Wiki Refactoring: an Assisted Approach Based on Ballots

Oscar Díaz
oscar.diaz@ehu.es
University of the Basque Country
San Sebastián, Spain

Gorka Puente
gorka.puente@ehu.es
University of the Basque Country
San Sebastián, Spain

Cristóbal Arellano
carellano001@ikasle.ehu.es
University of the Basque Country
San Sebastián, Spain

ABSTRACT

Wikis' organic growth inevitably leads to a gradual degradation of the wiki content/structure which, in turn, may entail recurrent wiki refactoring. Unfortunately, no regression test exists to check the validity of the refactoring output. Some changes, even if compliant with good practices, can still require to be backed by the community which ends up bearing the maintenance burden. This calls for a semi-automatic approach where "refactoring bots" interact with wiki users to confirm the upgrades. This paper outlines this as follows. First, a refactoring bot detects wiki degradation. Second, the community evaluates the severity of the degradation through voting. Finally, the refactoring bot takes control and enacts the appropriate changes, if so decided by the community. This lessens but does not exclude, the participation of the community. We aim at reducing the maintenance penalty that goes with the *laissez-faire* way that characterizes wiki contributions.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques; H.4 [**Information Systems**]: Information Systems Applications

General Terms

Human Factors, Design, Management

Keywords

wiki, refactoring, voting systems

1. INTRODUCTION

The difficulty in *automating* wiki refactoring stems from the very same nature of wikis: pervasive peer-review. Some changes, even if compliant to the guidelines, may need to be backed by the community. In software, "safe refactoring" is ensured through regression testing, i.e., the process of testing program changes to make sure that the older programming still works with the new changes [2]. However, the test for wiki refactoring is whether the community ends up happy [1]. Changes should be backed by the community. So far, wikis support it through "warning templates", i.e., messages that appear along wiki content. Warning templates are inserted *by users for users*. Users advise other users (i.e., the community) for "bad smells" . Detection and accomplishment of these enhancements are up to the community.

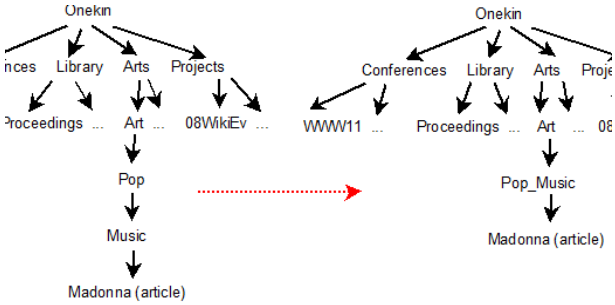

Figure 1: Sample case. Nodes stand for wiki pages.

Traditionally, wiki refactoring is either fully automated (e.g., through bots[1]) or totally manual (e.g., using warning templates). The former is not satisfactory for sensible changes that might require previous approval from the community. On the other hand, warning templates put all the burden on the users. This paper calls for an intermediate approach where users are assisted (but not substituted) during the refactoring process.

As an example, consider the *TooMuchStructureInDepth* guideline (see Fig. (1) for an example). The category *Music* seems to provide superfluous structure. The guideline recommends to create a new category whose name is obtained by concatenating its parent name (e.g., *Pop*) with its own name (e.g., *Music*). Children (e.g., *Madonna*) are moved upwards to the newly created category (e.g., *Pop_Music*) and the spurious categories (e.g., *Pop* and *Music*) are advised for deletion through a warning template. Note, the goodness of this outcome much depends on the user's mental model, and hence, it is debatable. This is handled through "discussion pages". The fate of the discussion is on the hands of the community. Neither detection nor enactment is automated. However, our contention is that when the discussion

ACM 978-1-4503-0909-7/11/10.

WikiSym '11 October 03 - October 05 2011, Mountain View, CA, USA

[1] en.wikipedia.org/wiki/Wikipedia:Bots

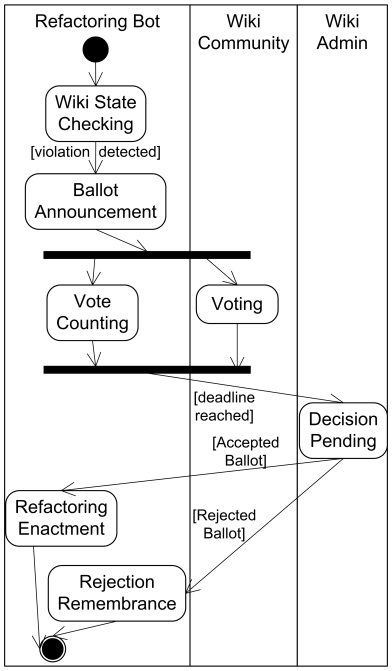

Figure 2: The *Ballot* Process.

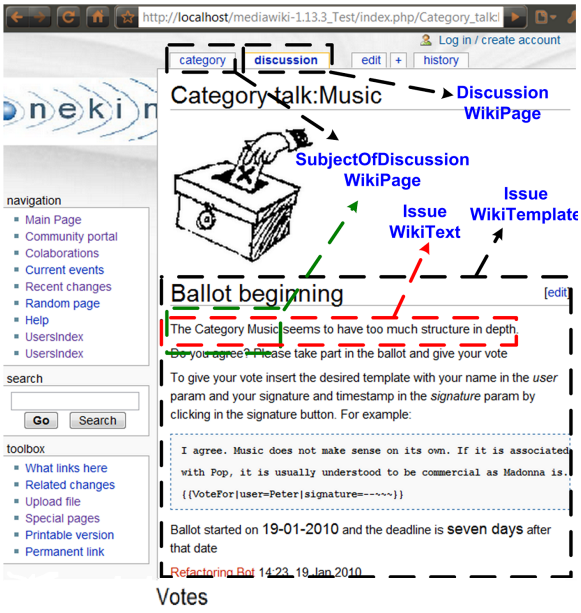

Figure 3: The *bot* generates a *discussion* page.

goes along well-established guidelines (as those proposed by *Wikipedia*), the community can retain the final decision but detection/enactment of the guidelines can be automated. This is "the ballot process".

2. THE BALLOT PROCESS

The ballot process intertwines actions from the refactoring bot (*"rebot"*) and the wiki community as follows (see Fig. 2):

1. The *rebot* monitors whether the wiki's current state violates some guidelines.

2. If so, the *rebot* notifies this situation to the wiki community by setting a *Discussion page*.

3. *Discussion* creation launches two parallel activities. First, the wiki community can now cast votes in the ballot for the issue at hand. Second, the *rebot* periodically monitors the discussion, and counts the votes.

4. If the ballot deadline is reached, the ballot is moved to the pending state. Now, the wiki administrator accepts or rejects the proposal based on the ballot outcome by using the *"AcceptedBallots"*, *"RejectedBallots"* tag.

5. If the ballot is accepted then, the *rebot* conducts the change.

6. If the ballot is rejected then, the *rebot* records the decision to avoid future ballots on this issue.

Broadly, *rebot* specification implies: an anomaly pattern, a communication medium and an enforcement action.

Anomaly Pattern. It is a description of a deviation from the guidelines in terms of wiki content/structure A wiki state is defined as a set of pages, categories and templates pertaining to a wiki at a certain moment. An anomaly pattern describes a predicate on a wiki state.

Communication Medium. It supports the interaction between the *rebot* and the community. We resort to discussion pages to notify guideline violation and, subsequently, conduct user communication. *Rebot*-to-community communication is achieved through categories playing the role of "tags". Process-oriented tags include: '*AcceptedBallots*', '*RejectedBallots*', '*PendingDecisions*' and '*VotedDecisions*'. E.g., if a ballot is rejected, the '*RejectedBallots*' page is tagged with the discussed category (i.e., *subjectOfDiscussion*). Similarly, the act of voting is also supported through tagging. The communication exchange follows:

* The *rebot* initializes the discussion with the *subjectOfDiscussion* (e.g., *"Music"* category) and the *issue* (*"to have too much structure in depth"*). Fig. 3 shows the *discussion* page generated by the *rebot* to communicate that category *Music* suffers from *TooMuchStructureInDepth*.

* Once the discussion page is created, users can vote. Ballots are supported through three wiki templates, namely, *VoteFor*, *VoteAgainst* and *Blank*. These templates are parameterized by a *Comment* (e.g., a *WikiText* adding some grounds to the vote), a *signature* (a wiki built-in facility) and the *user* name.

* Each vote causes the discussion page to be tagged with the namesake vote tag. For instance, the use of the *VoteFor* template by *Peter* leads to tagging the corresponding discussion with the tag *PeterVoteFor*.

* Once the ballot is over, the wiki administrator notifies the *rebot* of the output. To this end, the administrator tags the associated page (e.g., *Music* category) with either *"AcceptedBallots"* or *"RejectedBallots"*. Both tags are *protected categories* provided by the *rebot*.

Enforcement action. A set of wiki operations that returns the wiki back to a guideline-compliant state.

The aim is to lower the barriers for layman participation not only during editing but also at maintainance time.

3. REFERENCES

[1] W. Cunningham et al. WikiRefactoring. c2.com/cgi/wiki?WikiRefactoring [accessed 3Apr11].

[2] T. Mens and T. Tourwé. A survey of software refactoring. *IEEE TSE*, 2004.