Software Analytics for MDE Communities (Position Paper)

James Williams[1], Nicholas Matragkas[1], Dimitris Kolovos[1], Ioannis Korkontzelos[2], Sophia Ananiadou[2], and Richard Paige[1]

[1] Department of Computer Science, University of York, York, UK
{james.r.williams,nicholas.matragkas,dimitris.kolovos,richard.paige}@cs.york.ac.uk
http://www.cs.york.ac.uk
[2] NaCTeM, Manchester University, Manchester, UK
{ioannis.korkontzelos,sophia.ananiadou}@manchester.ac.uk
http://http://www.nactem.ac.uk/

Abstract. The MDE research community has grown significantly in size and maturity in the last decade. During these years, a plethora of languages and tools has been proposed to address different challenges of the MDE approach to software development. As the community continues to broaden its expertise, it is worthwhile to assess its practices in order to identify areas of improvement. To this end, we propose to apply software analytics techniques on open source MDE tools and languages. In this paper, we describe how such an analysis can take place, and we present a set of preliminary results on analysing the newsgroup activity of 10 Eclipse modelling projects.

Keywords: MDE, Data Mining, Open Source Software, Automated Analysis

1 Introduction

Since the first International Workshop on the "Unified Modelling Language (UML): Beyond the Notation" in Mulhouse in 1998 [2], the MDE research community has grown significantly in size and maturity. During these years, a plethora of languages and tools has been proposed to address the different challenges of the MDE approach to software development. Currently, the MDE community is at a stage, where it attempts to assess its impact on the industrial practice of software engineering. To this end, several empirical investigations have been conducted (e.g., [14, 5, 12]) and most of them report similar findings. More particularly, it is found that the industrial adoption of MDE has been slow despite the published verifiable evidence for enhancing both developer productivity and product quality [14].

Poor industrial adoption is attributed to technical, economic, and social factors. Technical barriers are both pragmatic and theoretical [14]. Pragmatic barriers are primarily concerned with the provided MDE tool support, namely its

usability, scalability and reliability [9, 15]. On the other hand the theoretical limitations have to do with our fundamental understanding of software and systems modelling. More particularly, [14] notes that we lack deeper understanding in the areas of language design, model synchronisation, model transformations, and model validation. Furthermore, economic barriers to the industrial adoption of MDE have to do with the training and the investment in time and tools required by an organisation in order to adopt MDE. Finally, the social factors influencing MDE adoption are related to the reluctance of organisations to try new things, and the lack of MDE skills in the workforce [20].

Although the aforementioned studies provide a valuable empirical assessment of MDE in the industry, they do not provide an assessment of the MDE landscape itself. This distinction is subtle but very important. For example, the existing studies have found out that organisations perceive that the barrier to entry in MDE is very high. However, there is no study to assess why this perception exists. Such a perception could be the result of poor documentation of MDE tools, of unhelpful communities, or a combination of both. In this paper, we propose the application of software analytics in order to analyse the MDE landscape and to gain a better understanding of its current state, as well as of its future potential. The combined understanding of the MDE landscape and of the industrial perceptions and practices can allow the MDE community to address any existing issues in order to make MDE part of the mainstream software engineering practices.

In addition to gaining understanding, an analysis of the MDE landscape could provide a useful tool to potential adopters, who could use the identified advantages and limitations of the various MDE tools during decision making. Deciding whether an MDE tool meets the required standards for adoption in terms of quality, maturity, activity of development and user support is not a straightforward process. This task involves exploring a variety of sources of information including the tool's source code repositories (if the tool is open source), communication channels, and bug tracking systems. Source code repositories can help the potential adopter to identify how actively the code is developed and how thoroughly the code is tested. Communication channels, such as newsgroups, forums and mailing lists, help identify whether user questions are answered in a timely and satisfactory manner, and help estimate the number of experts and users of the software. Bug tracking systems can show whether the software has many open bugs and at which rate bugs are fixed. By providing the results of such an assessment to possible adopters, we reduce the cost in terms of time and effort of performing this analysis themselves. Moreover, the risk of adoption is reduced since they can make an informed decision on whether to adopt or not a particular technology.

The rest of the paper is organised as follows: Section 2.1 identifies which a set of characteristics to measure. Section 2.2 describes how MDE open source projects could be used for the proposed analysis, while Section 2.3 describes briefly the infrastructure needed. Section 3 presents the preliminary results on

analysing the activity of the newsgroups of 10 Eclipse modelling projects. Finaly, Section 4 concludes the discussion and presents future work.

2 Software Analytics for MDE

In 1883 Lord Kelvin said "To measure is to know". This principle of quantified knowledge is at the crux of software analytics, which is the field of finding meaningful patterns in software engineering data. The application of software analytics can help answer important questions tool adopters ask such as "How effective is this project at fixing bugs?", "Is the community of this project active?" or "How quickly will I get help if I have a problem?". In the following sections, we briefly discuss on how we plan to apply software analytics to MDE.

2.1 Start with a Question

Every analytics exercise starts with a set of interesting questions to be answered. In the case of applying analytics to MDE, we propose that this set of questions should be inspired by the research literature on open source adoption, which reports on the factors, which make an OSS project successful, i.e. a project which is long lived and adopted by a large number of individuals. [10, 19, 3]. OSS success is affected by five broad categories of factors, namely software quality, community quality, project activity, user interest, and developer interest [10, 19]. By using the identified success factors as goals, and by using the Goal/Question/Metric (GQM) methodology [1], we can derive a set of metrics to use in order to measure aspects of OSS MDE projects.

Software Quality : Studies have found that software quality has a considerable influence on individual satisfaction [4]. In the case of OSS MDE tools, individuals can be simple users, or developers, who want to reuse, extend, or modify the tool. Traditional software quality models such as ISO-196 [7] propose to measure aspects of the software such as its maintainability, reusability and scalability.

Community Quality : In the context of OSS, community quality can be interpreted as the technical support provided by the members of an OSS project to other members. Following [10], community quality can be defined as "individual's perception about the reliability, responsiveness, assurance, and empathy of the service provided by the OSS development community". Studies have found that there is a strong correlation between community quality and individual satisfaction [13].

Project Activity : There are two main types of activity in an OSS project. First, it is the development activity and size, i.e. how fast bugs are fixed, how often new features are added, how often there is a new release etc. On the other hand, there is the community activity and size, which is related to the activity in the communication channels of the project. Communication channels include user

newsgroups, developer mailing lists, bug tracking systems, etc. Crowston et al. [3] showed that there is a strong correlation between project activity and project success. Subramaniam et al. [19] argues that OSS software displays direct and indirect network effects. OSS projects benefit from an increased number of users because the communication and the contributions potentially increase and the knowledge base for this project increases.

User Interest : User interest is one of the most relevant aspects of OSS success. Following [18], user interest is defined as the ability of a project to attract users to adopt the project. By having a large pool of users, a project enjoys better community support, faster bug fixes and bug reports, etc. Indicators of user interest include the number of downloads [19], number of individuals participating in newsgroups [3], or traffic on the project's Web site [3].

Developer Interest : In the OSS context, the distinction between a user and a developer is blurred. Users usually make use of the software but they are also developing it. However, researchers have argued that every project consists of a very small number of core developers, who are developing the main features of the software, and a larger number of 'peripheral' developers, who make minor contributions, fix bugs, and write documentation. In the context of our work, we regard the former as developers, and the latter as users. A large and steady number of core developers ensures active development. [17] suggest that developer interest is strongly correlated to user interest. Most particularly, developers' interest in a project increases as the user interest increases as their intellectual contributions have a wider audience.

2.2 Obtain the data

Once we have a set of questions, we need to obtain the appropriate data in order to quantitatively answer these questions. The MDE community consists to a large extent of open source projects. [6] provides a long list of MDE tools and languages used in the industry and more than half of them are open source. Moreover, some of these tools and languages hold such a prominent position in the MDE landscape, that they are considered to be de facto standards. The most well known of these is the Eclipse Modelling Framework (EMF) [16], which is considered to be the metamodelling de facto standard [8]. These projects are either hosted on public forges such as the Eclipse forge[3] or on private servers. To obtain the data for the propose analysis, three types of resources should be mined. First, version control systems such as SVN[4] or Git[5] should be mined to obtain data about the source code of a tool, the development activity and the size of the developer community. Second, bug tracking systems such as Bugzilla[6]

[3] `http://projects.eclipse.org/list-of-projects`
[4] `http://subversion.apache.org/`
[5] `http://git-scm.com/`
[6] `http://www.bugzilla.org/`

should be mined in order to obtain data about the number of bugs, communication activity, and average time to fix a bug. Finally, communication channels such as mailing lists and newsgroups should be mined to obtain data about the size of the user community, the user satisfaction, and user participation.

2.3 Perform the analysis

To measure and monitor the MDE landscape, we need a tool which is able first to extract the required data from the various information sources, and then analyse them in an automated manner. The OSSMETER project[7] provides a OSS extensible telemetry platform, which can perform this type of analysis. The platform integrates and manages a number of software measurement components, which extract and analyse software project data in regular time intervals. Furthermore, the platform facilitates the visualisation of measurements, as well as direct comparison of projects.

3 Preliminary Results

As a first step towards realising the proposed analysis, we have analysed the activity of the newsgroups of 10 open source modelling projects hosted by the Eclipse Foundation. Newsgroup activity relates to the project activity and user interest factors of project success identified in Section 2.1. The projects were selected due to having more than 2000 messages in their newsgroup (as of 16th April, 2014). For each project we monitored how the following metrics evolved over time:

- Number of new users
- Number of new threads
- Number of active users (measured by counting the number of users who posted in the previous 15 days)
- Number of messages classified as requests and as replies
- The average number of requests and replies per thread

A complete list of the analysed projects as well as the results of the analysis can be found at: `http://www.ossmeter.com/`. An illustration of the implemented visualisations is shown in Figure 1. Moreover, the raw data for this analysis can be obtained from: `https://github.com/ossmeter/oss4mde`.

Figure 2 illustrates two of the derived plots. More particularly, Figure 2a shows the number of active users of EMF[8] from February 2007 until August 2008. The dotted lines represent project milestones. We can observe that around milestone dates, the number of active users increases. Figure 2b overlays the number of EMF active users with the number of active users of the JDT Eclipse project[9], which is a very popular Eclipse project. From this plot we can observe

[7] `http://www.ossmeter.org/`
[8] `http://www.eclipse.org/modeling/emf/`
[9] `http://www.eclipse.org/jdt/`

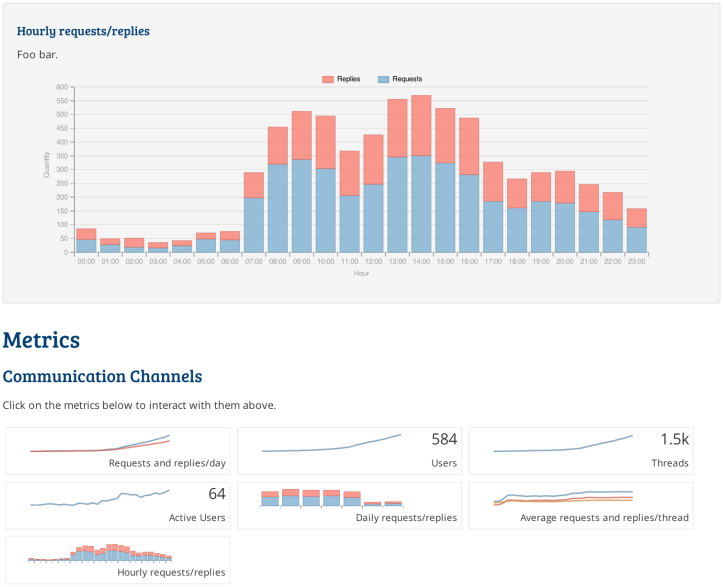


Fig. 1: Result visualisations.

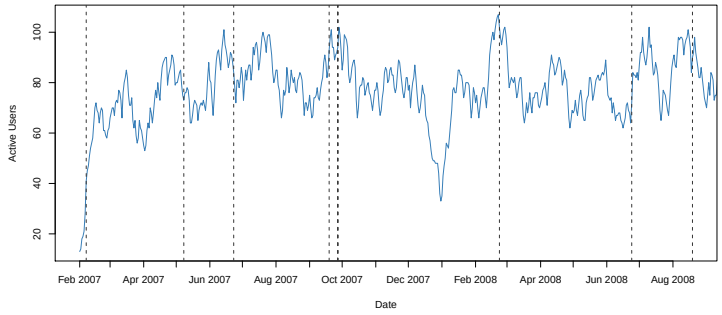
the EMF demonstrates similar levels of activity compared to JDT, which is a positive sign.

One interesting pattern, noticed is a decrease in the number of active users of the newsgroups. Table 1 lists the average number of active users from 2010 until April 2014. Seven out of ten projects demonstrate this pattern, meaning that less people are frequent contributors to the discussions on the newsgroups. This can be an indication of a shrinking community.

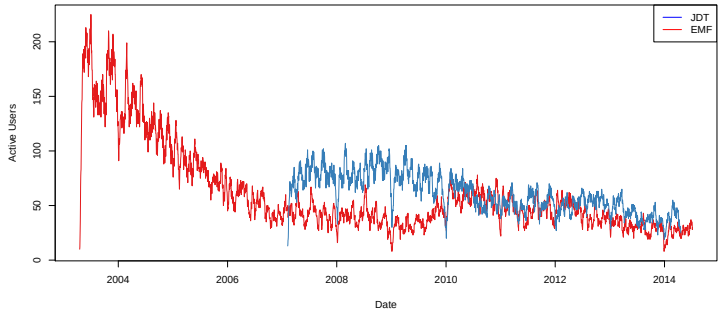
We have also observed that *active users* metric correlates with a decrease in the number of new users and new threads. This is particularly prevalent in the GMF project[10] - a set of libraries and tools for generating graphical editors for modelling languages. Back in 2008 the number of active users fluctuated between 70 and 100, whereas in the last month (April 2014) the newsgroup has approximately 10 active users. This suggests that the users of GMF may have begun to migrate to an alternative tool. One potential project that may be gaining GMF's old user base is Graphiti[11], which has shown steady growth in new users since it's creation in 2010.

Two languages that go hand-in-hand in the modelling world are UML and OCL. There are Eclipse projects that provide support for both of these languages (UML2, UML2Tools, OCL[12]). Whereas the OCL project has maintained a steady number of active users and still gains new users each month, UML2

[10] `http://www.eclipse.org/modeling/gmp/`
[11] `https://www.eclipse.org/graphiti/`
[12] `http://www.eclipse.org/modeling/mdt/?project=uml2`



(a) Number of active users for EMF.



(b) Active users for JDT and EMF projects.

Fig. 2: Plots related to the active users of EMF.

and UML2Tools are losing users and activity in the newsgroup is declining. As GMF users may be migrating to Graphiti, UML2 users may be moving to a project such as Papyrus[13] - a project providing graphical editors for UML (and other languages), with industrial backing. One potential reason for OCLs continued activity is that it can be used with EMF as well as UML. The requests/replies per day metrics for OCL shows the lack of developer response in the early years, where the requests outnumbered replies. Since August 2013, however, the number of replies has overtaken the number of requests, suggesting an active development team behind the project.

Similarly, Table 2 shows the number of average messages per month from 2010 until April 2014. The number of messages demonstrates a similar pattern to the number of active users. More particularly, half of the projects have a decreasing number of new messages in their newsgroups. This can be an indication of declining user participation on the newsgroups of the particular projects.

[13] `http://www.eclipse.org/papyrus/`

Table 1: Average number of active users per day.

	2010	2011	2012	2013	2014	Change
Papyrus	9	11	15	20	13	44% ↑
UML2Tools	8	3	3	2	2	-74% ↓
UML2	10	6	7	6	4	-60% ↓
OCL	5	7	7	7	7	36% ↑
M2T	27	20	16	11	8	-70% ↓
Graphiti	10	15	14	16	15	48% ↑
GMF	39	26	18	13	6	-84% ↓
Epsilon	11	8	7	10	10	-15% ↓
EMFT	19	13	12	9	17	-11% ↓
EMF	59	53	52	42	38	-35% ↓

On the other hand, there are projects such as Papyrus and Eclipse OCL, which demonstrate remarkable increase. This can possibly indicate an internal mobility of users within MDE projects.

Concerning the *number of threads* metric. the Epsilon project[14], has the longest message threads out of all the projects analysed. Currently it averages 4.8 messages per thread, consisting of 1.8 requests and 2.9 replies. Typically, the average thread length in all projects is between three and four. This is likely the original question followed by an answer then a "thank you". Extra messages may be asking for clarification, or follow up questions.

4 Conclusions

In this paper, we have argued for the use of software analytics for assessing the current state of the MDE landscape. With a better understanding of the MDE landscape and of the perceptions of MDE in the industry, the MDE community can take the necessary steps in order to make MDE a standard practice in industrial software engineering. To perform the required analysis we propose the use of the many open source MDE tools and languages, which are hosted on public forges and private servers. Furthermore, we propose the user of the success factors of OSS projects as an initial set of characteristics to measure. Finally, we report on some of our initial findings related to measuring newsgroup activity in 10 popular Eclipse modelling projects.

There is a great amount of work that can be done in the context of analysing MDE projects; we have only scratched the surface here. In addition to analysing the source code repositories and the bug tracking systems, we plan to further analyse each newsgroup message to determine its sentiment, allowing us to gain

[14] `http://www.eclipse.org/epsilon/`

Table 2: Average number of messages per month.

	2010	2011	2012	2013	2014	Change
	2010	**2011**	**2012**	**2013**	**2014**	**Change**
Papyrus	33	53	76	109	82	146% ↑
UML2Tools	25	7	7	2	1	-96% ↓
UML2	45	29	42	28	9	-79% ↓
OCL	40	45	69	68	69	75% ↑
M2T	179	125	101	45	37	-79% ↓
Graphiti	69	97	95	82	87	27% ↑
GMF	192	157	93	78	20	-89% ↓
Epsilon	-	-	-	-	-	-
EMFT	99	58	67	82	106	7% ↑
EMF	475	427	451	351	282	-40% ↓

insight into how happy users are, or whether they are frustrated with the software/support they are receiving. This will allow us to understand whether users are receiving good support, and may also enable us to answer interesting questions, such as "Do longer threads tend to have less happy users?". Finally, we plan to extend our analysis to more MDE tools and languages.

Acknowledgements

This research was part supported by the EU, through the OSSMETER FP7 STREP project (#318736).

References

1. Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. The goal question metric approach. In *Encyclopedia of Software Engineering*. Wiley, 1994.
2. Jean Bézivin and Pierre-Alain Muller, editors. *The Unified Modeling Language, UML'98: Beyond the Notation, First International Workshop, Mulhouse, France, June 3-4, 1998, Selected Papers*, volume 1618 of *Lecture Notes in Computer Science*. Springer, 1999.
3. Kevin Crowston and Hala Annabi. Information systems success in free and open source software development: Theory and measures. In *Software Process: Improvement and Practice*, pages 123–148, 2006.
4. Jamshid Etezadi-Amoli and Ali F. Farhoomand. A structural model of end user computing satisfaction and user performance. *Inf. Manage.*, 30(2):65–73, May 1996.
5. John Hutchinson, Jon Whittle, Mark Rouncefield, and Steinar Kristoffersen. Empirical assessment of mde in industry. In *Proceedings of the 33rd International*

Conference on Software Engineering, ICSE '11, pages 471–480, New York, NY, USA, 2011. ACM.
6. John E. Hutchinson. *An Empirical Assessment of Model Driven Development in Industry*. PhD thesis, Lancaster University, December 2011.
7. ISO/IEC. *ISO/IEC 9126. Software engineering – Product quality*. ISO/IEC, 2001.
8. Dimitrios S Kolovos, Louis M Rose, Saad Bin Abid, Richard F Paige, Fiona AC Polack, and Goetz Botterweck. Taming emf and gmf using model transformation. In *Model Driven Engineering Languages and Systems*, pages 211–225. Springer, 2010.
9. Adrian Kuhn, Gail C. Murphy, and C. Albert Thompson. An exploratory study of forces and frictions affecting large-scale model-driven development. *CoRR*, abs/1207.0855, 2012.
10. Sang-Yong Tom Lee, Hee-Woong Kim, and Sumeet Gupta. Measuring open source software success. *Omega*, 37(2):426 – 438, 2009.
11. Nigel Meade and Towhidul Islam. Modelling and forecasting the diffusion of innovation a 25-year review. *International Journal of Forecasting*, 22(3):519 – 545, 2006. Twenty five years of forecasting.
12. Parastoo Mohagheghi, Wasif Gilani, Alin Stefanescu, and Miguel A. Fernández. An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases. *Empirical Software Engineering*, 18(1):89–116, 2013.
13. Richard L Oliver. Cognitive, affective, and attribute bases of the satisfaction response. *Journal of Consumer Research*, 20(3):418–30, 1993.
14. Bran Selic. What will it take? a view on adoption of model-based methods in practice. *Softw. Syst. Model.*, 11(4):513–526, October 2012.
15. Miroslaw Staron. Adopting Model Driven Software Development in Industry A Case Study at Two Companies. In Oscar Nierstrasz, Jon Whittle, David Harel, and Gianna Reggio, editors, *Model Driven Engineering Languages and Systems*, volume 4199, chapter 5, pages 57–72. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
16. Dave Steinberg, Frank Budinsky, Ed Merks, and Marcelo Paternostro. *EMF: eclipse modeling framework*. Pearson Education, 2008.
17. Katherine J. Stewart, Anthony P. Ammeter, and Likoebe M. Maruping. Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects. *Info. Sys. Research*, 17(2):126–144, June 2006.
18. Katherine J. Stewart and Sanjay Gosain. The impact of ideology on effectiveness in open source software development teams. *MIS Q.*, 30(2):291–314, June 2006.
19. Chandrasekar Subramaniam, Ravi Sen, and Matthew L. Nelson. Determinants of open source software project success: A longitudinal study. *Decis. Support Syst.*, 46(2):576–585, January 2009.
20. Jon Whittle, John Hutchinson, Mark Rouncefield, Hkan Burden, and Rogardt Heldal. Industrial adoption of model-driven engineering: Are the tools really the problem? In Ana Moreira, Bernhard Schtz, Jeff Gray, Antonio Vallecillo, and Peter J. Clarke, editors, *MoDELS*, volume 8107 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013.