CRONUS: A TASK MANAGEMENT SYSTEM TO SUPPORT SOFTWARE DEVELOPMENT

Yura Ferreira[1], Sergio Assis Rodrigues[1], Divany Gomes Lima[1], Márcio Luiz Ferreira Duran[1]
José Roberto Blaschek[1] and Jano Moreira de Souza[1, 2]
[1]*COPPE/UFRJ – Computer Science Department, Federal University of Rio de Janeiro, Brazil*
[2]*DCC-IM/UFRJ – Computer Science Department, Mathematics Institute, Federal University of Rio de Janeiro, Brazil*

Keywords: Project Management, Software Metrics, Task Management.

Abstract: Currently, information technology professionals have become increasingly interested in factors that may have an impact on project management effectiveness and the success of projects. This article introduces a task management tool wich complements traditional tools to support the planning, controlling and execution of software development projects.

1 INTRODUCTION

There are theoretical and empirical researches that highlight the importance of metrics and the appropriate time to use them in software projects (Riguzzi, 1996). As there is a high probability that the real project execution differ from what was initially planned (Boehm, 1996), it is important to predict whether the milestones will be achieved as stated in the contractual plan. Keeping project status information updated demands great effort and high costs.

The CHAOS report (Standish Group, 2004) focused on commercial software industry analyzing 50.000 IT projects in 1994 and met that: 31% of the analyzed projects were cancelled before their closure, 53% exceeded more than 50% of the initially estimated costs and only 16% succeeded. Ten years later, this scenario has slightly improved, indicating that 29% of software projects were delivered according to the scope, time and costs planned, 18% of projects were cancelled before their closure and 53% had substantial changes in time and budget. According to those reports, low user interactions and lack of appropriate monitoring are among the main causes of software project problems. The Cronus tool was developed based on 10 years of experience in software project development to large institutions of Brazilian private and public sectors. These projects are characterized by the use of cutting-edge technology and applied research (Rodrigues et al, 2009). Cronus is a task organizer, acting as a coordination and communication tool among the professionals. Moreover, Cronus enhances the development process allowing a detailed control by team leaders. This work presents the Cronus tool, the process on which it is based and indicates its level of acceptance by users.

2 APPLIED CONCEPTS

2.1 Project Management

Generally companies select the best practices based on what succeeds or fails in project management; however, the best practices are not necessarily the same in other companies (Kerzner, 2003). Although some researchers claim that a pre-defined set of techniques and tools may lead a project to success, there are evidences that it does not happen even in projects of a single company (Shenhar, 2005). Choosing one methodology to each project is a big mistake. However, the chosen methodology should be improved so as to follow the changes in project management due to the constant evolution in the organizational environment and in technology (Kerzner, 2003).

Project management tools play an important role in methodology support. Researches have shown that since there are not methodologies that cover all

Ferreira Y., Assis Rodrigues S., Gomes Lima D., Luiz Ferreira Duran M., Roberto Blaschek J. and Moreira de Souza J. (2009).
CRONUS: A TASK MANAGEMENT SYSTEM TO SUPPORT SOFTWARE DEVELOPMENT.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Artificial Intelligence and Decision Support Systems*, pages 376-379
DOI: 10.5220/0002011803760379

possibilities, the tools are not supposed to play this role (Soroczak & McDonald, 2006). In the same way Cronus supports generic project management methodology although it favors a methodology of project management that incorporates the recommendations of the PMBOK (2004) with an iterative approach, as indicated in Figure 1.

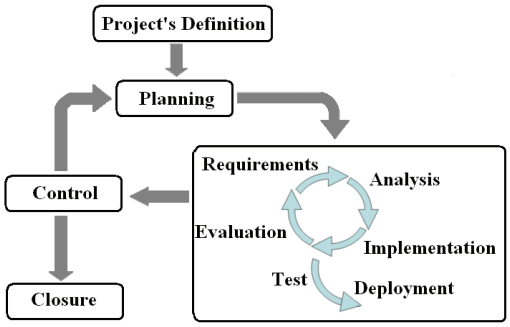


Figure 1: Model of Software Project Management.

2.2 Communication

Communication is the source of many problems in projects. Generally, teams are formed by people who have different background and levels of formation, which makes communication in projects particularly difficult (Forsberg et al, 2005). Coordination, visibility, communication and cooperation are adversely affected by the distance between team members and, if not properly handled, can lead to barriers and complexities in the project. The roles and responsibilities should be clearly indicated so that difficulties in communicating do not become a barrier to team's performance (Casey & Richardson, 2006).

2.3 Software Project Metrics

As you cannot control what you cannot measure (DeMarco, 1982), metrics data are important to gauge costs and benefits and emphasize quality.

Software metrics helps the identification and management of risks before they become critical, the flow of communication (in the team and organization), the evaluation of performance and also supports objective reasons for decision-making (Goethert and Brad, 2000).

Putnam & Myers (2003) defined 5 basic metrics that must be clearly defined and standardized: size, productivity, time, effort and reliability. The authors show that people working at the same level of productivity generate a number of functions or work products based on the reliability level of effort spent in a period of time.

There are two approaches to achieve the metrics: collecting of general indicators for managerial decision or collecting of detailed indicators to monitor specific aspects of the project. Both approaches are necessary. However, data should not be indiscriminately collected since it may be too expensive and may not offer any benefits (Kelsey, 2006). In fact, it is advisable to define an architecture of metrics that can express the monitoring indicators at the project opening as established in the measurement plan. The architecture of metrics should be based on basic measures. These are combined to get the derived measures that are submitted to a model to generate indicators (Kelsey, 2006).

3 THE CRONUS SYSTEM

Cronus is a system based on task management concepts and product oriented monitoring. It does not substitute traditional project management tools but intends to complement them. In the system, the schedule is linked to milestones which reflect the project deliverables. There are four types of users: manager, supervisor, developer and administrator. The manager view concerns managerial reports and task management. The supervisor profile contains several reports; task and effort register and reschedules interfaces. Developers access only effort register, which can be automatic. Administrator has wide access.

The Cronus, unlike other tools which restrict the level of strategic monitoring (activities), covers the tactical level, managing the control and project implementation. Figure 2 shows the Cronus concepts.

Firstly, the work breakdown structure can be stored in the system or extracted from other types of management tool (e.g., Microsoft Project, Excel, dotProject). Secondly, contractual constraints, human resources, and financial aspects are registered. Then, it is possible to control the project execution.

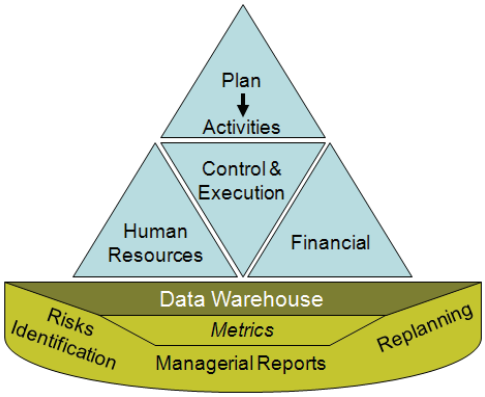
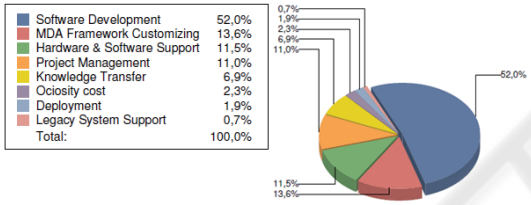


Figure 2: Cronus architecture.

Along the project a wide amount of data is gathered and stored in the Data Warehouse module.

Towards a group of managerial reports, the software uses several metrics to give managers inputs to identify risks and reschedule the project if necessary.



Type of Activity	Cost Per Man-Hour
Software Development	19.155,85
MDA Framework Customizing	5.004,16
Hardware & Software Support	4.229,15
Project Management	4.047,64
Knowledge Transfer	2.546,25
Ociosity cost	858,06
Deployment	705,91
Legacy System Support	276,10
Total	36.823,12

Figure 3: Strategic reports.

Cronus offers easy to use interfaces to stimulate communication and effort registration. From the effort stored, it is possible to manage project activities and distribute tasks to idle developers. The software provides reports extracted from a data warehouse mechanism. Figure 3 shows an example of tabulated data with relevant project information and graphics to facilitate the strategic understanding. These functionalities help managers to lead the software project successfully.

4 SYSTEM EVALUATION

Currently almost 150 team members allocate their effort daily in Cronus. Twenty of them were invited to evaluate Cronus and answer a questionnaire with 10 questions, 8 objective and 2 subjective. Figure 4 shows the schooling profile of the participants.

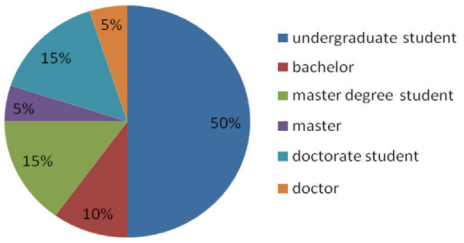


Figure 4: Participants' schooling profile.

When questioned about how difficult it is to use Cronus, around 60% of people agreed that its interface is easy or very easy. No one devaluated the tool and 40% said that the tool was usable. Almost 90% of Cronus' users answered they liked to use it, which means that the tool was well accepted.

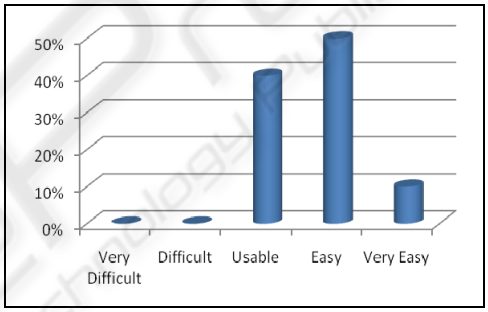


Figure 5: Cronus' usability.

Another important report crosses the participant's profile with utility, usability and the effort register. As mentioned in Table 1, the results are satisfactory once it is almost unanimous that Cronus is useful. In respect to usability, Table 1 shows good results and corroborates the Cronus principle: easy to use.

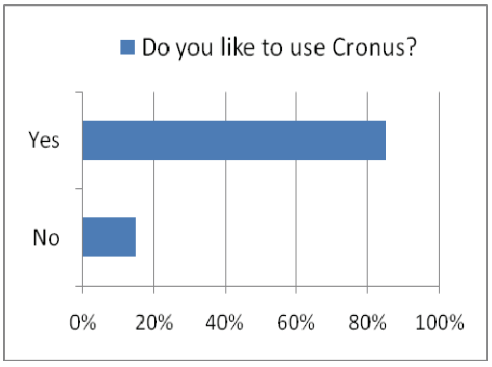


Figure 6: Cronus' appreciation.

Table 1: Crossing results.

Utility					
	No	Low	Yes	Very	N.A.
Leader			25%	75%	
Developer		8%	83%	9%	
Usability					
	Very Difficult	Difficult	Usable	Easy	Very Easy
Leader			24%	63%	13%
Developer			58%	33%	9%
Effort register acceptance					
	0-3	4-6	7-9	10	
Leader		24%	63%	13%	
Developer		17%	83%		

5 CONCLUSIONS

This work shows a task management approach and a computational system to support software development projects. Towards the efficiency in metrics, Cronus establishes its architecture in transforming project planning into activities which makes the challenge of controlling and monitoring tasks easier.

Cronus includes several task management tools to improve the chances to effectively deliver projects. By the use of this software, the expectation is to save time and record effort automatically, improve the deliverable quality, provide transparent financial reports and track changes, risks and issues.

As previously mentioned the system acceptance among its user reaches almost 90%. This good perception is a result of Cronus' usability and credibility, once 95% experimental participants believe in the usefulness of the system.

As future work, new reports will be developed and the automation of effort recording will be enhanced, so as to capture not only the period of time, but also the tools used by developers to perform the tasks.

ACKNOWLEDGEMENTS

This research was sponsored by Brazilian Defense Ministry and Armed Forces; Brazilian Planning Budgeting and Management Ministry; CAPES; CNPq and COPPETEC Foundation.

REFERENCES

Boehm, B., 1996. *Anchoring the software process*, IEEE Software, Vol.13, Issue 4, 73-82.

Casey, V., Richardson, I., 2006. *Project Management within Virtual Software Teams*. IEEE International Conference on Global Software Engineering (ICGSE'06).

DeMarco, T., 1982. *Controlling Software Projects:* Management Measurement and Estimation, Yourdon Press.

Forsberg, K., et al, 2005. *Visualizing Project Management:* Models and Frameworks for Mastering Complex Systems. Hoboken, NJ: Wiley, 3rd ed.

Kelsey, R., 2006. *Software Project Management*: Measures for Improving Performance. Vienna, VA: Management Concepts.

Kerzner, H., 2003. *Project Management:* A Systems Approach to Planning, Scheduling, and Controlling. Wiley, 8th edition.

Goethert, W., Brad, C., 2000. *Managing Software Projects with Metrics*, Software Engineering Institute, Carnegie Mellon University, SEI Symposium Conference.

PMBOK, 2004. *Project Management Body of Knowledge*, 2004 Edition. Project Management Institute, http://www.pmi.org, 2004.

Putnam, L., Myers, W., 2003. *Five Core Metrics*: The Intelligence Behind Successful Software Management. New York: Dorset House Publishing.

Riguzzi, F., 1996. *A survey of software metrics*. Technical Report DEIS-LIA-96-010, LIA Series n.17, DEIS, Università di Bologna.

Rodrigues et. al, 2009. *A Research Model to Support the Software Development Management for the Brazilian Government*. IADIS E-Society 2009, Barcelona.

Shenhar, A., et al, 2005. Toward a NASA-Specific Project Management Framework. *Engineering Management Journal*, December, 1. Disponível em <http://www.allbusiness.com/engineering-management-journal/20051201/3038574-1.html >

Soroczak, S., McDonald, D., 2006. Collaborating Over Project Schedules. in *Workshop on Supporting the Social Side of Large-Scale Software Development*. ACM Conference on Computer Supported Cooperative Work (CSCW'06).

Standish Group, 2004. *Chaos Report*, Standish Group, Disponível em <http://www.standishgroup.com>