Software Development Improvement with SFIM

René Krikhaar[1,2] and Martin Mermans[3]

[1] Vrije Universiteit Amsterdam, The Netherlands
[2] ICT NoviQ, The Netherlands
[3] Philips Medical Systems, The Netherlands
Rene.Krikhaar@ict.nl, Martin.Mermans@philips.com

Abstract. Most industries are challenging to increase productivity of software development. Often many process improvement activities are started with enthusiasm, unfortunately most of these are less successful than forecasted or improvements do not sustain for long. This paper presents the Seven Forces Improvement Method, SFIM, which claims to overcome unexpected disappointment in improvement results. SFIM is built upon different aspects that influence the success of software process improvements, such as culture, skills and organization. The method has been applied to improvement activities in a large software department for a number of years. The success of SFIM is compared with the compliance with the SFIM method. The paper shows that application of SFIM increases the success rate of software improvement activities in industry.

Keywords: Software Process Improvement, 7S model, Force Field Analysis, CMMI.

1 Introduction

Healthy organizations are continuously looking for ways to better serve the customer and improve their business in a never-ending cycle. Good is never good enough and what is good today may become unacceptable tomorrow because of the ever changing environment in which each organization operates. New technologies, merges with other companies, changing customer demands, employees with fresh ideas, there are numerous triggers for changing.

During more than a decade, most organizations use the Capability Maturity Model (CMM) [Hum89] to control and measure software improvements. CMM provides a framework consisting of Key Process Areas (KPA's) with a kind of recipe in which order the KPA's have to be developed. For some reason this model does not work properly in each organization. CMM mainly focuses on process, while other aspects play a role as well, e.g. the factors that play a role during realization. When changing an organization to establish an improvement, many more factors are influencing the success. In general, it takes a lot of effort and energy of many people to achieve a change.

A feeling of discomfort is rising in the software world. Only with great effort and difficulty organizations move towards higher maturity levels. Most of them never reach higher levels or when achieving a high level they have a large probability to fall back to a lower level. After 15 years of CMM only 18% of the organizations that report there CMM statuses to the SEI are at CMM level 4 or 5 [PMF05]. On the other hand, organizations at CMM level 5 still suffer problems that one might not expect at that maturity level such as projects that are still running late or providing unexpected results. The main objective of this paper is to provide means to realize sustainable software process improvements in an organization without restricting this to the domain of software process only. Here, we will answer the following questions:

– Why do software process improvements often not sustain in an organization?
– Why are these improvements slowly (or not at all) progressing in an organization?
– What are the influencing factors in software process improvements?

In section 2, we discuss various change management models to improve an organization. Two of the most influencing models, CMMI [CKS06] and 7S [PW82, PA81, WP80], are compared with each other in section 3. In section 4, we introduce the SFIM method, which encompasses good elements of multiple change management models. In section 5, we discuss SFIM in the context of an industrial case at Philips. In section 6, we discuss related work. Conclusions are drawn in section 7 including some suggestions for future work.

2 Change Management Models

In this section, we will discuss organizational models, which support a change in an organization. A lot of models have been published and still new models are developed in research and they are applied in industry [VBM06, SPI05, HHSE03]. Some models contain multiple viewpoints to address the organizational change. We will discuss a few of them: MOON [WW02] addresses cultural and human aspects, EFQM [HHH96] addresses quality in a full product lifecycle, CMMI [CKS06] identifies various process areas, the BAPO model [HKN+05] addresses four viewpoints, TOP is an integral development model [RHH06] and the 7S model [PW82] addresses seven points of view. The scope of operation of the above models ranges from culture to humans, from architecture to process and from skills to quality. We will shortly discuss these models.

Associates for corporate change developed the MOON-scan [WW02]. (MOON is a Dutch acronym, which means "Model Organizational Development Level"). MOON classifies the development of an organization into four increasing levels: *'reactive'*, *'active'*, *'proactive'* and *'top-performance'*. The model provides actions based upon human and cultural elements to move to a higher level. This model typically addresses the softer aspects of organizational improvement. MOON is applied in some (Dutch) industry, however not widely known. Interesting is that culture is one of the most important views in this model.

At the end of the eighties, the European Foundation for Quality Management developed the EFQM model as a joint activity of 14 European organizations [HHH96]. The EFQM model explicitly covers the 'soft' aspects of improvements such as leadership, strategy, policy and people management and sees them as important enablers for quality management in the whole product lifecycle. EFQM is applied in many European organizations. The model identifies a number of elements that have impact on quality. Metrics play an important role in EFQM and less attention is paid on causal analysis.

Software Engineering Institute developed the CMMI model [CKS06]. The Capability Maturity Model Integration covers the various capabilities of a development organization in 22 (CMMI version1.2) process areas. CMMI has a staged and a continuous model. The staged model distributes the process areas over 5 maturity levels thereby indicating the order in which process area to improve first. The continuous model puts all process areas on the same level and lets the organization decide which one to address first. One of the success factors of the staged model is the ability to compare maturity levels of organizations in an objective way. Level 5 organizations have the (software) development process completely under control and are able to improve in any direction they like. The industrial success of CMM in the software community resulted in the CMMI model for system development. CMMI is used all over the world in many different types of industry.

From engineering disciplines, we also know some models that put their activities in a broader scope. BAPO is a model developed at Philips Research to address Architecture (A) that fits in the context of Business (B), Organization (O) and Process (P) [HKN+05]. BAPO is based on several years of experience in developing architectures for large intensive software systems. Applying architecture without having in mind the business, organization and development processes will not make sense.

The TOP model identifies Technology, Organization and Process as dimensions in which system engineering is active [RH06]. The key idea behind the TOP model is that there should be a balance between Technology, Organization and Process for any development activity, for example architecting or configuration management. In case of introducing a new technology it should fit in the organization and process, which may require adaptation in any of the three dimensions.

The 7S organizational model of Peters and Waterman distinguishes the hard and soft aspects of an organization and divide them over 7 views with an "S" name [PW82]. The hard S's in the 7S model are:

- *Strategy*: the main objectives of an organization and the road to achieve them;
- *Structure*: the organization structure including the roles, hierarchy and coordination;
- *System*: the formal and informal rules

And the soft S's in the 7S model

- *Style*: the way one behaves and the way of cooperating;
- *Staff*: human resource matters like payment structure, education, motivation and behavior;
- *Skills*: most important and distinguishing skills;
- *Shared Values*: the shared values or the culture of an organization.

From the above model, the 7S model puts attention to the widest range of issues. With respect to the 7S model, the EFQM model also addresses many aspects, but does not explicitly include the organization (structure in 7S). EFQM is focusing on steering on some performance indicators instead of a thorough cause analysis. MOON only addresses culture (relates to *Shared Values* in 7S). CMMI mainly focuses on processes (relates to *System* in 7S) but also touches other aspects within the process areas, which are further, discussed in the next section. Software process improvement activities often mainly focus on process (as the term already indicates). We argue, as we have experienced during more than a decade of working on software improvements, that a single point of view will not result in success and/or will not sustain. In Fig. 1, we summarize the discussed models from a process perspective (gray areas related to more process focused elements).

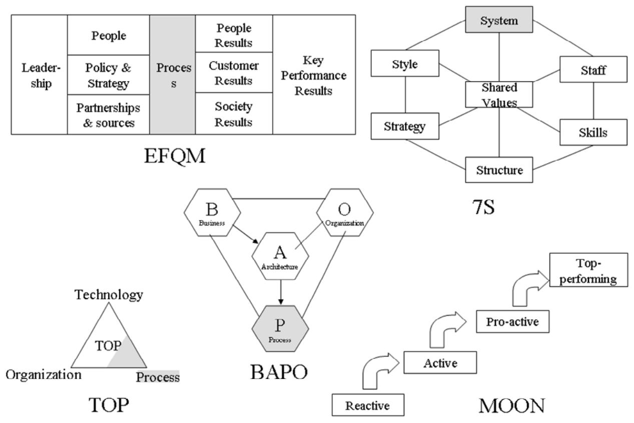

Fig. 1. Change Management Models (CMMI not included)

3 CMMI and 7S Model

In this section we compare CMMI with the 7S model. We show the communalities and differences that exist between the models.

CMMI distinguishes specific practices and generic practices. Specific practices are different for each of the process areas of CMMI. Generic practices however are the same for each process area and they determine the level of institutionalization that is achieved over each process. Common features organize the generic practices: *Commitment to Perform*, *Ability to Perform*, *Directing Implementation*, *and Verifying Implementation.* In Fig. 2, we show relations (dashed lines) between CMMI (ovals) and 7S (boxes).

The main focus of CMMI is on the *System* aspect and the prerequisites to make this work such as *skilled staffing* and organization. Aspects of the 7S model: *Strategy*, *Style* and *Shared Values* have to be in place but are not explicitly addressed in CMMI. Improvement actions guided by CMMI often fail despite apparent presence of all the prerequisites. Then the question arises: ‘why’ doesn’t it work.

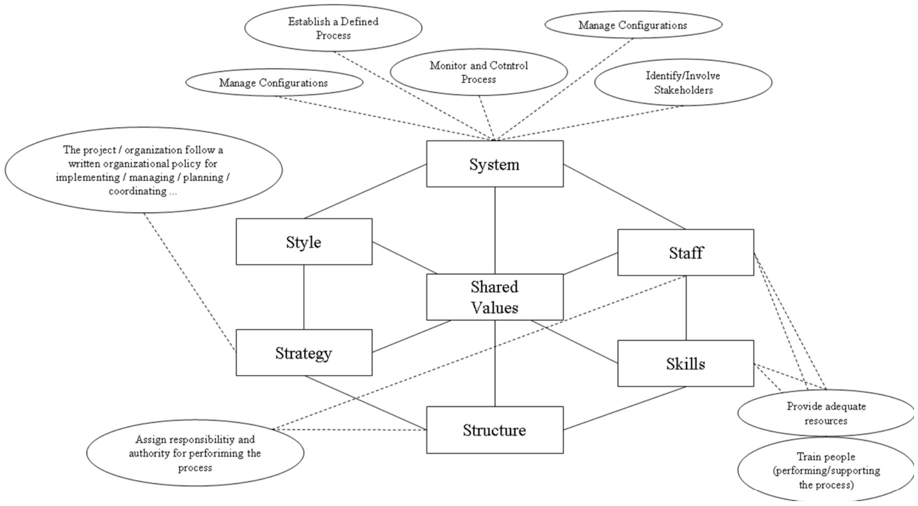
Establish a Defined Process
Manage Configurations
Monitor and Cotntrol Process
Manage Configurations
Identify/Involve Stakeholders
The project / organization follow a written organizational policy for implementing / managing / planning / coordinating ...
System
Style
Staff
Shared Values
Strategy
Skills
Structure
Assign responsibilitiy and authority for perforiming the process
Provide adequate resources
Train people (performing/supporting the process)

Fig. 2. Example of relations between CMMI and 7S model

The essence of the 7S model is that it specifically addresses the many different aspects of the organization and it emphasizes *harmonization*. Every aspect has to be aligned in the direction the organization wants to go. That might for instance be: move towards higher CMMI levels, improve quality of the product etc.

What we also notice is that CMMI provides Guidance at mainly the lower maturity levels. Even a Level 5 ‘continually improving process performance’ organization needs guidance. In the next section we introduce our method that assists at any level.

4 SFIM: Seven Forces Improvement Method

In this section the Seven Forces Improvement Method (SFIM) is introduced. The key principle of the SFIM method is to improve an organization’s performance in a sustainable way. This can be achieved by iteratively applying all SFIM steps for each Area of Attention. SFIM is heavily based upon existing successful methods and techniques in the field of to organizational change methods: the 7S model and Process Improvement models like CMMI.

SFIM starts with the identification of an improvement area. In organizations, it is often outside discussion which areas should be improved. In general, an improvement with a lot of management attention is a good starting point. For an improvement, which we call Area of Attention (AoA), the SFIM method proposes the following steps (see Fig. 3), inspired by the Deming Cycle (*Plan-Do-Check-Act*) [WD86].

1. Define the precise AoA objective. Be sure that the objective is SMART (Specific Measurable Ambitious, Realistic, Time driven), meaning Specific, Measurable, Ambitious, Realistic and Time-Driven.

2. Refine the AoA objective in terms of 7S. This means that the AoA objective is defined per S in the 7S model. This should be complete in the sense that it completely covers the AoA objective.
3. Analyze the 7S objectives with the Force Field Analysis technique [Lew51]. Force Field Analysis is a technique to identify the driving forces and restraining forces for a solution for a certain problem. The forces are identified for example in a brainstorm session. These techniques provide insight in all forces and opens possibilities to especially resolve the negative forces in an organization. This results in a T-table with the driving and restraining forces below the left respectively right part of the T.
4. Improve the organization; first focus on restraining forces. Starting with the hardest issues will pay back in the end. We experience that in many improvements restraining forces are ignored which results in non-sustainable results.

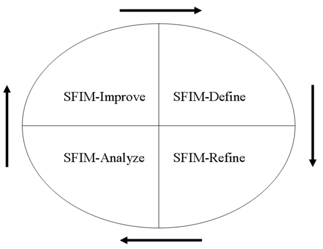

Fig. 3. Seven Forces Improvement Method

5 SFIM Case Study

We have applied the SFIM method in a development organization within Philips producing software intensive systems. In this section, we provide some characteristics, to be able to put the improvement activities as discussed in sections 6 in a better perspective.

The development organization operates worldwide at three sites. During the past ten years, the business has grown from about 100 systems to more than 500 per year. Time to market plays an important role so does Quality. The organization reached CMM level 2 in 1995, level 3 in 2002 (only software departments). Currently, we estimate that the organization is more or less at CMMI level 2 for all development departments.

About three hundred developers develop a highly innovative medical device. Developers have backgrounds in various disciplines from engineering to medicine. Innovative technologies are used to build from mechanical parts, hardware and software a proper working system. The organization is structured in a matrix, project management and line management. Line management is responsible for providing skilled resources and for the long-term quality of the system. Project management is taking care of running various projects in parallel. Each produces new functionality on time.

In this section, we describe the application of the SFIM method to the following Area's of Attention in the described organization: *Project Planning, Monitoring and Control* and *Software Quality*. Other Areas of Attention, with and without SFIM method, are briefly discussed in Section 5.3.

5.1 Project Planning, Monitoring and Control

Project Planning Monitoring and Control is an important process area of CMMI (level 2). It addresses issues such as estimations, risk management and progress tracking in order to control a project.

SFIM-Define: The organization wants to achieve mature Project Planning and Project Monitoring and Control processes at CMMI level 2 within 2 years. Of course, the main objective was to grow in process capability. Reaching CMMI levels is not a goal in itself.

SFIM-Refine: To achieve this the organization needs a strong and explicit project *structure*, with recognizable project management functions and roles and well-defined authorities and responsibilities. The *strategy* to use CMMI as guiding model has to be fully accepted and its consequences understood. Time was spent to convince leading people in the organization. This resulted in fewer discussions in the organization about the need for CMMI level 2. All people in a project management (operational) role or alike must be trained in estimating planning, tracking and managing projects. *(Skills*). The operational organizational axes should be sufficiently *staffed* with types of people that do belief in, and also intend to use a sound project planning and management approach. Management *style* should enforce starting projects or activities only with full commitment of those involved. The quality *Systems* of the organization should be adapted to support the project planning and project management activities. Procedures and manuals should provide the operational people with the right (level of) information to do their job well. The culture (*Shared Values*) of the organization must belief in the benefits of a sound project management approach and the added value of making plans.

SFIM-Analyze: The organization historically has a functional *structure* with many leading people on the functional axis of the organization and many hierarchical layers. There are several 'single' experts and they have to work on several projects in parallel therefore. The project structure also has a very hierarchical project structure with three or more layers: Project, segment and team layer. Project managers are organized in a Project Management Office. Segment and team leaders are located in the various departments of the development organization. Thus both the operational and functional axes are strong resulting in a struggle for power and people blaming each other for failures.

Within the organization the CMMI score is mentioned on the (one page) *strategy* as one of the results to achieve. The organization however tends to do many other improvement activities in parallel thereby not really following the essence of CMMI (Staged) which is to take one step at a time and begin at the beginning. Two forces: the one that tends to follow step by step approach of CMMI and the one that says do it all at the same time. The risk of the latter approach is that the effort that is put on changes on non-institutionalized processes is lost because the change did not sustain or did not bring the expected benefits.

Skills: Because of the technical background of most people that fulfill a project management role the calculations that are needed for creating schedules and the usage of the applicable tooling is no issue. Sometimes the ‘soft’ skills need extra training that is also provided. Standard courses on project management are provided to project managers and segment leaders. Team leaders require skills in technical and project management areas. People however who really have both skills tend to move onwards to either a fully technical function or either an operational or line function and stop being a team leader.

Lately the operational organization is *staffed* with more project managers, segment leaders and team leaders. Typical of an organization that is moving from CMMI level 1 to 2 is the change in staff types that is needed. A CMMI level 1 organization depends on its heroes. Types that do well in crises situations, led by gut feeling and people that do not care too much about careful planning and data collection because there is no time for that. Towards CMMI level 2 more of the latter is needed but even at higher maturity levels sometimes projects still run out of control and need to be rescued by the hero types. Opponents of the more structural working method that is required might say: you see we need leadership not administration to run a project. If this force becomes too strong this might result in putting the wrong type in the lead of every project and not only on those that really require this (the ones that ran out of control) and thus pulling the organization back to the CMMI L1 behavior.

We recognize two *styles*: ‘process’ style versus the ‘work hard; play hard’ style of management: forget about the planning just do what you can and work as hard as you can which of course does not help getting a proper planning and tracking process to work. Both styles are still present in the organization’. A way of objectively determining which style is winning would be to count how many projects start with real commitment about the targeted end dates of the project. A process style of leadership will only start projects with full commitment of those involved. The organization still tends towards the ‘work hard, play hard‘ style and thus does not benefit from reaching higher CMMI levels.

Systems: The organization had lots of Project Planning and Control procedures in place and reduced them to one small procedure fulfilling the needs of the CMMI and of the organization. The basic procedures and information are in place.

Shared Values: Within the organization it is really widely felt that plans should be based on thorough estimations and that they should be realistic before any commitment is given. On the opposite there are still some people who think that just working as hard as you can, will get the job done faster no matter what the outcome of the planning process is. Again these are two forces working in opposite directions.

The results of SFIM-analyze are visualized in Fig. 4. Dark gray means mainly restraining forces in this view; light gray means forces are neutral (or no issues) from this view and white boxes mean that there are mainly driving forces.

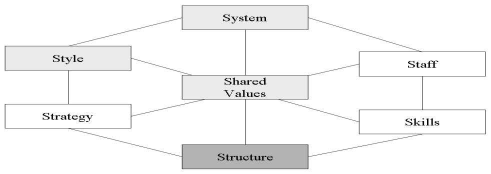

Fig. 4. SFIM-Analyze Project Planning, Monitoring and Control

SFIM-Improve: The organization has already made some changes to bring balance to the forces as mentioned above but still struggling with a few others. We considered the structure related issue the strongest force and there are changes happening already. *Structure*; the hierarchical line organization has been weakened already by merging two departments. Removing a hierarchical layer is considered but not done yet. *Strategy*; the organizations improvement is still focusing at many improvements at the same time. The awareness about the guidance that a model like CMMI can give is growing. *Skills*; training is now seen in the organization as an important driver for improving the output of people. *Staff*; 'Pushers' are lesser valued; what remains is the value for leadership in combination with good project administration keeping a discipline. *Style*; Management no longer pushes projects to the limit and accept that the impossible really cannot be done. Realism is the new key word. Projects do not start until it is clear that it can be done. *System* does not require any change. *Shared values*; Because of the change in management style people now dear to speak up in case they think a project or an assignment is not realistic.

5.2 Software Quality

This section Area of Attention that concerns improving the software quality in a sustainable way. In the mid 1990's the software departments of the studied organization were using coding standards consisting of several rules for the C language and a proprietary C dialect. Compliance with the coding rules was checked with some tools (QAC [PR06] and dedicated house-made scripts) and by manually reviewing the software. This way of working was successful for many years. However, after a change of product's operating system (from VMS to Windows) and a move to another programming language (from C to C++), during a few years less attention was paid to standards, so coding standards had to be introduced again. A lot of new people joined the organization that did not have the tradition of writing coding standard compliant code.

SFIM-Define. The main objective of this AoA is to introduce new coding standards in the software organization that have to be followed by the programmers. To be SMART in this sense, all the newly developed or modified software has to comply with the coding standards before software is checked in into to code archive. The rules only apply to newly developed code to have a realistic objective. Legacy code that does not comply with the rules is not taken into account. Only software that complies with the above objective may be considered in the daily build.

SFIM-Refine. The main Strategy behind the Software Quality objective is that it should be measurable. The software developers have to learn the coding rules (and rationale behind them) meaning that developers should have the right Skills for writing “error-free” software. For this it is of major importance to have coding rules that do not alter discussions (Style). In order to achieve this for a large group of developers, it also requires a Shared value of delivering a product with high quality software.

The organization, Structure, is built upon three dimensions: Line Management, Project Management and Technical Management. Technical Management address the technical aspects like what are the good rules to achieve high quality software. Line Management is responsible for quality in the long term, taking care of personnel with the appropriate skills and so on. Project Management is concerned to deliver a product on time having the specified functions with good quality. All supported by the right processes.

The above described organization structure requires to have the right people with the right responsibilities (Staff). The balance in the three concerns, Line, Project and Technique, has to be controlled by the right processes in the corresponding System. For example, legacy code should be treated in a different way than completely new developed components. In other words, coding rules should be applied with sense, sometimes rules have to be ignored and/or modified to serve the general objectives. Processes should be in place to change coding rules.

SFIM- Analyze: The organization is organized along three Structures: technical, line and project management. The leading technical people are willing to invest in software quality by means of coding standards; which is clearly supported by the line managers who are “enforcing” the employees to follow the coding rules. Project management is not always convinced about strict application of coding rules; however they share the overall goals of the organization.

In the overall business *Strategy*, the product’s quality is very important. High Quality Product is seen as one of the key selling points. Making software quality measurable by a coding standard and provide tools to automatically check these rules is supporting this business goal. Another positive impact, from strategic point of view, is that any good practice is upgraded to a consolidated (written) procedure. The organization is convinced that tools and processes walk hand-in-hand. Both tools and processes are embedded in the Integrated Development Environment (also illustrated in [BKP05]).

The organization is highly *skilled* in developing high tech products. A lot of software programmers have however less experience in programming languages. This becomes clear when certain coding rules are under discussion because not all programmers understand the rationale behind rules. Especially (at time of introduction) new software technology as C++, COM, C#, and .NET ask for more education. Highly experienced *staff* is hired to implement and support the coding rules and accompanying code-checking tools. The staff is supported by all kinds of monitoring tools to measure the current status of software quality, tuned for different “levels” of insight in coding standards. One software architect is championing the quality topic, supported by different engineers who support engineers at different levels (what is the rationale of this rule? Does this code comply with the rules? Can this code be checked-in into the code base?). All software engineers have an attitude to deliver high quality results. In the area of software engineering the style of the organization can be characterized as *laissez-faire*. Any initiative that makes apparently sense is accepted, any bad practice will not be removed by any measures of line management, but there may be forces from other engineers to remove it.

A *System* is operational for many years in the organization by means of a Quality Manual, describing the procedures in development. New processes, based on best practices, have been defined and engineers are really involved and aware of these procedures. In case software is submitted to the code base it may only pass when it did not introduce new violations in the code. For this purpose a quality database has been implemented containing the whole history of coding violation in all code files. Status is reported to quality assurance officers, line managers, software architects and software engineers at the corresponding level of interest. An important *shared value* in the organization is that they build the best system of the world. This value is supported by the experience of an earlier usage of coding rules in engineering. In fact, most people in the organization are convinced that quality can be made explicit and measured. On the other hand, the organization has to deal with the culture of discussing all decisions at any place at any time.

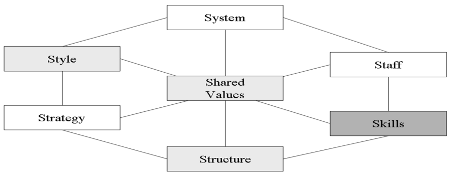

Fig. 5. SFIM-Analyze: Software Quality

SFIM-Improve: The above described S-in-7S describe the current status of the software quality AoA. In the past we have improved on a number of points which are briefly discussed below.

Structure: In various meetings and reports the value of coding standard compliant code is made clear to project management. Also the short term, meaning during the project's lifetime, value of coding rules is made clear by bad practices from the past and the ultimate impact on that project. This all resulted in some movement of project management to adhere to all coding rules in order to enlarge the final quality of the product. The restraining force at Structure level is diminishing.

Skills: Negative points of skills are partially solved by providing developer specialist courses (Microsoft Certified Sw engineering). Checking coding rules to signal the wrong programming attitude solves another part. Explaining the impact at the person's desk by expert helps to move into the right direction. We see that this costs a lot of time, but results are achieved. In the past all developers followed a coding standards course, which showed major acceptance. In the future we plan to organize such a course again.

Style: Management is involved in quality by providing quality performance indicators at a high level. A confidence factor is used to report results to managers [KJ03]. This resulted in improved attention (of managers) to software quality.
Shared Values: A very hard point is to stop discussions that do not contribute to the final result. Different organization wide workshops have been held to change this in the organization. We experienced that this requires a long breath.

In general, the introduction of coding standards was a success. We explicitly addressed the restraining forces at the different S's of the model: *Style*, *Structure* and *Shared Values*. Due to putting much attention to the *Style* aspect, management could be involved. This also motivated people to accept a change in *Sharing Values*. By embedding the processes in the integrated development environment, the organization was able to embed Software Quality in a *structure*. More details are described in [KJ03].

5.3 Other Areas of Attention

Many other improvement activities took place during the last decade in this organization. Many improvement activities were successful which resulted in a CMM level 3 in 2002. Nevertheless, the organization has many difficulties to keep this maturity level and is working on getting CMMI level 2 for all development departments (enlarging the scope). To illustrate SFIM, we discuss some major improvement activities taking place during the last 6 years. Note that this set is not representative.

System Testing (SFIM): the main objective of this AoA was to increase the performance of system testing, resulting in higher quality systems. The *Mean-Time-Between-Crashes* metric was defined to indicate the quality level of delivered systems. All 7 elements in SFIM were addressed to increase the level of success. Much attention was put on having a clear *strategy* for system testing, which was seen as the most restraining force in this AoA. Other elements like *Structure* (a separate group in the organization exists to address system testing), *Staff* (extra roles were defined, e.g. a project's test coordinator), *Skills* (people did follow test management courses, a dedicated test method was introduced) and *Systems* (processes for integration tests, alpha tests and beta tests are in place and well deployed) were in right shape and pointing to the right direction. During the period of improvement less attention was paid to *Strategy* change (although identified by SFIM). The main reason was that it is hard to change this and more attention was paid to the other six elements. The improvement activity was not successful in the sense that it did not result in a sustainable way of testing systems. A lesson learned was that one should stick to the most restraining forces to solve; otherwise the activity is doomed to fail.

Process Database (no SFIM): the main objective of this AoA was to introduce a process database containing all kind of figures of projects in order to achieve better results in future projects. For this AoA, the SFIM method was not applied. Much attention was paid on the technology (tool) and processes to fill the process database. At the end, the process database was filled for a number of projects and hardly anyone was using the content of this process data for new projects ("my project is completely different from the previous ones”). After a few years, filling the process database was not performed anymore, of course due to lack of a need for it. We have analyzed the failure of this AoA. We may conclude that the *Systems* were addressed well, but there was no *Shared Value* on this topic. Each project manager had his own new ideas when a project started. Furthermore, *Staffing* failed, after a person left the organization, there was no drive for replacement. The *structure* of the organization hampered because the people who had to cooperate were distributed over different organizational units.

General Structural System View (no SFIM): the main objective of this AoA was to define a single structural view that is used by the different disciplines in development (software, mechanics and hardware). After a number of attempts, this improvement slowly progressed. The main reason for slow progress was the absence of a *Shared Value*, but also the right *Structure*, people were not aware of responsibilities in defining and using this structure.

Peer reviews (no SFIM): the main objective of this AoA was to structurally introduce peer reviews of source code and documents. A very enthusiastic person was introducing peer reviews, but there was no (technical) champion in the organization to get the job done (*Staff*). A lot of bureaucracy was introduced which resulted in resistance from engineers. For example, after each review, minutes had to be written that did not introduce new facts. So the process (*Systems*) was not well thought off.

Use Case Introduction (no SFIM): the main objective of this AoA was to introduce a method to close the gap between the user (represented by the marketing department) and development. For this the Use Case approach from the object-oriented community was chosen and adapted. The Use Case Introduction failed after a few years of partial application. The main reason was that the marketing department was not *skilled* to understand the Use Cases (which stem from software community). Furthermore, the processes were not well defined and staffing was only properly addressed by development (and not at marketing).

5.4 SFIM Conclusions

We have applied SFIM on three improvement activities, two of them more successful than the third one. Non-SFIM improvement activities took place with different success rate. In this chapter we discussed some improvement activities that could not sustain in the organization. Unfortunately, we are not able to present exact figures of success and failures of improvement activities.

We conclude from this chapter:

– SFIM should be applied without excluding any of the seven elements. The System Testing AoA showed that relaxing the *Strategy* element resulted in a failure.
– In retrospect, we have analyzed a number of improvement activities in which we could easily identify, in terms of SFIM, which elements were underestimated during the organizational change. This provides us evidence for the SFIM method.

6 Related Work

In [WR94] the authors describe a software improvement program as performed at Schlumberger in the nineties. They demonstrate various elements that have impact on this program. The People-Process-Technology triangle is followed to describe the various points of concern when improving. The P-P-T triangle maps more or less on the earlier described TOP model. The paper states "*Software process improvement must involve the other parts of business with which software interacts, namely marketing, hardware development, sales, manufacturing, etc.*". It shows the importance to broaden the scope of software improvements. The authors indicate that cultural change may be required however hard to achieve. *Shared Values* in the SFIM method addresses the culture aspect. Looking at the discussed topics in this paper, we see that in this industrial improvement program 5 or 6 aspects of the 7S model are addressed although not explicitly referred to by the authors.

Siemens poses that a focus on quality lead to reductions in cycle-time, effort, and costs and thus to business benefit [AP03]. Various activities took place to increase the industrial strength of the company. Process, Quality and Test, Organization and People, Architecture and Agility are discussed in this paper. All these improvement activities lead to more focus on innovation (more time available for) and the long-term success of the company. The paper describes some AoA's that were addressed. In each of them a number of the 7S elements were discussed but not explicitly identified.

Dyba [Dyba05] investigated the effect of factors that influence SPI success. The impact of the factors is determined by means questionnaires send to about 120 managers. The paper concludes that organizational culture is important rather than entirely SW engineering tools and techniques. [Dyba05] did not address issues such as organizational structure nor does it provide means to determine on forehand which aspect to focus on primarily.

In [BMPZ02] 13 lessons learned are discussed during the 3 phases of their existence: emphasis is put on data collection, a learning organization and change. The lessons are grouped in: 1) Need for collecting project data, 2) Need for management buy-in on the process, 3) Need for a focused research agenda and 4) Need for continued staff support (on data collection). They provide some prerequisites for process improvement as also SFIM proposes. The four lessons address *Style*, *System* and *Staff* in the 7S model.

7 Conclusions

In this paper we have introduced the Seven Forces Improvement Method that is based on the 7S model, Force Field Analysis techniques and best practices in software development improvement projects in industry. We have applied SFIM in a large software department for three major improvement activities. Furthermore, we have analyzed some less successful improvement activities from the past having SFIM in mind to learn from it.

The major contribution of this paper is that we show that a wide range of factors play a role in improving software development in a complex organization. Furthermore, we proposed a technique to balance these various factors and how to prevent you, as improver, from spoiling time on the wrong actions. We experienced that a restraining force (e.g. the culture or structure of an organization is not in line with the desired change) is more influencing than pushing activities in other areas (e.g. only defining process guidelines in a quality system). This means that "easiest things first" is not working, but more "hardest things first".

From this we conclude, that software development improvements will not sustain in an organization when forces, related to one or more of the 7S-es, are pushing in the wrong direction (a restraining force). Meaning that, restraining forces have to be tackled first to be successful. Smoldering restraining forces in the organization may result in a fall back of an initially successful process improvement. In SFIM we identified that *Strategy*, *Structure, System, Style, Staff, Skills* and *Shared Values* are influencing the performance of software process improvements.

Future work: To increase the confidence in SFIM, the method should be applied in other industries as well (we are challenging readers of this paper and we are willing to discuss issues). Thorough analysis of more improvement stories (success or failure) from a SFIM perspective may help to improve the method itself. Elaboration of the SFIM method could also be achieved by more detailing the four SFIM steps.

Acknowledgement. We want to thank Hans van Vliet for his input on an earlier version of this paper. This work was supported by ITEA SERIOUS (if04032).

References

[Hum89] Humphrey, W.S.: Managing the Software Process. Addison-Wesley Publishing, London (1989)

[PMF05] Carnegie Mellon Software Engineering Institute, Process Maturity Profile, Software CMM, 2005 Mid-Year Update (2005)

[CKS06] Chrissis, M.B., Konrad, M., Shrum, S.: CMMI 2 edn.(R): Guidelines for Process Integration and Product Improvement, The SEI Series in Software Engineering (2006)

[PW82] Peters, T., Waterman, R.: In Search of Excellence. Harper & Row, New York, London (1982)

[PA81] Pascale, R., Athos, A.: The Art of Japanese Management. Penguin Books, London (1981)

[WP80] Waterman Jr., R., Peters, T., Phillips, J.R.: Structure Is Not Organisation in Business Horizons (1980)

[VBM06] http://www.valuebasedmanagement.net/ visited (December 31, 2006)

[SPI05] SPIder working group integral SPI strategies, Modellen: wanneer wat (Dutch) (2005)

[HHSE03] ten Have, S.W., Stevens, F., van der Elst, M.: Key Management Models. Prentice Hall, Englewood Cliffs (2003)

[WW02] Wanrooij, W.: Corporate Change: de weg naar topprestaties (Dutch), Scriptum (2002)

[HHH96] Hardjono, T.W., ten Have, S.W.: The European Way to Excellence: How 35 European Manufacturing, Public and Srvice Organizations Make Use of Waulit Management, DGIII Industry European Commission (1996)

[HKN+05] Hofmeister, C., Kruchten, P., Nord, R., Obbink, H., Ran, A., America, P.: Generalizing a Model of Software Architecture Design from Five Industrial Approaches, Succeedings of WICSA (2005)

[RH06] Reek, E., Hulshout, A.: TOP: Technology Organization and Process (company report) (2006)

[WD86] Walton, M., Deming, W.E.: The Deming Management Method, New York, Dodd (1986)

[Lew51] Lewin, K.: Field Theory in Social Science. Harper Row, New York (1951)

[PR06] http://www.programmingresearch.com visited (December 30, 2006)

[BKP05] Bril, R.J., Krikhaar, R.L., Postma, A.: Architectural Support in Industry: a Reflection using C-POSH, Journal of Software Maintenance: Research and Practice, 17(1) (2005)

[KJ03] en Krikhaar, R., Jansen, P. In zeven stappen naar een code standaard (Dutch). Informatie (2003)

[WR94] Wohlwend, H., Rosenbaum, S.: Schlumberger's Software Improvement Program, IEEE Transactions on Software Engineering, 20(11) (1994)

[AP03 Achatz, R., Paulisch, F.: Industrial Strength Software and Quality: Software and Engineering at Siemens. In: Proc. International Conference on Quality Software (2003)

[Dyba05] Dyba, T.: An emprical Investigation of the Key Factors for Success in Software Process Improvement, IEEE Transactions on Software Engineering, 31(5) (2005)

[BMPZ02] Basili, V.R., McGarry, F.E., Pajerski, R., Zelkowitz, M.V.: Lessons learned from 25 years of process improvement: The rise and fall of the NASA Software Engineering Laboratory, ICSE (2002)