JOURNAL OF SOFTWARE MAINTENANCE AND EVOLUTION: RESEARCH AND PRACTICE
J. Softw. Maint. Evol.: Res. Pract. (2010)
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/smr.470

Developing measurement systems: an industrial case study

Miroslaw Staron[1,*,†], Wilhelm Meding[2], Göran Karlsson[2] and Christer Nilsson[2]

[1]*IT University of Göteborg, SE-412 96 Göteborg, Sweden*
[2]*Ericsson SW Research, Ericsson, Sweden*

SUMMARY

The process of measuring in software engineering has already been standardized in the ISO/IEC 15939 standard, where activities related to identifying, creating, and evaluating of measures are described. In the process of measuring software entities, however, an organization usually needs to create custom measurement systems, which are intended to collect, analyze, and present data for a specific purpose. In this paper, we present a proven industrial process for developing measurement systems including the artifacts and deliverables important for a successful deployment of measurement systems in industry. The process has been elicited during a case study at Ericsson and is used in the organization for over 3 years when the paper was written. The process is supported by a framework that simplifies the implementation of the measurement systems and shortens the time from the initial idea to a working measurement system by the factor of 5 compared with using a standard development process not tailored for measurement systems. Copyright © 2010 John Wiley & Sons, Ltd.

Received 16 April 2008; Revised 25 February 2010; Accepted 17 March 2010

KEY WORDS: software metrics; software engineering; measurement

1. INTRODUCTION

Measuring in software engineering is as important as in any other engineering discipline. In addition to frameworks for establishing metric programs in software development organizations, like the Goal-Question-Metric (GQM, [1]), there exist standards defining how measurement processes should be defined and executed (ISO/IEC 15939, [2]) or what/how measures are to be defined (ISO/IEC 25021, [3]). The standards define components of a successful metric program; furthermore, the new ISO/IEC 25000 series of standards contain metrics for quality assessment and describe measurement methods for these metrics. In software development organizations, these standards need to be operationalized—i.e., conforming procedures for metrics data collection, analysis, and presentation need to be developed. The operationalization of these standards is an important aspect for a wide adoption of measurement systems. Therefore, in this paper we address the following research question:

> *Which activities and artifacts are involved in the process of building and deploying measurement systems in industry?*

The concept of a *measurement system* has been adopted from the existing standards on metrology [4] where it is defined as a set of measuring instruments assembled in order to measure quantities of specific kinds. In the case of software engineering, the quantities are dependent on the purpose of measurement and the measured entities. An entity can be a project, process, product, team,

*Correspondence to: Miroslaw Staron, IT University of Göteborg, SE-412 96 Göteborg, Sweden.
†E-mail: miroslaw.staron@ituniv.se

etc. and a quantity can be project length, number of activities in the process, lines-of-code of the product, team size, etc. Since the set of base quantities in software engineering is not as well defined as in other engineering disciplines (e.g., physics with the definition of such quantities as length), our discipline strives to define standards for defining quantities. Although the ISO/IEC 25021 [3] standard defines a set of base and derived measures (DM) for such aspects as quality measure elements, these metrics are still too abstract to be used for making decisions in large corporations[‡]. The standard introduces the concept of 'quality measure element', which is the basis for creating quality metrics, but is a quality metric *per se*. The consequence of the lack of standardized measures is that organizations develop custom measurement systems and metrics to satisfy specific information needs of stakeholders—not limited to quality metrics only.

In practice a successful[§] measurement system is always developed with a stakeholder in mind; examples of stakeholders are project managers, product managers, system owners, team leaders, etc. Each stakeholder has own goals for measuring, which is reflected in the measurement systems being tailor made for the stakeholders. In practice the main measurements—indicators—are specific for the stakeholders. The data-measures, however, should be as generic as possible to allow benchmarking and reuse of measures. These conflicting requirements are addressed in the ISO/IEC 15939 standard by defining three kinds of measures—base measures (BM): reusable measures measuring attributes of empirical entities from the real world, DM: measures combining BM using mathematical expressions, and indicators: derived and/or BM combined with the decision criteria (specific for the stakeholder and his/her information need). The reusability decreases from BM to indicators as the customizability increases.

In this paper we present a process of how to identify indicators, and base/DM in mature organizations adopting metric programs. After our evaluation at Ericsson, we found that using this process leads to developing measurement systems up to 5 times faster than using the standard, non-customized, software development process. The process starts with the identification of stakeholders, which has been indicated as the most important aspect in the literature (e.g., [5]), and by Ericsson's managers. It finishes with the deployed and the evaluated measurement system for that stakeholder. In addition to the process we also present a short evaluation of it at one of the projects at Ericsson, our industrial partner. For the reasons of confidentiality agreement, we cannot present a detailed study of measurement systems at Ericsson including the development effort data, but we can show that the development of measurement systems requires no company-external specialists' knowledge and makes Ericsson independent from costly consultancy in software metrics—a situation that is identified as an obstacle when adopting measurement programs in industry.

The paper is structured as follows. Section 2 presents the most related work in the field. Section 3 briefly characterizes the organization and its measurement practices (including historical timeline of the metric programs in the organization). Section 4 presents the process of developing measurement system, details the activities and deliverables from the process. Section 5 presents an example use of the process when developing measurement systems at Ericsson, including an example measurement system deployed at the company. Finally, Section 6 presents the conclusions.

2. RELATED WORK

The process presented in this paper solves the same problem as GQM paradigm [1], which is used to elicit the set of measurements that are to be collected in order to support decision processes in organizations. The main difference is that in our process model, we focus on following the ISO/IEC 15939 standard and emphasize reuse of measures. We also, according to the standard, perceive

[‡]ISO/IEC 25000 series of standards is a 'type 2 technical report', which means that the report is still under development or that there is a 'future, but not immediate possibility of an agreement on an International Standard' (from ISO/IEC 25021 p. iv–v).

[§]Successful = adopted, deployed, and used for a pre-defined period of time (e.g., during the whole project, for one product).

different measures to have different roles and we put emphasis on finding procedures (automated if possible) to collect and combine measures—concepts stressed in our previous work [6].

The concept of a measurement system is not new in engineering. Measurement instruments are one of the cornerstones of engineering. In this paper, we only consider computerized measurement systems—i.e., software products used as measurement systems. The reason for this are: the flexibility of such measurement systems, the fact that we work in the software field, and similarity of the problems—e.g., concept of measurement errors, automation, etc. An example of a similar measurement system is presented by Wisell *et al.* [7], where the concept of using multiple measurement instruments to define a measurement system is also used. Although differing in domains of applications, these measurement systems show that concepts that we adopt from the international standards (like [4]) are successfully used in other engineering disciplines. Therefore, our process of creating measurement systems can be applied when building these measurement systems.

Lowler and Kitchenham [8] present a generic way of modeling measures and building more advanced measures from less complex ones. Their work is linked to the TychoMetric [9] tool. The tool is a very powerful measurement system framework, which has many advanced features not present in our framework (e.g., advanced ways of combining metrics). TychoMetric provides a possibility of setting up advanced and distributed (over several computers) filters and queries for multiple data sources as it is intended to cover all (or at least many) kinds of metrics and projects. The configuration of a set-up of one measurement system is much more difficult than developing a custom measurement system using simpler tools (which is usually appreciated by managers). The plethora of features and the fact that we intended our framework to be simple, easy to use and interpret rendered the TychoMetric tool too advanced for building measurement systems for wide spread throughout our organization. A similar approach to the TychoMetric's way of using metrics was presented by Garcia *et al.* [10]. Despite their complexity, both the TychoMetric tool and Garcia's approach can be seen as alternatives in the context of advanced data presentation or advanced statistical analysis over time.

Wade and Recardo [5] indicate a strong need for custom-made processes for the development of measurement systems, which should assure that it is the 'business management' and not 'software management' who owns and steer the development of measures and software collecting them. In our paper, we present the process that addresses this issue by providing a customization of a general process (minimizing the learning curve) by minimizing the number of activities and adding guidelines that are specific for developing measurement systems and assuring that business management needs are addressed by metrics and software for their collection/presentation.

Meyer [11, pp. 99–122] claims that the need for customized measurement systems for teams is one of the most important aspects in the adoption of metrics at the lowest levels in the organization. Meyer's claims were also supported by the requirements that the customization of measurement systems and the development of new ones should be simple and efficient in order to avoid unnecessary costs in development projects. In our research, we simplify the ways of developing Key Performance Indicators exemplified by a 12-step model of Parmenter [12] in the domain of software development projects. Parmenter's process tackles a larger problem of creating indicators for managing organizations and therefore requires a significant amount of effort to be used in software projects. As per request of managers at Ericsson, we made sure that our process is compatible with the 12-model. The manager's requests were of key importance for the success of the measurement program, which is also reported in, e.g., [13–18].

We also investigated metric tools used in software industry to compare the differences between measurement instruments used in our research to measurement instruments used in other disciplines—in particular we studied publications on

- Using video recordings in the process of measurements of fluids (hydrology), where the problem of 'digitalizing' information from a physical world was discussed [19]. This problem is often encountered in software development organizations when using resource metrics.
- Network load measurements for a specific protocol using a dedicated measurement system (computer science), where we studied how a software measurement system is constructed and integrated with general software [20].

We deliberately chose the ISO/IEC 15939 (:2002 in the start of the study and :2007 when the study concluded) in our work. It is not the only standard, but it is accepted and was best suited for our purposes. A good comparison of other related measurement frameworks and models was conducted by Chirinos *et al.* [21], who compared several frameworks and models [1, 22, 23]. Despite the recommendations from Chirinos *et al.* to use their measurement framework (MOSME), the ISO/IEC standard was more applicable due to its wider adoption in industry. The use of standardized view on measurement processes also provided the possibility for future benchmarking with other organizations (e.g., as indicated in [24]). The standards are also based on the state-of-the-art in measurement theory, which can be found in [25–28].

Agresti [15] presents a framework for defining meaningful metrics for IT organizations focusing on 10 most important areas of metrics: problem, people, platforms, providers, process, product, performance, progress, plans, and partners. Although the generality of these areas can only guide the organizations to choose the right metrics, we see this work as an important input to our process—in particular for organizations without clearly identified stakeholders. Agresti's framework can support these organizations in identifying the stakeholders and as such complements our work to support full lifecycle of the products.

Umarji and Emurian [29] present another study related to the problem of defining metrics useful for the organizations. They identify five aspects that directly influence the acceptance of a metric program: ease of use, usefulness, control over the measurement program, attitude of the individual to the metrics, and intention (which is an overarching concept for the previous four aspects). Their results show that involving stakeholders in early stages of development of metrics significantly increases the chances of success of the metric program. Similar conclusion is supported by the studies of Jeffery and Berry [30] and Gopal *et al.* [31].

Kilpi [32] studied acceptance issues and the role of the software metric program at Nokia showing that using metric programs in a mature organization like Nokia can save considerable effort when fine-tuning software development processes. They describe a method: Nokiaway, which is a Nokia adaptation of the GQM approach. One of the main differences to the standard GQM approach is the need for using only pre-defined metrics in measurement systems, which might be a risk for an adoption of a metric program (a bottom–up approach). The reason for this risk is that the stakeholders (as mentioned in the studies above) might need customized metrics. Our work addresses this shortcoming by presenting a process for developing metrics that as part of measurement systems customized for particular stakeholders.

Niessink and van Vliet [33] stress the need for the metric programs to generate value. In particular, they stress the need for a clear mapping from organizations needs to metric programs, measurement systems, and metrics (a top–down approach). They also emphasize the importance of coupling between the metrics and organizational actions—i.e., actions to be taken based on the results of the metrics. Their conclusions are aligned with our intentions—to create a process for developing measurement systems generating the value by involving the stakeholders and addressing their needs. Since the stakeholders are persons who have the mandate and possibility to act upon the values of metrics, we perceive the V-process presented in our paper as addressing the needs identified by Niessink and van Vliet. The need for addressing of the needs of the company and this top–down approach was identified even in an earlier study by De Panfilis *et al.* [34] or Redmond and Ah-Chuen [35]. In the design of the V-process model, we were careful to take these experiences into an account.

3. MEASUREMENT SYSTEMS AT THE STUDIED ORGANIZATION

The studied organization is one of the units of Ericsson. Ericsson is one of the leading global telecommunication equipment providers. The studied organization is responsible for development of network products for the telecommunication networks. The group that we work with focuses on quality management, methods and tools supporting software development projects. Their daily tasks include collecting, maintaining, and presenting quantitative data (measurements) on processes, products, and projects in the organization. The organization consists of several hundred engineers. In this section, we briefly present the way in which measurements were used in the past in the organization and how measurement systems affect that situation.

3.1. Measurements in the past

The history of measuring at the studied organization started around 2002 when the need for quantitative data for decision support arose. Before that point, measures were used in the organizations supervising the studied organization at the company level. In 2002, the awareness and the need for controlled metric program came into existence. The discussions were continued for a number of years, including prototypes and early pitfalls. During 2003, the organization built various kinds of prototype metric programs including the presentations for decision makers, which results in the development of common performance reporting in 2004. The awareness of measuring during these years resulted in using different measures for same purposes and same measures for different purposes in the organization. During 2005, an effort was undertaken to systematize the measures and reduce the number of measures collected. In 2006, an effort was established to build a single metric program for the organization to control and monitor projects, products, and resources in the organization. Until now this effort has resulted in a partial implementation of this metric program. The incremental improvement of this program is planned for the future. Based on the current situation in the company, one could predict that the organization will require reactive measurement systems to support changing metric programs. Over the years, the measures used changed, although several are still valid (e.g., the In-Service-Performance (ISP)). It also shows that after improving in aspects currently in focus (and measured), the organization starts to improve other aspects and thus needs new measures.

In the starting days of measuring it was only a set of individuals who discussed the use of metrics for decision support. Currently, software metrics are discussed at all levels in the organization and the needs for using measures for decision support are growing continuously. These growing needs require infrastructure where measures can be used (e.g., combined, summarized) by end users (e.g., managers, system owners), in a considerably short notice. One of the requirements for the case study presented in this paper was to address the increased needs for measurement systems.

The studied unit works mainly with quality-related measures. These measures are usually specified in quality assurance plans for each project, collected during projects and presented at the project status meetings. The measures in quality assurance plans were collected in order to monitor project and product quality. And, since the maturity of the organization grew over time, the measures evolved. At the beginning (around 2002), there were around 20 different areas, containing between 70 and 100 measures. Some of the areas were intended to monitor whether a project adheres to process descriptions (e.g., measuring different activities in the projects). The measures were collected manually and then used in performance reports (which were created manually as well). Nowadays, as the organization is much more mature, these measures are not necessary—processes are followed and there is no need to spend resources on controlling that. Owing to this and similar factors, currently the organization collected measures in seven areas, and the number of measures decreased to approximately 20 metrics that monitor key aspects of projects. These measures are updated daily and always show the current status of the project, the measurement systems are available via web sites and there is no need for manual performance reporting on these measures. As the competence in the company w.r.t. metrics collection and analysis is significantly higher than before, the demands for measurement systems grows constantly and there is a need to make the knowledge how to build measurement systems available to more persons than just a few individuals (as it was at the beginning).

3.2. Current measurement systems

As part of our project, we performed an evaluation of the metric program at the studied organization and its chances of being successful, according to the framework of Jeffery and Berry [30]. The results showed that the metric program fulfills most of the requirements for being successful. The uniform process of creating measurement systems is one of the important aspects of that. The current measurement processes in the organization are based on ISO/IEC 15939:2007 standard, which is a normative specification for the processes used to define, collect, and analyze quantitative data in software projects or organizations. The central role in the standard is played by the information product, which is a set of one or more indicators with their associated interpretations that address the information need [2]. The information need is an insight necessary for a stakeholder to manage objectives, goals, risks, and problems observed in the measured objects [2]. These measured objects can be entities, such as projects, organizations, software products, etc., characterized by a set of attributes. We use the following definitions from ISO/IEC 15939:2007:

- BM—measure defined in terms of an attribute and the method for quantifying it. This definition is based on the definition of base quantity from [4].
- DM—measure that is defined as a function of two or more values of BM. This definition is based on the definition of derived quantity from [4].
- Indicator—measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs.
- Decision criteria—thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result.
- Information product—one or more indicators and their associated interpretations that address an information need.
- Measurement method—logical sequence or operations, described generically, used in quantifying an attribute with respect to a specified scale.
- Measurement function—algorithm or calculation performed to combine two or more BM.
- Attribute—property or characteristics of an entity that can be distinguished quantitatively or qualitatively by human or automated means.
- Entity—object that is to be characterized by measuring its attributes.
- Measurement process—process for establishing, planning, performing, and evaluating measurement within an overall project, enterprise, or organizational measurement structure.
- Measurement instrument—a procedure to assign a value to a BM.

The view on measures presented in ISO/IEC 15939 is consistent with other engineering disciplines, the standard states at many places that it is based on such standards as ISO/IEC 15288:2007 (Software and Systems engineering—Measurement Processes) [36], ISO/IEC 14598-1:1999 (Information technology—Software product evaluation) [37], ISO/IEC 9126-x [38], ISO/IEC 25000 series of standards, or International vocabulary of basic and general terms in metrology (VIM) [4]. Conceptually, the elements (different kinds of measures) that are used in the measurement process can be presented as in Figure 1.

One of the key factors for every measurement system is that it has to satisfy an information need of a stakeholder–i.e., there needs to be a person/organization who/which is interested in the information that the measurement system provides. Typical stakeholders are project managers, organization managers, architects, product managers, customer representatives, and similar [14, 32, 39, 40]. The indicator is intended to provide quantitative information along with the interpretation, which implies an existence of an analysis model that eases the interpretation. The analysis model is a set of decision criteria used when assessing the value of an indicator—e.g., describing at which value of the indicator we set a red flag signaling problems in the measured object. The DM (based on the definition of the derived quantity) and BM (based on the definition of the base quantity) are used to provide the information for calculating the value of the indicator.

4. DEVELOPING MEASUREMENT SYSTEMS: PROCESS

In this section we present the process—the activities and the related deliverables/artifacts of the process of building measurement systems. The actor who performs these activities is measurement specialist—usually a quality manager—who has the basic knowledge of metrics.

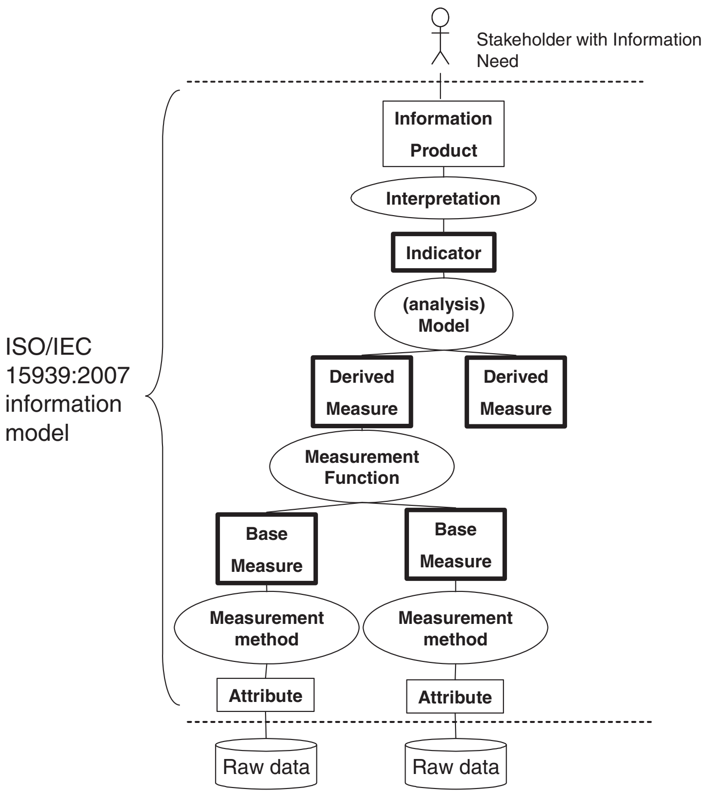


Figure 1. Measurement system information model (from ISO/IEC 15939:2007).

4.1. Overview

The process, which is presented in Figure 2, can be iterative in the middle (as indicated by the shaded area) in order to minimize the risks for misunderstandings between measurement specialists and stakeholders. The horizontal lines in Figure 2 show dependency between activities, and using the V-shape helps to visualize these dependencies in a more natural way. We use the parallel to the V-model in order to help the organization to easily understand it.

The V-shape allows us also to show the progression from high abstraction levels to very detailed specifications and then developing the measurement system in the bottom–up approach.

4.2. Activities

During the development of the measurement system, the actor needs to perform the following activities. There are two roles involved in these activities: developer of the measurement system (measurement specialist, quality manager) and stakeholder (who provides the developer with information). The developer is involved in all activities, whereas the stakeholder is involved in the activities 1–10 and 17–18.

1. *Elicit information need from stakeholders*: The stakeholders are the primary source of information needs as it is they who are the 'customers' for measurement systems. The information needs depend on the role of stakeholders, but the main characteristic is that the information need must be a product of one or more measures.
   (a) Questions to ask:
      (i) What is it that you need to know to monitor <your project>? (the part in brackets depends on the information need—here we use project as a 'template' example)
      (ii) Why do you need to know it?

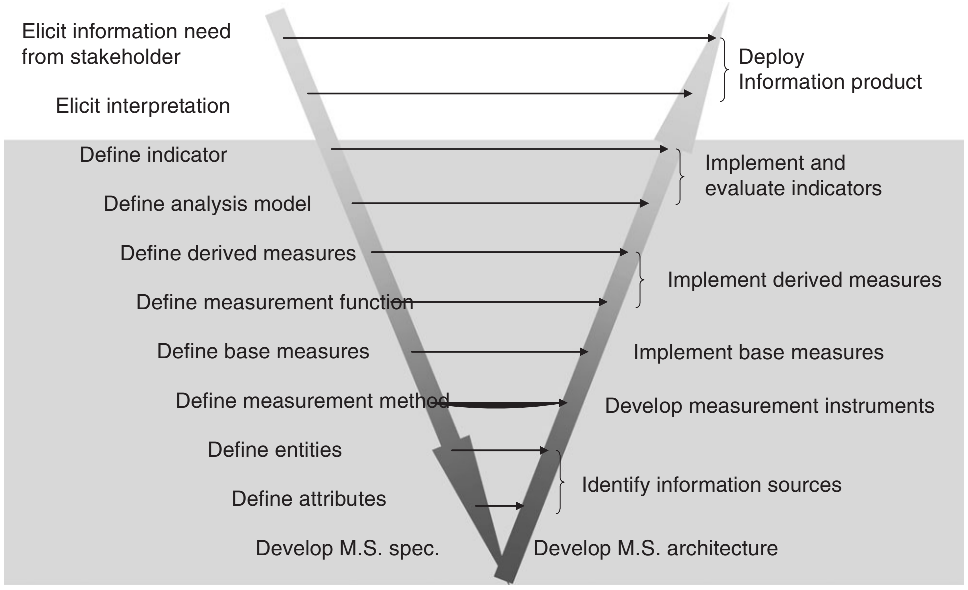

Elicit information need from stakeholder
Elicit interpretation
Define indicator
Define analysis model
Define derived measures
Define measurement function
Define base measures
Define measurement method
Define entities
Define attributes
Develop M.S. spec.
Deploy Information product
Implement and evaluate indicators
Implement derived measures
Implement base measures
Develop measurement instruments
Identify information sources
Develop M.S. architecture

Figure 2. Process for developing measurement systems.

2. *Elicit interpretation*: Once the information need is elicited, the stakeholder should be asked how the information is interpreted, including actions that are taken upon specific interpretation. The interpretation is to be used in the next step when 'color-coding' the decision criteria for the indicator.
   (a) Questions to ask:
      (i) How do you act based on the information provided from indicators?
3. *Define indicator*: By the time the interpretation is found, several candidate indicators are usually discussed. In this activity, the candidate indicators are assessed and the main indicator (or a limited number of them) is found.
   (a) Questions to ask:
      (i) What is the most important indicator that would notify you when there are problems with <your project>?
      (ii) Which other indicators help you to monitor that <your project> is according to your expectations/plan?
4. *Define analysis model*: After the indicator is found, the stakeholder is asked about the thresholds on the interpretation of the indicator. The thresholds are used to assess whether the indicator shows problems, shows potential problems, or shows that measured phenomena are progressing according to expectations or plans. The analysis model contains also the formula used to calculate the value of the indicator from derived and BM. The thresholds are usually color-coded to help stakeholders quickly see whether the status is 'red' or 'green' and also for non-stakeholders when interpreting the information.
   (a) Questions to ask:
      (i) When can you say that the indicator shows problems with <your project>?
      (ii) What are the values of the thresholds?
5. *Define DM*: After the indicators are found, define the DM, which are used in the formulas in the analysis model. These DM are defined in this activity.

   (a) Questions to ask:
      (i) Which measures are used in the formulas?
   (b) *Note*: *Not all indicators are calculated from DM* (*some can just be BM with associated decision criteria*). *If this is the case, then this activity is not applicable.*

6. *Define measurement function*: Each DM is calculated from one or more BM. In this activity, the function used to calculate the DM is defined and documented.
   (a) Questions to ask:
      (i) Which BM are used to calculate DM?
      (ii) How are the BM used to calculate DM (what is the formula)?
   (b) *Note*: *This activity is optional if no DM are found.*
7. *Define BM*: The BM are coupled directly to DM. The underlying raw data (see activities 13–14) are usually very detailed and the BM are only an excerpt of the data (e.g., the raw data can be per team member per project whereas the BM is supposed to be the average value per project—i.e., data re-configuration is needed).
   (a) Questions to ask:
      (i) Which BM are needed?
      (ii) Can these BM be reused?
8. *Define measurement method*: The most important part of the definition of the BM is how to assign the values to BM (i.e., how to measure)—the measurement process. This measurement process for a single BM is called measurement method. The measurement method is an algorithm that assigns value of a particular attribute of a particular entity to the BM.
   (a) Questions to ask:
      (i) Is the measurement method correct?
      (ii) Are all elements, which should be measured, measured?
9. *Define entities*: The entities that are measured need to be precisely defined since these are the main sources of information. The granularity and availability of BM depend on the definition of measured entities. Creating a domain model (similar to domain models known from the Unified Process [41] or Software Product Lines [42] can be used).
   (a) Questions to ask:
      (i) What are the entities in the measured organization/product/process (depending on the information need)?
      (ii) What are the relationships between these entities?
      (iii) How can the data for these entities be obtained?
   (b) Artifact: Domain model with entities which is a part of the specification of the measurement system
10. *Define attributes*: Each entity is characterized by its attributes and relationships. The purpose of this activity is to identify the relevant attributes and the methods to measure them. An attribute that cannot be measured should be replaced by an attribute that is easier to measure (or several other attributes).
   (a) Questions to ask:
      (i) What are the most important attributes (of the entities) that are relevant for the information need?
      (ii) What are the most important relationships between entities that are important for the information need?

11. *Develop measurement system specification*: After the entities, attributes, and measurements are identified, a specification of the measurement system is developed. The specification is a model of all measurements, entities, and their attributes. The model serves also as a documentation of measurements used and provided by the measurement system.

    (a) Artifact: measurement systems' specification

12. *Develop measurement system architecture*: The specification of the measurement system is a logical view of the measurement system. An implementation view of the measurement system is the specification of its architecture (in the form of a model): components used, e.g., files and databases.

    (a) Artifact: measurement system architecture model.

13. *Identify information sources*: The measurement method is directly coupled to the information sources. The information source determines how the measurement method is specified—it can be a list of questions (if an information source is a person) or a description of the structure of the database (if the database is the information source), or a procedure how to access/parse the file (if it is a text file).
14. *Develop measurement instruments*: Once the information sources are identified and described, the measurement instruments are developed. A *measurement instrument* is an algorithm that computes how the measurement value is assigned to the BM. A measurement instrument can be a user form in MS Excel or a webpage (if the information source is a person), or a software program integrated with measurement system (if the information source is a database). The measurement instrument implements the measurement method specified earlier in the process.

    (a) Deliverable: measurement instrument(s).

15. *Implement BM*: Once the measurement instrument for assigning the values to BM is in place, the BM are implemented. The implementation is creating the place where the measurement values are stored and how they are accessed.

    (a) Deliverable: BM

16. *Implement DM*: When BM are in place, the DM can be implemented. The implementation includes accessing BM, implementation of formulas, and methods for accessing DM.

    (a) Deliverable: DM.

17. *Implement and evaluate indicators*: The indicators are to be developed and evaluated. The development is similar to the development of base and DM. The evaluation, however, should be done together with the stakeholder; the values of the indicators reflect the stakeholder's expectations. The stakeholders are individuals or groups of specialists whose perception of the situation of ⟨the project⟩ (or other phenomena depending on the information need) is reflected by the status of the indicator (e.g., low or high value, 'red' or 'green' color). This can be done by using the historical data during the evaluation of indicators, or by having an indicator in the 'evaluation' phase for a period of time when the values are recorded and discussed with the stakeholder.

    (a) Questions to ask:

        (i) Are the thresholds correct?
        (ii) Do the thresholds reflect the stakeholder's perception of the situation in <the project>?

    (b) Deliverable: indicators.

18. *Deploy information product*: After the indicators are validated with the stakeholder, the measurement system can be made accessible in the organization. The usual way to do that

would be to create an internal webpage where the values of the indicators are displayed and explained. The color-coding helps in understanding whether an indicator indicates problems (e.g., by being colored as red) or lack thereof (e.g., by being colored as green).

(a) Questions to ask:

  (i) Where should the information be published?
  (ii) How often should the published information be updated?

(b) Deliverable: measurement system (information product).

4.3. *Artifacts and deliverables*

One of the requirements for deliverables is that a non-metric-expert is able to understand and use measures and indicators for his/her specific purposes (if needed). The main idea is that the process should be very light-weight and should minimize the amount of 'paperwork' and documentation to these artifacts that are going to be used even in the maintenance phase. In addition to the deliverables from this process, there are two important artifacts that are important in the process and for further maintenance of the measurement systems.

Deliverables

1. *Information product*: The main deliverable is the measurement system itself and in particular the indicators. The indicators are labeled as information products since they provide the information that satisfies the information needs of the stakeholder.
2. *Indicators*: Each of the developed indicators is also a deliverable that is intended to be reused for different stakeholders. The indicators can be packages with or without the analysis models.
3. *DM*: The measurement systems should provide the possibility to reuse DM—e.g., by importing values of derived measures from other measurement systems. This reuse is important in order to provide a uniform and consistent way of measuring the same phenomena across the company.
4. *BM*: Similar to DM.
5. *Measurement instruments*: The instruments used to collect measurements are also important deliverables. The organization is supposed to standardize the ways in which these are specified and through this standardization to develop uniform way of collecting measures.

Artifacts:

6. *M.S. specification*: The model of all measures provided by the measurement system and the scales and units of the measures. The model is also important for later maintenance of the measurement system.
7. *M.S. architecture specification*: The model of components and data sources in the measurement system is important when problems occur. An important aspect of the measurement system is that it provides accurate and up-to-date information. If a problem occurs, one needs to be able to easily track the information flow in the measurement system (e.g., which files are involved in calculating each particular indicator).

The dependencies between these two artifacts and the components of the measurement systems are presented in Figure 3.

The important part of the figure is the distinction between the specifications (models) and the realization of files, databases, and the measured entities. The deployed measurement system is usually built using several files (depicted as files on an ellipse), which correspond directly to the specifications (depicted on rectangles).

In the studied organization we use an in-house developed framework for building measurement systems. The framework is built on top of MS Excel 2003 and is intended to provide the basic structure of files and measurements. The framework is conceptually presented in [6].

The measurement systems are built on top of Microsoft Excel (MS Excel) in order to reuse the tools that are already used to collect measures and data in the company. The measurement system contains three main components, corresponding to elements in the measurement information model

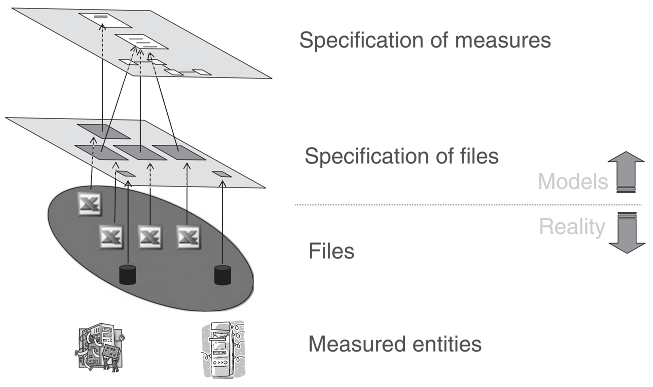


Figure 3. Dependencies between specifications, files (components of the measurement system) and the measured entities.

specified in the ISO/IEC 15939:2007 standard: indicators, DM, and BM. The dotted line around these components is intended to show that these three components might exist in one or several execution spaces—i.e., being separate MS Excel instances (any combination is allowed, e.g., BM and DM in one file whereas Indicators in another one). In the case of using several instances, the measures are imported when needed, e.g., BM are imported when needed for calculation of DM. Values are assigned to BM using measurement instruments (MI).

4.4. *Validation of measurement systems*

Verification of a measurement system should be done using standard software testing methods (e.g., following such standards as ISO/IEC 14598-1:1999 [37]). The measurement systems, however, also need to be validated in a specific way. Since measurement systems are based on software metrics; these metrics need to be validated both theoretically and empirically. The theoretical validation of metrics should be done using a theoretical validation framework—for example the framework of Briand *et al.* [43], or Zuse [44]. In our work we use Briand's framework since it allows checking properties of metrics useful in our context, e.g., null value or additivity. As DM and indicators are combinations of base and DM, these properties should be verified before measures are combined.

The verification whether measures expose certain properties is only a partial validation of software metrics. The more important part is the empirical validation, which is intended to check whether a measure actually measures what it is expected to measure [45, 46]. Although extensive experimentation is usually used to empirically validate the metrics, we advocate for a more pragmatic approach:

1. Develop and deploy a measurement system.
2. For a period of time, validate the indicators and measures with the stakeholder: this validation is done by observing the values of the indicators together with the stakeholder and ask the stakeholder to assess the same phenomena without using the measurement system (stakeholder's view).
3. After a period of time, if the stakeholder's view is consistent with the indicators values, then we can assume that the measures are empirically valid.

The above pragmatic approach is based on the following assumptions: (i) the stakeholder is an expert in the area and has the 'intuition' about the same phenomena that the indicator presents, (ii) the period of time contains both negative and positive values in the indicator. Although the period of time when the measurement system is observed can be substituted by using historical data, we strongly recommend using the data from on-going activities and current products. The reason is that the stakeholder's view is usually more up-to-date when considering situations in measured on-going projects, products, etc. Empirical validation of metrics on historical data is usually biased by the maturity effect.

4.5. Factors important in implementing measurement systems

Since measurement systems are intended to support metric programs in organizations, the usual factors important for metric program adoption are also considered crucial, for example management commitment, dissemination, etc. [14, 32, 33, 39, 47–50].

In addition to these factors, we have identified the following as success factors specific to building measurement systems in the studied organization:

1. Use of standards: the measurement systems should follow standards, e.g., ISO/IEC 15939, and structure measures in uniform way.
2. The specifications and implementations of the measurement systems should be similar to each other.
3. Reuse of measures: Measures defined in the measurement systems should be 'packaged' so that they can be reused in other measurement systems. Otherwise, the cost of measuring might get too large if the number of measures increases.
4. Information sources are known: When building a measurement system, the developer/design should know whether the information can be obtained and how. If the desired information is not obtained, a danger exist that the measurement system is going to be based on opinions (which might be subjective) and not objective data.
5. Maintainability: the infrastructure (information sources, executable files, definitions of indicators and measures) should be maintainable by other people in the organization than the developer of the measurement system.
6. Simple, succinct, and unambiguous presentation: The indicators should present the information in a very succinct way that is very easy to interpret. This means that the indicators should contain no details about the underlying measurements; these should be provided at a separate place (linked from the indicators presentation) and accessed only when needed.
7. Accurate and reliable information sources: The information collected by the measurement systems has to be accurate—the instruments used to collect the data should be precise and do not introduce measurement errors. The sources of information should also be reliable, which means that information is up-to-date and not biased—for example 'current' project status has to be measured at a specific time (e.g., at the moment when the indicator is presented) and does not come from a month before the moment when the indicator is presented.
8. Clear responsibility assigned: The responsibility of the measurement system should be clearly assigned. There should be a group or an individual responsible for development, deployment, and maintenance of the measurement system. If this is not fulfilled, there is a danger that the measurement system gets de-calibrated (does not measure accurately or measures are no longer required as the phase in the project has changed).

The above are only the most important factors, although there are several minor issues that should be taken into consideration when developing and deploying a measurement system. Most of these issues, however, have already been captured by Jeffery and Berry in their evaluation framework [30].

5. EXAMPLE USE OF THE MODEL

We exemplify the use of the model on an example measurement system for measuring ISP, which was developed using the process presented in this paper. The summary of the specifications of the measurement systems is presented in Table I. The table summarizes the results of activities from 'Elicit information need from stakeholder' to 'Find attributes' as presented in Section 4.

The specification of the measurement system contains the measures used in this measurement system. It is presented in Figure 4.

The figure shows the graphical version of the specification of the measurement system for the ISP. The important aspect of the figure is its layout that is similar to the ISO/IEC 15939:2007 information model. Such a layout makes it quite straightforward for the stakeholders to understand the specification. In particular, the stakeholders can 'learn' the nomenclature of ISO/IEC 15939 while

Table I. Summary of the specification of the measurement system for monitoring In-Service-Performance (ISP).

Stakeholder	Product manager
Information need	Are we fulfilling the X_1 availability promise?
Indicator	ISP: Unplanned downtime for all products
Analysis model	Green: X_1 to 100% availability Yellow: X_2 to X_1 availability Red: Below X_2 availability
Derived measures	PA: Product availability last month extrapolated for 1 year (min/year)
Measurement function	PA = sum(RA for all releases)/NR
Base measures	Object: Release RA: Release availability last month extrapolated for 1 year (min/year] Object: Product NR: Number of releases [number]
Measurement method	For RA: 1. Collect downtime logs from all nodes running the release for the last month 2. Calculate average downtime for all nodes running the release 3. Count the number of minutes in last month 4. RA = number of minutes in last month − average downtime for all nodes running the release For NR: 1. Count the number of releases of software that are in operations

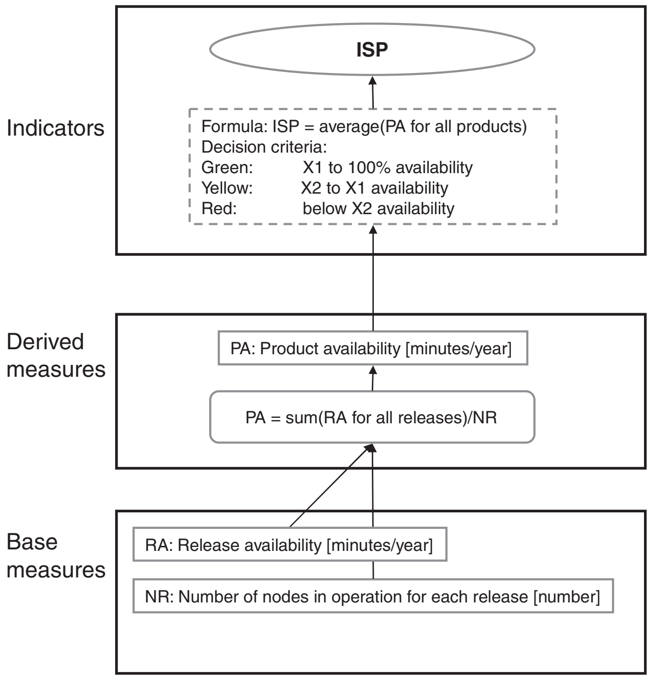

Indicators
ISP
Formula: ISP = average(PA for all products)
Decision criteria:
Green: X1 to 100% availability
Yellow: X2 to X1 availability
Red: below X2 availability
Derived measures
PA: Product availability [minutes/year]
PA = sum(RA for all releases)/NR
Base measures
RA: Release availability [minutes/year]
NR: Number of nodes in operation for each release [number]

Figure 4. M.S. specification for the ISP measurement system.

developing measurement systems that lead to such bi-effects as easier reuse of base/DM (the stakeholders can easily distinguish between base/DM and indicators) or easier customization of indicators (the stakeholders have a clear view of which measures are indicators and what their analysis model is).

The studied organization has developed several releases of the software for the products, which causes the organization to collect the information per each release for each product. In total the organization monitors three products (Product A, Product B, and Product C); hence, there are

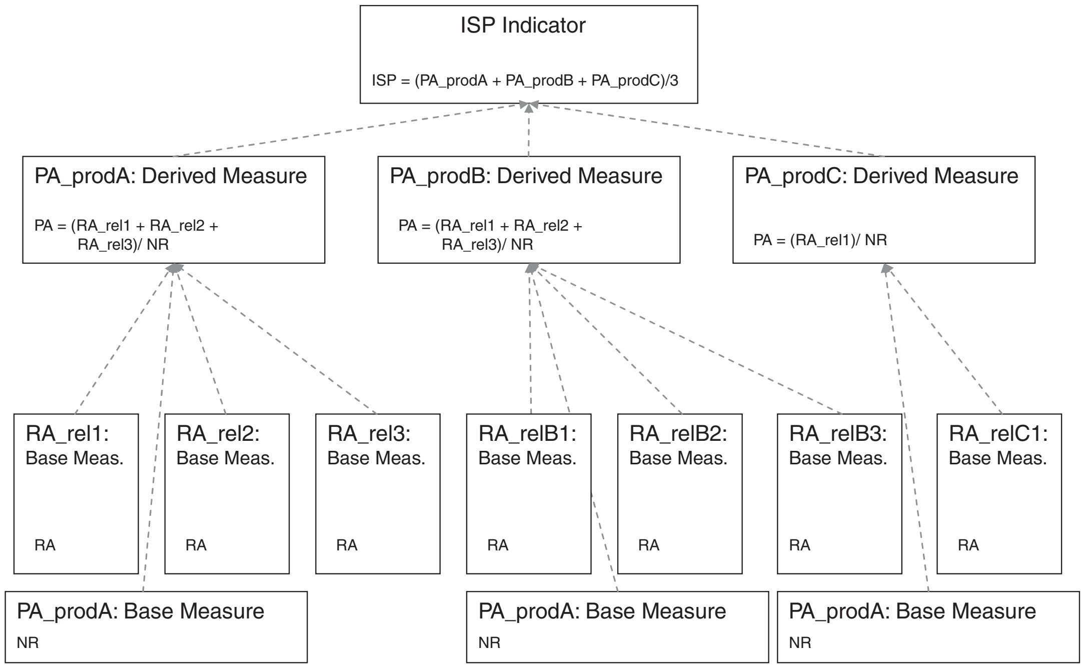

ISP Indicator
ISP = (PA_prodA + PA_prodB + PA_prodC)/3
PA_prodA: Derived Measure
PA = (RA_rel1 + RA_rel2 + RA_rel3)/ NR
PA_prodB: Derived Measure
PA = (RA_rel1 + RA_rel2 + RA_rel3)/ NR
PA_prodC: Derived Measure
PA = (RA_rel1)/ NR
RA_rel1: Base Meas. RA
RA_rel2: Base Meas. RA
RA_rel3: Base Meas. RA
RA_relB1: Base Meas. RA
RA_relB2: Base Meas. RA
RA_relB3: Base Meas. RA
RA_relC1: Base Meas. RA
PA_prodA: Base Measure NR
PA_prodA: Base Measure NR
PA_prodA: Base Measure NR

Figure 5. Instantiation of measurement system for ISP (M.S. architecture specification).

three instances of DM. Each product has a number of releases in operation (the releases that are no-longer maintained, out-dated are excluded): Product A—three releases in operation; Product B—three releases in operation; Product C—one release in operation. Therefore, the specification is instantiated as presented in Figure 5.

This model in Figure 5 shows that even for a simple measurement system, the number of files and data points can grow rapidly. This means that automation plays a major role in the success of adopting the metric program. In practice, each rectangle in Figure 5 is a separate file in this example (although there is no requirement that it should be a separate file). Each file measuring a release is identical in structure, but differs w.r.t. which object is measured. Finally, there is one file with the indicator that satisfies the information need of the stakeholder.

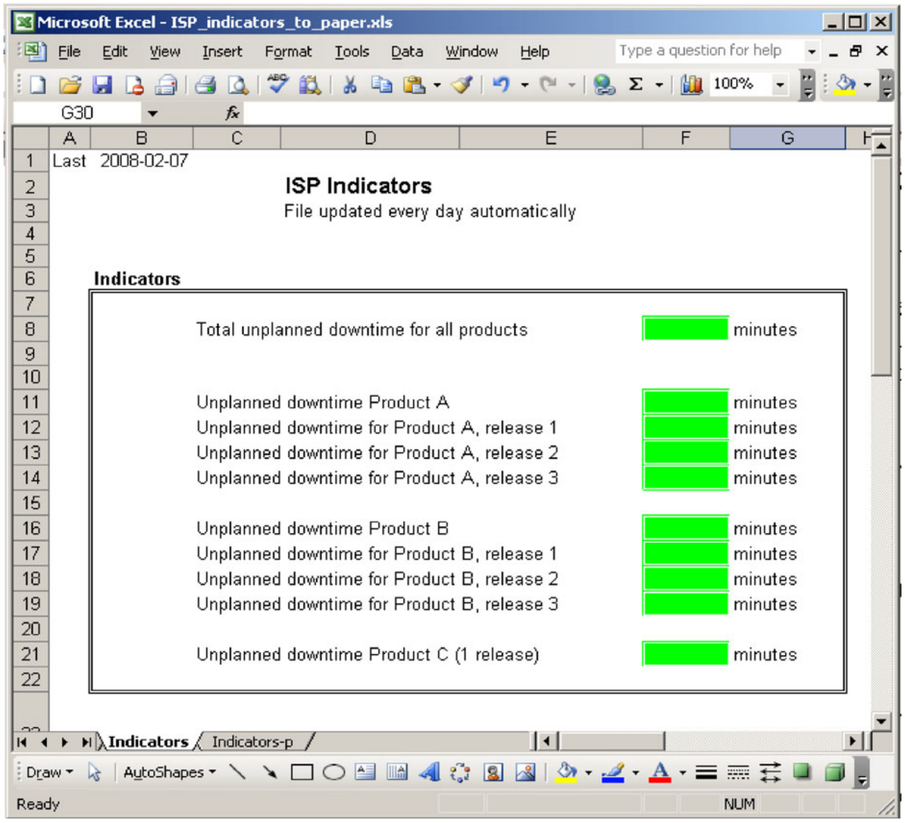
Finally, the measurement system instantiated from these specifications, which is used at the organization, is presented in Figure 6. The measurement system contains the total ISP indicator (the first indicator from the top), and several more detailed indicators, which use the same analysis model to present ISP indicators per product and per release. The values of the indicators (downtime) have been hidden due to confidentiality of the data.

The measurement system updates the information from databases daily and updates the presented information accordingly.

6. RECOMMENDATIONS FOR IMPLEMENTING ORGANIZATIONS

Based on the experiences of using the presented process for developing measurement systems at Ericsson, we can provide the most important recommendations for other companies willing to use metrics according to ISO/IEC standards: ISO/IEC 15939:2007, ISO/IEC 9126-x, ISO/IEC 25000, and ISO/IEC 14598-x. The recommendations are as follows:

1. Utilize top–down approach: start from the information need and not what can be measured. One typical mistake is to start from 'what can be measured', which leads to wasting resources for collecting data, which is not useful for making decisions.
2. Do not reason in terms of 'what are the main metrics', but in terms 'what do we need to know'; Start from 'what we need to know' and continuing toward what and how we can


Microsoft Excel - ISP_indicators_to_paper.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
G30
Indicators
Indicators-p
Ready
NUM

Last 2008-02-07

ISP Indicators
File updated every day automatically

Indicators

Total unplanned downtime for all products	🟩	minutes
Unplanned downtime Product A	🟩	minutes
Unplanned downtime for Product A, release 1	🟩	minutes
Unplanned downtime for Product A, release 2	🟩	minutes
Unplanned downtime for Product A, release 3	🟩	minutes
Unplanned downtime Product B	🟩	minutes
Unplanned downtime for Product B, release 1	🟩	minutes
Unplanned downtime for Product B, release 2	🟩	minutes
Unplanned downtime for Product B, release 3	🟩	minutes
Unplanned downtime Product C (1 release)	🟩	minutes

Figure 6. Measurement system for ISP: presentation of indicators.

measure to provide this information. This focuses the whole V-process on the goal at all times, not discussing the importance of the metrics.

3. The developer of the measurement system should not bias the stakeholder's opinion. The stakeholder is the person who needs the information to make the decision—therefore, it should be the stakeholder who decides how to interpret the information (the indicator).
4. Automate the processes of data collection, analysis, and presentation. It is of key importance that the information is always up-to-date and that the stakeholder can trust that the updates of the metrics data were correct. Manual collection of data should be allowed only in special cases as the costs of regularly collecting the data in a manual way very often exceed the benefits from the data itself. The manual data collection should be facilitated when new metrics are evaluated or the metrics data are collected sporadically.

Although there exist more detailed experiences from our work on implementing the metric programs, both we and our stakeholders perceive the above recommendations as the most important ones.

7. CONCLUSIONS

Effective and efficient measurement processes in industry require well-defined measures, processes for developing/using and maintaining measures and automation support. In this paper, we present a process for developing measurement systems and its industrial instantiation. The process is intended to operationalize the way in which organizations design, deploy, and use measurement systems. In the studied organization, we have developed and deployed several measurement systems in the manner exemplified in the paper.

The presented process can be used with several measurement system frameworks—e.g., the TychoMetrics tool or a custom-made framework. The results from our initial evaluations show that the process is efficient, produces the required documentation (but not too much documentation), and results in good-quality measurement systems. We naturally assume that the engineers working with the measurement systems have the required competence in working with software metrics and the required knowledge of the company.

Our further work includes empirical evaluation of the process with experiments outside of the company and the deployment of the process at our other industrial partners.

ACKNOWLEDGEMENTS

The project has been partially sponsored by the Swedish Strategic Research Foundation (www.stratresearch.se) under the program Mobility in IT. It was also partially sponsored by Ericsson Software Architecture Quality Center and Ericsson Software Research.

We would like to thank the involved managers and engineers at Ericsson for their support in this study.

REFERENCES

1. van Solingen R. *The Goal/Question/Metric Approach*: *A Practical Handguide for Quality Improvement of Software Development*. McGraw-Hill: New York, 1999.
2. International Standard Organization and International Electrotechnical Commission. *Software Engineering–Software Measurement Process*, ISO/IEC (ed.). ISO/IEC: Geneva, 2002.
3. International Standard Organization. ISO/IEC *25021*: *Software Engineering—Software product Quality Requirements and Evaluation* (*SQuaRE*)*—Quality Measure Elements*. International Standard Organization: Geneva, Switzerland, 2005.
4. International Bureau of Weights and Measures. *International Vocabulary of Basic and General Terms in Metrology = Vocabulaire International des Termes Fondamentaux et Généraux de Métrologie* (2nd edn). International Organization for Standardization: Genève, Switzerland, 1993; 59.
5. Wade D, Recardo RJ. *Corporate Performance Management*: *How to Build a Better Organization through Measurement-driven Strategic Alignment*. Improving Human Performance Series, Butterworth-Heinemann: Boston, 2001; xvi, 170.
6. Staron M, Meding W, Nilsson C. A framework for developing measurement systems and its industrial evaluation. *Information and Software Technology* 2008; **51**(4):721–737.
7. Wisell D, Stenvard P, Hansebacke A, Keskitalo N. Considerations when designing and using virtual instruments as building blocks in flexible measurement system solutions. *IEEE Instrumentation and Measurement Technology Conference Proceedings*, 2007. *IMTC 2007*. 2007; 1–5.
8. Lawler J, Kitchenham B. Measurement modeling technology. *IEEE Software* 2003; **20**(3):68–75.
9. Predicate Logic. TychoMetrics, 2007. Available at: http://www.predicatelogic.com [30 June 2008].
10. Garcia F, Serrano MA, Cruz-Lemus JA, Ruiz F, Piattini M. Managing software process measurement: A meta-model based approach. *Information Sciences* 2007; **177**(2):2570–2586.
11. Harvard Business School. *Harvard Business Review on Measuring Corporate Performance*. Harvard Business Review Paperback Series. Harvard Business School Press: Boston MA, 1998; 229.
12. Parmenter D. *Key Performance Indicators*: *Developing*, *Implementing*, *and Using Winning KPIs*. Wiley: Hoboken NJ, 2007; xv, 236.
13. D'Ambrogio A, Iazeolla G, Pasini L. A method for the production of simulation models with application to web interaction paradigms. *Simulation Modelling Practice and Theory* 2007; **15**(5):605–620. Available at: http://dx.doi.org/10.1016/j.simpat.2004.06.009.
14. Gopal A, Mukhopadhyay T, Krishnan MS. The impact of institutional forces on software metrics programs. *IEEE Transactions on Software Engineering* 2005; **31**(8):679–694.
15. Agresti WW. Lightweight software metrics: The P10 framework. *IT Professional* 2006; **8**(5):12–16.
16. Visser JK, Sluiter E. Performance measures for a telecommunications company. *AFRICON 2007*, 2007.
17. Mitani Y, Matsumura T, Barker M, Tsuruho H, Inoue K, Matsumoto K. Proposal of a complete life cycle in-process measurement model based on evaluation of an in-process measurement experiment using a standardized requirement definition process. *First International Symposium on Empirical Software Engineering and Measurement*, *ESEM 2007*, 2007.
18. Staron M, Meding W. Ensuring reliability of information provided by measurement systems. *Software Process and Product Measurement*. Springer: Berlin, 2009; 1–16.
19. Oberkampf WL, Trucano TG. Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences* 2002; **38**(3):209–272.
20. Feigin J, Pahlavan K. Measurement of characteristics of voice over IP in a wireless LAN environment. *IEEE International Workshop on Mobile Multimedia Communications*, 1999.
21. Chirinos L, Losavio F, Boegh J. Characterizing a data model for software measurement. *Journal of Systems and Software* 2005; **74**(2):207–226.

22. Briand LC, Morasca S, Basili VR. An operational process for goal-driven definition of measures. *IEEE Transactions on Software Engineering* 2002; **28**(12):1106–1125.
23. Kitchenham B, Hughes RT, Linkman SC. Modeling software measurement data. *IEEE Transactions on Software Engineering* 2001; **27**(9):788–804.
24. Schalken JJP, van Vliet H. Measuring where it matters: Determining starting points for metrics collection. *Journal of Systems and Software* 2008; **81**(5):603–615. Available at: http://dx.doi.org/10.1016/j.jss.2007.07.041.
25. Yingxu W. The measurement theory for software engineering. *Canadian Conference on Electrical and Computer Engineering*, *IEEE CCECE 2003*, 2003.
26. Fenton N. Software measurement: A necessary scientific basis. *Transactions on Software Engineering* 1994; **20**(3):199–206.
27. Allen MJ, Yen WM. *Introduction to Measurement Theory*. Brooks/Cole: Monterey, CA, 1979.
28. Wans Y. The measurement theory for software engineering. *IEEE Canadian Conference on Electrical and Computer Engineering*. IEEE: New York NY, 2003; 1321–1324.
29. Umarji M, Emurian H. Acceptance issues in metrics program implementation. *Eleventh IEEE International Symposium Software Metrics*, 2005.
30. Jeffery R, Berry M. A framework for evaluation and prediction of metrics program success. *First International Software Metrics Symposium*. IEEE Computer Society: Los Alamitos CA, 1993; 28–39.
31. Gopal A, Krishnan MS, Mukhopadhyay T, Goldenson DR. Measurement programs in software development: Determinants of success. *IEEE Transactions on Software Engineering* 2002; **28**(9):863–875.
32. Kilpi T. Implementing a software metrics program at Nokia. *IEEE Software* 2001; **18**(6):72–77.
33. Niessink F, van Vliet H. Measurement program success factors revisited. *Information and Software Technology* 2001; **43**(10):617–628.
34. De Panfilis S, Kitchenham B, Morfuni N. Experiences introducing a measurement program. *Computer Standards and Interfaces* 1999; **21**(2):165–166.
35. Redmond JA, Ah-Chuen R. Software metrics–A user's perspective. *Journal of Systems and Software* 1990; **13**(2):97–110.
36. International Standard Organization. Systems engineering—System life cycle processes, *15288:2002*, 2002.
37. International Standard Organization. Information technology—Software product evaluation, *14598-1:1999*, 1999.
38. International Standard Organization and International Electrotechnical Commission. *ISO/IEC 9126—Software engineering—Product Quality Part 1*: *Quality Model*. International Standard Organization/International Electrotechnical Commission: Geneva, 2001.
39. Umarji M, Emurian H. *Acceptance Issues in Metrics Program Implementation*, 2005.
40. Umarji M, Emurian H. *Acceptance Issues in Metrics Program Implementation*, 2005.
41. Rational. Rational unified process documentation, 2000. Available from: www.rational.com [25 January 2003].
42. Software Engineering Institute. Software product lines, 2006. Available at: http://www.sei.cmu.edu/productlines/index.html [14 November 2006].
43. Briand LC, Morasca S, Basili VR. Property-based software engineering measurement. *IEEE Transactions on Software Engineering* 1996; **22**(1):68–87.
44. Zuse H. *A Framework of Software Measurement*. Walter de Gruyter: Berlin, New York NY, 1998; xxix, 755.
45. Fenton NE, Pfleeger SL. *Software Metrics*: *A Rigorous and Practical Approach* (2nd edn), vol. XII. International Thomson Computer Press: London, 1996; 638.
46. Kitchenham B, Pfleeger SL, Fenton N. Towards a framework for software measurement validation. *Transactions on Software Engineering* 1995; **21**(12):929–944.
47. De Panfilis S, Kitchenham B, Morfuni N. Experiences introducing a measurement program. *Information and Software Technology* 1997; **39**(11):745–754.
48. Hall T, Fenton N. Implementing effective software metric programs. *IEEE Software* 1997; **14**(2):55–65.
49. Brökcers A, Differding C, Threin G. The role of software process modelling in planning industrial measurement programs. *METRICS*. IEEE: New York NY, 1996.
50. Grady RB, Slack TV. Key lessons in achieving widespread inspection use. *IEEE Software* 1994; **11**(4):46–57.

AUTHORS' BIOGRAPHIES

Miroslaw Staron is an Associate Professor of Software Engineering in the Department of Applied IT at the University of Gothenburg, Sweden. He obtained his PhD in Software Engineering in 2005 from Blekinge Institute of Technology. His research interests are centered around industrial software engineering with emphasis on software metrics, measurement processes and model driven software engineering. Dr. Staron has been collaborating with Ericsson, Volvo Information Technology, Telelogic, Volvo Car Corporation or recently RUAG Space.

Wilhelm Meding MSc EE, has been working as Quality Manager for almost 10 years at Ericsson and previously in similar positions at Volvo Buses and Volvo Cars. He has been actively conducting research in the area of software metrics since August 2006 and has published 12 papers so far. He is currently working as Measurement Program Leader at Ericsson, and has also cooperation with other companies in the area of metrics.

Christer Nilsson MSc EE, has been working as Quality Manager at Ericsson since 2000. Before that he worked as process manager also in Ericsson, and as well as line manager for almost 5 years. Christer was involved in establishing the measurement program at Ericsson since 2003.