SPI – Why isn’t it more used?

Tor Stålhane, Ph.D.,
Kari Juul Wedde, M.Sc.,
SINTEF, Trondheim, Norway

Introduction

SINTEF has for some years offered assistance to industry in Norway and Sweden in the area of SPI – Software Process Improvement. This assistance is based on two projects, namely:

- The QIS project – Quality Improvement in Scandinavia. This is an EU sponsored project that shall give help and assistance to PIEs in Norway and Sweden and market the concept of SPI to the software industry in these two countries.
- The SPIQ program – Software Process Improvement for better Quality. This is a national Norwegian program that is partly a national ESSI type project and partly a co-operation between academia and Norwegian industry to increase the use of process improvement methods in Norwegian software industry.

From the start we thought that it would be an easy job to sell SPI to the industry. However, this turned out to be a rather optimistic assumption. Quite a lot of companies did not believe that SPI would solve their problems. This was a general attitude and not limited to management or developers only.

The rest of this paper describes our findings related to why a large part of the industry rejected the very concept of SPI. We start by describing the situation as we see it and the reasons that we have found, both among management and among developers. Then we sketch some suggestions for solutions in order to improve the situation before we end the paper with some conclusions.
The current situation – and how we got there

Where we got our data

We base our conclusions on several data sets. The data are mostly qualitative and not collected in order to throw light over the problem at hand. This is a consequence of the fact that we got a clear picture of the sordid situation quite late in the process. The data sources are as follows:

- A survey of Norwegian companies - used to discover which factors the companies considered important for improvement [1].
- A survey of European companies - used to discover why companies do not do SPI and what they do instead in order to improve [9].
- A set of interviews with quality assurance (QA) personnel and developers in five Norwegian companies on the role of written procedures as a vehicle for organizational learning [2].
- Data collected in two Norwegian companies pertaining to where they are and where they want to be in the near future – also called a gap analyses. This includes, among other things, also a set of arguments for and against SPI.
- Discussions with developers in Norwegian and Swedish industry on SPI.

All this information gives clues to the problem, even though none of them present the full picture. Thus, instead of a data analyses – statistical or otherwise – we present a solution to a jigsaw puzzle consisting of all the small pieces of information that have been given to us.

The way we are – and why

Everything has a history, so also the current status of QA, SPI and software development. In the seventies, software development was a rather chaotic affair. The programmers considered themselves artisans – or even artist – that did not produce, but instead create. Project control, quality control and any other control was mostly absent: The project was finished when it was finished and the customers stood in awe and watched the blinking lights on the computer.

The managers did not think this was a good state for the business and project control seemed to be a solution that appeared just in time. When it was sanctified by ISO and given an official number, the managers were all too ready to jump on the bandwagon. ISO 9000 was not a bad idea; it was just that its introduction was badly timed. Since the management layer of the companies were screaming for control over hundreds of cowboy programmers, ISO 9000 lead to an intense focus on control. As a result of this rather myopic interpretation of the standard, two things happened:

- The management focused on control, since ISO 9000 promised to give them just that, and the QA department followed suite.
- The most important part of ISO 9001 – section 4.14 that focused on process improvement - was lost in the introduction. The fact that it in many cases was sold as statistical process control (SPC) did not help much either.
SPI – why isn’t it more used?

Thus, the big chance to introduce real quality improvement and SPI into software development was lost. Instead we could hear ghastly statements from QA managers like “The QA department’s most important job is to give the managers full control over the projects”.

Some data – and our interpretation

In order to see if there is a way of adding more details to the picture, we have looked at two contrasts that we hope will shed light on the status concerning SPI. These contrasts are small versus large companies and the way the QA department looks at SPI versus SPI as seen from the developers.

Small versus large companies

There seems to be a larger share of the big companies that do SPI than there is of the small companies. There are at least two explanations for that:

Either: SPI will lead to changes. It is thus a risk and larger companies can better afford to take such a risk. The main reason for this is that they have more resources. In addition, SPI will take time and larger companies usually have a longer planning horizon.

Or: Larger companies need SPI, small companies don’t. The reason for this is that small companies are more flexible, is less dependent on written procedures and has more efficient internal communication.

When considering SPI and small companies, there are also two other areas that should be taken into considerations. These are:

- Small companies live in an ever-changing world and they have little or no influence over the way their environment changes. Thus, anything that depends on a stable environment has little or no value for a small company. Since SPI to a large extent is a synthesis of previous experience, SPI becomes at least partly irrelevant.
- SPI will require a certain minimum of documented routines. However, written procedures and routines have a tendency to petrify the organization and thus remove some of the flexibility that a small company needs in order to compete efficiently.

Thus, small companies focus on being innovative and flexible, instead of using what they see as a large documentation overhead that they fear will take away all of their competitive advantages. This stance is, however, not limited to small companies. A manager from a large company that produces mobile telephones have stated that they are a CMM level 1 [7] company and intend to stay there for the foreseeable future. This does not mean that they do not do SPI – on the contrary, they do quite a lot of it. The point is that they look for ways to improve themselves that do not destroy
SPI – why isn’t it more used?

creativity and innovation.

**QA department versus the developers**

One of the observations that is rather consistent in all our material relates to the on-going discussion of what kind of information and knowledge that is important for a company. The programmers focus on how to write code, while the QA department focuses on rules and regulations. This was brought clearly forward when developers and QA personnel were asked about the value of procedures as a vehicle for knowledge dissemination in the company. The results are presented in Table 1.

<table>
<thead>
<tr>
<th>Personnel</th>
<th>Agree</th>
<th>Undecided</th>
<th>Disagree</th>
</tr>
</thead>
<tbody>
<tr>
<td>QA managers</td>
<td>64%</td>
<td>18%</td>
<td>18%</td>
</tr>
<tr>
<td>Developers</td>
<td>0%</td>
<td>50%</td>
<td>50%</td>
</tr>
</tbody>
</table>

Table 1 Procedures as a vehicle for knowledge transfer

The message is loud and clear – QA managers think that procedures, document forms etc is important knowledge, while the developers think it is not. This attitude also surfaces in another question, related to the use or no use of the company’s written procedures. The developers were asked about the amount of control and the amount of use of the company’s procedures. The results are summarized in Table 2.

<table>
<thead>
<tr>
<th>Degree of control</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>36%</td>
<td>55%</td>
<td>9%</td>
</tr>
<tr>
<td>Medium</td>
<td>33%</td>
<td>34%</td>
<td>33%</td>
</tr>
<tr>
<td>High</td>
<td>0%</td>
<td>56%</td>
<td>44%</td>
</tr>
</tbody>
</table>

Table 2 Use of company procedures

It is straightforward to see that the degree of control has a strong influence on the degree of use of the company’s procedures. A rather sad conclusion is that the developers do not use the procedures because they find them useful but because they get in trouble if they do not use them. A collateral to this conclusion is that the developers do not see the procedures as particularly useful. This result is consistent with the conclusions from the previous question.

If written procedures is not the answer, what is? We asked both the developers and the QA managers which alternative vehicles they would consider. As shown in Table 3, the degree of agreement between the two groups is good. We see that they agree on the most important items, which are – in order of importance:

- Experience bases – saving experience for later reuse in new projects. The experience base can be at any level of formalization – from a paper archive of experience reports to a complete Experience Factory (EF).
SPI – why isn’t it more used?

- Socializing – meeting at the coffee machine, in the cantina or somewhere else in order to exchange experiences, ideas, insights and complaints.
- Study groups – a more formalized version of “Socializing”.
- Reports and documentation – a less formalised version of an experience base.

These four areas include 70% and 71% respectively of all answers and show an important conclusion:

**People want to learn from others’ experiences by reading their reports and by interacting with them – either formally in a study group or informally through socializing. Written procedures are not the answer.**

<table>
<thead>
<tr>
<th>Vehicle</th>
<th>Developers</th>
<th>QA managers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number</td>
<td>Rank</td>
</tr>
<tr>
<td>Study groups</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Experience bases</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>Courses</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Socializing</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Discussions</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Teambuilding</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Individual studies</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Reports, documentation</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Job rotation</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Multidisciplinary group</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Working meetings</td>
<td>1</td>
<td>9.5</td>
</tr>
<tr>
<td>Include consultants</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 3 Vehicles for knowledge transfer

**Why they don’t think SPI is a good idea**

**Those that don’t do SPI**

The companies that participated in the ESPINODE industrial survey and answered that they didn’t do SPI were asked three important questions: Why don’t you do SPI, what do you do to improve and what kind of help would you need to improve yourself? Since the questions were open-ended, we received quite a lot of different answers. We have selected categories, starting with the most popular one and then kept on including categories of decreasing popularity until we have reached at least 70% of all respondents. The majority of the answers ended up in one of a small number of categories, as shown in Table 4.

When we look at the results, there are two things that catch the eye: There is a strong agreement among the companies on why they do not need to do SPI and on what they do instead. Only two categories are needed in order to include 70% of all responders. The agreement is less when we come to the help needed. This is as expected – it is
always more easy to diagnose a problem than to agree on a cure.
The fact that many companies claim that they do not have the necessary resources
available indicates that they do not believe that SPI is a sound business proposition.
When they say that they do not have enough resources to do SPI, they are really
saying that they do not believe that they will make money out of it. They lack a cost /
benefit analyses that will show how SPI will help their business. The problem may
partly be connected to the way QA has been introduced. QA was supposed to
increase efficiency and when the results did not materialize, this also contaminated
the idea of SPI which at least partly has been sold as a QA technique.

<table>
<thead>
<tr>
<th>Question</th>
<th>Category</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why don’t you do SPI?</td>
<td>Do not have the resources available</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td>SPI costs too much</td>
<td>30%</td>
</tr>
<tr>
<td>What do you do to improve (instead of SPI)?</td>
<td>Improve and increase our know-how</td>
<td>52%</td>
</tr>
<tr>
<td></td>
<td>Introduce new tools and methods</td>
<td>29%</td>
</tr>
<tr>
<td>What kind of help do you need?</td>
<td>Consultancy</td>
<td>24%</td>
</tr>
<tr>
<td></td>
<td>Networking</td>
<td>24%</td>
</tr>
<tr>
<td></td>
<td>Formative help on the concept of SPI</td>
<td>18%</td>
</tr>
<tr>
<td></td>
<td>Financial help</td>
<td>14%</td>
</tr>
</tbody>
</table>

Table 4 Those that don’t do SPI

**What do they need?**

In order to get an understanding of the companies’ needs, we asked them what they
considered their most important challenge for the next two to three years. The
responses are summed up in Table 5. By and large, all companies – whether they do
SPI or not - agree on their most important challenges for the future. Note, however,
that while the SPI companies talk about improving efficiency, the others talk about
improving innovation. Thus, it seems that the companies that do SPI focus on process
efficiency while the rest focus on being innovative. Many companies seem to believe
that SPI is a barrier for innovation – or at least that it doesn’t help. Being innovative
is much more critical for small companies, which depends on innovation in order to
keep their competitive edge.

In our opinion, the fear that SPI will destroy or hinder creativity stems mainly from
bad experiences with QA that has degenerated to a rigid control regime plus an
enormous amount of documents that nobody outside the QA department really needs.
To get out of this sordid state, we need to disconnect SPI from QA, at least until QA
move from being a controlling bureaucracy to being a service to the development
projects. In addition, we need to develop a way of basing SPI on a sound business
cost / benefit analysis so that the companies see that SPI really help them make
money.

<table>
<thead>
<tr>
<th>Most important challenge for the next years</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Companies that do SPI</td>
<td>Improve development efficiency</td>
</tr>
<tr>
<td></td>
<td>Increase customer satisfaction</td>
</tr>
<tr>
<td></td>
<td>Promote company growth</td>
</tr>
<tr>
<td>Companies that do not do SPI</td>
<td>Increase customer satisfaction</td>
</tr>
</tbody>
</table>
SPI – why isn’t it more used?

<table>
<thead>
<tr>
<th>Be more innovative</th>
<th>28%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promote company growth</td>
<td>19%</td>
</tr>
</tbody>
</table>

Table 5 Company needs

The SPC inheritance

The concept of SPI stems from production industry. Their concept is simple and straightforward to apply:

1. Get the process under control by identifying and removing sources of variation.
2. Run controlled experiments where one or a few process parameters are varied in a controlled manner and collect data.
3. Analyze the collected data in order to identify the best candidate for improvement and implement the necessary changes.
4. Collect data to verify that the changes have had the expected impact.
5. Repeat the process from step 1.

Several persons – at some point in time event including one of the authors of this paper – thought that this improvement paradigm could also be used for software. Some still believe so. Level 4 and 5 of CMM are almost pure SPC both in concept and attempts. Software developers by and large consider SPC to be close to irrelevant for software development and they are probably right. The most important reasons for this are that SPC assumes:

- A technologically stable process that is repeated a large number of times - typically several hundred times. This will probably never be the case for a software development process.
- That the variation factors can be brought under statistical control. Given that a large part of the variation in a software development process stems from the developers, this is a rather unrealistic assumption.

Thus, SPC is out. To the degree that SPI pushes the use of SPC or is in any way connected with SPC, it will score low with software developers, and probably also with software managers. QA managers, however, too often think that SPC is a great idea – also for software.

The problems - as seen from the manager

The main problem with SPI as seen from the manager’s side is that it consumes resources. This was given as the main reason by 40% of all managers asked. SPI is seen as an activity that takes resources – people and money - away from development which often is too late already due to lack of people. Even if managers believe that SPI will enable them to run the project within time and budget in the end, the end is too far away to interest them. A possible solution would be to hire more people but there are at least two obstacles to this solution:

- Good developers is a scarce commodity.
Hiring more people will cost money and the improvement promised by SPI will only materialize somewhere in the future. It is a general experience that a sure expenditure today will usually outweigh an unsure income somewhere in the future.
An extra problem – especially as seen from middle management - is that SPI can only succeed by involving the developers. This is seen as a threat to middle management since involving the developers mean to give them more power – power that usually is taken away from middle managers.

**The problems - as seen form the developers**

Since SPI is marketed as a part of QA, developers automatically associates it all the misguided attempts to implement QA as a document driven control regime. Both “control” and “document” are terms that score low on the developers’ scale of enthusiasm. They feel that they are hindered in being creative and innovative – which is what development is really all about.

In addition to all this, QA – and thus SPI – is seen as a source of a never-ending steam of procedures for this and routines for that in addition to an insatiable lust for more documents. The problem is not the documents, but the justified suspicion that these documents do not add to the product’s value – they are just there to satisfy somebody’s need for control and control translates into “I don’t trust you”.

A survey of working relation in the North Sea [8] illustrates this: “The Norwegians coursed the way they were forced to work. They were forced into what they felt were idiotic QA procedures. They were proud industrial workers who had to follow routines and a documentation regime that the on-shore industry had outgrown a long time ago. They had to wait for a working permit even for small jobs and the whole place was crowded with managers on all levels, with certificates and detailed working procedures”. In the offshore area this lead to eight years of continuos strikes and worker unrest. Software developers have not come that far – yet.

**QA managers versus developers**

One of the points that are driven home to us quite often is that the world looks quite different depending on whether you see SPI from the QA manager or from the developers’ side. Some times one wonders: Do they really work for the same company or do they just share a mail address?

Some times the differences are important – for instance that developers and QA managers agreed that developer involvement was important for the success of SPI, while they did not agree on what developer involvement really meant. For developers and development managers, developer participation consisted of the following factors:

- A strong influence on how their work is done.
- Participation in the improvement work by making improvement proposals.
- Participation in the development of procedures for development work
- Participation when the company sets improvement goals.
- Responsibility for part of the improvement work.
- Being regularly informed of the status on the company’s improvement work
- Discuss the principles for SPI regularly discussed with the management.
However, when we asked the QA managers, they produced the following, rather short list of participation factors:

- Participation in the development of procedures for development work
- Responsibility for part of the improvement work.

What is most revealing in the factors identified by the QA mangers is what is not included. These are factors such as participation by making improvement proposals, by setting improvement goals, and by being informed of the status in the improvement work. It is as if the QA managers say that SPI is the responsibility of the QA department. The developers should stand by and watch.

This problem was highlighted when we checked the two groups’ position regarding feedback sessions. It turned out that the QA managers considered feedback sessions to be sessions where the developers were allowed to see the data. The development managers, on the other hand, considered feedback session to be a part of data collection and analyses – which make sense. Our experience is that the developers’ participation in feedback session is absolutely essential for data analyses and interpretation.

**Where do we go from here – out of the quagmire**

Even though the situation in many ways is rather bad, there are hopes for the future. Firstly, there are companies that have succeeded in their SPI work. These companies have been able to make the QA department and the software development department co-operate in a way that contributes to increased quality for the customer and increased efficiency for the company. In these companies the developers are involved in the improvement work and the management supports it. Secondly, the SPI field has matured over the years, and we have today a better understanding of how to implement SPI so that it caters to the needs of software development and the needs of each specific company.

The problems discussed in the first part of this paper are diverse. This diversity could have resulted in a large set of solutions. We think, however, that there are a few areas that dominate, and we will therefore focus on solutions to these most important obstacles to SPI.

**SPI and QA**

W.S. Humprey introduced in [5] what he called Software Engineering Process Groups (SEPG's) and these groups should among other things, take care of process improvement activities. In the discussions of why these groups were needed; why not let the QA department take this job, he stated ”At least from a parochial development viewpoint, they (QA) are thus often viewed as the enemy, or at least as an unnecessary annoyance”. When we first read the book, we thought this was an American problem and not valid for the European way of working. Unfortunately, we were wrong. Our data and later experience has shown that this is a relevant statement, at least for part of the European software industry. The SPI community therefore has two alternatives:
SPI – why isn’t it more used?

Either: Keep away from the QA department as best they can.
Or: Be an integrated part of the QA activities.

The first alternative may work for some time, but is not a solution for the future. The second alternative is therefore our choice. The main reason for this is that SPI and QA overlap. QA shall verify that the employees have applied their expertise properly. ISO 9000 requires continuous improvement, etc. SPI is about learning to work smarter - changing the process to get an environment in which people can do a better job. The obstacles to SPI are thus not the tasks to be done or a lack of common interests. The problems stem from the implementation. There are two main areas where today’s implementation has led to conflicts of interest between QA and SPI.

- **Organisation**
  QA has traditionally been organised as a separate department - at least in large companies. This organisation has often resulted in little contact and co-operation between QA and the developers. SPI, on the other side, depends on close co-operation with the development department and the developers – preferably on a daily basis. SPI can not succeed without active participation from the employees.

- **Control versus help**
  The separation of QA from the developers can lead to a situation where QA become a part of the management - controlling the developers instead of helping them. SPI is about helping people to do a better job.

The solution to these problems is simple - at least in principle. Establish a common platform for SPI and QA where they can work together toward a common goal - looking for new improvement opportunities. In this setting, QA activities must be in-project activities and QA people should take part in the daily work of the development projects. This is the only way that QA people can get hands-on experience and thus learn about the developers’ needs - problems to be solved and opportunities to grab.

**Make SPI and QA unify their effort, looking for improvement opportunities.**

**SPI and software development**

In order to succeed in SPI, we have to understand and accept the nature of software development. We have to understand that software development is a creative and innovative process. In spite of all the methods and tools appearing on the marked at a high frequency, software development is in the end a people process. Another aspect of software development is the technology focus and rapid technology changes. These changes may be in conflict with one of the main fundaments of SPI - that it should be fact based. Facts come from measurements, and the question is how we can collect enough data to draw conclusions based on statistical data analysis in an ever-changing environment. The answer is simple - we can not. That does not mean that we have to give up. The solution is to use the data as indicators and include the experts - the people related to the measured process - and their interpretation of the data. This implies that you have to take risks. You can not be sure that the interpretations are correct. A few data combined with expert knowledge are, however, far better than expert knowledge alone. The measured facts
are there to aid the experts in reaching their conclusions and keeping this process on
the right track. The risks that rise from this way of reaching conclusions have to be
controlled. Thus, risk analyses has to be an integrated part of any SPI program.

**SPI should be based on a combination of measurement data and expert
knowledge - controlled by a risk analyses.**

**SPI and the developers**

Software development is one thing, software developers is quite another. How can
we motivate them to get involved with SPI? Rigid procedures and control are
definitely not the answer. This does not mean that developers do not want procedures
at all or that all project tracking is considered to be control. Firstly, lack of
procedures is often considered a problem by the developers, and definition of new
procedures can be the result of SPI activities. Procedures will reduce the time spent
thinking about small problems - giving more time to be used on creative tasks [10].
On the other hand, procedures are not considered to be an aid for knowledge transfer
(see Table 1). The problem with existing procedures is, however, that they not always reflect the needs of the developers. Secondly, project tracking means
measurements, and measurement data can be collected as long as they are considered
useful by the developers and not used as a vehicle for control.
The solution to both these problems is developer participation. Our experience is
clear; if the developers are involved, they will be motivated. In SPI, involvement
means to participate in:

- Defining the improvement strategy - problems to be solved and opportunities to
take advantage of
- Define the metrics and the data collection procedures
- Interpretation of the measurement data
- Identifying and implementing improvement activities, i.e. defining procedures.

**SPI can not solve the developers’ problems without their participation**

**SPI and the business**

To make it easier to sell SPI to the management, we should be able to document the
cost and benefit of SPI. SPI methods have been used to document the benefit of
inspections, the effect of new tools or methods etc. [3], [6]. Little have, however,
been done up till now in order to document the effect of the SPI itself. Some data has
been published [4] - all of them positive, but not all of them widely accepted.
The SPI community is fact based and they thus require evidence in order to
institutionalize the results of improvement activities. Why do they do not put the
same requirement on themselves - perform cost benefit analyses of the total company
SPI activity and document the results. This is a challenge that the whole SPI
community has to meet in the near future if they want a wider acceptance and use.

**Cost benefit analysis of SPI have to be put forward in order to sell SPI to
managers**
Conclusions

SPI is about learning to work smarter, and thus everybody in the software community should be interested. So, why do they not believe that SPI can help them? First, the discipline of software engineering has matured over the years, but is still to some degree based on hero programmers that believe that SPI will be like QA - more paper work and control. Second, SPI means investments and managers do not like to invest without at least some kind of documentation of the benefits.

Tomorrow’s organizations on the other hand, will be organizations that are continuously looking for improvement opportunities. The improvements should be based on a few vital elements:

- Facts - measurements combined with knowledge - in order to solve the right problems. Uncertainty should be handles through risk management.
- Participation - in order to motivate the developers and keep the improvement program alive.
- Learning - both on the individual and organizational level.
- Minimal paper work and overhead - keeping creativity and innovation alive.
- Cost/benefit analyses in order to defend the investments.

References

SINTEF is an independent, not-for-profit research foundation based in Trondheim and Oslo, Norway. Our role is to encourage innovation and improve competitiveness in Norwegian industry and public administration. In doing so, we maintain close links with the technical Universities in Trondheim and Oslo, collaborating on projects, and sharing equipment and other resources. SINTEF is not a publicly funded organization. A very small part (less than 4%) of our income is from a public grant; most of our operating revenues arise from contract research and development work carried out for industry and the public sector in Norway and elsewhere. With over 1800 employees and a turnover of NOK 1.4 billion, SINTEF is Scandinavia’s largest independent research organization. It is organized into eight separate research institutes, covering all major scientific areas and industrial sectors. Refer to our web site www.sintef.no for further information.

SINTEF has over the years been a leading company in the area of software engineering and have broad and deep experience in this area. This experience serves as a sound basis for our Software Process Improvement (SPI) work. Our major SPI activities are:

- The SPIQ program – Software Process Improvement for better Quality. This is a national Norwegian program that is partly a national ESSI type project and partly a co-operation with Norwegian industry to increase the use of process improvement methods in Norwegian software industry.
- The QIS project – Quality Improvement in Scandinavia. This is an EU sponsored project that shall give help and assistance to PIEs in Norway and Sweden and market the concept of SPI to software industry in these two countries.

**Tor Stålhane** was born in 1944 and became a M.Sc. at the Norwegian Institute of Technology, University of Trondheim (NTNU) in 1969. During 1969 to 1985 he worked at SINTEF - RUNIT, department for languages and compilers. From 1985 he worked on his Ph.D. studies and finished his thesis on software reliability in 1988. From 1988 he was back at SINTEF where he mainly worked with quality assurance and software safety and reliability. In 1997 he became professor in Computer Science at the Stavanger Polytechnic. During the latest decade he has been mainly been working with safety analyses of software intensive systems and measurement based process improvement.

**Kari Juul Wedde** was born in 1951 and became a B.Sc. at the Technical College of Trondheim in 1972. After that she has continuously updated herself by following graduate and postgraduate courses at NTNU. In 1973 she started to work for SINTEF. The work at SINTEF has given her a long and broad experience in software engineering, ranging from compiler construction to telecom applications. Her main areas of expertise during the last years has been software testing, quality assurance, measurement based software process improvement and safety analyses of software intensive systems, and she has several international publications in these areas.