Selecting a software process modeling tool: A quantitative comparison method applied

Pablo Szyrko, Mauricio Silclir, Diego Rubio
{pablo.szyrko, msilclir, rubiodiego}@gmail.com
Engineering and Software Quality Research Group
http://www.institucional.frc.utn.edu.ar/sistemas/lidicalso/
Information Systems Engineering Department
Universidad Tecnológica Nacional
Maestro M. López esq. Cruz Roja Argentina
Córdoba, ZIP X50165ZA, Argentina

ABSTRACT

Many aspects of our lives are governed by large, complex systems with increasingly complex software, and the safety, security, and reliability of these systems has become a major concern [1]. As a consequence, nowadays, an increasing number of companies are using process improvement models as a mean to improve the quality of their processes and therefore the quality of their products and services [2]

In this context of permanent process changes and improvements, organizations need a modeling language that allows them to define their own processes (a new process or a customization of an existent one), verify them and validate any change to be made on that process. In order to fulfill these requirements the modeling language used, must be not only friendly and unambiguous but should provide an automated tool that supports process modeling.

This paper establishes a set of requirements that any software development process modeling tool must fulfill. In addition, this work provides a quantitative comparison method that allows selecting the most suitable tool based on the analysis and study of a set of alternative available tools.

The formal process of analysis and selection of the most suitable tool is composed of three stages. The first one is the requirements definition and categorization. All the analysis of the modeling tools is based on how each of them fulfills every requirement defined in this step. The categorization defines a hierarchy between the requirements, to identify those with high value in relation to the final purpose from those which presence is desired but is certainly optional. In addition, a quantitative value is assigned to each category, to be used when applying the quantitative method of selection.

Once the set of requirements has been established and categorized, the objective of the second stage is to execute the analysis of each tool, in order to determine how their implementation accomplishes each requirement. There are three levels of compliance: Absolute, Partial, and Not Compliant. A quantitative value, the rating, is associated with each level of compliance, to be used at the moment of applying the quantitative method of selection.

As a third and last stage, the tools comparative analysis is performed. At that moment, all the results proceeding from the analysis of each tool are summarized, and the arithmetic operations of the model are applied, obtaining a general value based on two dimensions: the value applied to each requirement and the level of compliance of each tool for each one of those requirements. As an outcome, the model discards those tools that does not fulfill mandatory requirements, and after that proceeds with the calculation of the final quantitative value of each tool. Once all the calculations have been performed, a prioritized ranking related to the weighted requirements is presented.

This model is presented with the proposed values as default but it allows for tailoring and customization. It is possible to modify the requirement categories and ratings, according to each organization particular needs and objectives and it is also possible to add new requirements, breakdown the existent ones and eliminate some of them, being these tasks part of the customization stage of the model.

Finally, this paper presents step by step implementation of the proposed methodology along with the final evaluation outcomes. The evaluation was done as part of the activities of the research project "Development of an automated validation model for the definition and maintenance of software development processes"[3], which was started in 2009 as part of the "Software Quality and Engineering development and research Lab" at the "Universidad Tecnológica Nacional Facultad Regional Córdoba" [4]. For this particular example, the following tools were fully analyzed against all of the established modeling requirements: Eclipse Process Framework Composer (EPF-Composer) [5], Microsoft Team Foundation Server (TFS) [6] and IBM Rational Team Concert (RTC) [7]; presenting as a consequence a predefined set of values for those organizations analyzing the implementation of the mentioned software modeling tools.

Keywords: Software process modeling tool, Quantitative selection method and Requirements definition and analysis.

REFERENCES

[1] W. S. Humphrey. **The Software Quality Challenge.** The Software Engineering Institute. Crosstalk: The journal of Defense Software Engineering. Vol.21, No. 6. Page 4. June, 2008.

[2] **CMMI Appraisal Program.** Software Engineering Institute (SEI). Process Maturity Profile CMMI® SCAMPISM Class A Appraisal Results 2007 Year-End Update. Pittsburgh, Pennsylvania, USA: Carnegie Mellon University, Page 20. March 2008. http://www.sei.cmu.edu/appraisal-program/profile/pdf/CMMI/2008MarCMMI.pdf.

[3] P. Szyrko, M. Silclir, G. García Favre, D. Rubio. **Development of an automated validation model for the definition and maintenance of software development processes.** April 2009. http://www.institucional.frc.utn.edu.ar/sistemas/gidicalso/pub/file/Publicaciones/WICC09%20-%20Automatizacion_Procesos.pdf

[4] **Universidad Tecnológica Nacional – Facultad Regional Córdoba.** http://www.frc.utn.edu.ar/

[5] **Eclipse Process Framework Composer - Part 1Key Concepts.** 2007. http://www.eclipse.org/epf/general/EPFComposerOverviewPart1.pdf

[6] **Team Foundation Server.** http://msdn.microsoft.com/en-us/library/ms181238%28VS.80%29.aspx

[7] **Rational Team Concert.** http://www.ibm.com/developerworks/rational/products/rtc/