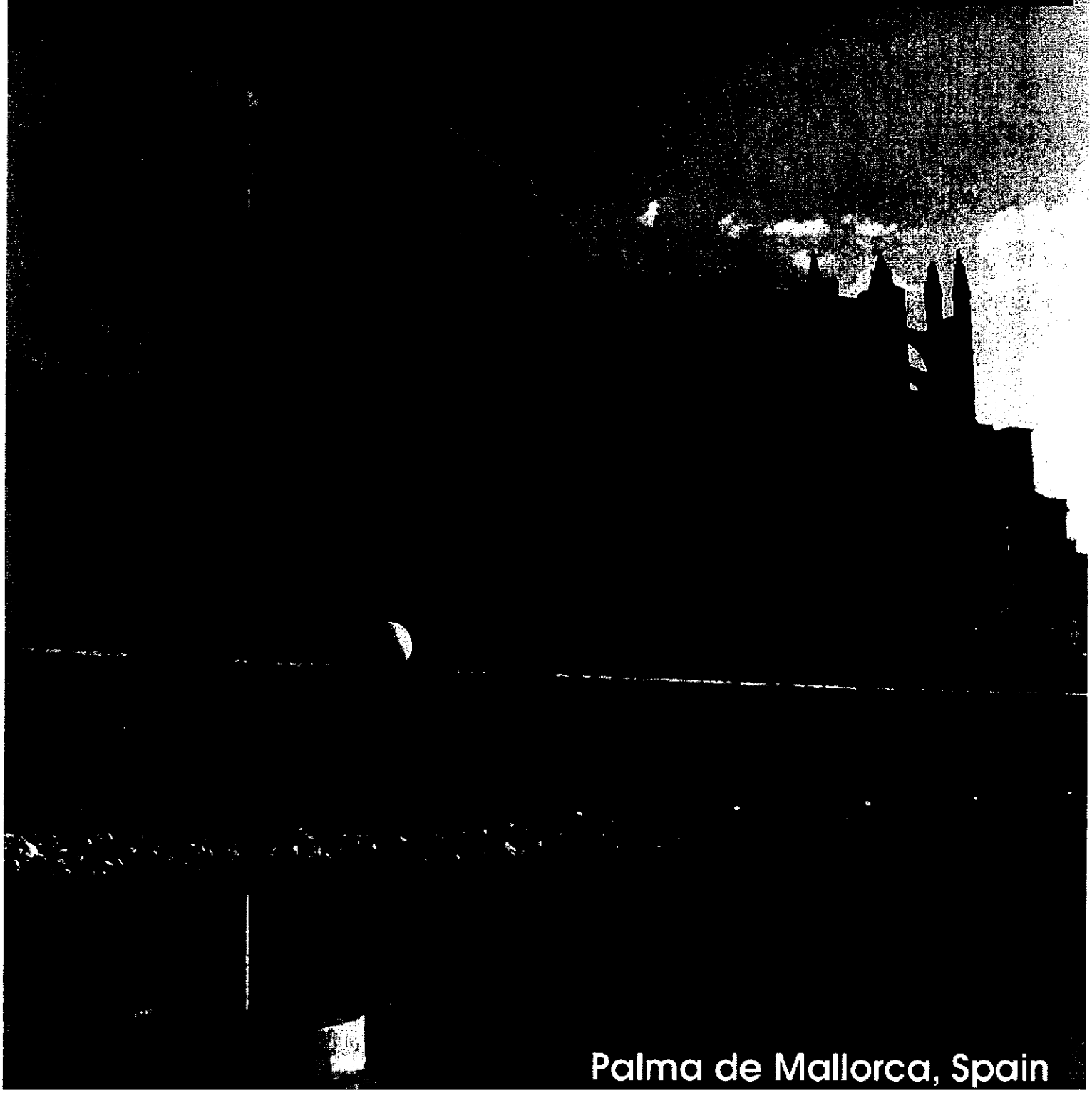
Proceedings of the

IWSM-Mensura 2007

International Conference on Software Process and Product Measurement



Palma de Mallorca, Spain

Editors

Alain Abran
Department of Software and IT Engineering
École de technologie supérieure
1100, Notre-Dame Street West
Montréal, Québec
Canada H3C 1K3
E-mail: Alain.Abran@etsmtl.ca

Reiner Dumke
Otto von Guericke University of Magdeburg
Postfach 4120
39016 Magdeburg
Germany
E-mail: dumke@ivs.cs.uni-magdeburg.de

Antonia Mas
Research Group for Software Process Improvement
Department of Computer Sciences and Mathematics
University of the Balearic Islands
Escola Politècnica Superior
Edifici A. Turmeda. Campus universitari
Cra. de Valldemossa, Km. 7.5
07122 Palma de Mallorca (Illes Balears)
Spain
E-mail: antonia.mas@uib.es

Edition: Edicions UIB
Campus de la Universitat de les Illes Balears
Cra. de Valldemossa, km 7.5
07122 – Palma (Illes Balears)
http://edicions.uib.es

ISBN: 978-84-8384-020-7
Dipòsit Legal: PM 2625-2007

Alain Abran
Reiner Dumke
Antonia Mas (Eds.)

PROCEEDINGS OF THE IWSM - MENSURA 2007

INTERNATIONAL WORKSHOP ON SOFTWARE MEASUREMENT - INTERNATIONAL CONFERENCE ON SOFTWARE PROCESS AND PRODUCT MEASUREMENT

PALMA DE MALLORCA – SPAIN
NOVEMBER, 5 – 8, 2007

Preface

This volume contents the papers presented at the IWSM-Mensura 2007 conference held in Palma de Mallorca, Spain in November 2007.

IWSM-Mensura 2007 is a continuation of two international conferences IWSM (International Workshop in Software Measurement) and Mensura (International Conference on Software Process and Product Measurement). They are conducted jointly for the first time.

The objective of the IWSM-Mensura 2007 conference is to bring to light the most recent findings and results in the area of software measurement and to stimulate discussion between researchers and professionals.

We would like to thank the many people who have brought this International Conference into being. On the one hand, we want to express our thanks to the Organizing Committee members who have made possible that all conference events, both technical and social, have been performed successfully.

On the other hand, we want to appreciate the work done by all the people that have participated in the organization of sessions: special sessions, workshops and industrial track. Finally, we also want to thank the Program Committee Members for their hard work in reviewing both the abstracts and the final papers.

The organizers would also like to thank the University of the Balearic Islands, the Research Group for Software Process Improvement and the Government of the Balearic Islands for supporting the conference.

November 2007

Reiner Dumke and Miguel Toro
Alain Abran and Antonia Mas
General and Program Chairs

Conference Organization

General Chairs

Reiner Dumke, *University of Magdeburg, Germany*
Miguel Toro, *University of Seville, Spain*

Program Committee Chairs

Antonia Mas, *University of the Balearic Islands, Spain*
Alain Abran, *University of Québec, Canada*

Proceedings Chairs

Daniel Rodríguez, *University of Alcalá, Spain*
Maya Daneva, *Univerity of Twenty, The Netherlands*

Program Committee Members

Moteoi Azuma, *Japan*
Manfred Bundschuh, *DASMA, Germany*
David Card, *USA*
François Coallier, *ÉTS, Canada*
Juan J. Cuadrado-Gallego, *University of Alcala, Spain*
Jean-Marc Desharnais, *ÉTS, Canada*
Ton Dekkers, *Shell Information Technology, Netherlands*
Javier Dolado, *University of the Basque Country, Spain*
Christof Ebert, *Vector Consulting, Germany*
Khaled El Emam, *University of Ottawa, Canada*
Juan Garbajosa, *U. Politécnica, Madrid, Spain*
Felix García, *University of Castilla-La Mancha, Spain*
Elena García-Barriocanal, *University of Alcalá, Spain*

George Ghinea, *Brunel Univesity, UK*

Naji Habra, *FUNDP, Namur, Belgium*

Nadine Hanebutte, *University of Idaho, USA*

Mark Harman, *Kings College, UK*

Rachel Harrison, *UK*

Ali Idri, *INSIAS, Morocco*

Natalia Juristo, *U. Politécnica, Madrid, Spain*

Adel Khelifi, *U. Al Hosn, URA*

Pedro Lara, *University Europea, Madrid, Spain*

Miguel Lopez, *Belgium*

Mathias Lother, *Germany*

Hakim Lounis, *Canada*

Fernando Machado, *Catholic University of Uruguay, Uruguay*

John McGarry, *USA*

Roberto Meli, *DPO, Italy*

Pam Morris, *Australia*

John C. Munson, *USA*

Olga Ormandjieva, *Concordia University, Canada*

Oscar Pastor, *Technical University of Valence, Spain*

Mario Piattini, *University of Castilla-La Mancha, Spain*

Tony Rollo, *UK*

Salvador Sánchez, *University of Alcalá, Spain*

Miguel-Ángel Sicilia, *University of Alcalá, Spain*

Manoranjan Satpathy, *General Motors, India*

Andreas Schmietendorf, *FH Harz Wernigerode, Germany*

Manuel Serrano, *University of Castilla-La Mancha, Spain*

Marcio Silveira, *EDS, Brazil*

Harry Sneed, *SES Munich/Budapest, Hungary*

Andreas Schmientendorf, *FHW Berlin, Germany*

Esperança Amengual, *University of the Balearic Islands, Spain*

Stanimir Stojanov, *Bulgaria*

Hannu Toivonen, *Nokia, Finland*

Claes Wohlin, *Blekinge Institute of Technology, Sweden*
David Zubrow, *USA*
Horst Zuse, *TU Berlin, Germany*
Marcus Ciolkowski, *Fraunhofer IESE, Germany*
Javier Aroba, *University of Huelva, Spain*
Dietmar Pfalh, *University of Calgary, Canada*
Andreas Jedlitschka, *Fraunhofer IESE, Germany*

Organizing Committee

Antonia Mas, *University of the Balearic Islands, Spain*
Esperança Amengual, *University of the Balearic Islands, Spain*
Ana Belén Petro Balaguer, *University of the Balearic Islands, Spain*
Carlos Guerrero Tomé, *University of the Balearic Islands, Spain*
Antoni Lluís Mesquida Calafat, *University of the Balearic Islands, Spain*

Special Sessions/Workshops Chairs

Mercedes Ruíz, *University of Cádiz, Spain*
Luis Fernández, *University Europea de Madrid, Spain*

Industrial Track Chairs

Luigi Buglione, *AtosOrigin, Italy*

Industrial Track Program Committee

Alfonso Romo, *El Corte Inglés, Spain*
Victoria de MENA, *El Corte Inglés, Spain*
Carmen Velasco, *El Corte Inglés, Spain*
Miguel Lopez, *Algorismi, Belgium*
Luca Santillo, *Independent consultant, Italy*
Sylvie Trudel , *CRIM, Canada*

Index

Implementing Software Measurement Programs in Non mature Small Settings

María Díaz-Ley[1], Félix García[2], Mario Piattini[2]

[1] Sistemas Técnicos de Loterías del Estado (STL)
Gaming Systems Development Department 28234 Madrid, Spain
Maria.diaz@stl.es

[2] ALARCOS Research Group
Information Systems and Technologies Department
INDRA-UCLM Research and Development Institute
University of Castilla-La Mancha,
13071 Ciudad Real, Spain
{Felix.Garcia, Mario.Piattini}@uclm.es

Abstract. Although measurement has been successfully applied in various areas, it has proved to be a complex and difficult undertaking in the field of software and especially in the context of small and medium enterprises (SMEs). Measurement programs in SMEs with a low maturity level should be tailored to their limitations (limited resources, experts, etc.) in order to carry out the measurement initiatives efficiently. In this paper we report the method, principles and practices followed in our experience for defining and implementing a measurement program in the development department of Sistemas Técnicos de Loterías del Estado (STL). We also show the characteristics of this company which guided our approach towards tackling the problem, and the resulting software measurement program. As a result of the application of certain practices in the company, some significant benefits were obtained, which could be replicated in similar environments.

Keywords: Software measurement program, small and medium settings, case study.

1 Introduction

Although measurement is applied in various areas, it has proved to be a complex and difficult undertaking in the field of software, especially within the context of small and medium enterprises [1]. Small and medium enterprises have some common characteristics which eventually become obstacles in establishing software measurement initiatives Some of these are: limited resources and training, poor software measurement knowledge, restricted cash flow, a restricted mentality as regards software measurement, etc. [2-4].

This paper aims to report our experience in setting up a measurement program in

the software development department of a medium-sized company with a low software measurement maturity level. We show the methodological approach and implementation strategy chosen which fits the characteristics of the company, and we briefly present the resulting measurement program. Finally it is exposed the benefits of using these practices in similar settings.

The paper is organized as follows: Section 2 sets this work in context by showing some measurement program implementation experiences. Section 3 specifies the characteristics of the company, the measurement program definition framework chosen, the organizational approach, the goals of the measurement program, the resulting measurement program and some practices for implementing the measurement program are also expounded. Section 5 shows the conclusions and lessons learned from the experience and Section 6 shows further research.

2 Related Work

In this section we provide an overview of the implementation of measurement programs in software technological companies and we address some studies which identifies good practices for implementing software measurement programs successfully.

Daskalantonakis presented a company-wide software measurement initiative which was implemented to fulfill organizational quality policy [5]. In this work some cultural issues were identified and an organizational approach through which to carry out measurement programs was defined. Some benefits were an improvement in terms of defect density and customer satisfaction. Kettelerij studied the possibility of establishing measurement at Daniro J-Technologies in the context of software development projects and proposed a measurement program definition [6]. In [7] Kilpi shows how Nokia organizationally carries out its measurement activities.

With regard to the implementation of measurement programs in small and medium settings, in [8] MacDonell et al. presented a measurement program whose purpose was to define a framework to determine the metrics that should be defined in an organization that develops multimedia systems. Lavazza et al. presented a measurement program experience which took place in the Banca Conboto where GQM was used and adapted due to the operational, budget and time constrains. The resulting measurement program gave valuable information about the quality of the maintenance process [9].

These experiences gave us a valuable insight into the matter but there is very little information about experiences related to small and medium settings in defining and implementing measurement programs.

In the frame of measurement programs good practices, Gopal et al. [10] identified and proved some success factors by analyzing its effects on the measurement programs success. The success of a measurement program was measured using two variables: use of metrics in decision-making and improved organizational performance. The success factors selected were divided in two sets: organizational and technical factors. Daskalantonakis also stated some good practices form its experience in Motorola [5, 11] and Fenton et Hall [12] identified from the experience fifteen success factors for implementing software measurement programs. However none of these

studies show good practices to follow especially when the measurement program is implemented in small and medium software companies and its characteristics are typical of these environments: low measurement maturity level, poor measurement knowledge, measurement not integrated in the culture, limited resources and budget, etc.

3 Development of the Measurement Programs in STL.

In this section we describe the company in which the case study was conducted, the methodological and organizational approach set out, the principles followed; the process improvement goals supported by the measurement program and a summary of the resulting measurement program.

3.1 Description of the Unit in Terms of Measurement and Measurement Program Constraints

Sistemas Técnicos de Loterías del Estado (STL) is a company which was created by the Spanish government and which provides the operations and IT development services for the national lottery. Software measurement initiatives have been encouraged for many years by the software development and maintenance department in this company, which is formed of 39 people. Unfortunately the measurement process which was defined was not accurate enough and had not been properly established throughout the department. Some of the outstanding problems were the scarce amount of resources available for this task. However the need to measure continued and the need of defining accurate measurement programs re-emerged

The development and maintenance department is in charge of developing and maintaining the bet on-line systems including the software of the related terminals and payment channels; and the invoicing and informative systems. They support 33 products of which 21 are critical and 18 of them are on-line systems. Most of the code is written in FORTRAN, C, C++, some Java and Pl/SQL. The core critical product amounts to 2000 KLOC.

The development projects where normally carried out by less than 12 people. It usually takes five or six months long and rarely above 15 months.

Some other characteristics of the company related to measurement are as follows:

- **C1**: The resources were limited and therefore we could not spend too much effort on defining and implementing the measurement program.
- **C2:** Some project managers were reluctant to use the measurement initiative
- **C3:** Measurement was not established in the company culture.
- **C4**: Measurement knowledge was quite limited throughout the company.
- **C5**: The software measures collected were few, there was no established measurement process and therefore the measurement maturity was quite low.

3.2 Measurement Program Definition Framework

The methodology proposed by MIS-PyME model was used for the software measurement program definition. We shall now give a brief introduction to this methodology framework. A detailed description of MIS-PyME can be found in [13].

The main characteristic of MIS-PyME, among the outstanding software measurement models, is that it fully covers the requirements of a model suited to small and medium settings with a low measurement maturity level. MIS-PyME framework is formed of three main modules: MIS-PyME work products, MIS-PyME methodology and the MIS-PyME measurement maturity model. MIS-PyME Methodology is based on GQ(I)M [14, 15], but it is designed to define basic indicators which are commonly used and required in most small and medium software development settings. These indicators are adapted to the measurement maturity of the setting. Like GQ(I)M, it is a top-down methodology since it develops the measurement program with the goal in mind but restricts the domain of those goals solely to software process improvement and may be conditioned by the MIS-PyME measurement goals table and the indicator templates provided. This methodology also makes use of a database of measurement program definitions related to software process improvement. MIS-PyME methodology also stresses and helps the integration of the measurement program in the software processes.

In terms of organizational approach, the principles supported by MIS-PyME are the following:

1. The “Reuse and project-specific tailoring” principle, stated by Basili and Rombach [16, 17]. This principle indicates that measurement planning should reuse models and metrics which have been defined for the whole organization. However these models might be tailored to the project or product specific characteristics. In our approach we define measurement programs which will be valid and useful in almost all products or projects, by doing so it makes measurement programs establishment easier in the development unit and make possible the development process control.

2. To implement measurement programs which are adapted to the measurement maturity of the setting and with the sole purpose of giving support to the software process improvement initiatives as are also followed in [5]. If the measurement process is still not well integrated in the organization culture and it is not mature enough, it is costly to establish measurement programs. STL could not afford to develop measurement programs which come from just any business goal and was focused on supporting process improvement goals.

3.3 Settings

The roles determined to carry out the measurement program and suggested by MIS-PyME model were as follows: the first was that of the measurement analyst who had some knowledge about software measurement but she was not too experienced. The usual work of this person was consulting, coordinating projects, defining requirements and testing. The second role was simply played by the top manager (the director of the development department) who supports the measurement program initiative. He had an in-depth knowledge of the software processes and process improvement needs. The third ones were the reviewers. This group was formed of all the project managers (most of them are also the sub-department managers) and some key developers.

This approach was designed in order not to disturb project managers in their usual tasks. The top manager knows the main measurement needs and he is capable of determining them. The measurement analyst will work only with the top manager during the entire definition process phase. At the "verifying measurement process" phase the measurement analyst worked with the reviewers.

3.4 Software Measurement Program in STL

The highest priority software Process Improvement Goals (PIG) were the following:

- **PIG 1:** Improving project monitoring and control. We particularly wished to improve the monitoring of the project's progress in comparison to the plan and to understand and manage any deviations from the plan at the project's closure.
- **PIG 2:** Improving process and product quality. This goal focused on understanding, monitoring and evaluating the development service and product quality exploited.

Measurement Program Resulted: For the first goal (PIG1, see section 4.1) two sub-goals through which to improve the software process are identified: PIG 1.1- Improving project progress monitoring in comparison to the plan; PIG 1.2 - Understanding and managing any deviations from the plan at the project's closure. For the first sub-goal we emphasized the indicators as follows: the first indicator which gives the information about the effort spent against what it was estimated. The next indicator shows the progress of the coding phase by showing the number of requirements coded against the total, and the planned schedule. Another indicator shows the progress of the verification phase by showing the number of open incidences and its severity and the planned schedule. For the acceptance phase progress, an indicator shows the number of requirements verified with regard to the total of requirements, the number of open incidences and its severity, the planned schedule, and the defect density. For the second sub-goal four indicators were defined in order to measure the deviation at the protect closure: deviation regarding the effort, the size of the software, the duration of the project and the total developing cost.

Besides, for each of the indicators related to the PIG 1.2 sub-goal, an indicator was defined in order to analyze the development process. All the projects were thus globally analyzed during a period of time.

From PIG 2, two sub-goals were identified: PIG.2.1 - understanding, monitoring and evaluating the development service provided and PIG.2.2 –understanding and monitoring the quality of the product exploited.

For PIG.2.1 a first level indicator called IND-PRJ-QUALITYDEV was defined. This indicator was simply formed from two other indicators, IND-PRJ-FIABIMPL and IND-PRJ-INEXACDURACION. IND-PRJ-FIABIMPL aimed to evaluate the reliability of the software developed and released under a project. IND-PRJ-INEXACDURACION aimed to evaluate the deviation in time as regards what was planned. There is also an indicator which globally measures the development process as regards the development service quality. This indicator is called IND-PROC-QUALITYDEV.

PIG.2.2 was represented by three indicators which monitor the quality of the products provided by our clients: IND-PROD-FIABCRIT, IND-PROD-FIABSOPORT, IND-PROD-FIABINF. These indicators show the density of failures in production for each product during the period analyzed. Each indicator includes the products related to its product classification: critical, support, informative.

In summary, the resulting measurement program defined 31 indicators, 29 measures, 6 estimations and 9 concept criteria. This measurement program was designed to be used by the whole department. The measurement program was created in two phases which lasted almost three months. PIG.1.1 was tackled in a second phase after PIG.1.2, PIG.2.1 and PIG.2.2 were already implemented and in use.

Software Measurement Process: The measurement program mentioned above formed the initial measurement process defined for the development department in STL. Three different sub-processes were identified: the project management measurement sub-process, the process management measurement sub-process and the product management measurement sub-process. As far as the integration of the measurement process is concerned, the analysis activities related to the indicators were included in the development process and its related templates (e.g. close of project report, project progress monitoring report, etc.).

4.5 Software Measurement Program Implementation

In this section we briefly explain some important issues related to the implementation of the measurement program and we finally show an example.

- **Instrumentation:** It was intended to avoid data that could not be obtained from the tools already in use. The software measurement program collects the information from five development and configuration management tools: the Microsoft Project Manager, ActiTime (a free tool with which to register the effort dedicated to each task), Remedy (an incident management tool) and IRQA (a requirement management tool). Only Remedy needed to be tailored in order to automatically launch the queries to the database and to create reports containing the required information. We briefly studied some software measurement tools such as MetricFlame, MetricCenter, ProjectConsole, etc. But we preferred to use simple than complicated tools since complicated tools may make the understanding of the basic measurement process steps more difficult; and, after weighing up the difference between the effort needed to study a complex tool, adapt it and train people,

and the benefits provided, we chose to develop three simple Spread Excel Sheets to give support to each of the sub-processes.

- **Verification:** The measurement program was first reviewed in two sessions. In the first session the measurement analyst gave an overview of the measurement program defined and suggestions were only received in the second session after the measurement program was analyzed. Afterwards the measurement program was tested in the department. The results obtained from the measurement programs during the verification phase did not become known outside the reviewers group. For the indicators related to the products, an analysis was done based on the four months product activity. For the indicators related to PIG 1.2 and 2.1 an analysis was done using the projects which were finished between the previous six moths. The mentioned indicators were implemented quite fast but PIG1.1 related to the monitoring of the project and therefore its verification took the duration of it. This part of the measurement program is still at the verification phase.
- **Final results and Example**: The measurement program defined in STL finally overcame the goals stated in section 4 (pending the verification phase of PIG 1.1 sub goal). In Figure 2 an example of an indicator (IND-PROC-QUALITYDEV) after its first analysis is shown. This indicator not only evaluates the development service quality, but additionally this indicator made us to ascertain whether the on-time release of the software product had a negative impact on software reliability.

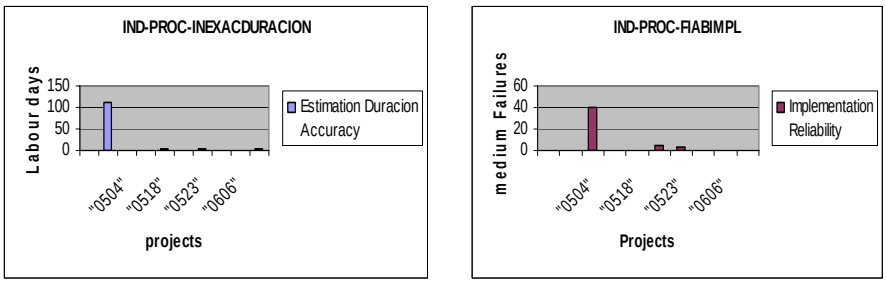


Fig 2. Service quality indicator (IND-PROC-CALIDADSRV) which contains two input indicators: IND-PROC-INEXACDURACION and IND-PROC-FIABIMPL.

5 Conclusions and Lessons Learned

In this paper we have reported an experience in defining and implementing a measurement program in the development department of Sistemas Técnicos de Loterías del Estado (STL) whose aim was to consolidate a useful and simple software measurement process.

The interest of this paper is to show the strategy followed to develop and implement the measurement program, taking into account the characteristics of the company and the good practices identified which could be applied in similar environments.

The characteristics of the company were as follows:

- People involved in the measurement program, including the measurement analyst are from inside the company and not too expertise in the field.

- Poor measurement culture in the company, poor knowledge and therefore poor measurement maturity.
- Some project managers and developers reluctant to use measurement.
- The measurement program should be established in a small or medium software development company or unit with less than 50 people approximately.
- People involved in project, in which the measurement program is established, may be normally less than 12 people.
- The duration of the project should not exceed 15 months and should normally take 5 or 6 months.

Regarding the above characteristics we suggest to follow the practices as follows:

1. The definition of Measurement Programs should not focus on defining measurement programs for certain projects or products, since it will be costly, difficult to handle and of little worth for future developments. The organization should focus on reusing its existing measurement models.
2. Measurement programs should focus on supporting software process improvement goals instead of on business goals. Low measurement maturity level settings cannot afford measurement programs from just any business goal if the aim is to define successful measurement programs effortlessly, accurately and consistently.
3. Measurement Programs definition should be adapted to the measurement maturity of the company. As an example a software measurement low maturity organization cannot expect empirical prediction results. We used MIS-PyME indicator templates to guide us in this issue.
4. Using common models for software project measurement related may be of use in order to detect the measurement goals which may support software process improvement goals. We made use of the measurement goal table provided by MIS-PyME and we found them useful.
5. With regard to the organizational approach (supported by MIS-PyME) and the roles involved, the benefits detected by our experience were as follows:

- The extra work that the measurement program definition implies for the project managers is reduced by using this approach. The aim is to seek common useful measurement goals that support software process improvement. The top manager is able to support the measurement analyst in defining the first approach of the measurement program.
- Project managers make more objective suggestions and effective modifications about the measurement program definition if they review the first approach. As some project managers are reluctant to use these initiatives, the first approach of the measurement program can show them its usefulness and motivate them to review it.

6. We discovered that the way in which the revisions were made by means of two preliminary verification sessions, and the pilot test, was quite useful: By doing so people give better suggestions, analyze better the problem and the usefulness of the measurement program, get more involved in the measurement program, and they easier agree with it.
7. The means of documenting software measurement programs and integrating them in the rest of the software process is essential for the acceptance and use of the measurement process. Our measurement process is documented in a way which

clearly answers the questions: “what aspects of the projects, products, resources and process are measured?”, “what for?” and “what do I have to measure at each moment of the software development and maintenance process?” Moreover, measurement activities are included in the software development and maintenance model, and in the output report templates involved. Our measurement process was documented in a “.pdf” format but we will, however, modify it to make it accessible via WEB as this is the format of the software development and maintenance model.

8. Excel Spread Sheets or familiar databases are recommended for this type of settings. First because before having powerful tools it is better to understand the process and to control the essential activities. Furthermore, the benefits that the tool provides may not make up the cost of evaluating the tool and training people. Once the company is mature enough, other more powerful tools can be acquired.
9. We also recommend taking advantage of the data provided by already existing development tools and attempting not to collect ambiguous or difficult (as regards data collection) data.

Some of the practices suggested above have been already identified as success factors for implementing measurement programs by some authors: the first recommendation was indicated by Basili and Rombach [17], the second and seventh were identified by Daskalantonakis [5, 11] and the ninth was specified by Fenton and Hall [12]. We agree with these practices too and we propose others that made the definition and implementation of our measurement program easier.

Since the information related to measurement programs definition and implementation in small and medium settings with similar characteristics is scarce and it covers a wide industry sector, our work is worthy of consideration.

However most of the practices suggested in this study may not be valid when the measurement program is not aimed to be implemented in a software development small and medium company or single unit, when the measurement maturity level is high and the software measurement is integrated in its culture or when the budget and resources assigned are rather good.

6 Further Research

Our future research will focus upon guiding small and medium settings towards increasing their maturity with regard to software measurement and also upon observing the benefits of measurement in the development process and its consequences in businesses in this kind of environments. By doing so, we shall continue with the refinement of MIS-PyME methodology framework.

Acknowledgment. We would like to thank the staff of Sistemas Técnicos de Loterías del Estado (STL) for their collaboration. This research has been sponsored by the COMPETISOFT (CYTED, 506AC0287) and ESFINGE (Dirección General de Investigación del Ministerio de Educación y Ciencia, TIN2006-15175-C05-05) projects.

References

1. Gresse, C., T. Punter, and A. Anacleto. *Software measurement for small and medium enterprises*. in *7th International Conference on Empirical Assessment in Software Engineering (EASE)*. 2003. Keele, UK.
2. Briand, L.C., C.M. Differding, and H.D. Rombach, *Practical Guidelines for Measurement-Based Process Improvement.* Software Process - Improvement and Practice, Dec. 1996. 2(4): p. 253-280.
3. Mondragon, O.A. *Addressing Infrastructure Issues in Very Small Settings*. in *Proceedings of the First International Research Workshop for Process Improvement in Small Settings*. 2005.
4. Emam, K.E. *A Multi-Method Evaluation of the Practices of Small Software Projects*. in *Proceedings of the First International Research Workshop for Process Improvement in Small Settings*. 2005.
5. Daskalantonakis, M.K., *A Practical View of Software Measurement and Implementation Experiences Within Motorola.* IEEE Transactions on Software Engineering, 1992. 18(11): p. 998-1010.
6. Kettelerij, R., *DESIGNING A MEASUREMENT PROGRAMME FOR SOFTWARE DEVELOPMENT PROJECTS*, in *Daniro System Integration and Development B.V.* May, 2006, University of Amsterdam: Amsterdam.
7. Kilpi, T., *Implementing Software Metrics Program at Nokia.* IEEE software, 2001. 18(6): p. 72-76.
8. MacDonell, S.G. and T. Fletcher. *Metric Selection for Effort Assessment in Multimedia Systems Development*. in *Fifth International Sotware Metrics Symposium*. 1998. Bethesda, MD, USA.
9. Lavazza, L. and M. Mauri. *Software Process Measurement in Real World: Dealing with Ioerating Constraints*. in *Workshop on Software Process Simulation and Modeling*. 2006.
10. Gopal, A., et al., *Measurement Programs in Software Development: Determinants of Success.* IEEE Transactions on Software Engineering, 2002. **28**(9): p. 863-875.
11. Daskalantonakis, M.K., R.H. Yacobellis, and V.R. Basili, *A Method for Assessing Software Measurement Technology.* Quality Engineering, 1990: p. 27-40.
12. Hall, T. and N. Fenton, *Implementing Effective Software Metrics Programs.* IEEE software, 1997. 14(2): p. 55-65.
13. Diaz-Ley, M., F. García, and M. Piattini. *Software Measurement Programs in SMEs - Defining Software Indicators: A methodological framework.* in *PROFES'07*. 2007.
14. Park, R.E., W.B. Goethert, and W.A. Florac, *Goal-Driven Software Measurement-A Guidebook*. 1996: Carnegie Mellon University Pittsburgh: Software Engineering Institute.
15. Goethert, W. and J. Siviy, *Applications of the Indicator Template for Measurement and Analysis*, in *Software Engineering Measurement and Analysis Initiative*. September 2004.
16. Basili, V.R. and D.H. Rombach, *The Tame Project: Towards Improvement-Oriented Software Environments.* IEEE Transactions of Software Engineering, June, 1988. 14(6): p. 758-773.
17. Basili, V.R. and H.D. Rombach, *Support for comprehensive reuse.* IEEE Software Engineering Journal, 1988: p. 758-773.