Cover Feature

Process Improvement for Small Organizations

To introduce and sustain a software process improvement initiative, a smaller organization must minimize the limitations of its size and maximize the benefits inherent in its culture. The authors describe how this was done at Silicon & Software Systems.

Declan P. Kelly
Bill Culleton
Silicon & Software Systems

Software process improvement (SPI) has been a hot topic within the software industry for a number of years. Its high profile has been due in part to the introduction and industry acceptance of standard improvement models, most notably the Capability Maturity Model (CMM)[1] developed by the Software Engineering Institute at Carnegie Mellon University. Large organizations, such as Motorola's Government Electronics Division with 1,500 software engineers, have successfully implemented SPI initiatives and reaped significant benefits.[2] For smaller organizations, these benefits are no less significant, but smaller organizations often operate under different constraints.

For a small organization to introduce and sustain a process improvement initiative, it must minimize the limitations of its smaller size and maximize the benefits inherent in its culture. In this article, we describe an approach to SPI that has been used successfully in an organization of approximately 150 software engineers. Our approach involved as many software engineers as possible while avoiding disruption to ongoing projects.

SMALL-COMPANY ISSUES

The ratio of the effort required to implement a CMM-based SPI initiative in a company of 1,500 software engineers and a company of 150 software engineers will not be 10:1. In fact, for the core activities, excluding training and support, the effort required will not be significantly different. Clearly, the investment that a larger company can afford will usually be significantly greater than the investment that a smaller company can justify. Therefore, smaller organizations have an even more acute need to use SPI resources efficiently; if they don't, the cost of SPI can become prohibitively expensive.

Significant cultural differences exist between small organizations and larger ones. In smaller organizations, employees expect to be involved in all aspects of the software engineering process. Often, this practice is a result of the company's history. When a software company starts off with a handful of people, by necessity, everyone is involved in all aspects of software development. As these start-up companies grow, they often retain this culture of involvement. On the other hand, large organizations can leverage significant efficiency improvements by, for example, having a dedicated group choose computer-aided software engineering (CASE) tools or standards. Therefore, in a large organization, it is not unusual for decisions about the software process to be made outside the software engineering groups. In contrast, software engineers in smaller organizations expect to influence decisions that affect the way they work.

The culture of smaller organizations can often be characterized as creative, dynamic, and innovative. The success of these organizations is often due in no small part to the creativity and innovation of their employees. SPI is frequently viewed as the antithesis of these qualities, leading to bureaucracy that restricts the freedom of individuals. This aspect of small-company culture affects the SPI initiative in two ways. First, the SPI initiative should use the creativity of individuals within the organization to provide innovative solutions to SPI problems. Second, the result of the SPI initiative should not stifle creativity; it should focus creativity on project-specific problems, not standard process issues.

S3'S SPI PROJECT

Silicon & Software Systems (S3) has been providing design services in silicon, software, and hardware design since it was established in 1986. Within the software division, application areas include telecommunications, consumer electronics, Internet, and digital broadcasting.

0018-9162/99/$10.00 © 1999 IEEE

Why the Capability Maturity Model?

We chose the CMM as the basis for our software process improvement because

- it provides a road map for process improvement through the five levels (see Figure A),
- it is an accepted industry standard and therefore allows easy comparison with other companies, and
- its key process area structure is flexible enough to tailor KPA implementation to suit the organization.

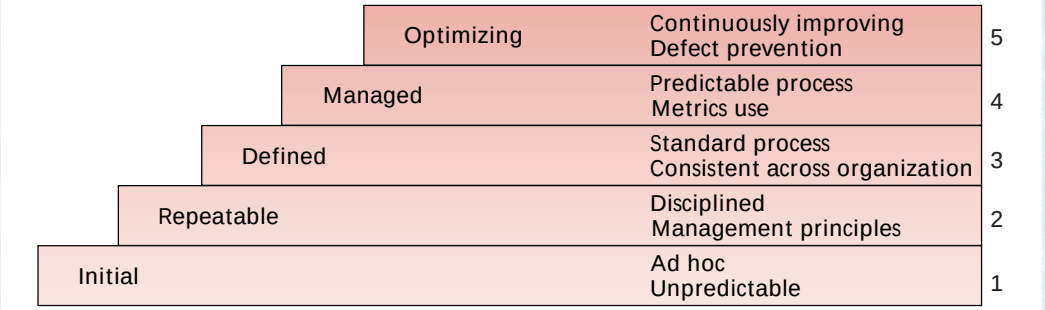

Figure A. The levels in the Capability Maturity Model.

The customer base is diverse and includes many major multinational companies. Because software projects in S3 cover a range of application areas and involve working with many different customers, the SPI process must be flexible enough to accommodate S3's diverse and constantly evolving projects. It would certainly be simpler to mandate a common programming language, design methodology, and set of CASE tools for all projects, but such an approach clearly is not an option for a company like S3. The diversity of S3's projects at first seems a complicating factor, but in practice it results in simplified procedures. This may seem surprising. However, applying quality procedures to such a diverse range of projects forces the procedures to focus on the essential details. The result is that the quality procedures mandate requirements essential to the quality of all projects, but they do not bound the freedom of projects by including arbitrary restrictions.

The starting point for our SPI initiative was an existing quality system—a set of quality procedures and tools to support their use. Throughout its history, S3 has used quality procedures, in many cases based on IEEE software engineering standards. The existing quality system had been certified ISO 9001-compliant.

Goals

The business objectives of our process improvement initiative were to reduce project lead time and improve the quality of the results, which in turn would increase customer satisfaction to the point that customers would keep using S3 as a preferred supplier. We had four main goals.

Maximize involvement, minimize disruption. In pursuing our SPI program, we wanted to involve as many of the software engineering staff as possible. There were two reasons for this. First, it would not be practical to have a dedicated group assigned full-time for the entire project. Second, given the organizational culture, it would not be acceptable to have solutions imposed by a small group.

All the people we wanted to involve were active on projects, which we did not want to disrupt. To deal with this, we broke the SPI work into small tasks, and we defined exactly what was required and how much effort it would take for each task. This approach allowed people to evaluate their project commitments and then commit to the SPI work only if it would not conflict with other project requirements. A single full-time coordinator was assigned to manage the project and coordinate the work of part-time participants.

We made sure that we involved people with varying degrees of experience from all sections of the software group. In particular, we tried to involve people who were skeptical about the need for a defined quality system and for a CMM-based SPI program. The skeptics generally think they know better and so object to what they perceive as restrictions on how they work. In practice, they often do have good ideas about ways of working; the trick is to channel their skepticism into improving the quality system. Involving them can often dissolve their skepticism. As they become immersed in the SPI initiative, they begin to see that it is not an attempt to impose restrictions but rather a way to improve the efficiency and effectiveness of the complete software group. Skeptics can also be some of the most influential individuals within the organization. Consequently, a benefit of converting them to SPI is that they influence others to become involved.

Stress quality, not CMM compliance. Although we were basing our initiative on the CMM, we did not focus exclusively on moving up the levels (see the "Why the Capability Maturity Model?" sidebar). Level 1 has no defined key process areas (KPAs) because it is the starting point; there *is* no defined process. In CMM terms, we were at Level 1 (although we had partially addressed all Level 2 KPAs), and we wanted to be formally assessed for Level 2. First and

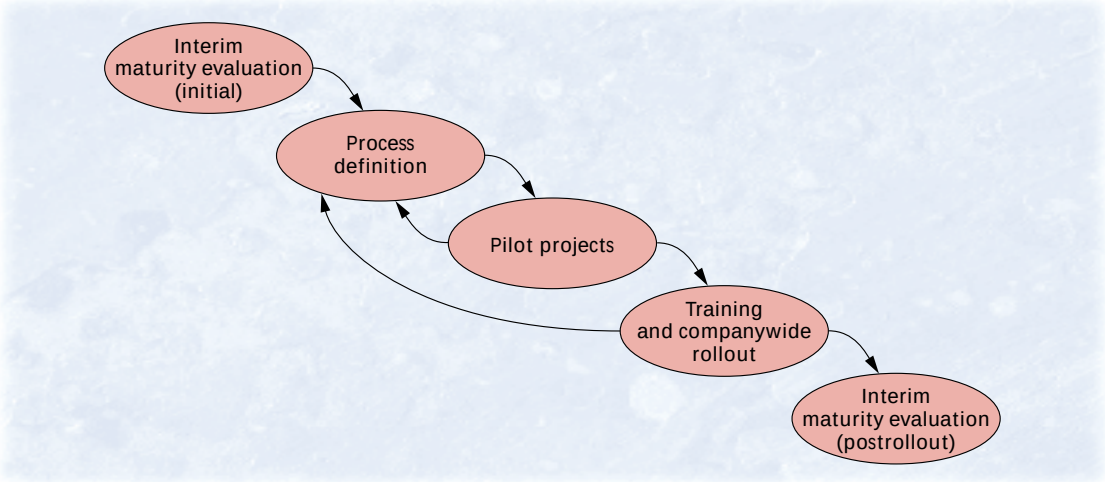

Figure 1. S3's software process improvement initiative. Improvement efforts began in September 1997 with an interim maturity evaluation to determine how S3 assessed relative to the CMM and its key process areas. The next step was to define a proposal for improving the weaker areas. Pilot projects helped validate the proposals. In June 1998, the team was ready to expand the process companywide and establish training. Shortly thereafter, we performed another interim maturity evaluation to gauge progress in addressing the key process areas.

foremost, however, we wanted an efficient and flexible quality system that would promote continuous process improvement.

Although achieving a particular CMM level will no doubt improve software quality,[1,3] there is always the danger of introducing a CMM-compliant process that does not fit your organization or that involves too much overhead. Thus, you have short-term process improvement, but it will be hard to sustain. If, however, the emphasis is on an efficient, flexible, quality system, the process improvement team is likely to keep working on the process definition until that efficiency and flexibility are achieved—even after a CMM-compliant process has been defined.

Also, CMM compliance should never be used to justify a process change. It is always tempting to counter objections to process changes by stating that the changes are a CMM requirement. It is more difficult, but ultimately more rewarding, to explain where the requirement comes from and then discuss the objections with a view to finding an acceptable solution.

Emphasize the advisory role. Most people view a quality system as a set of requirements to be enforced. We wanted the engineers to view it as both defining mandatory requirements and providing helpful advice. Our motto was, "Quality standards should seek to help the user do the job well, rather than just prevent the user from doing it badly."

To follow this motto, we structured the quality procedures in two layers. On the bottom is a set of mandatory requirements (which all projects follow). On the top is a set of optional guidelines, or best practices, which are based on the requirements. We deliberately put best practices in the guidelines layer because not all practices apply to all projects. The choice of which best practice to use is left to the project leader and the project team under the guidance of an independent quality assurance group. This is important because the final responsibility for a project's quality rests with that project's leader and team.

Promote efficiency. To make the quality system efficient, we wanted to be sure that the quality procedures were well written and clearly structured. People should be able to find the information they need without searching through multiple documents or following a chain of cross-references. We also wanted adequate tool support to make the quality system easy to use.

Interim maturity evaluation

Figure 1 shows the steps in our SPI initiative. The first step was a maturity evaluation. Before embarking on the SPI project, we wanted to evaluate our existing process against CMM KPAs. This would give us a baseline against which to measure our improvement. We based our evaluation on an Interim Maturity Evaluation Questionnaire—a set of questions, derived from the KPAs, about the way the organization worked. The evaluation consisted of a discussion based on the questionnaire and guided by an external consultant. The participants first individually scored each question and then as a group discussed questions they disagreed on. The discussion continued until participants agreed on the appropriate score for the organization for each question. We calculated an estimate of compliance for each KPA by averaging the scores for individual questions. In addition to providing us with an objective measure of our current status, the evaluation helped raise awareness of the issues we needed to address to achieve compliance at the next CMM level.

Where we were. In terms of CMM levels, S3 was a Level 1, but the typical description of a Level 1 organization as "unpredictable" (see Figure A) didn't apply. In our opinion, at that time, S3 most closely resembled the CMM Level 3 description: "standard process, consistent across organization." This situation is probably found in many Level 1 organizations: The company has not fully addressed all the Level 2

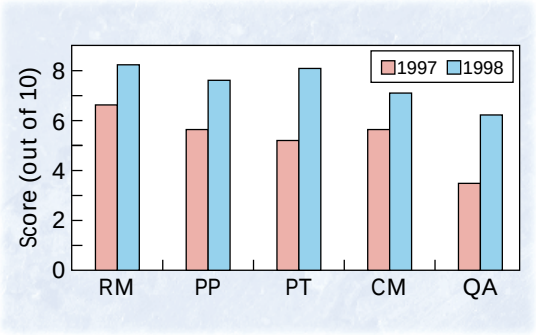

Figure 2. S3's compliance rating in the CMM Level 2 key process areas in September 1997, when the initiative began, and in June 1998, when the initiative was ready to go companywide. The Level 2 KPAs are requirements management (RM), software project planning (PP), software project tracking and oversight (PT), software configuration management (CM), and software quality assurance (QA). The software subcontract management KPA, also in Level 2, is not shown because S3 does not use subcontractors.

KPAs and so remains at Level 1, but it has partially addressed Level 2 and higher KPAs.

Figure 2 shows our compliance rating in Level 2 KPAs when we started our initiative. As the figure indicates, the existing quality system partially addressed all of these issues. It also covered some Level 3 KPAs, such as peer reviews. Many organizations starting a CMM-based SPI initiative will have some level of documented process. Starting an SPI initiative in a pure Level 1 company would be much more difficult.

Where we wanted to go. To achieve CMM Level 2, we needed to address requirements management, software project planning, software project tracking and oversight, software quality assurance, and software configuration management.

In addition to these CMM KPAs, we defined two non-CMM KPAs: completed work analysis and software metrics. CMM purists would object to including these with the CMM Level 2 KPAs because each represents a topic covered by a KPA at a higher CMM level. The CMM philosophy is that the KPAs at each level form a foundation for the process improvement at higher levels. Therefore, it isn't effective to implement higher level KPAs before the required foundation is in place. Although this may be true for a pure Level 1 organization, we started with a quality system that partially addressed all the relevant Level 2 KPAs and some Level 3 KPAs. We thus took a pragmatic approach and defined KPAs for issues we believed could offer significant benefits.

The completed work analysis KPA was a central part of our continuous process improvement plan. Its goal was to define a procedure for analyzing the work done on projects and for feeding back the experience so that it could be used in future projects. The scope was thus wider than process issues, encompassing tool use and communication and coordination on individual projects. We wanted to use the procedure from the completed work analysis to ensure that we learned not just from our mistakes but also from our successes. By analyzing the experience on all projects, we hoped to ensure that the lessons learned from individual projects were also learned by the complete organization. A group of 150 software engineers gains 150 person-years of experience every year. This represents a significant source of knowledge for the organization if it can be tapped.

The Level 2 software project tracking and oversight KPA covers some metrics, but we added a non-CMM KPA to define additional metrics. The goal was to define data to be collected by projects that would identify weaknesses in our existing process. Knowing the weaknesses would allow us to focus our process improvement effort on them. For example, we collected metrics on the defects introduced per project phase. This let us focus our improvements on the phases where the most serious defects are introduced. When defining the metrics to collect, we wanted to be clear about why we were collecting them, so for each metric, we also defined what we planned to conclude from the data.

Process definition

For the KPAs we defined ourselves (complete work analysis and software metrics), we had a clear understanding of what we wanted to achieve and the requirements. For the Level 2 KPAs, we chose to buy expertise so that we could get the project moving as quickly as possible.

KPA definition workshop. We organized a workshop and chose participants from across the software group so that all types of projects were represented. We did not assume that the participants had any knowledge of CMM, so the workshop started with a CMM introduction. The rest of the workshop focused on the Level 2 KPAs that were relevant for S3. The workshop's goal was to produce proposals for implementing these KPAs in the S3 environment. The consultant first described the requirements of two KPAs and gave some ideas about how to implement them. The participants then broke into two groups and worked out an implementation proposal. This approach was very effective. The groups were able to consider various options in the context of how people work in S3, and the consultant was there to answer any questions about CMM.

Once the two groups had agreed on a proposal, each group presented its proposal to the other, and the proposals were discussed further. At the end of the workshop, the participants had produced concrete proposals

Table 1. How the task force tailored an activity from the CMM Level 2 key process area (software project planning) to meet S3's needs.

Key process area	Description
Activity	The software engineering group participates on the project proposal team.
S3 context	The participants in the proposal are the sales team and software development team in conjunction with management. The sales department and senior management deal with the commercial and financial aspects.
Interpretation	Experienced software engineers prepare the technical aspects of the project proposal. These engineers normally form the core of the project team that will perform the work. At this stage, this team prepares an initial plan that documents the main aspects of the project. Once the customer and S3 senior management make a commitment in the form of a signed contract or letter of intent, the plan is further developed. Any further development or modifications at later stages must always involve the project team as well as any organizations outside the project that are affected by it.

for implementing two CMM KPAs. The success of the workshop was due in large extent to the consultant, who not only understood the CMM theory but also had experience with its practical application. This was a great help to us as we designed an implementation of the KPAs to suit our working environment.

One striking outcome of the workshop was that all the participants, without exception, were enthusiastic about the SPI project and were highly motivated to contribute further. They all agreed to meet again the following week to work on the remaining KPAs. This follow-on workshop also resulted in implementation proposals.

Task forces. To involve as many people as possible in the SPI project, we used task forces. In each task force, one person was responsible for implementing the changes, and approximately five people reviewed and guided the implementation. Each CMM Level 2 and internally defined KPA was assigned to a task force. This ensured that a large proportion of the organization was directly involved in the SPI project. Indeed, after this step in the initiative, approximately 45 percent of the software group had been directly involved.

Each task force followed a standard approach to KPA implementation. First, they wrote a proposal describing how the KPA would be implemented. For CMM KPAs, they produced a detailed proposal document that was based on the workshop results. For the non-CMM KPAs, they wrote a proposal document. In each case, the task force reviewed and discussed the KPA proposal and agreed on how to implement the KPA. The next step was to implement the agreed-on changes. This involved updating existing quality procedures, writing new procedures, and, in some cases, providing tool support. The task force also reviewed any new or updated quality procedures. Table 1 shows how the task force proposed to implement an activity from the software project planning KPA.

Pilot projects

Once the task force approved the changes for a KPA, we were ready to introduce them on real projects. For KPAs that had significant changes, we chose to test the changes on a pilot project. This allowed us to monitor the effects of the changes very closely and take corrective action if problems arose with the KPA requirements.

We tested our changes to the software project planning KPA on an Internet application project that had a short life cycle. The project let us not only verify the completeness of the process, but also test the downward scalability of the KPA definition to a small project. The project team implemented all the improvements and tracked their effects. From the customer and the S3 management point of view, the documented project plan provided excellent visibility and greatly aided tracking. The project team was more confident in their plan and, because they spent less time replanning, as had been typical of previous projects, they had more time to concentrate on technical issues and product quality. Also, increasing the level of detail (relative to previous plans) produced a more complete schedule, thus reducing the number of tasks that might otherwise have been forgotten.

In general, the pilot project test of the project planning KPA definition went very smoothly, and the KPA definition did not change significantly.

In contrast, the pilot project to test the quality assurance KPA definition did encounter problems. After less than a month, it became apparent that the KPA definition was not practical, although it had been set and reviewed by many people and seemed to be a reasonable approach. The amount of overhead to both the project and supporting infrastructure was not acceptable, and the return on invested time was questionable. When we analyzed the KPA definition, we found that we had spent too little time tailoring the KPA to S3's environment, and we had not sufficiently defined

We tried to get as much feedback as possible from the software engineers who would use the updated quality system.

the supporting infrastructure. The project immediately stopped using this KPA definition, and the original task force, in conjunction with the pilot project team, redesigned it. We put the new version and an improved infrastructure in place only after the project was completed. The updated KPA definition has since been tested and is proving effective.

Training and rollout

Before introducing the updated quality system throughout the software group, we arranged training on the new system for all software engineers and relevant managers. To stimulate discussion, the training was conducted in groups of 12. The training course for each group lasted a full day. During the course, we tried to get as much feedback as possible from the software engineers who would use the updated quality system. Also, we tried to instill an understanding of the context and purpose of the changes so that the engineers would not only understand what was required of them but also why. The training courses included a walk-through of some of the updated quality procedures, which generated much useful discussion and many suggestions for improvements.

The training course initiated the rollout of the new processes to all projects. Process mentors were assigned to each project to support the introduction of the new processes. For existing projects, the process mentor and project leaders decided case by case which parts of the new processes to adopt and at what point to introduce the changes.

Infrastructure. To make the updated quality system easy to use, we developed a supporting infrastructure and used the companywide intranet to support it. We provided an easy-to-use Web interface to the quality documents and tools to support specific changes that resulted from KPA implementation. Among these was a lessons-learned database that provided easy access to suggestions arising from the completed work analysis performed on projects. A software process group also reviews these lessons and looks for ways to provide feedback into the quality system.

Version control. Our goal to start a continuous improvement process conflicted with the need for stability within the quality system. If the quality procedures are constantly changing, the multitude of versions will cause confusion. To avoid this, we released modified documents only when we released a new quality system version, which was at most every three months. With each quality system version release, we provided a full description of all changes. This gave projects visibility into any changes that had been introduced. When a project starts, it uses the latest version of the quality system. A project might later adopt a newer version of the quality system, depending on how significantly it has changed. Typically, projects will change versions only at the start of a major new phase.

RESULTS

In evaluating an SPI project, you cannot consider only objective measurements because the software process is ultimately about helping people do their jobs. Trying to measure process improvement while ignoring the people involved provides a very one-sided view. If the people who use the quality system do not feel that the SPI is improving things, there is a problem.

Measurements

At the start of our SPI project, we conducted an interim maturity evaluation to assess our status with respect to CMM KPAs. After we introduced the new quality procedures and tool support, we again performed an interim maturity evaluation. Figure 2 shows the results of this evaluation.

When we started our SPI initiative, we were reasonably strong in all areas except quality assurance. The strength in the other KPAs was due to our ISO 9001-compliant quality system, which in our opinion was already a very strong foundation. Figure 2 shows real improvements in all KPAs. The most striking improvement is in quality assurance—understandable because this was our weakest area initially and hence where we could gain the most. The large improvement is due to CMM's emphasis on a supportive role, for example, support at the planning stage, in contrast to ISO 9001's audit-based approach. The figure also shows that we can benefit even more, especially in quality assurance. We decided that before seeking a formal CMM assessment, we should score at least nine for each KPA in the interim maturity evaluation. Moreover, given that our primary goal was to achieve real improvements, with CMM certification as a secondary goal, we also decided to look at further improvement benefits before seeking the assessment.

Since the first rollout in June 1998, all projects have moved to the new process definitions, with the help of the training course and mentoring by process experts. The feedback from customers has been very positive. Before the project, most of our customers required that we use their process; now, after one and a half years, most customers are happy to see projects use the S3 process. An excellent example is S3's recent recognition as a qualified software supplier to a CMM Level 4 company. The positive feedback from customers proves that the SPI program is contributing to S3's business goals.

Subjective evaluation

The subjective evaluation of our SPI project is also very positive. Software engineers feel greater owner-

ship of the quality system and are more inclined to suggest improvements through a structured change proposal mechanism, as evidenced by the increased number of change proposals. They also greatly appreciated our addition of a strong support infrastructure. Managers feel that the SPI initiative has given them greater visibility, though not yet complete, into projects and their progress.

The effort expended to date, excluding the project leader, is 180 person-days on process redefinition, 70 person-days on training, and 20 person-days on evaluations. Two new procedures were introduced, one on requirements management and the other on risk management (as part of project planning). We have revised six existing procedures, one of which is the project planning documentation procedure.

Our approach to process improvement has proved very effective. There has been a noticeable shift in attitudes toward the quality system. Where once it was seen as something static to be tolerated, it is now seen as a living thing that is constantly evolving. Thus, although not everyone agrees with all aspects of the quality system, people are far more willing than in the past to make constructive suggestions.

The SPI approach described in this article was targeted for S3's software division. A process improvement initiative that follows a similar approach is now being introduced across S3. ❖

Acknowledgments

We thank Dave Murrells of Silicon & Software Systems for reviewing an earlier version of this article and providing valuable comments.

References

1. K.M. Dymond, *A Guide to the CMM: Understanding the Capability Maturity Model for Software*, Process Transition Int'l, Annapolis, Md., 1995.
2. M. Diaz and J. Sligo, "How Software Process Improvement Helped Motorola," *IEEE Software*, Sept./Oct. 1997, pp. 75-81.
3. J. Herbsleb et al., "Software Quality and the Capability Maturity Model," *Comm. ACM*, June 1997, pp. 30-40.

Declan P. Kelly is a research scientist at Philips Research Laboratories in Eindhoven, The Netherlands. Before joining Philips Research, he worked in the software division at Silicon & Software Systems as software process improvement coordinator, among other roles. His research interests include digital video processing, storage applications, software engineering, formal methods, and software process improvement. He received an MSc in computer science from Trinity College, Dublin, and a joint honors BSc in mathematics and computer science from University College, Dublin. He is a member of the IEEE Computer Society and the ACM.

Bill Culleton is process and quality manager at Silicon & Software Systems, Dublin, where his responsibilities include managing a CMM-based process improvement project in the software division as well as project management mentoring and training. He also supports process improvement in S3's other divisions. He received an MSc in microelectronics design and a BA BAI in telecommunications engineering from Trinity College, Dublin. He is a member of the Institute of Engineers of Ireland. Contact Culleton at bill_c@s3group.com.