Abstract

Our society in general, and large organisations in particular, has become more and more dependent on software systems to be able to function efficiently. This ever increasing dependence on software systems leads to larger and more complex systems, which become increasingly expensive to develop and maintain. The majority of the costs of these software-intensive systems does not consist of hardware costs (the computers and network), but instead from the required labour effort to develop, implement, and improve the software systems.

With the increasing complexity of software systems, also the lack of predictability of developing that software increases. Projects that develop or maintain software systems, run a large risk to require more time and many than was anticipated and do not deliver exactly what was required.

A company can tackle aforesaid problems by providing software developers with better software tools to develop software, it can decrease the scope and complexity of a computer program (less is sometimes more), it can hire better software developers, but it can also improve also the work processes of both the software developers and their managers.

The fields software process improvement (software process improvement) and software methodology study the manner in which software developers (and their managers) tackle their work. Software methodology research develops complete frameworks, guidelines and step-by-step instructions to build or improve software in order to make this development of software more efficient. Software process improvement research, on the other hand, investigates how IT organisations can become more mature and it investigates the characteristics of mature IT organisations.

With most of the computer science research one can see at the end of a study if the objectives have been reached. Tests of the newly developed computer system can assess the strong and weak aspects of the solution with little difficulty: “a system works, or does not work”. The validity of the results of a software process improvement or software methodology study cannot be reliably judged at face value. The procedures and instructions must prove themselves, as soon as the IT professionals have mastered the new skills. Moreover, one cannot directly observe whether a method or improvement initiative has produced actual results. One observe whether the employees follow the new procedures, but this does not show if the procedures result in actual positive results. Every IT project is differently and that makes the comparison of projects and processes difficult.

As a consequence, there exists no consensus on acceptable development methods and procedures within the field of computer science and software engineering. Software
development methods succeed each other with the same relentless pace as the pace with which technological innovations succeed each other. Following the newest trends a company can choose for a flexible, light-weight method today and tomorrow it can choose to follow a more disciplined method that places more emphasis on documentation and standardised procedures. From time to time passionate discussion between IT professionals can be observed regarding the merits and pitfalls of software development methods and the strength and weaknesses of software process improvement frameworks.

An additional disadvantage, resulting from the uncertainties concerning the pay-off of software process improvement, is that sometimes companies stop a software process improvement program halfway. These programs are killed because the beneficial effects of SPI are not easy to observe, whereas the costs have a clear impact on the bottom-line of an organisation. In these situations software process improvement initiatives are killed, even though positive results might be achieved easily.

Given the large social need for good, reliable and payable software, it is undesirable that the efficiency of development methods and frameworks for software process improvement remains unknown.

As stated before, reviewing the efficiency of a software development method or a framework for software process improvement is completely different from typical computer science research. Instead of using an engineering take on the problems, one needs a perspective that is more akin to the social sciences. Identifying positive and negative factors, observed in experiments and pilot studies, one gradually obtains to a better picture of the workings of a method. Instead of falling back on on maths and logic (as a respectable computer scientist should), one has only assumptions and theories, which can corroborated or refuted by means of observations, measurements and statistics.

In my research I have looked at the impact of a software process improvement initiative and I have looked at consequences of the implementing a software development method. The ABN Amro Bank has allowed me to study the impact of implementing the Capability Maturity model (CMM) and the Dynamic Systems Development Method (DSDM). CMM is an accepted framework for software process improvement and DSDM is modern, light-weight software development method. Within the context of the Inspiration programme, the ABN Amro Bank has introduced these methods in its own organisation.

As a result of an improvement program that has been executed previously, the ABN Amro Bank possessed a database with detailed data on completed and running projects. On the basis of this project information the majority of my research has been carried out.

My research has been guided by the following research question:

• How can empirical data be used to strengthen a software process improvement initiative?

Empirical data can both be used to provide feedback about the current impact of a SPI program on the efficiency and effectiveness of SPI programs as well as to provide feedback on which IT processes still leave room for improvement. Therefore it is possible to split the original research question into the following two subquestions:

1. How can empirical data be used to get feedback about the effects of software process improvement?
2. How can empirical data be used to get feedback about the software development processes that still have room for improvement?

We have seen that when assessing the impact of software process improvement that one runs the risk of comparing apples with oranges. To escape these problems we can take two approaches: design comparison procedures that take the differences between projects into account (in other words to make apples and oranges comparable) or we can design new measurement devices that are sensitive to differences between the IT projects (in other words to make a separate scale for apples and a separate scale for oranges and make a sorting device to differentiate between the two of them.)

Therefore we can split the question “How can empirical data be used to get feedback about the effects of software process improvement?” again into the following two questions:

1.a How can data about software projects be made comparable?

1.b How can important aspects, for which no current metric exists, be quantified?

In addition to developing methodologies that help to evaluate software process improvement and that help to diagnose underperforming software development processes, we have also applied these methods in case studies at the ABN AMRO Bank. The outcomes of these case studies do not only provide an answer with respect to the efficacy the research method, but the case studies also provide insight into the effects of software process improvement. These results contribute to the answer of the following research question:

3. What are the quantitative effects of software process improvement on software development productivity?

One of the most important findings of my research lie in the field of statistics required to analyse the project data in a meaningful manner. Using the technique of Hierarchical Linear Models makes it possible to compare different IT projects, implemented in different departments of a company, without comparing apples with oranges. The usage of Box-Cox transformations, also a mathematical technique, can neutralise the disturbing influence of differences in size of the projects. More on these techniques can be found in chapter 3.

We have also examined the influence of the manner of communication between IT professionals and other employees. A new method of consulting stakeholders in the IT project has been introduced: facilitated workshops. Facilitated workshops are a technique of DSDM. We have found that facilitated workshops provide better results for large projects, whereas for smaller projects the old method leads to lower costs. The results of this research can be found in chapter 4.

Besides examining empirical data from existing sources, we have also created new sources during the research. A method has been developed to assess the impact of software process improvement when solutions are implemented on the basis of existing software modules (i.e. COTS systems). The new metric that is defined for these systems, so-called Infrastructure Effort Points, is described in chapter 7.

In addition we have looked at ways to obtain information from project evaluations. Evaluating every project after completion results in valuable data, which the organisation
could help increase performance in the future. The problem that remains is how to extract these helpful insights from hundreds of written evaluations. Using Grounded Theory, we have developed a method to extract these insights, which lurk in the depths of the project evaluation database. More concerning this method can be found in the chapters 5 and 6.

Chapter 8 describes how one can deal with different sources of information on a software development process. One might start with feedback scores stored in the project administration database, progress with written project evaluation reports, and conclude with data from a new source of data. This method of investigation and analysis deviates from the accepted methods used by social scientists to organise their experimentings. To allow a professional to deal with these ever-changing sources of data, a research framework has been developed that can be used to iteratively obtain ever-improving information on the software development process. This information can assist management in swiftly adjusting working processes when required.

Eventually the burning question remains: Did software process improvement improved matters at the ABN Amro Bank? The answer on this is positive: by climbing to CMM maturity level 2+ and by introducing DSDM, the ABN Amro Bank is able to work with 20% improved efficiency. This means that the ABN Amro Bank can make the same software for only 80% of the costs, when the new process and method is used.