Forecasting the number of changes in Eclipse using time series analysis*

Israel Herraiz, Jesus M. Gonzalez-Barahona, Gregorio Robles
Grupo de Sistemas y Comunicaciones
Universidad Rey Juan Carlos, Spain
{herraiz, jgb, grex}@gsyc.escet.urjc.es
Libresoft team

Abstract

In order to predict the number of changes in the following months for the project Eclipse, we have applied a statistical (non-explanatory) model based on time series analysis. We have obtained the monthly number of changes in the CVS repository of Eclipse, using the CVSAnalY tool. The input to our model was the filtered series of the number of changes per month, and the output was the number of changes per month for the next three months. Then we aggregated the results of the three months to obtain the total number of changes in the given period in the challenge.

1 Introduction

There have been some cases of proposals of predictive models for libre (free / open source) software projects. In our opinion, the phenomenon of libre software development is quite complex as to obtain a satisfactory explanatory model. For instance, in spite of the proposed models in the literature [9, 4, 1], little empirical validation of these models have been done, so failing on the prediction of the actual evolution of libre software projects.

Using the "low-level" approach taken by the mentioned papers is a difficult task, because the events that happen within a project are random-like. All the interactions (a change made to the source code, a new message to the mailing list, a new developer coming to the project, a developer leaving the project) that we may find in a project can not be predicted, because they involved people. However, if we look at the macroscopic level, the aggregation of all these random-like interactions is not random, and despite containing noise, can be predicted by means of statistical methods. Think for example of the stock market. It is really difficult to predict how individual actors will behave, because of the many factors that may impact their actions. But when the global stock market is considered, several statistical methods may be used to predict the near future in absence of impacting external events.

The idea of using time series analysis to predict software is not new. Already in the period from 1985 to 1988 several papers [11, 12, 13] used statistical methods, including time series analysis, to model software evolution. For instance, in [13] time series ARIMA models are used to predict the evolution in the maintenance phase of a software project, using sampling periods of about one month.

Later, Kemerer and Slaughter [6] followed this line of research proposing an ARIMA model which is able to predict the monthly number of changes of a software project. However, they did not obtain very good results, because the phenomenon studied (monthly number of changes, the same than in the challenge) is quite noisy. To avoid the problems found by Kemerer and Slaughter, we applied *kernel smoothing* in the hope of reducing the amount of noise in the source data.

There have been several other research papers using time series methods to predict the evolution of software projects. Because of the space restrictions, we just cite them here [2, 3, 5].

2 Methodology

We obtained the monthly number of changes for Eclipse, classified by plugins (using the mapping table provided for the challange), using the tool CVSAnalY [8]. From the database created by CVSAnalY, we mined the revisions that were not deletions, and added up all the revisions in each one of the months, from the beginning of the history in the CVS until the last of January 2007.

Therefore, we obtained a list with the number of changes for every month since the beginning of the history of each

*This work has been funded in part by the European Commission, under the FLOSSMETRICS (FP6-IST-5-033547) and QUALOSS (FP6-IST-5-033547) projects. Israel Herraiz has been funded in part by Consejería de Educación of Comunidad de Madrid and European Social Fund, under grant number 01/FPI/0582/2005.

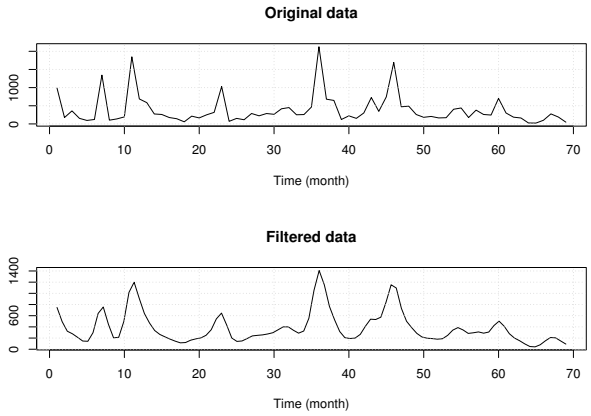

Figure 1. Original data and filtered data for the case of org.eclipse.core. Kernel smoothing with a bandwidth of 2.

plugin in the CVS. The data was actually obtained for each one of the subdirectories in the CVS. We mapped subdirectories to plugins adding up the changes for every subdirectory being part of the same plugin, in the hope of reducing some noise by aggregation. We needed to filter the data, though. We filtered this data using *kernel smoothing* with a bandwidth of 2, and the filtered series was the input for the ARIMA model. Figure 1 shows the original data and the the filtered data for the plugin `org.eclipse.core`. We have used [7] as a guide. That book includes some examples and charts to select the right model based on the values of some parameters that we describe below.

The ARIMA model has three parameters:

- d, which is the number of differences needed to make the data stationary. In our case it was $d = 1$ for all the plugins.
- p, which is the *auto-regressive* part of the model. The right value is obtained by inspecting the autocorrelation coefficients and partial autocorrelation coefficients plots. In all the plugins, it was value was between 2 and 3.
- q, which is the *moving average* part of the model. Again, it is obtained by inspecting the autocorrelation coefficients and partial autocorrelation coefficients plots. This time, it was 0 for all the plugins.

We inspected the autocorrelation plots and partial autocorrelation plots for all the plugins. Then we selected the values for p and q for each case, obtained the model, obtained the predictions for the next three months.

We then added up the results for the next three months for every plugin, and those were the results that we submitted to the challenge.

For more details on how to apply this methodology we recommend to read [7, 10].

References

[1] I. Antoniades, I. Samoladas, I. Stamelos, L. Aggelis, and G. L. Bleris. Dynamical simulation models of the open source development process. In S. Koch, editor, *Free/Open Source Software Development*, pages 174–202. Idea Group Publishing, Hershey, PA, 2004.

[2] G. Antoniol, G. Casazza, M. D. Penta, and E. Merlo. Modeling clones evolution through time series. In *Proceedings of the International Conference on Software Maintenance*, 2001.

[3] F. Caprio, G. Casazza, M. D. Penta, and U. Villano. Measuring and predicting the Linux kernel evolution. In *Proceedings of the International Workshop of Empirical Studies on Software Maintenance*, Florence, Italy, 2001.

[4] J.-M. Dalle and P. A. David. The allocation of software development resources in Open Source production mode. Technical report, SIEPR Policy paper No. 02-027, SIEPR, Stanford, USA, 2003. http://siepr.stanford.edu/papers/pdf/02-27.pdf.

[5] E. Fuentetaja and D. J. Bagert. Software Evolution from a Time-Series perspective. In *Proceedings of the International Conference on Software Maintenance*, pages 226–229, 2002.

[6] C. F. Kemerer and S. Slaughter. An empirical approach to studying software evolution. *IEEE Transactions on Software Engineering*, 25(4):493–509, 1999.

[7] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman. *Forecasting: Methods and Applications*. John Wiley & Sons, Ltd., January 1998.

[8] G. Robles, S. Koch, and J. M. González-Barahona. Remote analysis and measurement of libre software systems by means of the CVSAnalY tool. In *Proceedings of the 2nd ICSE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS)*, pages 51–56, Edinburg, Scotland, UK, 2004.

[9] G. Robles, J. J. Merelo, and J. M. Gonzalez-Barahona. Self-organized development in libre software: a model based on the stigmergy concept. In *Proceedings of the 6th International Workshop on Software Process Simulation and Modeling (ProSim 2005)*, St.Louis, Missouri, USA, May 2005.

[10] R. H. Shumway and D. S. Stoffer. *Time Series Analysis and its Applications. With R Examples.* Springer Texts in Statistics. Springer, 2006.

[11] C. C. H. Yuen. An empirical approach to the study of errors in large software under maintenance. In *Proceedings of the International Conference on Software Maintenance*, 1985.

[12] C. C. H. Yuen. A statistical rationale for evolution dynamics concepts. In *Proceedings of the International Conference on Software Maintenance*, 1987.

[13] C. C. H. Yuen. On analyzing maintenance process data at the global and detailed levels. In *Proceedings of the International Conference on Software Maintenance*, pages 248–255, 1988.