Employing Information Tags in Software Development

Karol RÁSTOČNÝ*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
karol.rastocny@stuba.sk

1 Introduction

A management of software development process is a crucial part of the software engineering, from which the success of software projects is dependent. This management mostly relays upon quality and freshness of software metrics and analysis over these metrics. Software metrics can be based on source code or empirical data about software developers. Code-based metrics are well known and many approaches based on them have been proposed. But empirical software metrics are still uncovered part of software engineering even though they contain important information about software development process and they can be used e.g. for forecasting significant trends, similarly as empirical data (e.g., implicit user feedback) in the web engineering. Reasons of this state are time expensive and erroneous collecting empirical data [2].

2 Management and maintenance of information tags

We proposed solution of these problems based on collecting, storing and maintenance of developer-oriented empirical data abstracted to information tags and also empirical software metrics. But empirical data stored in information tags can be invalidated by developers' activities, so information tags have to be managed and maintained. For this reason we proposed approach for information tag management and maintenance based on event stream processing, in which we evaluate stream (C-SPARQL) queries and execute tagging actions, when queries are evaluated successfully.

We performed preliminary performance evaluation of the proposed approach, during which we have evaluated non-trivial C-SPARQL query with 2s window over stream of RDF triples and we have measured response time. In out testing environment (with 2x2.8GHz CPU), we reached satisfactory response (1s with 100ms tolerance), when we generated up to 300 triples per second (see Figure 1). It seems to be

* Supervisor: Mária Bieliková, Institute of Informatics and Software Engineering

Spring 2014 PeWe Workshop, March 21, 2014, pp. 73–74.

unsatisfactory result, but bottleneck was caused by overhead of testing sandbox, that consumed more than 75% of resources. In addition, we receive 2 RDF triples per second from 10 developers.

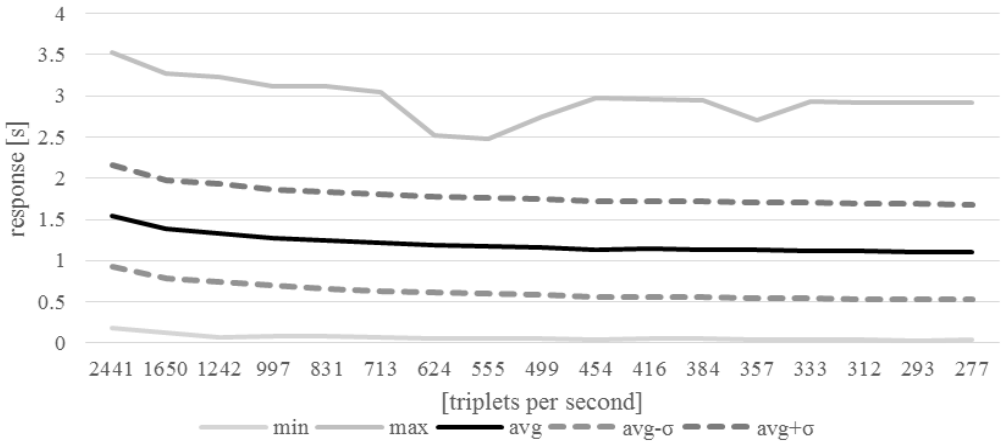

Figure 1. Response times of the Tagger by a number of RDF triplets posted to the Tagger, with C-SPARQL query set up to 2s sliding window.

3 Future work

Nowadays we are proposing approach for automatic learning tagging rules based on analysis of developers' activity streams and changes in information tag space. We have two working versions that we plan to evaluate in parallel. The first one is based on clustering of similar developers' activities that similarly affected information tags. The second approach is based on process discovery techniques [1]. This solution does not analyse modifications of information tags but it gets new versions as combination of adding new information tags and removing old information tags.

Extended version was published in Proc. of the 10th Student Research Conference in Informatics and Information Technologies (IIT.SRC 2014), STU Bratislava, 513-520.

Acknowledgement. This contribution is the partial result of the Research & Development Operational Programme for the project Research of methods for acquisition, analysis and personalized conveying of information and knowledge, ITMS 26240220039, co-funded by the ERDF.

References

[1] Van der Aalst, W.: Process Mining: Overview and Opportunities. *ACM Trans. Manag. Inf. Syst.*, vol. 3, no. 2, (2012), pp. 7:1–7:17.

[2] Johnson, P.: You can't even ask them to push a button: Toward ubiquitous, developer-centric, empirical software engineering. In: *The NSF workshop for new visions for SW design and productivity*, Nashville, (2001), p. 5.