MMT – a Project Tool for Observing Metrics in Software Projects

Pekka Mäkiaho, Katriina Löytty, Timo Poranen

Abstract: *In this paper, we present the Metrics Monitoring Tool, a tool for project members, project managers and upper management for reporting and following projects' metrics. We introduce how the tool was developed and how it was evaluated in students' software development projects. As a result, we show how the tool supported different user groups, suggest some new features, and propose some guidelines on how this tool could be utilized in software development and teaching of software project work.*

Key words: *Metrics, Project management, Software development.*

INTRODUCTION

Project work and basic project management skills are essential to all computer science students [2]. In many universities, group work skills are learned as a part of different course assignments starting from the first year of studies. During the third or fourth year, when students have studied enough core courses, there can be a larger capstone project as a stand-alone course.

In addition to group work and communication skills required in the capstone project, students have a possibility to combine their knowledge from different courses, like programming, databases, software and user interface design, into practice when they implement a software product. The student teams need to find a suitable combination of software tools [7] to utilize in the project for programming, requirements management, communication, user interface design and other main software development activities.

When course staff organizes different capstone projects to a larger group of students, there are challenges in following and supervising many projects at the same time. The projects also generate many standard metrics: whether a specific deadline was met or not, whether a specific deliverable was returned on time, how many hours of work has been done so far by different students, etc. Then, depending on the field of study, there can be many other process and product related metrics. In software development projects, there can be metrics like the number of written code lines, number of planned features to be implemented, number of features under development at a given moment, number of implemented features so far, number of passed test cases, etc.

In this paper, we present the Metrics Monitoring Tool (MMT) for observing and visualizing project metrics in software development projects. The tool helps project managers in reporting, team members to be more aware of the state of the project and course staff to compare and follow how all projects are progressing. Based on the research needs, for example, the tool can be extended to collect new data in future.

The rest of the paper is organized as follows. In the next section, we give a brief introduction to software metrics. Then we describe the development process and implementation of MMT and its features. After that we introduce the evaluation of MMT. The final section concludes the work.

SOFTWARE METRICS

A software metric is a measurement concerning a particular characteristic of software, or a measurement of a software project and its process. Software metrics are categorized to predicting metrics and controlling metrics depending on whether they are measuring and predicting the status of a software product itself, or measuring the process to develop the product [9]. The control metrics can also be categorized to project metrics and process metrics depending on whether they are measuring an individual project and supporting tactical decisions, or if they are measuring the process and supporting strategic decisions. A direct metric means the raw data that has been measured; indirect (aka derived) metric means that the metric is created based on the existing metrics [4].

One of the most important reasons that a project fails is poor reporting of the project's status [3]. Because of the poor reporting, the management does not know the state of the project and thus does not execute right actions. There is a hypothesis [6] that the level of reporting the metrics is related to how much the project management itself utilizes the metrics. Following the project metrics gives rigorous information on the project to the management. For example, by following the current statuses of the requirements, history of effort needed for moving a requirement from one state to another, working hours' estimations and the resources left on the project, the management can prioritize requirements, move resources from development to testing, etc. If the metrics are followed and the meaning of them is understood (monitoring phase) the right actions can be done in time (controlling phase).

THE METRICS MONITORING TOOL – MMT

The motivation for developing MMT originates from the University of Tampere Project Work (PW) and Software Project Management (SPM) courses and the related challenges in collecting and monitoring projects' metric data. During the courses, undergraduate PW students act as software development team members, and graduate SPM students act as project managers for those teams. The overall objective of the teams is to design and implement a functioning piece of software for a real client during one semester.

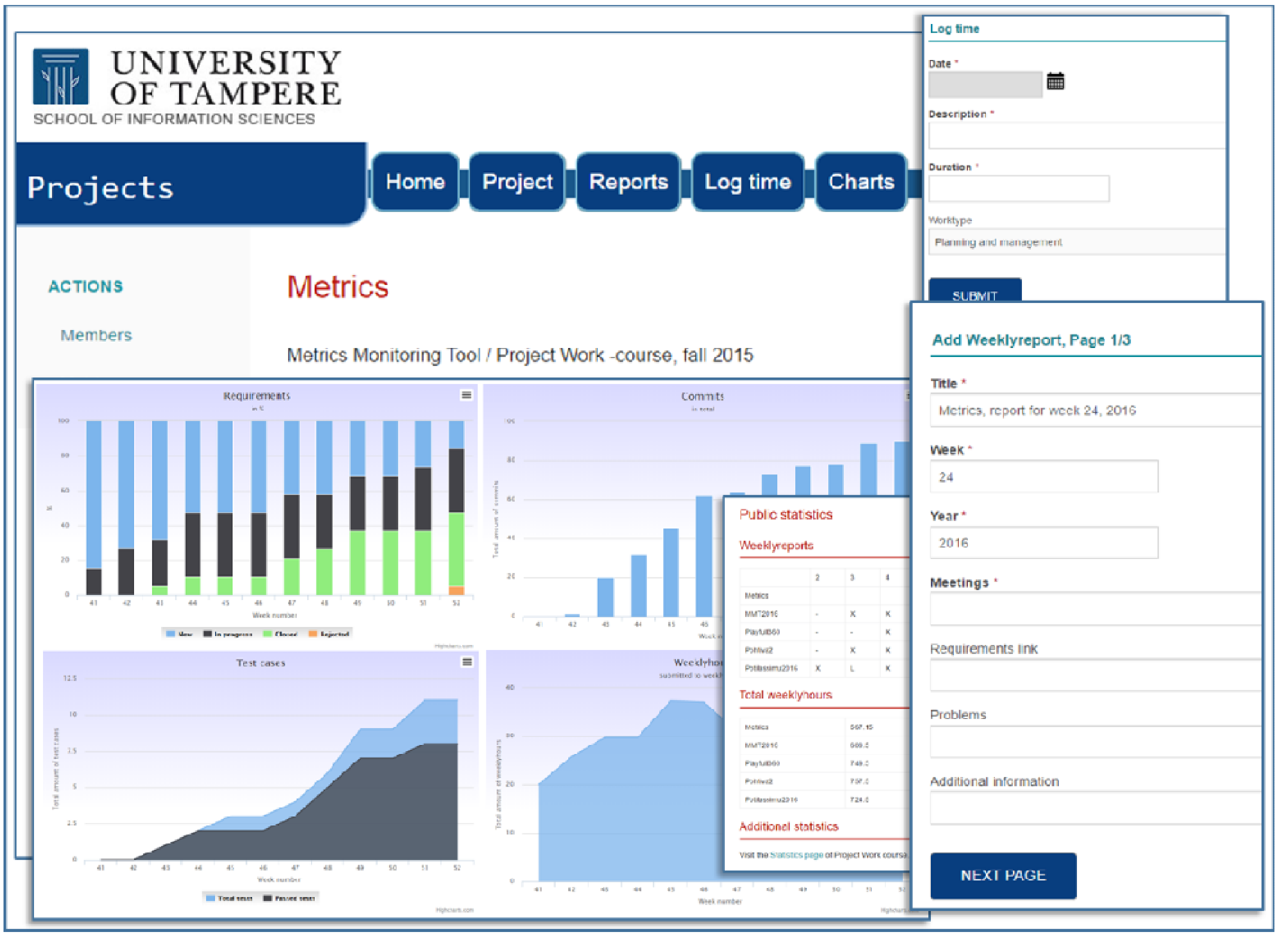
MMT is a web-based repository for logging, monitoring and aggregating various software development project metrics, such as working hours, code commits, requirements statuses as well as passed and failed test cases throughout the project lifecycle. In the first instance, MMT serves the purposes of the PW and SPM -courses. In future, the use of the tool can potentially be extended to other software project management and research purposes. The tool enables project team members, managers, supervisors and eventually clients to view the progression of their own and their peers' projects based on the collected metric data. The tool facilitates early identification of factors that indicate project success, difficulties or failure, which eases taking timely corrective action, as necessary. MMT thus supports team performance tracking, project progress control as well as project management research.

Prior to implementing MMT, project metric data was gathered by weekly reports submitted by project managers via a text based email template. The main challenges in this conduct included the high amount of manual work in aggregating the metric data, inconsistencies in and varying formats of the submitted data, lack of visibility to projects' overall progression versus their goals, as well as limited visualization capabilities.

MMT tool is being developed in various phases by several teams as part of the PW and SPM courses, where the preceding team hands the work over to the next team. The first construct of the tool was initiated in the fall 2014 through the summer 2015. This early pilot, a proof-of-concept development (version 0.1), provided knowledge for upcoming work on desired features, deployable methods and anticipated pitfalls. After the pilot version, development was started afresh with revisited requirements, revised database design and a feasible model-view-controller (MVC) framework. This stage provided the basis for the current solution (versions 1.0 through 1.3), which will be enhanced by teams in the coming PW/SPM -courses (versions 2.0 and up).

Currently MMT runs on a virtual server provided by the University of Tampere School of Information Sciences [5]. It relies on an SQL database and is implemented in PHP. MMT utilizes CakePHP, an open source MVC framework for PHP, and Highcharts, a JavaScript library for presenting various kind of plots and charts. From a data entity perspective, MMT firstly contains users and projects. The users then have inherent user roles and attached project roles according to their memberships in projects. The projects have weekly reports and related metrics. Similarly, the project members have weekly and daily working hours. The tool presents projects' metrics, progression and status in graphs derived from these data. Notification, commenting and other enhanced functionalities are planned for future versions.

Figure 1 portrays the layout and feel of the application with some of its current functionalities. MMT features are available to users based on their user and project roles: A public projects listing, public statistics and frequently asked questions (FAQs) are offered for all visitors of the web site. Project developers can view reports and charts of their own and public projects as well as log their own daily working hours. Project managers can additionally compose weekly project reports, log hours on behalf of team members and assign and remove members of their projects. Project supervisors have similar rights as managers bar the hour logging functionality. The tool's user and project basic data is maintained by admin users.



Metrics Monitoring Tool / Project Work -course, fall 2015

Log time

Date *

Description *

Duration *

Worktype: Planning and management

Add Weeklyreport, Page 1/3

Title *: Metrics, report for week 24, 2016

Week *: 24

Year *: 2016

Meetings *

Requirements link

Problems

Additional information

Public statistics

Weeklyreports

	2	3	4
Metrics			
MMT2016	-	K	K
Playful360	-	-	K
Pohiva2	-	K	K
Potilassimu2016	K	L	K

Total weeklyhours

Metrics	567.15
MMT2016	600.5
Playful360	748.0
Pohiva2	707.0
Potilassimu2016	724.0

Additional statistics

Visit the Statistics page of Project Work course.

Figure 1. Layout and feel of MMT

EVALUATION

The data was collected from the SPM and PW -courses at the School of Information Sciences, University of Tampere during the spring semester 2016. The projects started in January and ended in May. There were 4 project teams, 8 students (project managers) on the SPM-course and 15 students (developers) on the PW-course.

Using MMT was mandatory to all teams for, at least, logging hours and weekly reporting. During the course, the teams had 5 reviews with the supervisor and the client and they also formed a final report at the end of the course. The supervisors observed the project teams by using MMT, participating in reviews and communicating in face-to-face - meetings and by email.

There were also two mandatory Moodle questionnaires for the students with questions on MMT and its usage. The first questionnaire was executed in the beginning of March, after the projects had started. The second questionnaire took place at the end of

May, when all the projects had finished. The versions of MMT are related to the questionnaires so that the version 1.0 was in use before the first questionnaire. MMT was updated to the versions 1.1 and 1.2 before the second questionnaire. The version 1.3 was deployed just after the second questionnaire.

The four main features of the tool were logging hours, composing and viewing weekly reports and viewing the visual charts of the project metrics. Table 1 shows the average of the grades given by the students to each feature and MMT as a whole regarding usability, functionality and usefulness as well as the total grade. The scale was (1) poor, (2) fair, (3) good, (4) very good and (5) excellent.

Table 1. The average grade (1-5) given to each feature and to feature's attributes

Feature/ Attribute	Logging hours	Viewing reports	Viewing charts	Composing reports	MMT-tool as a whole
Usability	4.5	4.3	4.75	4.3	3.2
Functionality	4.5	4.0	4.5	4.0	3.6
Usefulness	5.0	4.0	4.5	4.3	4.0
Overall	4.2	3.9	3.7	3.3	3.8

Logging hours

Students were asked to use MMT for logging their working hours: whenever they worked for the project, they logged the date, the hours, the type of work, and the description in MMT. As this feature was mandatory to use for all students, it is not surprising that most of the comments related to this. When a weekly report was composed, the hours were copied to the weekly report's hours. Because of this, there were restrictions in editing the logged hours afterwards: In the version 1.0, a developer was not allowed to change the logged hours afterwards but had to ask his/her manager or supervisor to do it. In the questionnaire 1 this was complained about by two managers and two developers. This restriction was changed to version 1.1 so that also developers were allowed to edit the hours until the weekly report was sent. However, also this restriction was seen as a problem by one developer.

Overall, the logging hours -feature was seen as very useful: The managers said that it saved their time in comparison to a situation where they would have had to collect the hours manually or extract them from some other tool for the report. The developers reported that logging hours helped them to observe their own time usage.

Viewing reports

The only reports one could get from MMT are the weekly reports composed by the managers. Viewing (weekly) reports was not seen very useful by the developers. However, there were comments on extending this feature: project managers from all teams, and also five developers, requested additional features for getting different kinds of reports from MMT, especially on working hours, by using various filters.

It was extremely useful for the supervisors to have all the reports from different weeks and different projects in one tool. Getting this information from one place reduced the time required for preparing to project meetings from 60-90 minutes to 15 minutes.

Viewing charts

The charts show the state and the proceeding of the project in one view. One can see the current state and the history of the working hours, requirements, test cases and commits. For the supervisor, this feature is very important, as from the graphs one can not only see the history and the current state but also predict the project's progression and give some suggestions to the team.

The project members did not see the charts very useful now, but they would like to have additional charts on individual level, especially for seeing how the amount of working hours has built up.

The managers from all teams reported this feature to be very useful for seeing the status and proceeding. However, one team commented that viewing charts was not so useful from a project management point of view but it was interesting to compare their own project to other projects.

Composing reports

Composing weekly reports is done by the project managers by the following Monday of each week. The managers reported the amount and states of test cases and requirements as well as the amounts of weekly hours per person, commits to the code repository, meetings and possible issues.

When creating a weekly report, each project member’s working hours were automatically copied to the report as a template, and then the managers could modify these if needed. So, in the system there were reported weekly hours and working hours. This was seen very confusing. There were also some other usability issues. However, the feature was seen useful and it was reported to make reporting easier compared to writing and sending emails. In the previous semesters, the reports were sent by email and the average time for composing the report was 42 minutes. Now the average time was 16 minutes, varying from 2 to 30 minutes.

To be developed

In both questionnaires, we asked which properties or features of the tool should be further developed. In several of the general comments on increasing usability (from four developers and two managers) Bootstrap [1] or similar technology and better support for mobile interface were requested.

Restricting the editing of the logged hours afterwards was seen as a problem by both managers and developers. In the next versions, the restrictions could be removed.

The reports seen from the tool are limited only to the weekly reports composed by the managers. Also the amount of different types of charts is quite limited. More flexible reports and charts could be available in the next versions.

Developing the tool into a real project management tool, like Redmine [8], for not needing to use many different tools on the same project, was suggested by four managers. One manager asked for a feature that could predict the success of the project and to give tips to do the right actions. This feature has been in the product backlog from the very beginning and will certainly be implemented in the future.

The two supervisors of the project wanted a feature to give feedback and comment the projects. In the version 1.3 there is a feature that allows all the project members, managers and supervisors give comments on weekly reports.

Currently, there is no role *client* or *customer* in the tool. However, the clients of the projects were asked if they want to have an access to the tool. None wanted, but in one project the client was also the project’s supervisor and naturally he observed the team using MMT. One client was sent screenshots of the weekly reports and two other clients did not get any specific weekly reports at all. Having the possibility to send the reports also by email and/or having a client role in the tool was asked for by two managers and the supervisors.

CONCLUSIONS AND FUTURE WORK

Metrics Monitoring Tool benefits different user groups. Logging hours feature helps the project members to observe their own time usage. MMT reduces the time for creating weekly reports. Visualizing the metrics helps to monitor the projects. Especially from the supervisor's point of view, it is easy to catch issues, such as too slow accumulation of code commits or working hours, or if the requirements stay too long in certain states.

When using MMT in teaching of software project work or in software development, there could be some material (lecture, video, online help) for introducing the tool to the users to help and motivate them to utilize the useful features.

Using and evaluating the tool will be continued in the next semesters and new features will be developed on the base of this evaluation. It could also be worth researching further how the new features, like interactive commenting of weekly reports (implemented in version 1.3) or predicting the project's state (proposed to be developed) would help in teaching and software development.

REFERENCES

[1] Bootstrap (2016). A framework for developing responsive, mobile first projects on the web. Available: http://getbootstrap.com/.

[2] Computer Science Curricula 2013. The joint task force on computing curricula, Association for Computing Machinery (ACM) and IEEE Computer Society, 2013.

[3] Charette, R.N (2005). Why Software Fails. Available: http://spectrum.ieee.org/computing/software/why-software-fails.

[4] Fenton, N. Software Metrics - A Rigorous Approach. Chapmann & Hall, London, 1991.

[5] Metrics Monitoring Tool. School of Information Sciences, University of Tampere. Available: http://metricsmonitoring.sis.uta.fi/.

[6] Mäkiaho, P., T. Poranen, A. Seppi. Software metrics in students' software development projects. In Proceedings of International Conference on Computer Systems and Technologies (CompSysTech'15), pages 75-82, 2015.

[7] Portillo-Rodríguez J., Vizcaíno A, Piattini M, Beecham S, Tools used in Global Software Engineering: A systematic mapping review, Information and Software Technology, Volume 54, Issue 7, July 2012, Pages 663-685.

[8] Redmine. Web-based project management and issue tracking tool. Available: http://www.redmine.org, 2016.

[9] Sommerville, I. Software Engineering, 9th edition. Addison-Wesley, 2010.

ABOUT THE AUTHORS

PhD student Pekka Mäkiaho, MSc, School of Information Sciences, University of Tampere, Finland, Phone: +358 50 556 6266, E-mail: pekka.makiaho@uta.fi.

MSc student Katriina Löytty, BBA, School of Information Sciences, University of Tampere, Finland, Phone: +358 50 509 9008, E-mail: katriina.loytty@uta.fi.

University lecturer Timo Poranen, PhD, School of Information Sciences, University of Tampere, Finland, E-mail: timo.t.poranen@uta.fi.

The paper has been reviewed.