MANAGEMENT SYSTEM BASED ON OPEN SOURCE TOOLS

I. Verstovšek[1], J. Kamenik[1]
[1]*Cosylab, Teslova 30, 1000 Ljubljana, Slovenia*

ABSTRACT

Can cosiness and project management go together? As a result of the free spirit in research institutions, project management is mostly considered a necessary but awkward task. We present a working solution tailored to academic projects that requires only a minimum of effort and discipline and results in huge benefits, which will be presented in this article. Cosylab is of academic origin, therefore the spirit, organization and work procedures are very much like in research institutes. In addition, we work on about a dozen projects simultaneously for customers on four continents, which requires a lot of travel and on-site work. Commercially available project management tools are not suited to manage such diversity. We have therefore adopted a set of open source tools, such as Request Tracker, Wiki, Sympa mailing lists, mySQL and maven, implemented some custom additions and integrated the tools into a coherent product to suit our purpose. It enables developers to track their work and communicate effectively; project managers to monitor progress of individual projects; and management to supervise critical parameters of the company at any time. In the article, the experiences gained by using the system are presented. As it has turned out in practice, the product is ideal also for research institutes, as it is demonstrated by its use at the Swiss Light Source (SLS).

INTRODUCTION

Any management system that would be used within a research institute has at least three different kind of users that it must satisfy: **developer**, **project manager** and **The Big Boss™**. As we are used to in control system software, we find a familiar tree tier architecture. Each user of the system has specific demands for it:

- Developer: "I don't want overhead. I want the system to help me manage my tasks, my time and cooperate easily with other members of the team and the project manager."
- Project manager: "I want to monitor the progress of the project as a whole easily and in real time. I want to help the developers manage themselves. I want to spot and predict problems easily. I want The Big Boss™ to be happy and leave me alone as much as possible."
- The Big Boss™: "I am the abstract entity who has defined the budget and scope of the project. I want to get periodic updates on the progress of the project as a whole. I don't want the details, just the big picture."

In this article we will try to argue that the system we have developed within Cosylab meets the demands of all the parties involved.

We have started as a team for control systems in the largest Slovenian research institute Jozef Stefan. After a successful completion of the control system for ANKA [1], we have established a spin-off company Cosylab [2].

In the last two years, the company has grown significantly, from a team of eight to a bunch of twenty full time employees and additional twenty part-time students. As it seems at the moment, the number will increase for a further 10 this year. Also the number of projects in progress has risen from a few at the time to about a dozen at a time. Although this certainly is good news from the business perspective, it adds another level to complexity of the organisation that is not trivial to handle.

Already as a research group, we became aware of the need to have some sort of project management and adequate electronic support for it. We adopted a set of open source tools to help us manage ourselves [3, 4, 5]. Since then, the requirements on the project management were constantly increasing, and as a consequence, our tools were evolving - we began including new tools and wrote additional software, some for customization of existing software, but mostly for integration of different tools into a coherent whole.

All the time we try to keep in mind our vision from the early days - to retain the spirit of a research organisation that has infected us when we worked within a research institute, after all, most of us

have scientific background and mentality. In other words, we never want to become a dull assembly-line kind of a company. It is true that in a company it is easier to see why is it good to keep within the budget and meet deadlines, mostly because the metric that tells you whether you are on the right track is trivially simple - the pay checks. But then again, in research institutes it also does not hurt being on time and on budget.

Therefore, the solution that we have created can be extremely useful also for research institutes. We have indeed proven this to be the case, by introducing our project management system at SLS [6] and to DESY controls group.

COSYMANAGEMENT SYSTEM OVERVIEW

Request Tracker - Define Basic Units of Work

Any project is composed of a number of more-or-less well defined units of work, assigned to developers, usually by the project managers. There is a multitude of tools that manage "work units", many of them freely downloadable from the Internet. We chose the Request Tracker (RT) [6] because of one simple reason – this tool can manage e-mails really well – not only sending but also receiving. RT was picked as our main tool and we adapted all the other ones to it. Every now and then, when experiencing problems with RT's code written in Perl, a question appears why we did not rather write such tool by ourselves, but the final statement remains that RT is really well structured and extremely useful.

RT is (as described in its manual) an "enterprise-grade ticketing system which enables a group of people to intelligently and efficiently manage tasks, issues, and requests submitted by a community of users".

Its main unit is a ticket, which represents a specific task to be done. It has several fields:

- status (new, open, resolved, stalled, dead),
- estimated time,
- time spent (increases whenever a user reports a transaction),
- dates: start date, due date,
- priority,
- keywords (severity of a bug, type of ticket: QA ticket, Master ticket for the project, etc.), and
- links - how does this ticket relate to other tickets (parent - child, "depends on" and "refers to" relationships are supported).

Each ticket can have one or more parents, children or brothers. Setting also dependencies, a clear structure can be made. RT can warn us by email of a creation or modification of some ticket. The best feature of RT is the possibility of managing tickets via e-mail – every project has its e-mail address to which we can send a request for creation, correspondence or comment and set just about every field of the ticket.

RT has easy-to-use search functionality, which allows one to quickly find a ticket. With all its features RT can be used for handling support requests, ordinary tasks and bug tracking. We create about 10.000 tickets per year in Cosylab.

From a Set of Tickets to a Well Organized Project

RT and its web interface, however extremely useful, do not have everything one would wish for when managing and monitoring a project in progress, especially if this project is larger than a few man-weeks of work. The central entity of RT's web interface is a ticket, and just by browsing through different tickets, one can easily get lost by looking at the trees, and not seeing the entire forest.

In addition, a project is much more than a collection of tickets, and the ability to easily manage projects that are run on the basis of a ticketing system is where we have not found an easy open source solution and decided to implement one ourselves.

In order to introduce the smallest possible overhead and use all the data already available in the RT database, we have added project management functionality to RT database itself by introducing a small number of special tickets (defined via existing RT keywords) and by adding additional fields to RT tickets. This has provided the basis for transformation of a ticketing system into a powerful project management tool. The additional information that defines a project is:

- project group (developers and the project managers),

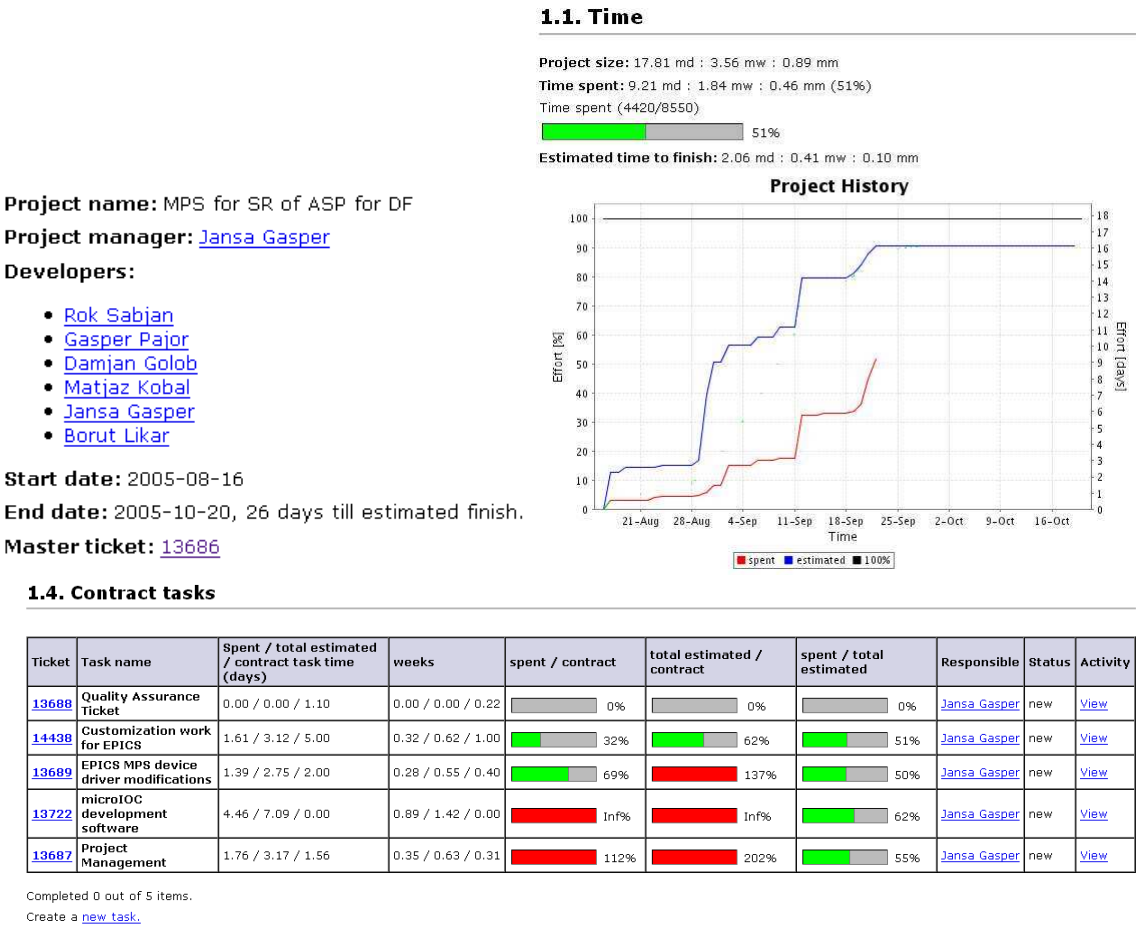
- budget (in terms of work hours or / and money),
- deadlines and milestones,
- communication (team meetings, meetings with the customer or The Big Boss™),
- deliverables (SW and HW releases, etc.),
- quality assurance, and
- project management.

Project Reports

When a project is started, all the project information known at project start is entered into RT database in the form of a few basic tickets for the project:

- *Master ticket*: the root ticket, defines the budget, project deadline, and the project manager,
- *Contract task*: children of master ticket, these are the main tasks of the project.

Project report tool is written in Java and generated entirely from the data in the RT database. Project report is a HTML generated from CosyDoc XML [3] that we use for writing documentation. The XML is transformed by means of a XSLT transformation and ANT scripts into HTML and PDF format. Figure 1 below shows some sections of this report.



Project name: MPS for SR of ASP for DF

Project manager: Jansa Gasper

Developers:

- Rok Sabjan
- Gasper Pajor
- Damjan Golob
- Matjaz Kobal
- Jansa Gasper
- Borut Likar

Start date: 2005-08-16

End date: 2005-10-20, 26 days till estimated finish.

Master ticket: 13686

1.1. Time

Project size: 17.81 md : 3.56 mw : 0.89 mm

Time spent: 9.21 md : 1.84 mw : 0.46 mm (51%)

Time spent (4420/8550)

51%

Estimated time to finish: 2.06 md : 0.41 mw : 0.10 mm

Project History

1.4. Contract tasks

Ticket	Task name	Spent / total estimated / contract task time (days)	weeks	spent / contract	total estimated / contract	spent / total estimated	Responsible	Status	Activity
13688	Quality Assurance Ticket	0.00 / 0.00 / 1.10	0.00 / 0.00 / 0.22	0%	0%	0%	Jansa Gasper	new	View
14438	Customization work for EPICS	1.61 / 3.12 / 5.00	0.32 / 0.62 / 1.00	32%	62%	51%	Jansa Gasper	new	View
13689	EPICS MPS device driver modifications	1.39 / 2.75 / 2.00	0.28 / 0.55 / 0.40	69%	137%	50%	Jansa Gasper	new	View
13722	microIOC development software	4.46 / 7.09 / 0.00	0.89 / 1.42 / 0.00	Inf%	Inf%	62%	Jansa Gasper	new	View
13687	Project Management	1.76 / 3.17 / 1.56	0.35 / 0.63 / 0.31	112%	202%	55%	Jansa Gasper	new	View

Completed 0 out of 5 items.

Create a new task.

Figure 1: Some of the sections generated by the report generator tool. Basic project data, time progress and status of individual tasks are shown. Using this report, one is able to easily determine the status the project and the progress of each individual contract task.

Figure 1 shows the basic information on the project progress, relevant both to the project manager and The Big Boss™. But the report does not stop here, it also allows the project manager to supervise the project proactively, as displayed on the figure 2.

2.1. Errors

Ticket	Owner	Status	Subject	Description	Quick fix
13993	Aljaz Podborsek	resolved	prepare test microIOC for ASP storage ring	Ticket has estimated time set to 60 min, however it was resolved without any efford entered. Either set estimated time to zero or enter missing time.	Set estimated time to zero.
14753	Jansa Gasper	resolved	Write documentation	Ticket has estimated time set to 180 min, however it was resolved without any efford entered. Either set estimated time to zero or enter missing time.	Set estimated time to zero.
13686	Jansa Gasper	open	**MPS for SR of ASP for DF**	Sum of contract tasks for child tickets (4640 min) does not equal master ticket contract time (8550 min).	

2.2. Warnings

Ticket	Owner	Status	Subject	Description	Quick fix
13685	Jansa Gasper	dead	test	Ticket in project queue has no parent	
14283	Jansa Gasper	resolved	add new features for storage ring MPS	Ticket has estimated time set to 960 min, however it was resolved in only 380 min.	
13722	Jansa Gasper	new	**microIOC development software**	Contract ticket has zero contract time.	
13738	Borut Likar	resolved	Modfiy DF-ASPBooster's maven build	Ticket is moderately over time: worked: 845, estimated: 480.	
14242	Borut Likar	resolved	This is for debian package creating stuff	Ticket has estimated time set to 1440 min, however it was resolved in only 330 min.	

Figure 2: Report also spots errors in project organisation and sources of potential trouble for a projects: tickets resolved too late and way over (or under) estimated budget, etc.

Gantt Chart

For complex projects, one would also need a tool for oversight how the task depend one on another, how much work is assigned to developers over time (resource utilisation), etc., in other words, to be able to take a "helicopter view" on the project. The functionality we look for is exactly the one of a Gantt chart.

There are plenty of commercial and open source tools that provide such functionality. We sought for an open source tool that would easily be integrated with RT. We have chosen Gantt Project (GP), written in Java. We have implemented a two way integration with RT by means of *Import* and *Export* features of GP. On import, one must enter a parent ticket number, and subsequently, this ticket and all its child tickets will be imported into GP. The mapping between the RT ticket and GP task is simple since they are conceptually the same.

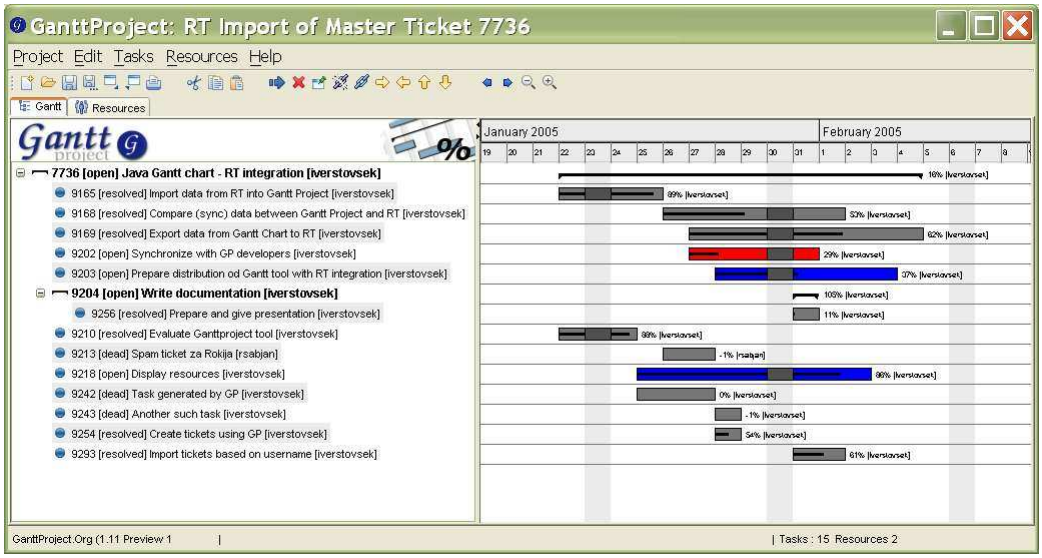


Figure 3: Gantt Project with tickets, imported from the Request Tracker.

After import, it is possible to see the progress of the project on a Gantt chart, where RT ticket status is colour coded on the Gantt chart. Another useful feature is the actual completion of the task in percents (calculated automatically from the data entered into RT). Here our solution adds value - there are probably not many project managers that would actually take care that accurate completion percentages are displayed on the Gantt chart.

Furthermore, it is possible to set starts and due dates of tickets, change ticket status, write comments or perform any other manipulation allowed by RT. It is even possible to make new RT tickets!

When finished with editing, a synchronisation with RT can be performed. Here, all the changes are

enumerated for preview and it is possible to select a subset to commit back to the RT database. Synchronisation is performed via RT's mail interface - the program sends emails with appropriate commands to RT.

Integration of Gantt Project and RT brought something that most of the commercial project management tools lack - the ability to look on a project on different levels: from the level of a single ticket, to the level of several projects, where one can monitor the workloads on engineers also into the future.

RT Analyser and OLAP Cubes

RT Analyser was developed entirely in-house. It offers on-line analytical processing (OLAP) capabilities over the RT database. The data is filled into the so-called OLAP cubes, multi dimensional entities where its dimensions can be whatever we find it to be useful, such as effort (actual and estimated) over projects, resources (engineers) and time.

Through RT Analyser, it is possible to extract and summarize the information from the OLAP cubes. The user requests results using simple drag-and-drop operations, and RT Analyser swiftly responds with graphs and tables. By looking at different "cross sections" of this cube, it is possible to gather many statistically useful data. For this purpose, also commercially available tools for viewing OLAP data can be used, such as the Microsoft Excel.

For example, when looking at the organisation as a whole, one question to ask is how much time spent went for production projects (these are the project that we get paid for), research ("basic" research, development of new products) and overhead (administration, sales and marketing, reading emails, drinking coffee, etc.).

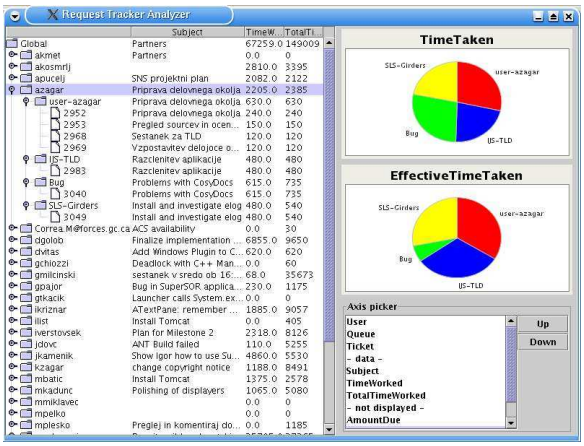


Figure 3: Screenshot of the Request Tracker Analyzer, showing distribution of time the given user (azagar) had spent on his projects (IJS-Girders, bug fixing, IJS-TLD and other tasks).

EXPERIENCE AND FURTHER PLANS

In Cosylab we are using the system for over 3.5 years, it was evolving along with our project management procedures. The system is now universally accepted in Cosylab, we have around 30 production projects (for external customers) and around 15 internal (research projects) running in parallel. We have also categorized all of the ongoing activities as projects - such as administration, IT, marketing, education and HRM.

The system has matured and can be regarded as a stand alone product that can also be used at facilities by other research groups. For example, the system will be used in DESY controls group for the project of control system upgrade of Petra III into a synchrotron.

Tracking Developer's Time and Progress

In Cosylab, the system is used to track time in minutes. This means that after finishing working on a given ticket, the developer must write a reply to RT on what he or she did and how long did it take. In order to reduce overhead for this activity, we have implemented a stopwatch application: it is an icon in the system tray, when clicked upon, it displays a list of all your tickets from which you select one. When switching a task, you switch it also in the stopwatch. At the end of the day, you just need

to commit all the recorded times to the db and write comments where necessary.

In DESY, the goal of the system is not to track the time spent of individual developers, but to track their *progress* on the tasks. It was not necessary to modify the system a lot to accommodate for this functionality - instead of entering time, developers enter their progress in relative units, where 100% is a completed task.

Tracking Product Releases

Specific to software development is the notion of developing in iterations, to have subsequent releases (versions) of a certain product. To facilitate this kind of development, we have implemented a simple tool that uses the data from the RT database to track the progress of a certain product release: lists open and closed issues with comments and generates complete release notes in HTML.

CONCLUSION

Every project management system is a trade off between flexibility and ease of use on one hand and order and strictness (that inevitably introduces some bureaucracy) on the other.

CosyManagement system, described in this article is (as its name suggests) orientated more into the direction of flexibility, but it still gives enough order that it is possible to spot possible problems early, which is one of the goals of project management.

Each organisation must choose its place on this balance, based the nature of its work. Requirements of Cosylab and research institutes are very similar - we all like to do research, but we must still meet deadlines. Therefore we believe that our solution is ideal also for academic projects.

It requires only a minimum of effort and discipline and results in huge benefits, presented in this article, hopefully making the developer, project manager and The Big Boss™ happy.

ACKNOWLEDGEMENTS

To Reinhard Bacher from DESY, for funding the adaptation of CosyManagement software to DESY and for several key feature requests, such as the progress graph and introduction of relative units.

REFERENCES

[1] I. Verstovšek et al., ANKA Control System Takes Control, PCaPAC 2000, Hamburg, Germany.
[2] Cosylab, http://www.cosylab.com
[3] I. Verstovšek et al., Cosylab Management Sytem, PCaPAC 2005, Hayama, Japan.
[4] G. Milčinski et al., e-Management and Quality Assurance, ICALEPCS 2003, Gyeongju, Korea.
[5] G. Milčinski et al., Developing a Control System from a Divan Bed, PCaPAC 2002, Frascatti, Italy.
[5] Swiss Light Source Controls Group, http://www.sls.psi.ch/controls/controls-home.html
[6] Request Tracker, http://www.bestpractical.com/rt/
[7] Gantt Project, http://ganttproject.sourceforge.net/