SO's success, the current editing system can be enhanced by addressing the shortcomings we found. Primarily, SO can take proper measures to reduce the dominant inconsistencies, such as presentation, status, and temporal inconsistencies. As we showed, temporal inconsistency rejects thousands of approved edits. Thus, it can be detrimental to the overall quality of the shared content and the affected users' motivation. Such inconsistencies also may cause more suggested/rollback edits,

TABLE II: Average values of popularity metrics across posts with inconsistent vs consistent rollback edits

Metric	Consistent	Inconsistent	Decrease
View	272663.8	196727.3	39%
Favorite	400.0	202.9	97%
Score	693.1	403.3	72%
Answers	18.6	11.1	68%

while very little of those could be meaningful. From a site management perspective, this inefficiency can cost both the performance and the overall reputation of SO. Consequently, as we noted in Sections 1 and II, inconsistencies in rollback edits can frustrate SO users. Users' frustration due to inconsistencies and the potential loss in content quality is evident when we compared the SO posts with inconsistent vs consistent rollback edits. In Table II, we summarize four popularity metrics from Fig. 10. We see that there is a significant decrease in the number of views, favourites, scores, and answers in all the posts that went through the inconsistent rollback edit.

Therefore, SO needs to enforce guidelines to reduce the inconsistencies, in particular the dominant ones. For example, if a user attempt to rollback an edit with temporal inconsistency, SO could recommend the user go through a proper review channel. SO already uses several review channels to make decisions on various forum attributes, such as closing a question. Therefore, an addition of a new review channel is not challenging or unusual.

• **Forum Users.** Interactive or on-demand systems, such as browser plug-ins, can be developed to warn forum users of potential inconsistencies in their rollback edits. For example, we find that 29.5% of the rollback inconsistencies were related to presentation inconsistencies. Thus, a browser plug-in can be developed to check whether the rollback edits reject/accept a change in code term formatting. The plug-in can use historical rollback edit data with similar formatting and consult SO editing guidelines to suggest users on their rollback actions. Besides, the plug-in can also monitor the content owner's action, who is responsible for deciding on the suggested edit. The plug-in can recommend to the content owner whether to approve or reject the suggested edit by consulting previous edits, guidelines, etc. Intuitively, such a plug-in can then be extended to cover all the six user-based inconsistencies in rollback edits. Given that about 70% of rollback edit inconsistencies are user-based, such a system could significantly reduce the occurrence of such inconsistencies in SO. Consistent rollback edits could increase users' satisfaction and overall quality of the shared contents.

• **Researchers.** The content quality in SO has been the subject of several recent studies [72, 64]. Such studies traditionally look at the current version of a question or answer to develop tools and techniques, such as assessment of API misuse patterns [72], or producing live API documentation [67]. Given the editing history of a post is available and given that we see quality edits of a post can get rejected due to inconsistencies, researchers can use our developed algorithms to pick good quality suggested edits from the post contents that were rejected by rollback edits. Existing machine learning systems can use additional data aiming to assist developers in the edits of SO posts [11, 10].

VI. Threats to Validity

External Validity threats concern the generalizability of our findings. We focus on SO, which is one of the largest and most popular developers forums. Our findings may not generalize to other forums that are non-technical. To minimize the bias when conducting our qualitative analysis, we took statistically representative samples of all relevant revisions with a 95% confidence level, and a 5% confidence interval [7]. **Internal Validity** threats concern experimenter bias and errors while conducting the analysis. To reduce bias in our qualitative analysis, the first 200 rollback edits were labelled by two of the authors. Then we measure the agreement using Cohen's Kappa [13, 12] that represents a perfect agreement strength (i.e., $\kappa = 0.98$). The distribution of inconsistency types is almost similar between our manually labeled dataset and the entire 102K SO rollback edits on which we applied our automated classifier. This observation informs of the soundness of our manual labels. However, the algorithms were developed by analyzing the manual labels. Inadvertent bias could have permeated into the analysis of algorithm performance due to use of same dataset for algorithm development. We mitigated this threat by splitting our manually analyzed dataset into development and validation sets. Future work will focus on developing machine learning algorithms by consulting larger set of unseen data. Our survey participants range from novice to experienced and constitute mainly software developers but also other related professions. Such diversity in the survey participants offers validity and applicability to the survey findings. Moreover, the difference between agreement and disagreement on the impact of our inconsistencies is quite large and statistically significant. However, any individual bias in the survey responses should be mitigated via a large sample of 44 users. **Construct Validity** threats relate to the difficulty in finding data relevant to identify rollback edits and inconsistencies. Hence, we use revisions of the body of questions and answers from the Stack Exchange data dump, which we think are reasonable and reliable for capturing the reasons and inconsistencies of revisions.

VII. Related Work

Collaborative editing systems are common in Wikipedia [35, 33], GitHub code editing [14], webcasts [41], scientific contents [36, 9], and so on. Studies show that collaborating editing positively impacts towards the improvement of shared contents [35, 33]. A recent study by Wang et al. [70] in SO found that users are motivated to edit more when they are closer to getting a badge. Indeed, offering incentives as reputation scores is found to be useful to improve post quality [35]. Similar finding was also observed webcasts by Munteanu et al. [41], who tested the effectiveness of engaged users to collaborate in a wiki-like environment to edit/correct transcripts that are produced from webcasts through an automated speech recognition system. Kittur et al. [33] find that the increase in the number of editors does not guarantee the quality of the articles in Wikipedia. Our study findings offer another dimension to the above studies by showing that there are several inconsistencies that are negatively affecting the rollback edit mechanisms and user engagements in SO.

The focus of collaborative editing is to improve the quality of the shared contents based on user engagement [1]. Chen et al. [10] observed that most of the edits in SO are small sentence edits. In a follow-up study, Chen et al. [11] predicted whether a post needs to be edited. While developing their SOTorrent database, Baltes et al. [6] also observed that majority of edits in SO are relatively small. The quality of question is important to get an answer: lack of clarity, relatedness, and reproducibility of the problem, as well as the too short question could dissuade developers from answering to the question [3, 39]. The reputation and past activity of an asker could also factor into the likelihood of a question getting resolved [65]. As such factors of good questions are investigated, e.g., code to text ratio, etc. [8, 15]. However, depending on the platforms and user characteristics these factors can vary [32]. As such, it is important to detect content quality automatically [63, 64, 71].

Wang et al. [70] found that users who make more edits in a short time, are likely to get more edits rejected. Thus bad edits can harm the content quality. Our study offers complementary viewpoints to the above studies, in particular to Wang et al. [70]. Unlike the above work, we report with empirical evidences, for the first time, on the types and impacts of rollback edit inconsistencies.

VIII. Conclusions

The editing system in SO encourages developers to improve the posted content. Suggested edits can be rejected or brought back by rollbacks. Inconsistencies in rollback can arise when similar guidelines are followed to both accepted and rollback edits. With a view to understand the types and prevalence of such inconsistencies in rollback edits, we qualitatively analyzed 777 rollback edits in SO using standard principles of open coding. We found eight types of inconsistencies in the rollback edits. In both empirical and user studies, we find that the inconsistencies can negatively impact the popularity and quality of the posts. To offer actionable support from our research, we develop algorithms to automatically detect the inconsistencies. The algorithms show 0.99 precision (on average). Our findings can guide 1) forum designers to properly develop editing system to promote and ease the sharing of quality contents, 2) forum users to understand the contributing and inconsistent factors towards rollback edits, and 3) researchers in software engineering to investigate tools and techniques to guide forum users and designers to handle the rollback edits properly.

Acknowledgement: This research is supported by grants from Natural Sciences and Engineering Research Council of Canada (NSERC) and Canada First Research Excellence Fund (CFREF) from the Global Institute for Food Security (GIFS).

References

[1] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 183–194, 2008.

[2] Syed Ahmed and Mehdi Bagherzadeh. What do concurrency developers ask about?: A large-scale study using stack overflow. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, page Article No. 30, 2018.

[3] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. Answering questions about unanswered questions of stack overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 87–100, 2013.

[4] Mehdi Bagherzadeh and Raffi Khatchadourian. Going big: A large-scale study on what big data developers ask. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, pages 432–442, New York, NY, USA, 2019. ACM.

[5] Richard P. Bagozzi and Utpal M. Dholakia. Open source software user communities: A study of participation in linux user groups. *Journal of Management Science*, 52(7):1099–1115, 2006.

[6] Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl. Sotorrent: reconstructing and analyzing the evolution of stack overflow posts. In *Proceedings of the 15th International Conference on Mining Software Repositories*, pages 319 – 330, 2018.

[7] Sarah Boslaugh. *Statistics in a nutshell: A desktop quick reference*. "O'Reilly Media, Inc.", 2012.

[8] Fabio Calefato, Filippo Lanubile, and Nicole Novielli. How to ask for technical help? evidence-based guidelines for writing questions on stack overflow. *Journal of Information and Software Technology*, 94:186–207, 2018.

[9] Rafael A. Calvo, Stephen T. O'Rourke, Janet Jones, Kalina Yacef, and Peter Reimann. Collaborative writing support tools on the cloud. *IEEE Transactions on Learning Technologies*, 41: 66–99, 2005.

[10] Chunyang Chen, Zhenchang Xing, and Yang Liu. By the community & for the community: A deep learning approach to assist collaborative editing in q&a sites. In *Proceedings of the ACM on Human-Computer Interaction*, page Article 32, 2017.

[11] Chunyang Chen, Xi Chen, Jiamou Sun, Zhenchang Xing, and Guoqiang Li. Data-driven proactive policy assurance of post quality in community q&a sites. In *Proceedings of the ACM on Human-Computer Interaction*, page Article 33, 2018.

[12] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[13] Jacob Cohen. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.

[14] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. Social coding in github: transparency and collaboration in an open software repository. In *Proceedings of the ACM conference on Computer Supported Cooperative Work*, pages 37–46, 2012.

[15] Maarten Duijn, Adam Kucera, and Alberto Bacchelli. Quality questions need quality code: Classifying code fragments on stack overflow. In *Proceedings of the IEEE/ACM 12th Working Conference on Mining Software Repositories*, pages 410–413, 2015.

[16] Stack Exchange. Meta stack exchange, 2008. URL https://meta.stackexchange.com. Online; Last accessed February 2020.

[17] Stack Exchange. Should 'hi', 'thanks', taglines, and salutations be removed from posts?, 2009. URL https://meta.stackexchange.com/questions/2950. Online; Last accessed February 2020.

[18] Stack Exchange. Are taglines and signatures disallowed?, 2009. URL https://meta.stackexchange.com/questions/5029. Online; Last accessed February 2020.

[19] Stack Exchange. What is a 'rollback'?, 2009. URL https://meta.stackexchange.com/questions/17038. Online; Last accessed February 2020.

[20] Stack Exchange. What are the terms and conditions for editing posts on stack overflow?, 2012. URL https://meta.stackexchange.com/questions/149839. Online; Last accessed February 2020.

[21] Stack Exchange. Editing policy is contradictory and unclear, 2012. URL https://meta.stackexchange.com/questions/138262. Online; Last accessed February 2020.

[22] Stack Exchange. Is it right to rollback an edit that only removed "thanks a lot!", 2012. URL https://meta.stackexchange.com/questions/146530. Online; Last accessed February 2020.

[23] Stack Exchange. When is "edit"/"update" appropriate in a post?, 2012. URL https://meta.stackexchange.com/questions/127639. Online; Last accessed February 2020.

[24] Stack Exchange. What's wrong with putting "edit: ...." in the body of a post?, 2013. URL https://meta.stackexchange.com/questions/202472. Online; Last accessed February 2020.

[25] Stack Exchange. Issues with stack overflow users editing my questions, 2013. URL https://meta.stackexchange.com/questions/165463. Online; Last accessed February 2020.

[26] Stack Exchange. What are some guidelines for editing questions, and how to respond to massive edits?, 2015. URL https://meta.stackexchange.com/questions/253784. Online; Last accessed February 2020.

[27] Stack Exchange. Lack of consistency with offensive posts (multiple communities), 2016. URL https://meta.stackexchange.com/questions/274042. Online; Last accessed February 2020.

[28] Stack Exchange. Clarify editing guidelines, 2017. URL https://meta.stackexchange.com/questions/300211/clarify-editing-guidelines. Online; Last accessed February 2020.

[29] Stack Exchange. What is the etiquette for modifying posts?, 2018. URL https://meta.stackexchange.com/questions/11474. Online; Last accessed February 2020.

[30] Stack Exchange. Database schema documentation for the public data dump and sede, 2019. URL https://meta.stackexchange.com/questions/2677.

[31] Stack Exchange. StackExchage API, 2019. URL http://data.stackexchange.com/stackoverflow.

[32] Nathaniel Hudson, Parmit K. Chilana, Xiaoyu Guo, Jason Day, and Edmund Liu. Understanding triggers for clarification requests in community-based software help forums. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*, pages 189–193, 2015.

[33] Aniket Kittur and Robert E. Kraut. Harnessing the wisdom of crowds in wikipedia: quality through coordination. In *Proceedings of the ACM conference on Computer supported cooperative work*, pages 37–46, 2008.

[34] Karim R Lakhani and Eric von Hippel. How open source software works: free user-to-user assistance. *Journal of Research Policy*, 32(6):923–943, 2003.

[35] Guo Li, Haiyi Zhu, Tun Lu, Xianghua Ding, and Ning Gu. Is it good to be like wikipedia?: Exploring the trade-offs of introducing collaborative editing model to q&a sites. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 1080–1091, 2015.

[36] Paul Benjamin Lowry, Aaron Mosiah Curtis, and Michelle Rene Lowry. A taxonomy of collaborative writing to improve empirical research, writing practice, and tool development. *Journal of Business Communication*, 41:66–99, 2005.

[37] H. B. Mann and D. R. Whitney. On a test of whether one of

two random variables is stochastically larger than the other. *The annals of mathematical statistics*, 1947.

[38] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge Uni Press, 2009.

[39] Saikat Mondal, Mohammad Masudur Rahman, and Chanchal K. Roy. Can issues reported at stack overflow questions be reproduced?: an exploratory study. In *Proceedings of the 16th International Conference on Mining Software Repositories*, pages 479–489, 2019.

[40] Saikat Mondal, Gias Uddin, and Chanchal K. Roy. Replication package, 2021. URL https://bit.ly/3c2dSpR.

[41] Cosmin Munteanu, Ron Baecker, and Gerald Penn. Collaborative editing for improved usefulness and usability of transcript-enhanced webcasts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 373–382, 2008.

[42] Stack Overflow. How do you change the size of figures drawn with matplotlib?, 2008. URL https://stackoverflow.com/posts/332311/revisions?page=1. Online; Last accessed February 2020.

[43] Stack Overflow. C# common library, 2008. URL https://stackoverflow.com/posts/498249/revisions?page=1. Online; Last accessed February 2020.

[44] Stack Overflow. How do i make an html page print in landscape when the user selects 'print'?, 2008. URL https://stackoverflow.com/posts/37162/revisions?page=1. Online; Last accessed February 2020.

[45] Stack Overflow. Meta stack overflow, 2008. URL https://meta.stackoverflow.com. Online; Last accessed February 2020.

[46] Stack Overflow. Is there a performance difference between i++ and ++i in c++?, 2008. URL https://stackoverflow.com/posts/24904/revisions?page=1. Online; Last accessed February 2020.

[47] Stack Overflow. What does the term "canonical form" or "canonical representation" in java mean?, 2008. URL https://stackoverflow.com/posts/280121/revisions?page=1. Online; Last accessed February 2020.

[48] Stack Overflow. How do you compare two version strings in java?, 2008. URL https://stackoverflow.com/posts/11024200/revisions?page=1. Online; Last accessed February 2020.

[49] Stack Overflow. Grouping activerecord query by a child attribute, 2008. URL https://stackoverflow.com/posts/125523/revisions?page=1. Online; Last accessed February 2020.

[50] Stack Overflow. What's wrong with the positioning in this very simple example? (ms ie 8), 2010. URL https://stackoverflow.com/questions/3774926. Online; Last accessed February 2020.

[51] Stack Overflow. Php/mysql: How to know if update was successful? (when data sent to update was the same as that already in db), 2011. URL https://stackoverflow.com/posts/6218171/revisions?page=1. Online; Last accessed February 2020.

[52] Stack Overflow. put imageview src to hashmap<string, string>, 2011. URL https://stackoverflow.com/posts/30150411/revisions?page=1. Online; Last accessed February 2020.

[53] Stack Overflow. I am trying to export mysql data to a file using java, but i am not able to get the table headers, 2012. URL https://stackoverflow.com/posts/22529397/revisions?page=1. Online; Last accessed February 2020.

[54] Stack Overflow. Rate google play application directly in app, 2012. URL https://stackoverflow.com/posts/11270668/revisions?page=1. Online; Last accessed February 2020.

[55] Stack Overflow. Mysql results issue in php array, 2013. URL https://stackoverflow.com/posts/14391452/revisions?page=1. Online; Last accessed February 2020.

[56] Stack Overflow. How to access the current subversion build number?, 2013. URL https://stackoverflow.com/posts/110175/revisions?page=1. Online; Last accessed February 2020.

[57] Stack Overflow. Ios - remove all padding from uitextview, 2013. URL https://stackoverflow.com/posts/20269793/revisions?page=1. Online; Last accessed February 2020.

[58] Stack Overflow. Provide more guidelines for reviewing edits, 2015. URL https://meta.stackoverflow.com/questions/295319. Online; Last accessed February 2020.

[59] Stack Overflow. How do i make a good edit?, 2015. URL https://meta.stackoverflow.com/questions/303219. Online; Last accessed February 2020.

[60] Stack Overflow. Swift : Programatically create a uibutton with some specific font and size, 2016. URL https://stackoverflow.com/posts/39640499/revisions. Online; Last accessed February 2020.

[61] Stack Overflow. Swift : Programatically create a uibutton with some specific font and size, 2016. URL https://stackoverflow.com/questions/39640499. Online; Last accessed February 2020.

[62] Chris Parnin, Christoph Treude, Lars Grammel, and Margaret-Anne Storey. Crowd documentation: Exploring the coverage and the dynamics of api discussions on stack overflow. Technical report, Georgia Tech, 2012.

[63] Luca Ponzanelli, Andrea Mocci, Alberto Bacchelli, and Michele Lanza. Understanding and classifying the quality of technical forum questions. In *Proceedings of the 14th International Conference on Quality Software*, pages 343–352, 2014.

[64] Luca Ponzanelli, Andrea Mocci, Alberto Bacchelli, and Michele Lanza. Improving low quality stack overflow post detection. In *In Proceedings of the 30th International Conference on Software Maintenance and Evolution*, pages 541–544, 2014.

[65] Mohammad Masudur Rahman and Chanchal K. Roy. An insight into the unresolved questions at stack overflow. In *Proceedings of the 12h Working Conference on Mining Software Repositories*, pages 426–429, 2015.

[66] Martin P. Robillard and Robert DeLine. A field study of API learning obstacles. *Empirical Software Engineering*, 16(6):703–732, 2011.

[67] Siddharth Subramanian, Laura Inozemtseva, and Reid Holmes. Live api documentation. In *Proc. 36th International Conference on Software Engineering*, page 10, 2014.

[68] Gias Uddin and Martin P. Robillard. How api documentation fails. *IEEE Softawre*, 32(4):76–83, 2015.

[69] Gias Uddin, Olga Baysal, Latifa Guerrouj, and Foutse Khomh. Understanding how and why developers seek and analyze API-related opinions. *IEEE Transactions on Software Engineering*, pages 1–40, 2019.

[70] Shaowei Wang, Tse-Hsun (Peter) Chen, and Ahmed E. Hassan. How do users revise answers on technical Q&A websites? a case study on stack overflow. *IEEE Transactions in Software Enginering*, page 19, 2018.

[71] Yuan Ya, Hanghang Tong, Tao Xie, Leman Akoglu, Feng Xu, and Jian Lu. Detecting high-quality posts in community question answering sites. *Journal of Information Sciences*, 302 (1):70–82, 2015.

[72] Tianyi Zhang, Ganesha Upadhyaya, Anastasia Reinhardt, Hridesh Rajan, and Miryung Kim. Are code examples on an online q&a forum reliable? a study of api misuse on stack overflow. In *Proc. 32nd IEEE/ACM International Conference on Software Engineering*, page 12, 2018.