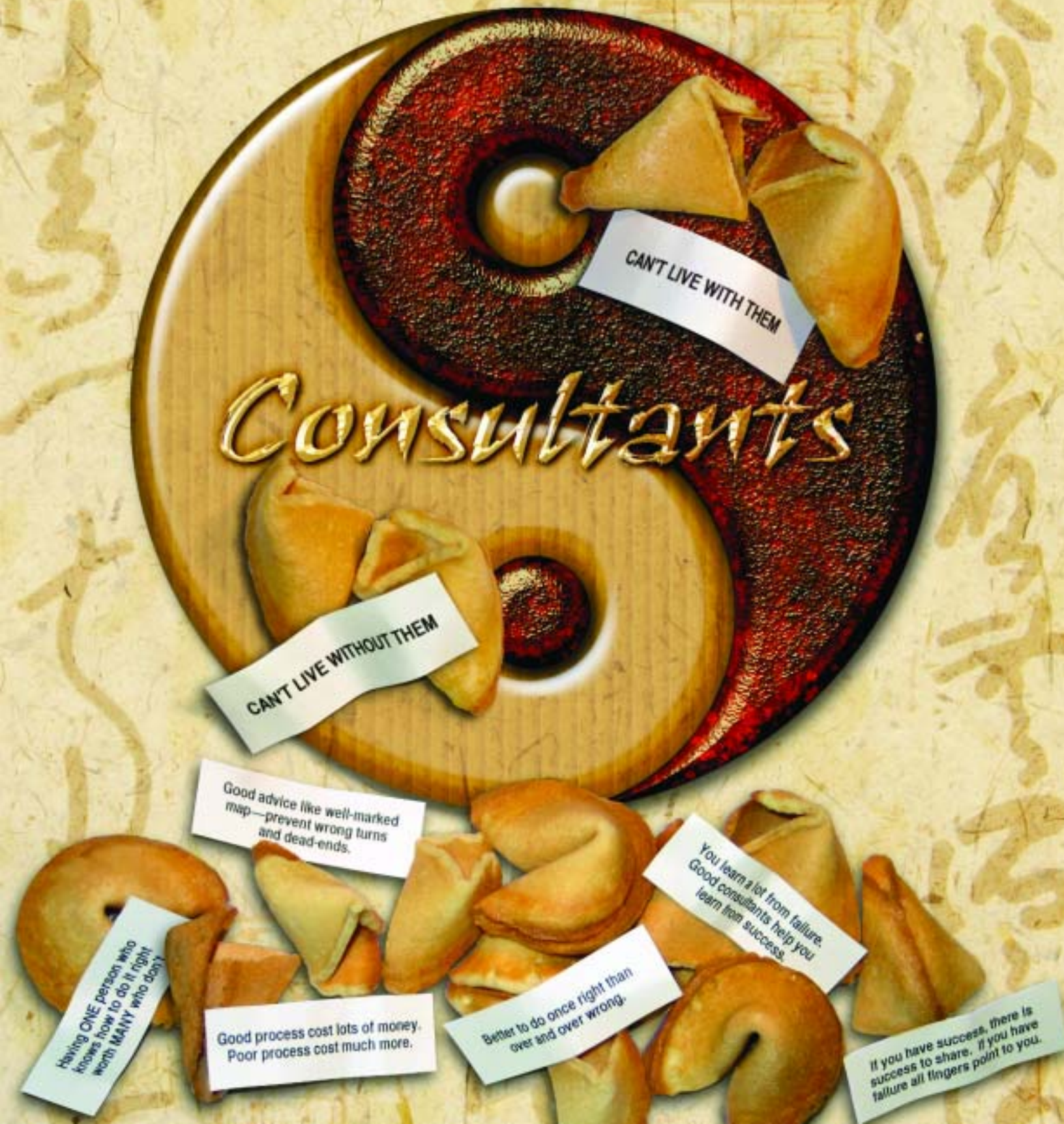
CrossTalk

February 2004 **The Journal of Defense Software Engineering** Vol. 17 No. 2



Software Consultants

4 Lessons Learned From Software Engineering Consulting
Based on their combined 15 years as consultants with the Software Technology Support Center, these authors talk about the patterns that emerge from observing and analyzing many different organizations and the lessons they have learned.
by Dr. David A. Cook and Theron R. Leishman

7 Overcoming Training Dilemmas Brings Greater Training Value
If you are planning training for your employees, read this author's advice on how to get more value from technical training by planning and determining training objectives and following up to evaluate progress.
by Gregory T. Daich

11 Ten Key Techniques for Process Improvement Consulting in a Challenging Environment
These authors outline techniques that consultants can use to encourage clients to do what needs to be done to help make a process improvement effort successful.
by Sarah A. Sheard, Suzanne Zampella, and Albert J. Truesdale

ON THE COVER
Cover Design by Kent Bingham.

Software Engineering Technology

14 Army Taps Vernon M. Bettencourt Jr. as Next Deputy CIO/G-6
Here is an opportunity to learn the background of the newly appointed Deputy Chief Information Officer/G-6 Bernon M. Betttencourt Jr.
by Patrick Swan

15 The Human Dynamics of IT Teams
The results of a research study show that today's information technology leaders need relationship management skills, insight into their own interpersonal needs, and awareness of others preferences to successfully deliver products.
by Jennifer Tucker, Abby Mackness, and Hile Rutledge

Open Forum

22 Making Meetings Work
Ever wonder why some meetings are successful and others are not? These authors offer some explanations based on the results of an industrial measurement study, including optimum number of people, meeting length, the affects of hierarchy, and more.
by Michael Ochs and Rini van Solingen

26 Verification and Validation People Can Be More Than Technical Advisors
This author proposes that some companies and government agencies recognize that the benefits of verification and validation can extend to non-technical areas like project management, resource management, finance, and scheduling.
by George Jackelen

Online Article

30 Information Assurance in Wireless Residential Networking Technology: IEEE and Bluetooth
by Ambareen Siraj and Rayford B. Vaughn

Departments

3 From the Publisher

10 Coming Events

19 Web Sites

20 SSTC 2004 Conference Registration

29 Letter to the Editor

31 BACKTALK

CROSSTALK

PUBLISHER	*Tracy Stauder*
ASSOCIATE PUBLISHER	*Elizabeth Starrett*
MANAGING EDITOR	*Pamela Palmer*
ASSOCIATE EDITOR	*Chelene Fortier-Lozancich*
ARTICLE COORDINATOR	*Nicole Kentta*
CREATIVE SERVICES COORDINATOR	*Janna Kay Jensen*
PHONE	(801) 586-0095
FAX	(801) 777-8069
E-MAIL	crosstalk.staff@hill.af.mil
CROSSTALK ONLINE	www.stsc.hill.af.mil/crosstalk

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail or use the form on p. 29.

Ogden ALC/MASE
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056-5820

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the CROSSTALK editorial board prior to publication. Please follow the Author Guidelines, available at <www.stsc.hill.af.mil/crosstalk/xtlkguid.pdf>. CROSSTALK does not pay for submissions. Articles published in CROSSTALK remain the property of the authors and may be submitted to other publications.

Reprints and Permissions: Requests for reprints must be requested from the author or the copyright holder. Please coordinate your request with CROSSTALK.

Trademarks and Endorsements: This DoD journal is an authorized publication for members of the Department of Defense. Contents of CROSSTALK are not necessarily the official views of, or endorsed by, the government, the Department of Defense, or the Software Technology Support Center. All product names referenced in this issue are trademarks of their companies.

Coming Events: We often list conferences, seminars, symposiums, etc. that are of interest to our readers. There is no fee for this service, but we must receive the information at least 90 days before registration. Send an announcement to the CROSSTALK Editorial Department.

STSC Online Services: www.stsc.hill.af.mil
Call (801) 777-7026, e-mail: randy.schreifels@hill.af.mil

Back Issues Available: The STSC sometimes has extra copies of back issues of CROSSTALK available free of charge.

The Software Technology Support Center was established at Ogden Air Logistics Center (AFMC) by Headquarters U.S. Air Force to help Air Force software organizations identify, evaluate, and adopt technologies to improve the quality of their software products, efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery.

Finding the Right Consultant Brings Mutual Success

Some consultants have been known to be expensive, time consuming, and even egotistical; initial impressions may be that they are sending customers down seemingly fruitless paths. However, many consultants bring an array of successful experiences to the table, that when shared properly can provide substantial benefits that outweigh the costs. While there is no crystal ball to tell which consultant will most benefit your organization, checking references and setting balanced expectations of knowledge, cost, and ability to work with your organization will go a long way to ensure you have an arrangement that is mutually beneficial.

One of my previous supervisors believed that consulting, when done right, puts the consultant out of work. A consultant's task is not only to guide, train, and execute, but also to enable the customer to perform on their own. If not, then the consultant is outsourcing, not consulting. I liked the analogy he shared of improving a basketball team. At first the consultant focuses on the team's weaknesses, often jumping on the floor to rebound, score, or play defense. Realizing a consultant can't play all positions, the next step is for the consultant to become the point guard, a coach on the floor to direct and motivate the team. Eventually the consultant wants to step off the floor and coach from the sidelines, allowing players to increase their skills and experience. Once management has learned and implemented the new processes and skills, the consultant will step off the bench and watch the game from the skybox, occasionally providing counsel on fine-tuning the team. Eventually the organization will heal and run itself. The successful consultant is no longer needed.

In *Lessons Learned From Software Engineering Consulting*, Dr. David A. Cook and Theron R. Leishman share some of the insights they have acquired from numerous years of consulting. While the focus of this article is on the more common problems that they have encountered, the closing suggestions for improvement are useful for almost all problems: Don't be afraid to ask for insight from others. This can work by asking a new person in the group with a fresh perspective, talking to someone from another group who may be showing success, or finding the right help from outside the organization.

Gregory T. Daich discusses ideas for getting more for your training time and dollars in *Overcoming Training Dilemmas Brings Greater Training Value*. It is usually not easy to find the necessary funding for training, and finding the time to go to the training can be even more difficult. When the training results in improvements that outweigh the costs it is great, but too often the stress on funding and time only results in little improvement with less funding and time now available.

We also have an article for our readers working in the consulting area. Sarah A. Sheard, Suzanne Zampella, and Albert J. Truesdale write about their lessons from challenging situations in *Ten Key Techniques for Process Improvement Consulting in a Challenging Environment*.

In this month's supporting articles, I wanted to expand on addressing organizational issues, so the line-up includes *The Human Dynamics of IT Teams*. In this article Jennifer Tucker, Abby Mackness, and Hile Rutledge remind us that people make software, and we must determine how to best work with these people to achieve the most success. In *Making Meetings Work*, Michael Ochs and Rini van Solingen share some insights from analysis of over 315 meetings in their own organization. While the results may not apply to all organizations, this could be the starting point for similar studies over a broader data set. In *Verification and Validation People Can Be More Than Technical Advisors*, George Jackelen shares one perspective for thinking outside the box when looking for help. In his example, a verification and validation group can also provide support in non-technical areas of software development. Finally, in our online article *Information Assurance in Wireless Residential Networking Technology: IEEE and Bluetooth*, Rayford B. Vaughn and Ambareen Siraj compare these two technologies, discussing both their strengths and weaknesses.

There are many different reasons to hire consultants. Organizations might want a sanity check, training, an overhaul, or even someone to do the work for them. Whether you find this support from internal mentors or from outside consultants, I hope this issue will help you consider new alternatives.

Elizabeth Starrett

Elizabeth Starrett
Associate Publisher

Lessons Learned From Software Engineering Consulting

Dr. David A. Cook[1]
AEgis Technologies Group

Theron R. Leishman
Software Technology Support Center/Northrop Grumman

The Software Technology Support Center (STSC) has provided consulting services for Department of Defense organizations since the late 1980s. During the last 15 years, literally thousands of visits have been made to various organizations in an effort to help them build and buy software better. This article talks about some of the lessons learned by two of the STSC's many consultants.

During the past years, many Software Technology Support Center (STSC) consultants have conducted program reviews and provided consultations for numerous Department of Defense (DoD) programs. After having observed many different organizations and analyzed their problems, certain patterns emerge. One of these is that there really are not any new problems. In almost all organizations that we visit, we hear, as part of the briefing, "We have unique problems here." It is difficult not to smile. In almost 100 percent of the cases, the problems are neither unique nor even difficult to uncover.

The Problems

The problems organizations have are usually grouped into the following categories:

- Requirements.
- Schedule.
- People.

According to G. Weinberg, "No matter how it looks at first, it's always a people problem" [1]. Of course, it is a bit simplistic to say that it's always a people problem. *All* problems are related to people. However, certain types of problems, while people-related, are actually specific to requirements and schedule.

Requirements Problems

It comes as no secret to developers that requirements problems are common in almost every large project. It should come as no secret to managers, either. However, we are still attempting to build software by using models that require exact requirements. While organizations often speak glowingly of iterative and spiral models, and of the Rational Unified Process, the software continues to be built using waterfall models, at least from a high level. For one thing, money tends to trickle down in a single lump sum, at least in terms of a high-level budget.

Unfortunately, budgets often are set and programs are funded prior to a full understanding of the correct requirements. Requirements issues continue to plague projects, often past the coding phase and *well* into testing, integration, and even sustainment [2, 3].

The simple rule is this: Expect to have incomplete requirements well into design, and often even well past it. To mitigate this risk, these obvious actions can be taken:

- Use a life-cycle model that permits iterative requirements gathering and incremental releases. In most projects, software needs to be developed and delivered in small increments. Yes, this often increases the cost. However, as we have learned many times, you will have to throw away at least one release – so do not make the final copy the throwaway [4]! It is usually not critical that the first version of the software is correct; it is probably not going to be, and you are going to end up throwing it away. It is much more important that the last version is correct.
- Management must be able to shift, reallocate, and if necessary request additional budget as requirements are eventually ferreted out and documented. Early in the requirements process, it is difficult to know the extent of the complete requirements; you are unlikely to know the implementation cost. The old way – padding the budget in hope of covering unforeseen requirements – has proven inadequate. Allowing the budget to grow as requirements expand allows ramping-up of the initial version or iteration of the development, with room (and money) to grow as needed.

Schedule Problems

One of the authors recently attended a software estimation workshop that covered a few well-known rules of thumb in the cost and schedule estimation world. One is that the average programmer produces 100 lines of code per month. Another is that by taking the cube root of the lines of code and multiplying by 3, you can estimate the basic schedule of a project in months.

For example, a project estimated at 100,000 lines of code (LOC) would take approximately 1,000 staff months (100,000 divided by 100). The cube root of 1,000 is 10, and multiplying this by 3 equals a nominal time schedule of 30 months. (Again, these are very rough rules of thumb – and good software developers will have more accurate and validated rules for their particular organization. Still, as general rules of thumb, they have been independently *discovered* and validated many times).

These rules of thumb can be used to establish a rough estimation of man months and time, given a reasonably good estimate of code required. One premise of these rules is that the cited nominal schedule cannot easily be shortened. The multiplier *3* used in the example varies among schedule estimation experts. However, most experts agree that anything below a multiplier of *2.25* creates an impossible schedule. In other words, if you know the line-of-code estimate of the project, it is relatively easy to establish a minimal schedule, which according to leading experts, cannot be lowered. This permits a relatively easy *idiot test* of the schedule.

The remarkable thing is that many projects do not have rough estimates of LOC (or function points, or some other size measurement), and therefore cannot justify (or even explain) their schedules. If your project has a schedule (and delivery date) dictated by anything other than a reliable (size, schedule, and cost) estimation method, then you are likely to have an impossible schedule. Just because the software is needed by a certain date does not imply that it can be ready by that date.

Develop an understanding and respect for the complexities of software development. This understanding is invaluable in comprehending the challenges of software estimation. Have you ever noticed how simple things are, if you are not the one actually doing the work? While remodeling their house, the wife of one of

the authors considered many of the tasks associated with the remodeling as *simple, little projects*. Such simple, little projects included moving a load-bearing wall and extending an exterior wall to make rooms larger. The author's wife could not understand why the project schedule and budget were so large. These are, after all, simple, little projects! Simple – but only in the mind of a person who does not understand the principles of construction and will not be doing the work. Software development is like this!

Program/project managers and customers of software-intensive systems need to understand that software development is not a *simple, little project*. Estimates and schedules developed using sound software estimation processes and approaches should be respected by managers and customers alike, and not randomly adjusted due to whims and desires of external influences.

As a final note, Brooks Law [4] (which paraphrased, says that adding additional people to a late software project only makes it later) does not just apply to adding people. Reorganization, changing contractors, or reassigning personnel in a late project will most likely not improve the situation. However, it will provide a convenient excuse for management when things start to fail!

When the schedule starts to slip in a major way, there exist only two viable solutions. First, change the schedule. Second, reassess the requirements and provide less functionality. Of course, a poorly designed system that is into development or testing cannot easily have functionality removed without major recoding. Therefore, often the only viable solution is to change the schedule.

People Problems

As consultants at the STSC, the authors have participated in many programs' reviews. In addition, we have been asked to consult and provide help for many programs. If there is one common theme that runs throughout all of the problems we have examined, it is this: Bad management (and bad management decisions) can cripple even the best programs. The interesting thing is that in literally all of the consultations in which we have participated, the problems are known by the *folks in the trenches*. Bad management decisions, unreasonable (and impossible) demands, and poor staffing decisions could easily be *discovered* by asking developers.

Unfortunately, developers often do not have an avenue to anonymously report problems and/or concerns. Afraid of complaining (and equally afraid of retribution) the developers gripe among themselves, but have no way to resolve the conflicts. When the problems become severe enough, outside consultants are called in, discover the problems, and summarize and report to management. The interesting thing is that management is often aware of the problem and typically makes such comments as, "We knew there were issues, but didn't know how severe the problems were." Unfortunately, by the time consultants are called, the problems are usually so severe that significant opportunities have been lost.

Do not be afraid of the truth! We have experienced some programs where it was apparent that the program mangers did not really want to hear the truth. They were content to manage the program with their heads in the sand. This type of program management is sure to kill almost any program.

To put it simply, not only are people your most important resource, but also they are your source of information. Managers, you need to set up avenues for your developers to voice issues and address concerns. Some of these avenues need to be anonymous. If you can convince your developers that valid complaints and problems are going to be addressed – with no fear of retribution – then you will have a handle on learning what the problems are.

Other Ways to Improve Your Chance of Success

As consultants, we are always amazed that the problem, while seemingly hidden from the program, is almost blindingly obvious to an outsider. This observation and others lead to the following solutions.

Do Not Forget the Simple Things

Once you successfully step back from the problem to observe, it is amazing how often the solutions rely on simple things. Past issues of CROSSTALK have covered the many simple things necessary to manage a program: risk management, requirements management, and configuration management [2, 3, 5]. Not that any of these topics are simple (far from it), but it is a simple fact that most programs will not succeed unless you have implemented risk management, requirements management, and configuration management.

Yet as consultants, we often see programs that are ignoring very fundamental areas. Someone once said of metrics that "they don't have to be 100 percent right to be useful." Well, when it comes to managing your risks, requirements, and configuration , you do not have to be 100 percent correct to be useful. As long as you manage with the right goals in mind, your project has a much better chance of succeeding [6].

Do Not Be Afraid to Ask for Help

As consultants, we often find that a major problem in programs is that many levels of the development effort, from programmers to upper management, feel that it will make them look bad if they ask for help. Managers do not want to admit that they do not have a handle on all facets of their program. Developers do not want to admit that they are not absolute masters of the nuances of either the development or target environment. The problem is rooted in the fact that most DoD projects are different from prior projects, so managers and developers alike do not have prior projects to make comparisons.

The solution is to find someone with experience. Experienced developers know that having an experienced program manager improves the chances of successfully completing a project on time and under budget. Unfortunately, experience comes from two sources – good experience, and bad experience. While there is certainly an argument to be made that a bad experience teaches very good lessons, nobody wants bad experiences on their project. To find others with good experience you have to admit that you need help. This is often easier to do with an impartial, outside observer than with peers.

Impartial observers do not necessarily have to come from outside your company – only from outside of your immediate group. In most large organizations, you can find mentors or other sources of help. This type of *cross-pollenization* not only helps your group, but can also help the entire organization by sharing viewpoints and experience.

Do not be afraid to request (and in some cases, demand) necessary training. Having new tools and languages do not help if your people do not know how to effectively utilize them.

Call In an Outsider

When you are deeply part of the problem, sometimes you cannot step back far enough to see the solution. In many software projects, independent verification and validation (IV&V) is used to assure quality. However, IV&V is usually called in after the software is finished. It helps, instead, to have an outside observer assist prior to program completion. Some projects are like this scene in Winnie-the-Pooh:

> Here is Edward Bear, coming downstairs, bump, bump, on the back of his head, behind Christopher Robin. It is, as far as he knows, the only way of coming downstairs, but sometimes he feels that there really is another way, if only he could stop bumping for a moment and think of it. [7]

We could often solve our problems if only we could stop banging our heads against a wall and think about it. However, without a truly objective outside observer, we often cannot see how to stop banging our heads.

It is a well-known fact that outside consultants are more influential. This feeling goes back to the days of the Bible: "Prophets are honored by everyone, except the people of their hometown and their own family" [8]. Outside observers are often useful in getting across the same message that you have been trying to convey for years – their status as an outsider gives them the credibility to get your message across.

As with outside observers, mentors, trainers, and expert advice, consultants can be used to *help you over the rough spots*. The cost of hiring a consultant is often paid back many-fold by the savings in preventing rework.

Summary

Are these guidelines enough to keep your program on track? Of course not! In reality, they are at best, general guidelines. They are based on an examination of many software projects under varying constraints. Many important issues such as design and implementation have not been covered – these usually are not major issues that the STSC sees. Other issues such as testing and verification and validation are very important, but the authors feel these issues have been adequately covered in other CrossTalk articles.

We do feel that the above guidelines are valid advice that might help you in your project. Certainly, software development is far too complex to be summed up in a few so-called simple rules.

- Have a strong risk management and risk mitigation plan.
- Implement configuration management.
- Use risk management and configuration together to proactively predict upcoming changes, thus mitigating major cost and schedule impacts.
- Focus on the product and quality, and modify the process to accommodate the needs of your program.

Do not be afraid to ask for help – it is far easier to learn from the experiences of others than to be forced to repeat all of the same mistakes. Also it is usually far cheaper to use a consultant's experience to help you avoid mistakes and errors.◆

References

1. Weinberg, Gerald M. The Secrets of Consulting. New York: Dorset House, 1995.
2. Leishman, Theron R., and Dr. David A. Cook. "Requirements Risk Can Drown Software Projects." CrossTalk Apr. 2002: 4-8.
3. Van Buren, Jim, and Dr. David A. Cook. "Experiences in the Adoption of Requirements Engineering Technologies." CrossTalk Dec. 1998: 3-9.
4. Brooks Jr., Frederick P. The Mythical Man-Month, Anniversary Edition. New York: Addison-Wesley, 1995.
5. Leishman, Theron, and Dr. David A. Cook. "But I Only Changed One Line of Code!" CrossTalk Jan. 2003: 20-23.
6. Cook, Dr. David A. "Confusing Process and Product: Why the Quality Is Not There Yet!" CrossTalk July 1999: 27-29.
7. Milne, A. A. The Complete Tales of Winnie-the-Pooh. 1926. Penguin USA, Oct. 1996.
8. The Bible. Clear English Version, Matthew 13:57.

Note

1. At the time this article was written, Dr. David A. Cook was the principle engineering consultant for Shim Enterprise, Inc., and under contract for the U.S. Air Force Software Technology Support Center.

About the Authors

David A. Cook, Ph.D., is a senior research scientist at AEgis Technologies Group, Inc., working as a verification, validation, and accreditation agent in the modeling and simulations area. He is currently the modeling and simulation liaison between the Missile Defense Agency and the Airborne Laser System Program Office. Previously, he was the principal engineering consultant for Shim Enterprise, Inc., and under contract for the U.S. Air Force Software Technology Support Center for over six years. Cook has more than 30 years experience in software development and management. He was formerly an associate professor at the U.S. Air Force Academy, a former deputy department head of the Software Professional Development Program at the Air Force Institute of Technology, and has published numerous articles on software-related topics. Cook has a doctorate degree in computer science from Texas A&M University, and is an authorized Personal Software Process[SM] instructor.

AEgis Technologies Group, Inc.
6565 Americas PKWY NE
Albuquerque, NM 87110
Phone: (505) 881-1003
Fax: (505) 881-5003
E-mail: dcook@aegistg.com

Theron R. Leishman is a consultant currently under contract with the Software Technology Support Center at Hill Air Force Base, Utah. Leishman has 19 years experience in various aspects of software development. He has successfully managed software projects and performed consulting services for the Department of Defense, aerospace, manufacturing, health care, higher education, and other industries. This experience has provided a strong background in systems analysis, design, development, project management, and software process improvement. He is a Level 2 Certified International Configuration Manager by the International Society of Configuration Management, and is employed by Northrop Grumman. Leishman has a master's degree in business administration from the University of Phoenix.

Software Technology
Support Center
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056
Phone: (801) 775-5738
Fax: (801) 777-8069
E-mail: theron.leishman@hill.af.mil

Overcoming Training Dilemmas Brings Greater Training Value

Gregory T. Daich
Software Technology Support Center/Science Applications International Corporation

This article discusses how to avoid training dilemmas such as "they came, they taught, they left, and nothing changed." (It does not include how to train a dilemma, which might be of interest to some managers.) Fundamentally, this article addresses getting more value from technical training. Organizations do not get the full value of the skills taught to their people without planning to determine training objectives and following up to evaluate progress. Many skills can be learned and readily applied on an individual basis, while other skills require an established organizational business process within which to effectively apply. Finally, students and managers may benefit from training on how to receive training effectively.

Many organizations have invested a significant amount of their resources in training because effective training has given them vital knowledge. Timely training can provide a significant edge over competitors' or opponents' capabilities. However, some training does not result in recognizable improvements to individual skills or organizational practices. Without careful planning and follow-up, information acquired during training often dissipates into oblivion. Except for a course handout lying around collecting dust, no sign remains in some organizations after a few weeks that anyone received training on a particular topic.

Managers often invest thousands of dollars and several days of work effort for their employees to take a particular course. They often have high expectations for improving skills from training. However, some managers seem to be unaware that many individual skills that are taught cannot be applied unless a *supporting business process* is in place within which employees can perform those skills. The skills taught to employees are often evaluated as ineffective since the organization did not readily adopt them.

For example, software document review practices require a supporting organizational process to be consistently and effectively performed. Some document review skills can be applied individually. But my experience is that until an organization plans and conducts a disciplined document review process improvement effort, which includes appropriate training, software document quality will not significantly improve.

Since training is so important, estimating its return-on-investment (ROI) could provide important insights to guide further training efforts. However, a Software Engineering Institute (SEI) study found that leading software development organizations rarely try to compute training ROI [1]. (See the sidebar "Measuring Training ROI" on page 8 for information on computing training ROI.) They also found that only 18 percent of respondents indicated that their training resulted in significant improvement in their software engineering skills. What can be done to improve the value of training and how can training ROI be estimated to help in understanding its value? One thing is clear: There are many opportunities for improving the delivery and reception of training in both government and industry.

Determining Training Success

Training success is addressed here in terms of learning success and teaching success. A common measure of *learning success* is for students to demonstrate mastery of individual skills taught in class [2]. Learning success can be partially evaluated through exercises and tests. However, few business people like tests during training so instructors generally rely more on discussions and exercises.

An important measure of *teaching success* must include the level of adoption of advocated individual skills. In other words, do students actually accept and use the skills and technologies on the job that they learned? A key responsibility of trainers is to try to significantly *affect* students' attitudes so that they actually try out new skills on the job. Of course, the training might be covering an outdated technique with respect to some new practices in the industry. If a student does not adopt the methods because he or she knows a better way, then the training was certainly not successful.

Measuring skill adoption is very difficult especially when many individual skills support an overall business activity. How much does individual skill competency contribute to overall productivity? That certainly depends on many factors, including, but not limited to, skill complexity and the likelihood of performing the skill incorrectly (e.g., injecting defects into the product).

Another method of measuring individual skill adoption and training success is to conduct post-training surveys. If a student professes to be performing the skill as taught, then it may be assumed that training was basically successful. However, some performance monitoring and evaluation is prudent to ensure the student performs the new skills effectively. If the student has not had a chance after several months to perform the skill or has refused to perform the skill for whatever reason, then the success of the training is questionable at best and could potentially be considered a total waste of time. If the organization has adopted supporting practices wherein individual skills can be practiced and observed, the likelihood of achieving training success is increased.

A key measure of training success is the level of adoption of organizational practices that are taught. Adopting improved skills and practices is a significant challenge often requiring weeks of effort to plan, define, pilot, and implement. However, without a concerted adoption effort, many useful practices never get off the ground in some organizations. Their method of improving practices never seems to get beyond the *desire to improve* stage.

To continue the earlier example regarding adopting disciplined document review practices, it is generally easy to understand the mechanics (the process) of a disciplined document review. Also, individual skills to effectively participate in disciplined document reviews are fairly easy to learn. Understanding the documents under review is certainly the hard part. However, when people are first introduced to disciplined document reviews, they are often amazed at the number of defects they find in their technical documentation. Without the process infrastructure, people often revert to poor review practices such as

Measuring Training ROI

Just because measuring a return on investment (ROI) for training may be difficult does not mean we should not try to do it. One method for measuring training ROI is to try to measure resultant project savings following training. A Software Engineering Institute study observed that "it is not always possible to attribute improvements entirely to training" [1]. However, we might be able to estimate an ROI for training and process improvement combined.

How do you measure the ROI for process improvements? One perspective is that you need to know something about the size of the effort and amount of time and resources it costs to perform that effort. Then you need to know how the new or improved skills and practices improve the organization's capability to perform that effort. This means that you need to estimate and then track short-term and long-term gains and losses from the improvement effort. ROI changes throughout the project life cycle. Initially it is often less than one (costs more to change than the value received) but then it grows to be greater than one (gets more value than the investment). Some skills and practices show an ROI greater than one fairly early such as disciplined document reviews [9, 10, 11].

One method for determining ROI is to estimate the number of defects created by current practices and the effort it takes to fix those defects, i.e., determine the cost of rework before improvement (CRB). Then measure the number of defects after adopting new and improved skills and practices and the effort it takes to fix these defects, i.e., determine the cost of rework after improvement (CRA). Then normalize CRB and CRA to reflect the amount of rework per 1,000 lines of code (KLOC [or some other relevant normalizing factor]). I used KLOC and the sample numbers below as an example.

Normalized CRB (NCRB) = $150,000 / 25 KLOC = $6,000 / KLOC (before improvement)

Normalized CRA (NCRA) = $100,000 / 50 KLOC = $2,000 / KLOC (after improvement)

Next, calculate the normalized cost of training and process improvement (NCTP). If $50,000 was invested in training and process improvement and 50 KLOC received the associated benefits, then:

NCTP = $50,000 / 50 KLOC = $1,000 / KLOC

Finally, calculate an ROI for training and process improvement (ROITP):

ROITP = (NCRB - NCRA) / NCTP = ($6,000 - $2,000) / $1,000 = 4

In this example, $4 were saved for every dollar invested in training and process improvement. This only considers the current project. Conceptually, the process only needs to be changed once to implement an improved capability. However, training is needed for each new staff member as he or she enters the organization, i.e., it happens as needed. The ROI will increase as additional projects receive the benefits of the training and process improvements. If you lose people and you have to train new people, then the ROI changes.

The key point and basis for ROI estimates is that defects cost a lot in terms of time and money if we do not find and fix them early. We should be able to estimate current practices and compare them to new practices that have been implemented in the organization and on which the staff has been trained. Thus, rework avoidance should help us gain a realistic perspective of the value of training and improved practices.

skim reviewing for finding defects. Skim reviewing is briefly reading a document without taking the time to (1) check for consistency and completeness against all source information and to (2) check against appropriate evaluation criteria (e.g., checklists).

Some discipline is always required to effectively perform organizational processes. Unless the organization adopts a business process infrastructure within which employees can perform many learned skills, training cannot be successful because many skills will not be performed. I call this *The Process Not in Place to Support the Training* dilemma. The following section addresses a few other training dilemmas.

Training Dilemmas

I have observed several dilemmas over nine years of providing software quality and test-related technology training. The following training dilemmas have inhibited adoption of individual skills and organizational practices.

They Came, They Taught, They Left, and Nothing Changed Dilemma

Managers often have not assessed training needs adequately before preparing plans for implementing needed skills and practices. Trainers often have not advocated planning for implementing individual skills and organizational practices that they teach. Students often have not made a sincere effort to learn and practice the skills they are taught. Many organizations need training on how to receive training effectively. Students should know their objectives and strive during and after class to achieve them. In other words, students need to be proactive learners to support their organization's training goals. Getting the most value from training is a shared responsibility between trainers and managers and their staff.

The SEI reported, "When employees were involved in the training process and the associated needs analysis, they felt that they were getting skills improvement that would be beneficial to them in their careers" [1]. Managers need staff input in skill needs analysis and process improvement planning. Effective training must consider what should happen after class. Maintaining the status quo back on the job will not achieve the ROI desired from training. Managers and students must plan to change and implement appropriate improvements to minimize this dilemma.

On-Site Training Room Dilemma

When training is held in the same building where the employees work, they will often arrive late or return from breaks and lunches late because they go to their desk and are caught up in the normal work activities. In addition, when managers give extra assignments or require previously assigned projects to be completed during the training period, employees may miss several hours of important instruction. Working long (often unpaid) hours into the night to finish projects can take its toll on employee comprehension and participation. Managers should not require their employees to work on projects during the period that they are attending class. Also, managers should not require more than a normal day's effort (eight hours maximum) during training periods.

Effective training can occur in on-site training rooms but it often requires a concerted effort to avoid the temptations to continue project effort during training. Trainers should include some fun incentives to encourage on-time attendance such as providing interesting (but not essential) information in the first five minutes after breaks or lunch [3].

I Am Here Because I Was Told to Be Here Dilemma

Occasionally, a trainer is blessed with one or more students who come to class with their arms folded and a look on their face that challenges the instructor to try to teach them anything. You know the type – they were told to be there, and they would obviously rather not be.

It is the trainer's responsibility to hold attendee attention by providing interesting and informative material and experiences. All effective trainers continue to work on that. However, the attitudes of some students can infect others to the point that progress can be inhibited. Establishing training objectives ahead of time can help with this dilemma. If some staff members do not want to attend, then maybe there are underlying issues that need to be dealt with before training will be effective.

Martha Kelly, course leader for several Langevin Train-the-Train courses, said:

> In any training you conduct, the learner should work harder than the facilitator. Training is a place where people come to practice their jobs. It isn't prison; it isn't a vacation; it isn't home. It's an extension of the office were real-world problems should be discussed and potential solutions to problems should be learned and practiced. [4]

One way to overcome a complacent attitude is for trainers to show they sincerely care about attendees' issues and problems. People often will not listen to you until they know you care about them. Trainers should get to know attendees, when the class size and time permits, by talking personally with each attendee at breaks, lunch, and after hours. This can go a long way to overcoming this dilemma. Activities to help students get to know each other can help as well.

Bad News Dilemma

Sometimes it does not take much to turn a group of students against a trainer. Bad news travels faster than good news. Even experienced, highly entertaining trainers can have difficulty answering some questions, leaving some students dissatisfied. If a student then chooses to share his or her opinion or concerns with others, the door to communications and effective teaching closes to some degree with the other students as well. What can be done in this case? Trainers could answer all questions correctly with evidence to back claims and do so delightfully. That is a tall order.

Certainly, if a trainer does not know the answer to a question, he or she should say so. Langevin's course on Advanced Instructional Techniques reminds trainers that they should not take themselves too seriously, and that they should adopt the role of a leader and guide rather than an expert [4]. Opinions should be expressed as opinions rather than facts to help students understand that we are all still learning. When an opinion is not *sold* adequately to a student, we can then agree to disagree and move on. Sometimes trainers can invite other attendees to respond or give added attention to the matter after class hours. The point is that we need to encourage students to discuss their concerns and not harbor them with resentment.

Not a Jay Leno Dilemma

How much value is the entertainment factor in instructing technical courses? As mentioned above, it is the trainer's responsibility to hold student attention by providing interesting and informative material and experiences. However, most students today have always had ready access to TV and the movies. Usually student expectations are high with respect to the entertainment factor in the courses they attend. Some trainers move into this industry with little prior experience in training let alone in stand-up comedy. All of a sudden, these new trainers not only have to teach someone how to perform a technical skill in the context of an organization's business practices, they have to be sensational to the students they are teaching.

Part of the answer is for trainers to receive training on becoming more effective and engaging. Perhaps part of the answer may also reside in students somehow valuing the technical content of the courses a little more than the entertainment factor. This could be done in part through a measure of the student adoption of new skills. Again, the ability of the instructor to affect attitudes is vital in adopting new skills.

There are dozens of books dedicated to increasing the *entertainment factor* in training. They are often based on the actual experiences of practicing trainers and many of their ideas are great. But when it comes down to it, each trainer needs to be authentic by being themselves and not someone else. Trainers can be taught to increase their level of animation to be more engaging and entertaining. Trainers do not need to be a Jay Leno-type entertainer to be effective. The key is certainly the level of enthusiasm a trainer shows for the material being taught.

No Management Endorsement Dilemma

Having no policies, no champion, no process, or no improvement plans are each indicative of a lack of management endorsement. Policies are required to identify and establish management support for key business practices [5]. Champions are needed to demonstrate capabilities and get people excited about new and improved skills and practices. Active management involvement in identifying and empowering process improvement leadership is vital to success. Documented processes are required to establish (1) the sequence of events or phases to be performed, (2) the associated entrance and exit criteria for those phases, (3) the inputs provided and the outputs expected from each process phase, and (4) key measures to be collected to evaluate the process success [6]. Improvement plans are needed to change old business practices to new.

Organizations need to consider the difficulties inherent in changing an organization's way of doing business. See the SEI's IDEAL[SM] Model for information on effectively changing processes [7]. The SEI has also published a People Capability Maturity Model that states, "The most common reason for the failure of improvement programs is lack of executive support" [8]. Get management endorsement for the specific training or do not train.

Responsibility

Make no mistake; the responsibility is squarely on the trainer's back to deliver effective workshops. However, too many managers have the mistaken perspective that their employees will *automatically* adopt skills that were learned as their new way of doing business. This may lead to the conclusion that the training was not effective if their employees and organization did not readily and *automatically* adopt what was taught.

Trainers should warn students that certain skills would require management endorsement and effort to become the

[SM] IDEAL is a service mark of Carnegie Mellon University.

Coming Events

March 1-3
17th Conference on Software Engineering Education and Training (CSEET 2004)
Norfolk, VA
www.cs.virginia.edu/cseet04

March 8-11
Software Engineering Process Group Conference 2004
Orlando, FL
www.sei.cmu.edu/sepg

March 29-April 1
Defense Technical Information Center Annual Meeting and Training Conference
Alexandria, VA
www.dtic.mil/dtic/annualconf

March 30-31
3rd Annual Southeastern Software Engineering Conference
Huntsville, AL
www.ndia-tvc.org/SESEC

April 19-22
2004 Systems and Software Technology Conference
Salt Lake City, UT
www.stc-online.org

May 17-21
STAREAST
Orlando, FL
www.sqe.com/stareast/

May 23-28
26th International Conference on Software Engineering
Edinburgh, Scotland
www.jupiterevents.com

accepted practice. Trainers can also prepare students to effectively adopt improved practices. However, process improvement is a business decision that many managers have not delegated, and unless management initiates a process improvement effort (which they often do not because of product delivery pressures), processes remain as status quo. This fundamentally means that the organization should not have acquired the training in the first place since all that happened was money and time were spent with no recognizable benefit.

Recommendations

Organizations that are aware of the issues surrounding these training dilemmas work toward gaining more value from their training decisions. Many training dilemmas were not listed in this article. However, many publications exist that identify similar and other training challenges and how to deal with them. Be careful not to make the assumption that the burden for training success is completely on the trainer, and that students automatically know how to get the most value from training. These are dangerous assumptions no matter how much training has been acquired.

Many organizations could benefit from training in how to receive training effectively. Train-the-trainer programs can answer this need by teaching students and managers how to get more value from training. These programs should not just be for trainers. Train-the-trainer programs can help students learn how to be better students.

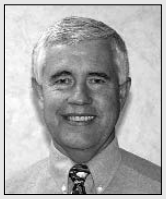
Do not expect the status quo to help your organization remain competent and competitive. Acquire training that gets adopted and makes an economic difference.◆

References

1. Mead, Nancy, et al. Best Training Practices Within the Software Engineering Industry. Pittsburgh, PA: Software Engineering Institute, Nov. 1996.
2. U.S. Air Force. Guidebook for Air Force Instructors. Air Force Manual 36-2236, Personnel, 15 Sept. 1994.
3. Pike, Bob. Dealing with Difficult Participants. Creative Training Techniques Press, 1997.
4. Kelly, Martha. "Advanced Instructional Techniques: The Art of Facilitation." Training Course, 12-14 Nov. 2002.
5. CMMI Product Team. Capability Maturity Model® Integration, Ver. 1.1, Continuous Representation. Pittsburgh, PA: Software Engineering Institute, Jan. 2002.
6. Perkins, Timothy K. "Process Definition Workshop." Ver. 1.4. Software Technology Support Center, Hill Air Force Base, UT, 14 Sept. 2000.
7. Gremba, Jennifer, and Chuck Myers. "The IDEAL℠ Model: A Practical Guide for Improvement." Bridge (3) 1997 <www.sei.cmu.edu/ideal/ideal.bridge.html>.
8. Curtis, Bill, et al. People Capability Maturity Model℠. Pittsburgh, PA: Software Engineering Institute, Sept. 1995.
9. Daich, Gregory T. "Disciplined Document Reviews Course." Ver. 2.4. Software Technology Support Center, Hill Air Force Base, UT, 12 Jan. 2001.
10. Gilb, Tom, and Dorothy Graham. Software Inspections. Addison-Wesley, 1993.
11. Dion, Raymond. "Process Improvement and the Corporate Balance Sheet." CrossTalk Feb. 1994.

® Capability Maturity Model and Capability Maturity Model Integration are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

About the Author



Gregory T. Daich is a senior systems engineer with Science Applications International Corporation (SAIC) currently on contract with the Software Technology Support Center (STSC). He supports STSC's Software Quality and Test Group with more than 26 years of experience in developing and testing software. Daich has taught more than 100 public and on-site seminars involving software testing, document reviews, and process improvement. Daich has also developed two Air Force training programs: Software-Oriented Test and Evaluation, and Disciplined Document Reviews. He has a master's degree in computer science from the University of Utah.

Software Technology Support Center
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056
Phone: (801) 777-7172
Fax: (801) 777-8069
E-mail: greg.daich@hill.af.mil

Ten Key Techniques for Process Improvement Consulting in a Challenging Environment©

Sarah A. Sheard, Suzanne Zampella, and Albert J. Truesdale
Software Productivity Consortium

When organizations put extreme pressure onto process improvement success, sponsors and change agents have little room to experiment and even less room to fail. Consulting in these environments can feel like a no-win situation, both for the employees and for the consultant. The consultant must discover all objectives and be genuine in responses. The consultant needs to take advantage of his or her position outside of internal politics to tell the truth, both negative and positive. Patience and a willingness to take the heat will go a long way, as will working in pairs and being willing to cease the engagement altogether. This article describes 10 key techniques to handle the difficulties.

Some organizations fear losing a competitive edge when their competitors achieve a high maturity level on a capability model. Therefore, these organizations seek to achieve a similar or higher level on a fast track. Demonstrating a capability model Level 1 attitude, management believes that by pushing individual heroes to work very hard, they can make the impossible happen [1]. Consequently, a software (or systems) engineering process group (SEPG) may request consultants to help meet process improvement goals while dealing with this extreme pressure.

Symptoms of a challenging environment that appear in assessment team members, sponsors, and interviewees include the following:

- Lying to assessors and to themselves about progress.
- Denying that model requirements apply.
- Arguing with an assessor or instructor that certain practices in the model ought not to be considered requirements[1].

In some fast-track process improvement organizations, true improvement is perceived as negative, rather than as the goal. They see the ideal situation as achieving the banner without changing the way work is done.

Other organizations have a difficult culture: Incentives are awarded to individuals, and each employee seeks solely to improve his or her own standing. Executive challenges to work as a team are perceived as patronizing attempts to fool the masses. Managers are told, "Find a way," and that way often involves deceit (e.g., back-pocket schedules) and exhortations to heroism (e.g., extremely long hours *just this once*).

Another challenging environment occurs when management has what employees call a Silver Bullet of the Month [2]. Employees perceive a process improvement effort to be temporary, only to be abandoned as soon as a banner is achieved. This leads to poor engineering of process documentation, which is then unusable, fulfilling the prophecy.

Techniques

Below are ten key techniques consultants can use to help make a process improvement effort successful. The second through fourth techniques are basic consulting techniques that are most critical in a challenging environment.

> *"The consultant therefore has a responsibility to the client to be truthful, especially when the client may not want to hear the truth."*

1. Be Aware of Explicit and Hidden Agendas

Known objectives are the primary key to success. Most process improvement efforts will explicitly set an objective for a higher capability maturity level, but secondary objectives are often hidden. Consultants who behave as if the explicit objective is all there is are likely to experience resistance or even sabotage when they suggest actions that go against hidden objectives.

Typical secondary objectives include a manager or the SEPG members earning a bonus if the effort succeeds, or being fired if it does not; the organization being able to market at the same level as competitors; and bidding on particular contracts that require a specific maturity level. The authors are also aware of one company that wanted a maturity level because the company was about to be sold. A successful consultant will be aware of these secondary objectives, and will try to meet them, to the extent they are ethical and not in conflict with true improvement.

Internal politics are another kind of hidden agenda. Any true change will modify someone's field of control. If that person perceives this as negative, he or she will fight hard to prevent or reverse the change. It is important to understand who these people are, how hard they are likely to fight back, and how and what can be done about it.

2. Be Authentic

Peter Block, in "Flawless Consulting" [3], discusses the two most important questions[2] consultants should ask themselves. The first, which is most important in a difficult environment, is, "Am I being authentic with this person now?" In addition to requiring truth as described in the next technique, authenticity requires the consultant to understand and express his or her own feelings and hesitations. This includes uncomfortable feelings like, "You are excluding me from the decision-making process," or "I feel I am being seen as a judge on this project, and that is not the best role for me."

Most managers know, but do not say, that they know more than any consultant does about their own particular business. It is refreshing to have the consultant confirm that. The consultant should then explain his or her specific role in the engagement; typically, the role is bringing in a fresh pair of eyes, highlighting risks, and suggesting practices that have worked in other places.

3. Call It Like It Is

In a difficult organizational climate, a consultant has the advantage of being an outsider. An outsider need not bow to internal politics, and can speak the truth that insiders are afraid to say. An outsider cannot be hurt (much) by innuendo and is not required to appear to be a team player. The consultant therefore has a *responsibility* to

© 2003 Software Productivity Consortium NFP, Inc.

the client to be truthful, especially when the client may not want to hear the truth. "No, your people are not praising your actions." "No, I do not think this plan is reasonable." "No, I don't think it's a good idea to tell the group a date you think is unreasonable so they'll work harder. They will just get cynical when you slip it again." Sponsors may not act like they appreciate such honesty, but it really is what they pay for.

Asking targeted questions may appear less confrontational than simply stating opinions, yet make the same point. Edgar Schein, in "Process Consultation" [4], suggests questions that focus on the project's goals whenever criteria for decisions become vague or when the group appears to be bypassing consensus. Questions such as, "How will unilateral pronouncements on our part help convince employees that their input is required?" may cause the sponsor to rethink some plans.

4. Support and Encourage

Most clients who hire a consultant are at least a little insecure about their ability to do the work. Especially after the euphoria of starting an effort, when the shock of reality hits, the consultant must reassure the client that this is part of the process; he or she is doing well and will get through it. There is no reason to despair. Specific examples of things that the client did well are welcome, reminding the client that worse things would have happened had the client not intervened. The consultant needs to be quite genuine and preferably specific in the encouragement, but also flexible.

For example, an intelligence agency senior manager asked one of the authors what he should do about emphasizing process improvement after Sept. 11, 2001. The consultant replied that although process improvement all too often is put on hold for fire fighting, there are exceptional times when there is no choice but to let emergencies take over. An organization has to react decisively and quickly, putting strategic efforts like process improvement on hold. The consultant reassured the client that what he was doing was as good as could be expected. The consultant reminded him that the important thing is not to drop process improvement altogether, but deal with the emergencies first and refocus on process improvement later.

5. Repeatedly Reset Expectations

Clients in denial, or clients in a hostile culture, frequently forget what you have told them, particularly when they did not want to hear it in the first place. Consequently, it is important for the consultant to revisit frequently the expectations he or she has of the organization and the improvement project. For example, "You realize, with this time frame so short, we can't do a pilot before rolling out to the organization? That means the projects may well find major flaws in the documentation that would require a rerelease and a slip in the appraisal. Remember, we talked about this when we made the original plan."

6. Take the Heat

Most process improvement efforts in challenging environments will benefit from agreement between the insiders (say, the SEPG), and the consultants. A smart consultant will team up with internal change agents. If the insiders cannot afford to say something that needs to be said but may be threatening, then the consultant should say it. For example, "Compared to other organizations I have seen, this organization has a lot of duplicity and deceit in its actions, for example …" Then the SEPG can be seen as the *good guys* responding to the bad news. It is more predictive of lasting improvement if the organization perceives the insiders as being helpful.

> *"If the insiders cannot afford to say something that needs to be said but may be threatening, then the consultant should say it."*

7. Double-Team

Having two or more consultants really helps in some situations. Of course the two consultants must plan in advance to be sure to say the same thing, but the advantage is, the story is much more likely to be believed if two different people are saying it. Furthermore, when one is worn down with arguing, the other can take over and provide a break.

Another technique that has worked, even unintentionally, is a variation of the approach known as *good cop/bad cop*. One author naturally tended to be considerate and flexible, but when backed into a corner, could call on her colleague who could be relied upon to come down hard in that situation. "Look, do you want to deal with me or *him*?"

8. Nudge-Nudge-Kick-Repeat

A basic mantra with difficult situations, the *nudge-nudge-kick-repeat* technique uses repeated gentle nudging toward the right answer, with occasional stronger attempts to get attention. The consultant must be extremely patient to continue to nudge as a matter of course.

Nudges are gentle reminders of the right way. A consultant should frequently remind the client of the point of process improvement and the need for buying in. Additionally, a consultant can review the way the effort has occurred to date and suggest different activities that may accomplish goals more easily. Nudges tend to sound like, "Typically clients find that …"

Kicks are more direct. "We told you six months ago to XXX. You said you would. Now you still haven't done anything, and as a result you have created YYY problems, and you're also four months behind schedule. Are you interested in this effort or not?" Note that the more specific the detail in such a statement, the better.

By understanding that the need for continuous nudges and occasional kicks is normal, the consultant can take in stride the number of times this technique needs to be repeated.

9. Document

In some very challenging environments, consultants can run into *process improvement saboteurs*. Intentionally or unintentionally, these people attempt to undo what the consultant recommends. Some take the form of passive-aggressive resistance, namely, appearing to agree and comply but working hard off-line to make sure that the recommendations are not implemented. The authors have also seen instances of internal team members consistently misrepresenting the statements of consultants: "My SEPG team leader tells me you said we wouldn't need to document these things." ("I most certainly did not!") Although it seems defensive and reactive to cover oneself with memorandums, in such cases as consistent misrepresentation the consultant does have to act.

A common technique is for the consultant to write minutes of the meetings that occur and send them to everyone on the team. This way the consultant can be sure everyone has the consultant's exact words. Perhaps a better technique is to have the insiders document the minutes, insisting that they be sent to the consultant for concurrence. This way, the insiders are not only taking charge (and getting practice documenting decisions), but also

realizing over time that they need to listen to the consultant's views and accurately represent them, or they will have to keep doing the minutes over and over again.

10. Be Willing and Ready to Bow Out

A consultant's greatest negotiating power comes when he or she is most free to leave the table altogether. In a difficult engagement, feeling able to terminate conveys additional power to compel the client to be reasonable. "I have to follow the assessment rules. If you want to break them, you will need to find a different consultant." Besides, a good consultant does not want his or her name associated with an effort that is so doomed to failure that the most important points cannot be implemented. Always having other work to do allows the consultant to be firm with any particular client.

® CMM and CMMI are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

SM SCAMPI is a service mark of Carnegie Mellon University.

Conclusions

There are some stubborn cultures that even excellent consultants cannot affect. Short of impossible situations, however, consultants can use these techniques to encourage clients to do what needs to be done. General support and encouragement go a long way. Truth allows the client to believe that the support is genuine. Truth also provides the client with bad news, sometimes saving face for the insiders. Occasionally, the consultant has to deal with denial more strongly, sometimes by teaming up, and sometimes by fairly direct statements of detailed fact. Finally, the consultant needs to have an alternate plan; it keeps paychecks coming, and makes the consultant's advice more compelling.◆

References

1. Sheard, Sarah A. What Is Senior Management Commitment? Proc. of the 11th International Symposium of the International Council on Systems Engineering, Melbourne, Australia, 2001.
2. Sheard, Sarah A. "Life Cycle of a Silver Bullet." CROSSTALK July 2003: 28-30.
3. Block, Peter. Flawless Consulting: A Guide to Getting Your Experience Used. San Diego: Pfeiffer & Company, 1981.
4. Schein, Edgar H. Process Consultation, Volume II: Lessons for Managers and Consultants. Reading, MA: Addison-Wesley, 1987.

Notes

1. Features of both the Standard CMMI® Appraisal for Process Improvement (SCAMPI℠) and its predecessor, CMM®-Based Assessment for Internal Process Improvement (CBA IPI), were instituted to prevent these symptoms from giving false positive assessments.
2. The second question is, "Am I completing the business of the consulting phase I am in?"

About the Authors

Sarah A. Sheard is currently a chief technologist leading the systems engineering effort at the Software Productivity Consortium. She received the 2002 INCOSE Founder's Award for her work in INCOSE, including publishing over 20 symposium papers, chairing the Measurement Technical Committee and the Communications Committee, and serving as program chair and director of the Washington Metropolitan Area chapter. Sheard has worked in systems engineering and process improvement for more than 20 years.

Software Productivity Consortium
2214 Rock Hill RD
Herndon, VA 20170
Phone: (703) 742-7106
Fax: (703) 742-7350
E-mail: sheard@software.org

Suzanne Zampella is a lead appraiser for the Capability Maturity Model® Integration (CMMI®) and is an authorized instructor for the Introduction to CMMI course at the Software Productivity Consortium. She has 20 years of experience in software engineering, process engineering, project management, teaching, and consulting. Between appraisals and teaching assignments, Zampella develops courses such as Transitioning to the CMMI and maintains the consortium's Quagmap mapping tool.

Software Productivity Consortium
2214 Rock Hill RD
Herndon, VA 20170
Phone: (703) 742-7139
Fax: (703) 742-7350
E-mail: zampella@software.org

Albert J. Truesdale is a member of the technical staff at the Software Productivity Consortium (SPC) where he consults with members and customers on process improvement and systems and software engineering activities. Truesdale is an SPC-authorized Lead Systems Engineering Assessor, and Software Engineering Institute-authorized Capability Maturity Model® Integration (CMMI®) Lead Appraiser and CMM®-Based Appraisal for Internal Process Improvement Lead Assessor. He participates in systems engineering and software capability maturity model assessments, teaches team workshops and other courses offered by SPC, and prepares guidebooks and related instructional materials for SPC member organizations. Truesdale has more than 30 years of experience in systems engineering and software intensive system development assignments.

Software Productivity Consortium
2214 Rock Hill RD
Herndon, VA 20170
Phone: (703) 742-7306
Fax: (703) 742-7350
E-mail: truesdal@software.org

Army Taps Vernon M. Bettencourt Jr. as Next Deputy CIO/G-6

Patrick Swan
U.S. Chief Information Office

Vernon M. Bettencourt Jr. has been named deputy chief information officer (CIO)/G-6 for the U.S. Army. As the deputy CIO/G-6, Bettencourt will run the day-to-day operations for the office of the CIO/G-6 and provide senior-level advice to Lt. Gen. Steven W. Boutelle, the Army's CIO. Bettencourt was previously the director of Analysis and CIO to the deputy chief of staff, G-3, for the Department of the Army. He began work at CIO/G-6 on Nov. 17.

Lt. Gen. Steven W. Boutelle, the Army's chief information officer (CIO)/G-6 announced today that he has selected Vernon M. Bettencourt Jr. as his deputy.

Bettencourt is currently serving as the director of Analysis and CIO to the deputy chief of staff, G-3, for the Department of the Army. He ensures that Office of Deputy Chief of Staff G-3 priority information requirements are identified and effectively supported using analysis; analytic simulations; information management; and command, control communications, and computers for information infrastructure systems. Bettencourt provides senior analysis support to the Force Development director of the deputy chief of staff, G-8.

As the deputy CIO/G-6, Bettencourt will run the day-to-day operations for the office of the CIO/G-6 and provide senior-level advice to Boutelle.

Bettencourt succeeds David Borland, the Army's current deputy CIO/G-6, who retires at the end of this year after more than 30 years of federal service.

Bettencourt began work at CIO/G-6 on Nov. 17.

"We are extremely fortunate to bring Vern Bettencourt on board," Boutelle said. "He brings a vast understanding of the issues involved with transforming the Army into a knowledge-based, network-centric force. He will be a key player in our efforts to realign and optimize our networks to support the joint warfighter."

"I'm excited to be joining the CIO/G-6 team," Bettencourt said. "Networking the force is a key focus area for the Army chief of staff and is an essential element in winning the global war on terrorism. I look forward to this challenge."

From June 1998 to February 1999, Bettencourt served as director, U.S. Army Modeling and Simulation Deputy Chief of Staff, G-3, where he provided vision, strategy, oversight, and management of modeling and simulation across the Army. From October 1995 to May 1998, Bettencourt served as special assistant for Forces and Program Evaluation, deputy under secretary of the Army (Operations Research). There he was responsible for policy, oversight, and guidance of analyses, test and evaluation, and experimentation activities associated with force structure requirements and readiness, selected materiel systems, and Army plans, programs and budgets.

Upon appointment to the senior executive service in 1995, Bettencourt served as the assistant deputy chief of staff for Combat Developments, U.S. Army Training and Doctrine Command (TRADOC) from January to October 1995. He was responsible for combat development and acquisition plans, programs, policies, and procedures in Training and Doctrine Command and their integration into Department of the Army and Department of Defense acquisition systems.

In 1990, Bettencourt joined the MITRE Defense Command, Control, Communication (C3) Intelligence Federally Funded Research and Development Center where he founded the Synthetic Environments Applications Department, designing and applying simulations to support systems engineering for sponsors in the Office of the Secretary of Defense, Army, Navy, and U.S. Marine Corps. He then became the Army Program area manager for the Information Systems and Technology Division, coordinating the efforts of scientists and engineers supporting the Army with information technology. His final position at MITRE was associate director of Army Programs, where he was responsible for strategic planning, development, and control of C3 intelligence, surveillance, reconnaissance support to the Army.

Bettencourt's final assignment in a 20-year Army career was director of TRADOC Analysis Center-Monterey, where he supervised simulation research. Prior to that, he served as military assistant to the deputy under secretary of the Army from 1985 to 1988. During his Army officer career, Bettencourt also served in various field artillery command and staff assignments in Germany, Vietnam, Korea and the continental United States. He earned his commission from the U.S. Military Academy at West Point, N.Y., has a Master of Science in operations research from the Georgia Institute of Technology, a Master of Business Administration in finance from C. W. Post University, and is a licensed professional engineer in industrial engineering.

He is a past president and a current fellow of the Military Operations Research Society, has chaired the Military Operations Research Society annual symposium and several workshops, and is a member of several academic and managerial honor societies. Bettencourt has been published and presented papers in numerous national and international operational readiness, management, and military forums. Additionally, he has received the Senior Executive Services Presidential Rank Award of Meritorious Executive.

For more information, contact Patrick Swan, public affairs officer for the Army's Chief Information Office/G-6, at <patrick.swan@us.army.mil>.◆

"Networking the force is a key focus area for the Army chief of staff and is an essential element in winning the global war on terrorism."

— Deputy Chief Information Officer Vernon M. Bettencourt Jr.

The Human Dynamics of IT Teams

Jennifer Tucker and Abby Mackness
Booz Allen Hamilton

Hile Rutledge
OKA

This article presents the results from a recent research study investigating the personality dynamics of information technology (IT) professionals and teams. Our research identified diverse needs within IT teams: strong needs for objectivity and individual contribution and independence, coupled with a desire for effective team relationships and cohesion. This article focuses on study results in more detail and highlights implications for IT leaders.

Much has been written about the importance of teams that gel among information technology (IT) professionals. For many, a gelled team symbolizes the power of team dynamics in today's complex and uncertain environment. Unfortunately, many IT leaders find this level of team connection and performance easier to envision than to achieve. Furthermore, the team-based vision does not always connect with the more commonly cited portrait of the IT worker – as a lone professional preferring to work independently and in relative isolation, as long as someone occasionally shoves pizza under the door.

This dichotomy of visions – a *gelled team versus a lone coder* – recently motivated the Defense Acquisition University to support a research study to examine the intricacies of IT dynamics at both the individual and team levels. This research study investigated the personality and team dynamics of 621 IT professionals working in 77 IT teams [1]. It assessed quantitative and qualitative variables related to IT demographics, personality, success factors, workplace satisfaction, retention factors, and communication patterns, resulting in a unique perspective on the IT individual and team.

Ultimately, the study confirmed the importance of human dynamics within IT teams but offered a new perspective on factors contributing to team success. The study concluded that IT professionals have a statistically different personality composition than the general population, and share unique perspectives on the effective working relationships that may lead to teams gelling. The following is a summary of key findings of the study.

Study Overview

Several IT writers (e.g., [2, 3, 4, 5]) have recognized the importance of personality characteristics on team performance and success. We used three published instruments, introduced below, to quantify these personality characteristics among IT professionals. We also designed two additional surveys: one to gather information about each team and another to quantify team communication processes and patterns.

The Myers-Briggs Type Indicator Assessment

The Myers-Briggs Type Indicator™ (MBTI™) Assessment, built upon the theories of psychological type, was used to both describe IT personality dynamics and to contrast the distribution of IT personality types against the general population. The MBTI sorts an individual's personality preferences based upon four distinct dichotomies. Table 1 lists the preferences associated with each [6].

Four specific pairings of the MBTI preferences also result in four unique temperaments, which map well to the following specific behavioral styles [6]:

- **Sensing Judgers (SJ):** Stabilizers – preferring structure, order, accountability, reliance on existing systems, policies and procedures, and the proven way of doing things.
- **Intuitive Thinkers (NT):** Visionaries – preferring non-conformity, theory, conceptualization, independence, objective complexity, and change for the sake of change if it produces learning.
- **Intuitive Feelers (NF):** Catalysts – preferring interpersonal support, relationships, possibilities for people, interaction, cooperation, imagination, and supportiveness.
- **Sensing Perceivers (SP):** Troubleshooters – preferring hands-on action and experimentation, practical solutions, variety and change, immediacy, flexibility, and adaptation.

Fundamental Interpersonal Relations Orientation-Behavior Survey

The Fundamental Interpersonal Relations Orientation-Behavior™ (FIRO-B™) Survey focuses upon interpersonal needs, and was used to investigate how IT professionals typically behave toward other people, and how they generally expect others to behave. The instrument assesses three scales along two dimensions, described in Table 2 (see page 16) [7].

Work Environment Scales

The Work Environment Scales (WES) broadens the view to the IT team level and reports information about the workplace *social climate* [8]. Ten WES scales assess worker satisfaction across a broad range of dynamics, grouped into three key categories (see Table 3 on page 16). The instrument assesses these along two parallel dimensions: perception of the *real* work environment (as things are), and perception of the *ideal* work environment (how the respondent imagines the perfect workplace to be).

Table 1: *Myers-Briggs Type Indicator Personality Scale Descriptions*

	Scale Descriptions	
E/I: Energy Source	**Extravert (E)** Gain energy from outer world of people, action, and things.	**Introvert (I)** Gain energy from inner world of concepts and ideas.
S/N: Perceiving Function: "Data Gathering"	**Sensor (S)** First perceive the immediate, practical, real facts of experience. Collect here and now sensory information.	**Intuitive (N)** First perceive possibilities, patterns, and meanings of experience. Collect information through impressions.
T/F: Judging Function: "Decision Making"	**Thinker (T)** Objective decision making. Seek clarity by detaching from problem; cause-effect oriented.	**Feeler (F)** Subjective decision making. Seek harmony with inner values by going within problem.
J/P: Outer World Orientation	**Judger (J)** Show external world judging mental function. Prefer to live in a decisive, planned way.	**Perceiver (P)** Show external world perceiving mental function. Prefer to live in a spontaneous flexible way.

™ Myers-Briggs Type Indicator and MBTI are registered trademarks of the Myers-Briggs Type Indicator Trust.
™ Fundamental Interpersonal Relations Orientation-Behavior and FIRO-B are registered trademarks of CPP, Inc.

	Inclusion	Control	Affection
Expressed	Extent to which you feel need to include others in activities.	Extent to which you feel need to exert control and influence.	Extent to which you feel need to express warmth and closeness.
Wanted	Extent to which you want others to include you in activities.	Extent to which you want to be in well-defined situations.	Extent to which you want warmth and closeness from others.

Table 2: *FIRO-B Scale Descriptions*

Now, we turn to the findings from the study.

The Diversity of IT Teams

Ten years ago, a development team consisted primarily of programmers. Today, the typical team includes a broad range of technical specialists – many of whom also play management roles. In our research, only 12 percent of IT professionals reported their role as programmer or developer [1]. Conversely, 32 percent of IT team members report their job as a *leader* or *manager*, indicating the practical need for a broader skill set beyond the technical realm.

What is the impact? IT teams that once shared a common technical base for building relationships are now coming together from separate specialties and backgrounds within their own field. This diversity can lead to profound opportunities for collaboration, or to miscommunication and stovepiped efforts if not managed effectively by the IT leader.

The force of technological change has also focused shared expertise at the team level. "So rapid are technological developments that the core is now the team, the only unit small enough to maintain its intellectual edge" [9]. Unfortunately, our research revealed that the average IT team has been together for only two years, and 45 percent of teams have been together a year or less [1].

The following are the implications for IT leaders:

- Carefully focus on the staffing process; look for diverse *and* specialized technical skills sets and the ability to acclimate to and work within the new team.
- Foster active listening skills and critical thinking skills within your team; this is vital to effective communication among diverse specialists.
- Consider expanding training programs to include management and human dynamics skills; something not typically included in the technical curriculum.

The IT Personality and Behavioral Styles

Results from the MBTI Assessment and the FIRO-B revealed the following intriguing personality dynamics among IT professionals.

Objective Decision Makers

More than three quarters (77 percent) of our sample reported a preference for *thinking* decision-making, with only 23 percent preferring *feeling* decision-making. This is significantly higher than in the general population, where the split between these preferences is generally even. Thinkers, as they are termed, generally prefer logical, objective, impersonal decision-making, focused upon cause-effect relationships and the clarity that comes from objectivity (problem first, people second).

Lone Gun Professionals

Forty-one percent of the IT professionals surveyed reported being *introverted thinkers* (combination of introversion and thinking preferences), nearly twice the percentage in the general population. Introverted thinkers often prefer a *lone-gun* approach to work, often avoiding teams, collaborative efforts, and the training that support such structures. This group is least likely to engage and connect interpersonally with others, and may avoid creating personal bridges of trust and openness with colleagues. This finding was supported with results from the FIRO-B, with more than half (55 percent) of the IT professionals reporting low, or highly selective, wanted inclusion scores (a low need to be included in the activities of others) [1].

Conflicting Behavioral Styles

The two most prevalent temperaments among IT professionals are the *intuitive thinking* (NT) and the *sensing judging* (SJ) temperaments, accounting for 75 percent of the total group. These are represented at 27 percent and 48 percent in IT teams, respectively, compared with 13 percent and 39 percent in the general population.

Interestingly, these two behavioral temperaments are those that tend to conflict most often. SJ groups may value established tried-and-true policies and procedures, proven standards, chain-of-command accountability, and respect for organizational tradition. These groups may see the NT's as disrespectful of tradition, irreverent, and simply stirring up the pot by constantly reinventing the wheel.

NT groups may value systems that reward future-focused, innovative thinking, and loose structure with minimal formal procedures and policies. These groups may see the SJ's as the ball-and-chain traditionalists who stifle creativity by their inability to think outside the box [10].

Implications

What are the impacts of these findings? IT professionals often prefer objective, impersonal dimensions of a problem, and may focus too heavily on the technical realm of the IT problem, neglecting user-based concerns. For example, some introverted thinkers fail to always consider the impacts of new systems on the people of the receiving organization, and may need reminders to connect on a personal level with key stakeholders.

Teaching teams about the differences

Table 3: *Work Environment Scales Descriptions*

Dimensions	Scales
Relationship	Involvement - Concern and commitment to job. Coworker Cohesion - Friendliness and supportiveness. Supervisor Support - Management supportiveness.
Personal Growth	Autonomy - Self-sufficiency; individual decision-making. Task Orientation - Planning, efficiency and task completion. Work Pressure - Work demands and pressure.
System Maintenance/ Change	Clarity - Communication of policies and expectations. Managerial Control - Use of rules to keep control. Innovation - Emphasis on variety, change, new approaches. Physical Comfort - Pleasantness of physical environment.

that stem from NT and SJ temperaments can help leverage those differences for better balance. For example, a temperament-based perspective on the dotcom era suggests that an over-emphasis on loose structure and innovation (NT), without a balance of practical policy and procedure (SJ), may have contributed to the eventual failure of brilliant start-ups. Respecting both can mitigate this risk.

Understanding personality dynamics and control needs may provide useful insights in selling new IT initiatives to teams. For example, the acceptance of new IT process tools (such as the Capability Maturity Model®) is highly dependent on developer acceptance. A 1999 Software Engineering Institute research study found that IT process tool adoption can be linked to three key factors: perceived control over the work, perception of the new tool, and perception of the tool/process impacts [11].

These results also have important implications for the leader trying to build a team closely connected to its stakeholders. As the IT role shifts toward the team-based and user-driven nature of today's development environment, personnel may need to engage more with the *people side* of the preference equation to meet IT needs.

We can teach these skills. When IT leaders and professionals understand personality preferences and needs, they can use this to identify team strengths, potential blind spots, and potential interaction dynamics. A leader's first step is to first recognize, and then communicate the need for the *touchy-feely* dimension of the IT process – providing concrete objective evidence of the benefits gained from building strong interpersonal, communications, and team-based skills.

Perceptions of Success, the Ideal Team, and Team Needs

Team Success and Turmoil

One of the key factors assessed during our research was manager and team member characterization of team success. Specifically, we asked managers and team members to (1) classify their team as successful or in turmoil, and (2) select three factors driving that rating. Interestingly, both managers and team members selected the same top three factors regardless of whether the team was called successful or in turmoil:

- **Work together effectively.** (Or do not work together effectively, for teams in turmoil.)
- **On-time delivery.** (Or do not deliver on time, for teams in turmoil.)
- **High quality services.** (Or do not deliver/provide high quality services, for teams in turmoil.)

Needs for Autonomy and Independence		Needs for Cohesion and Support
MBTI: High proportion of objectively-focused professionals.	⟷	**Survey:** Ability to work together effectively a key success factor.
FIRO-B: Unusually low wanted inclusion needs scores and low expressed and wanted control scores.	⟷	**WES:** Teams desire a higher level of peer cohesion than currently seen.
WES: Teams desire a high level of ideal autonomy.	⟷	**WES:** Teams desire a high level of supervisor support and clarity.
FIRO-B: Successful teams report lower wanted inclusion scores and lower expressed control scores than teams in turmoil.	⟷	**WES:** Successful teams report higher supervisor support than teams in turmoil.

Figure 1: *Potentially Contradictory IT Team Needs*

These points deserve discussion. First, consider what is missing! Although meets and exceeds client/user requirements were available as factors for both managers and team members, these factors were not among the top five in either group. Consider this omission given the industry's growing emphasis on user involvement and the continuing struggle that IT projects experience with requirements creep and management!

Second, two of the key factors – time and quality – can be assigned metrics, making them easier to define and manage. Third, *working together effectively*, is harder to systematize and requires the team to ask, "What does working together effectively mean to us?" Ultimately, given the link between working together effectively and success, IT leaders who have asked this question may have taken an important first step to improving IT project performance.

The Real and the Ideal Work Environment

The link between working together effectively and team success was also reflected in the WES results. Our data reported that IT professionals have many consensus on what the *ideal* environment could be than on what the *real* environment is. This was particularly marked for variables related to team involvement and dedication, emphasis on innovation, and degree of supervisor support.

Many IT leaders have found the WES useful for initiating discussions about team improvement – if we know where we want to be (the ideal team), we have a starting point.

Understanding IT Team Needs

Figure 1 brings these findings together, reporting the interesting tension within IT teams: a desire for autonomy and independence, coupled with a desire for peer cohesion and support, i.e., "the team can gel, as long as I can work by myself."

Implications

Our study findings reveal an important question for IT leaders: How do we best support teams that clearly value effective team relationships, while also fulfilling strong needs related to objectivity, individual contribution, and independence?

First, recall our findings about the IT personality. As previously described, the high representation of thinkers and introverted thinkers, with collectively low inclusion needs, suggests a general team orientation toward independent activity and objective decision making. In fact, a common philosophy among those with these preferences is, "side by side is binding." This suggests that, for these individuals, when a group is dedicated toward a shared vision, high levels of team face time are not a prerequisite for success. With these types of teams, effective IT leaders often serve their team best through roadblock removal – provide the teams with the right tools, communicate the mission, and check to see that the barriers to effectiveness are removed.

In reality, many managers respond to team challenges by implementing control mechanisms. In fact, most improvement models are designed with the goals of controlling process so that it can be tracked, documented, and managed. Our research suggests that this regulation comes at a cost. Most IT professionals report low control scores on the FIRO-B, and want more personal autonomy (i.e., individual decision-making and self-sufficiency) than currently experienced.

Furthermore, teams with low needs for external control, led by a manager with high control needs, are more likely to report themselves as in turmoil.

Conversely, IT professionals report wanting a significantly higher level of clarity in their work than currently experienced. This means that they want to know what to expect, and want policies to be more explicitly communicated than they currently are.

Leaders understanding the tensions between control, clarity, and support stay focused upon providing a roadmap for the IT team, without dictating how to drive. This requires recognizing the fine line between *delineating* the road forward (describing all of its roadblocks and speed traps), and *directing* the team on how to drive in order to avoid them.

Observing IT Teams at Work

Building a Robot

Although most study data were gathered using self-report instruments, we also observed each participating team's communication patterns and process in a simulated development environment. Specifically, each team was given a Lego Mindstorm Robot Kit and these specifications: "Construct a robot that moves around the dark circle within 30 seconds, stops, reverses direction, and goes around the dark circle in the opposite direction, also within 30 seconds. You have 25 minutes. Creativity and elegance of design count."

This task poses a specific team challenge, for it is nearly impossible to build an automated solution in the time allotted. To be successful, the team must explore the terms *robot* and *moves around*. If the team agreed that the robot could be non-automated (since we did not give them a better definition) and manually moved any object around the circle and back, they generally considered themselves successful. If they assumed the term robot meant automated, they were unable to complete the process.

Here are the key findings from the robot exercise, and their connection to the WES instrument:

- Half of the teams never questioned the definition of the term robot, and proceeded with the assumption that a robot must be automated – an interesting commentary on requirements analysis!
- Teams were highly task-oriented, with approximately 85 percent of team behaviors oriented toward providing information, suggesting solutions, and task problem solving. Far less time was spent in maintenance behaviors such as encouraging others, offering words of support, building harmony, and compromising.
- Teams identifying a non-automated robot solution (50 percent) reported higher innovation scores (measuring openness to new approaches) on the WES than teams that did not.
- Teams successfully implementing a non-automated solution (24 percent of teams) reported lower managerial control scores (measuring extent to which manager controls team activity) on the WES than teams that did not.

Implications

Generally, teams quickly saw parallels between their daily work and their robot effort, leading to fruitful discussion about team dynamics, critical thinking, communication pathways, and requirements management. For example, teams with a strong task-focused approach to requirements elicitation recognized that they often do not engage in interpersonal, collaborative elicitation approaches that may be more effective with users. As a second example, teams discussing baseline assumptions about the term robot often raised the question, "What hidden assumptions aren't being discussed in our team?" The robot exercise can be a powerful tool for leaders wanting to spark discussion about these dynamics.

Conclusions

Today's IT professionals are no longer solely technical specialists; they are also educators, facilitators, and consultants working as teams with end users to solve business needs. Amid these new roles, IT teams are under increasing pressure to create and deliver products and services that are on time, within budget, and of high quality. This reality leads to the key concluding points from our research.

First, successful IT leaders know how to communicate effectively, manage conflict, and influence others. Because IT is an inherently group-oriented activity, leveraging interpersonal skills is a critical success factor in achieving specific goals. Our research shows that relationship management skills, not always taught in technically focused environments, need to be highlighted as a key capability in today's IT toolset.

Second, IT leaders need insight into their own cognitive preferences and interpersonal needs. Personal style impacts both job performance and effectiveness with others. Understanding this and knowing preferences and team needs is an important first step in exercising managerial strengths and blind spots.

Third, successful IT leaders use both self-awareness and an awareness of others' preferences to maximize team performance. Leveraging interpersonal connections and deploying relationship management skills appropriately are critical in maximizing the utilization of team resources. Table 4 offers some practical questions for IT leaders.

In closing, our research has shown that tools and models from the field of organization development – applied strategically and practically with IT teams – yield benefits that enhance both the process and product of technical work.[1] We use these lessons in our own work with development teams and continue to see the power of this approach on team effectiveness, productivity, and satisfaction.◆

Table 4: *Practical Questions for IT Leaders*

- What data do you find most compelling when solving a problem? What might be left out?
- Are problems or people more important in your decision-making? How does this impact your problem-solving and your relationships with stakeholders?
- How important is it to be included in others' activities? What is the impact of either over- or under-including others in your work?
- Over whom do you exercise control? Who do you allow to control you? To what effect?
- Which is more important to you: orderly goal achievement, or passion and innovation? How could both coexist?

Special Acknowledgement

The authors are grateful to Dr. Dave Scibetta, Dr. James Dobbins, Dr. George Prosnik, and Dr. Beryl Harman at Defense Acquisition University for their support. We also thank key study researchers Dr. Philippe Gwet (statistician), Phyllis Griggs, and the Institute for Scientific Research, including Dr. Rebecca Giorcelli, David Harris, Amy Jacquez, Anita Meinig, and Robert Morgan. Special thanks are also extended to U.S. Rep. Alan B. Mollohan for his vision and support.

References

1. Booz Allen Hamilton, The Institute for Scientific Research, and Otto Kroeger Associates. "Team Dynamics in the Information Technology Industry." Ft. Belvoir, VA: Defense Acquisition University, 2003.
2. Weinberg, G.M. The Psychology of Computer Programming. NY: Dorset House, 1998.
3. DeMarco, Tom, and Timothy Lister. PeopleWare: Productive Projects and Teams, 2nd ed. Dorset House, 1999.
4. Humphrey, W.S. Managing Technical People: Innovation, Teamwork, and the Software Process. Addison Wesley Longman, 1997.
5. Yourdon, E. Death March: The Complete Software Developer's Guide

to Mission Impossible Projects. Upper Saddle River, NJ: Prentice Hall PTR, 1997.
6. OKA. The Typewatching Toolkit. Ver. 2.0. Fairfax, VA: Otto Kroeger Associates, 2000.
7. Waterman, J., and J. Rogers. Introduction to the FIRO-B. Palo Alto, CA: Consulting Psychologists Press, 1996.
8. Moos, R.H. Work Environment Scale Manual. Palo Alto, CA: Consulting Psychologists Press, 1994.
9. Zachary, G.P. "Armed Truce: Software in an Age of Teams." Information Technology and People 11(1): 62-65, 1998.
10. Kroeger, O., J. Thuesen, and H. Rutledge. Type Talk at Work: How the 16 Personality Types Determine Your Success on the Job. New York: Dell Publishing, 2002.
11. Green, G., and A.R. Hevner. Perceived Control of Software Developers and Its Impact on the Successful Diffusion of Information Technology. Pittsburgh, PA: Software Engineering Institute, 1999.

Note

1. This research project generated significant data related to IT team dynamics, retention factors, and success. Readers interested in learning about the study can contact the authors for more information or detail.

About the Authors

Jennifer Tucker is a senior associate with Booz Allen Hamilton. Her current work focuses on information technology (IT) project management, and applying organization development models with the specific needs of IT projects across the development life cycle. Tucker has a Bachelor of Arts in environmental science from Wesleyan University and a Master of Science in management from Purdue University. She is currently working toward a doctorate degree in science and technology in society at Virginia Tech.

Booz Allen Hamilton
8251 Greensboro DR
McLean, VA 22102
Phone: (703) 902-5840
E-mail: tucker_jennifer@bah.com

Abby Mackness is a principal with Booz Allen Hamilton. She has over 23 years of experience in managing information technology programs for federal, defense, and commercial organizations. Her current contracts focus on software engineering and the development of user-friendly knowledge management systems. Mackness has a Doctor of Jurisprudence from Catholic University, a Master of Science in computer science from George Washington University, and a Bachelor of Arts in experimental psychology from the University of North Carolina.

Booz Allen Hamilton
8251 Greensboro DR
McLean, VA 22102
Phone: (703) 902-4746
E-mail: mackness_abby@bah.com

Hile Rutledge is managing partner of Otto Kroeger Associates. He is an experienced organization development consultant, trainer, and public speaker with a background in management, sales, adult education, and leadership development. Rutledge is coauthor of the revised "Type Talk At Work." He has a Bachelor of Arts in humanities from Hampden-Sydney College and a Master of Science in Organization Development from the American University.

OKA
3605 Chain Bridge RD
Fairfax, VA 22030
Phone: (703) 591-6284
E-mail: hrutledge@typetalk.com

Web Sites

Software Technology Support Center

www.stsc.hill.af.mil

The Software Technology Support Center is an Air Force organization established to help other U.S. government organizations identify, evaluate, and adopt technologies to improve the quality of their software products, efficiency in producing them, and to accurately predict the cost and schedule of their delivery.

Stickyminds.com

www.stickyminds.com

Stickyminds.com was created as a place for software managers, testers, and quality assurance people to gather (and gather information). The site is sponsored by Software Quality Engineering, the same company that produces *STQE*. The purpose of the site is to gather knowledge and provide a forum to share it through discussion boards and roundtables, news departments, commenting on content and books, and sharing individuals' work with a the industry by posting papers, templates, and more. General content includes testing, requirements, project management, measurement, defect tracking, and more.

International Federation of Consulting Engineers

www.fidic.org

The International Federation of Consulting Engineers (FIDIC) represents the international business interests of firms belonging to national Member Associations of engineering-based consulting companies. The member firms of each national association comply with FIDIC's code of ethics, policy statements and statutes. The FIDIC federation Web site provides useful information and services to those working in the engineering-based consulting industry or seeking information about it.

The Knowledge Base

http://www.topten.org

The Knowledge Base is a source for ideas, solutions, or inspiring words. Users can search from thousands of Knowledge Nuggets and Top Ten lists full of information based on research and experience. Tips and inspirations include the following categories: Business Success, Personal Development, and Coaching Skills Orientation. Knowledge Nugget samples include Quotation, Tidbit, Wordz, and more.

STC Systems & Software Technology Conference

Technology: Protecting America

19 - 22 APRIL 2004 • SALT LAKE CITY, UT

Have you heard the news? Have you seen the change? The Software Technology Conference (STC) has expanded its scope and focus to include not only the software side of defense technology but the systems side as well. STC has changed its name but we haven't changed our content. We've only added to it. Now we have something for everyone - systems and software. So spread the word. STC is for you and your colleagues. Come one, come all to the "premier systems and software technology conference in the Department of Defense."

About STC

This year's conference will include more than 180 events to choose from, including general sessions, luncheons, plenary sessions, and presentation tracks including the newly added systems engineering track. If you work with systems or software, STC provides outstanding training and networking as well as the opportunity to view newly developed products by top defense technology providers.

In its sixteenth year, STC is the premier systems and software technology conference in the Department of Defense and is co-sponsored by the Defense Information Systems Agency (DISA), United States Army, United States Air Force, Department of the Navy, United States Navy, United States Marine Corps, and Utah State University Extension. We anticipate over 2,500 participants this year from the military services, government agencies, defense contractors, industry, and academia.

STC is Endorsed by:

Lt Gen Harry D. Raduege, **Jr.**, Director, Defense Information Systems Agency

LTG Steven W. Boutelle, Chief Information Officer/G-6 U.S. Army

John M. Gilligan, Chief Information Officer, U.S. Air Force

David M. Wennergren, Chief Information Office, Department of the Navy

RADM Kenneth D. Slaght, Commander, Space and Naval Warfare Systems Command U.S. Navy

BGen John R. Thomas, Director, Command, Control, Communications, and Computers (C4)/Chief Information Officer U.S. Marine Corps.

IEEE Computer Society CSDP Preparation Course and Examination

STC once again is partnering with the IEEE Computer Society to offer the preparation course and examination for the Certified Software Development Professional (CSDP) program at STC 2004. The CSDP is the Computer Society's certification program for software professionals developed by industry experts. The CSDP credential is intended for software engineers, software developers, software program managers, and other professionals. The CSDP is the only certification for computing professionals that carries the brand, reputation, and standards of the IEEE Computer Society. Complete details about CSDP are available at http://www.computer.org/certification. Register for the CSDP course online at www.stc-online.org.

Conference Highlights

STC is pleased to feature the following guest speakers in the 2004 conference agenda:

Opening General Session	**Jon S. Ogg**, Director, Engineering and Technical Management Headquarters, AFMC
Luncheon #1	**Steve McConnell**, Chief Software Engineer, Construx Software
Luncheon #2	**LTG Keith Kellogg**, USA (ret.) (invited), Senior Vice President, Homeland Security Solutions, Oracle Corp
Luncheon #3	**Gregory S. Shelton**, Vice President, Engineering, Technology, Manufacturing, and Quality, Raytheon Co.
Tuesday Plenary	**Co-Sponsors' Panel Discussion**
Wednesday Plenary	**U.S. Government's Top 5 Quality Software Projects for 2003** Sponsored by the Secretary of Defense
Thursday Plenary	**Dr Charles J. Holland** Deputy Under Secretary of Defense (Science and Technology)
Closing General Session	**Bill Neugent**, Chief Engineer, The MITRE Corp.

Track Topics on the Conference Agenda

- Acquisition
- Architectures
- Advanced Methods & Technologies
- Effective Development & Methods
- Homeland Security
- Management
- Metrics
- Net Centric
- Processes
- Requirements
- Security
- Software Intensive Systems
- Systems Engineering
- Testing

Special Sessions

Sponsored track presentations will be offered throughout the week by the following organizations:

- Institute of Electrical and Electronics Engineers (**IEEE**)
- International Council on Systems Engineering (**INCOSE**)
- Joint Strike Fighter (**JSF**)
- Missle Defense Agency
- Navy Open Architecture
- Office of the Secretary of Defense (**OSD**)
- Software Technology Support Center (**STSC**).

Registration

The conference fee structure for STC 2004 is as follows:

Discounted registration fee (paid by 19 March 2004):

Active Duty Military/Government*	$665
Business/Industry/Other	$795

Regular registration fee (paid after 19 March 2004):

Active Duty Military/Government*	$735
Business/Industry/Other	$865

* *Military rank (active duty) or government GS rating or equivalent is required to qualify for these rates.*

Completed registration form and payment must be received by 19 March 2004 to take advantage of the early registration fees. Credit cards will be charged on 19 March 2004.

Housing & Travel

The Housing Bureau of the Salt Lake Convention and Visitors Bureau (SLCVB), using the online Passkey system, handles housing reservations. To access the Passkey system, log on to the STC Web site at www.stc-online.org and select the Housing Reservation button.

Delta Airlines is the official host airline for all STC 2004 attendees. Book your flight by calling Delta Meeting Network® Reservations at 1-800-241-6760, Monday-Sunday 8:00 a.m. – 11:00 p.m. Eastern Time, or have your travel agent call for you. You must refer to File Number **200247A** when making your reservations.

Trade Show

STC 2004 will again feature its accompanying trade show, providing 180+ exhibitors the opportunity to showcase the latest in systems and software technology, products, and services. This year's schedule has been adjusted to allow participants more time to interact with the vendors without conflicting with conference presentations.

Trade show registration, rules, regulations, and updated hall layout are available on the STC Web site www.stc-online.org.

STC 2004 Exhibiting Organizations (As of 12/02/03)

ACM SIGAda
Ada Core Technologies, Inc.
AFIT Software Professional Development Program
Arxan Technologies
Borders Books & Music
CDWG
Compliance Automation, Inc.
Crystal Decisions
Defense Contract Management Agency
Defense Information Systems Agency
Defense News Media Group
Department of the Navy
Digital Harbor
EDS PLM Solutions
Galois Connections, Inc.
Galorath, Inc.
Green Hills Software, Inc.
IBM Software Group
Integrated System Diagnostics, Inc.
International Institute for Software Testing
L3 Communications
Lockheed Martin
Microsoft Corp.
Objective Interface Systems, Inc.
Pentagon Information Technology Service Center
PeopleSoft
Praxis Critical Systems, Ltd.
QSM, Inc.
Quality Plus Technologies, Inc.
Raytheon Co.
Sciforma Corp.
Seaweed Systems
Shim Enterprise, Inc.
Software Engineering Institute
Software Productivity Consortium
Software Technology Support Center
Task Force Web
Telelogic
The Aerospace Corp.
U.S. Air Force
U.S. Marine Corps
U.S. Navy
USA CECOM
USAA
Utah State University Conference Services
Verocel, Inc.
Vitech Corp.
Yrrid Software, Inc.

www.stc-online.org

Source Code: CT4A

General Information
stcinfo@ext.usu.edu
435-797-0423

Technical Content Inquiries
stc@hill.af.mil
801-777-9828

Trade Show Inquiries
stcexhibits@ext.usu.edu
435-797-0047

Media Relations
stcmedia@ext.usu.edu
435-797-0089

Making Meetings Work

Michael Ochs
Fraunhofer Institute for Experimental Software Engineering

Rini van Solingen[1]
LogicaCMG Technical Software Engineering

Every one of us has spent many hours, days, maybe even years in meetings. We all have experienced good meetings and bad meetings. Do software engineers spend large portions of their time in meetings? What factors make such meetings successful? This article presents the results of an industrial measurement study conducted to determine why some meetings are successful while other are not.

Software engineering is more than writing and debugging code. Software engineering is basically a *human* activity [1]. Software engineers participate in meetings, discussions, trainings, and other types of social interactions [2]. Software engineers only spend 30 percent of their time working alone. Fifty percent of their working time is spent in groups of two to three people, and the remaining 20 percent in larger groups and travel [3, 4]. This indicates that software engineers spend more than half their time interacting with other people. This is the reason that we investigated the effectiveness of one such interactive activity: meetings.

No literature was found on the total amount of time spent by software engineers in meetings. This article presents the results of an industrial measurement program that investigated the factors that influence the (perceived) quality of meetings in one department of a telecommunications company in Germany [5]. The analysis is based on a data set of 315 registered meetings, collected in a department with about 50 people over a period of one and a half years. For details on the design of this study, see the sidebar "Setting and Design of the Study," page 24. Based on a total of 1,600 labor hours per person per year, the engineers spent more than 6.5 percent of their time in registered meetings.

Figure 1: *Distribution of Result Quality Across Meetings Types*

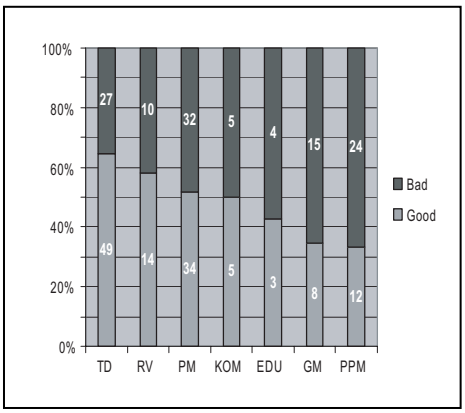
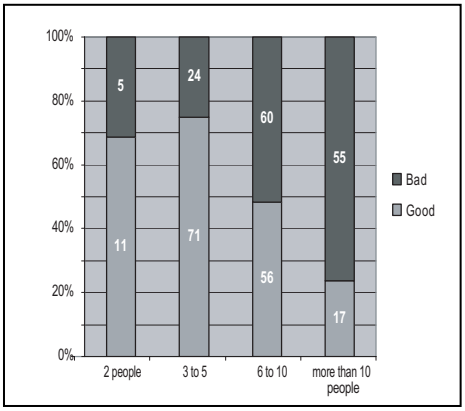


Figure 2: *Distribution of Result Quality Versus Number of Meeting Participants*



The study's objective was to determine why some meetings are successful and others not, and to find out how meeting success could be influenced. Although this success is of course difficult to quantify, we defined a binary metric called *meeting result quality* (values: good and bad)[2]. Please note that we asked the participants for their opinion, meaning that we measure *perceived* meeting quality from a participant's viewpoint. The measurement program was set up in a goal-oriented fashion using the goal/question/metrics philosophy [6, 7]. The data are presented in this article along with the following most commonly asked questions with respect to meeting quality.

Which Meeting Types Are Better?

Several meeting types can be determined, each for different purposes with the goals of the meetings strongly attached to their purposes. The meeting types examined in this study are (with their distribution) as follows:

- **Technical Discussions (TD).** Clarification of technical issues (30 percent).
- **Project Meetings (PM).** Project status and project future direction (28 percent).
- **Project Planning Meetings (PPM).** Project planning (16 percent).
- **Reviews (RV).** Reviews such as code inspections, etc. (10 percent).
- **Group Meetings (GM).** Meetings of organizational units, not project-wise (9 percent).
- **Kick-Off Meetings (KOM).** Initiation of projects (4 percent).
- **Education and Training (EDU).** Training the engineering department, e.g., seminars (3 percent).

The communications culture of the company was very project-driven[3]: 88 percent of the meetings were project related, 12 percent were not project related. The relation between the meeting type and its quality is shown in Figure 1.

A definite correlation between meeting type and its quality could not be found. Figure 1 shows that the probability for a good quality meeting is 35 percent to 65 percent. As such, the meeting type alone does not influence its quality. It seems that the quality of meetings is predominantly determined by other factors than by meeting type.

How Many People Are Present in a Good Meeting?

Meetings take place with a different number of attendees, varying from two persons up to more than 100 persons. In the measurement program, the exact number of participants was measured. For analysis purposes, we divide this number among four categories: meetings of two persons, meetings of three to five persons, meetings of six to 10 persons, and meetings with more than 10 participants. The relation between the number of participants and the percentage of good quality meetings is shown in Figure 2.

In general, the measurements are clear: Good meetings have a limited number of participants. The more people attending, the worse a meeting becomes. Detailed statistical analysis shows that the inflection point lies at eight to 10 partici-

pants, meaning, if a meeting has over 10 participants, it is likely to be perceived as bad. Exceptions were technical discussions, technical review meetings, and group meetings, for which detailed analysis showed that for those, the number of participants does not influence meeting quality.

How Much Speaking Time Is Necessary in a Good Meeting?

As the main purpose of a meeting is interaction, each participant should have sufficient time to explain his or her opinion. Average time per participant is therefore expected to influence how good a meeting is. In the study, the speaking time per participant was estimated from the meeting duration and the number of participants. We grouped these numbers into four categories: below five minutes, five to less than 10 minutes, 10 to less than 20 minutes, and 20 minutes or longer average speaking time per participant. The relation between speaking time and meeting quality is shown in Figure 3.

In order to make the chances for a successful meeting higher, the measurements indicate that at least 15 minutes speaking time is required for each participant. When less time is available, the meeting is likely to fail. Detailed statistical analysis of the measurements showed that speaking time does not influence the quality of review meetings. Group meetings required less time per participant; 10 minutes are sufficient, but this is clarified by the fact that those meetings were mainly used for communicating facts and not so much for discussions.

Of course, the average time per participant is not an exact measure because it is possible that one person speaks, e.g., 50 percent of the meeting time while others do not contribute at all. However, in this study it was not feasible to capture individual participant speaking time in detail. Nevertheless, lessons can be learned from this approximate measure.

How Does the Number of Roles Influence Meetings?

Many roles are present in organizations. Every role has its own responsibilities, goals, and interests. In the measurement program, we measured the influence of the number of roles on the quality of a meeting. Typical roles include developer, project leader, team leader, product manager, quality assurance manager, department head, etc. Figure 4 shows the number of roles present in a meeting and the subsequent meeting quality.

The overall trend shows a decrease in meeting quality with an increase of the number of roles involved. This general trend shows that when more than three roles are present, the worse the meeting. Detailed statistical analysis showed that for technical discussions, reviews, and group meetings, the number of roles had no impact on the meeting's quality.

How Does Hierarchy Affect Meetings?

Besides the number of roles, the amount of hierarchy also was expected to have an influence on meeting quality. Hierarchy was measured by maximum hierarchical distance (XHD), meaning the number of hierarchy layers between the highest and lowest representatives. An XHD of zero means a meeting with peers. An XHD of three means, for example, a meeting in which a software engineer, project leader, department head, and research and development director participate. The relation between hierarchy and meeting quality is shown in Figure 5.

The measurements showed a quite equal distribution when maximum hierarchical distance was zero to two. In those cases, 60 percent of the meetings were good, 40 percent were bad, implying a lower impact of hierarchy on meeting quality. As soon as, however, the XHD was three, the worse the meetings were perceived. Again, this relation was not present for technical discussions, reviews, and group meetings for the same reasons as given before.

How Long Does a Good Meeting Last?

This is a nice and practical question. It would be good to know generally how long meetings should last to make them good. In Figure 6, the relation between meeting duration and quality is shown.

The result is counter-intuitive. You would expect that short meetings are appreciated above long ones. The measurements actually show the opposite: The longer a meeting lasts, the better its quality perception. Even more so, the measurements indicate that meetings are only of good quality when they last at least two hours. What does this mean? Does it mean that only long meetings should be held? Do people like talking? Is it an indicator that this organization has a meeting culture?

Several reasons explain this trend. First, meeting quality is largely influenced by speaking time. Therefore, when many people participate it is better to have a

Figure 3: *Distribution of Result Quality Versus Participant Speaking Time*

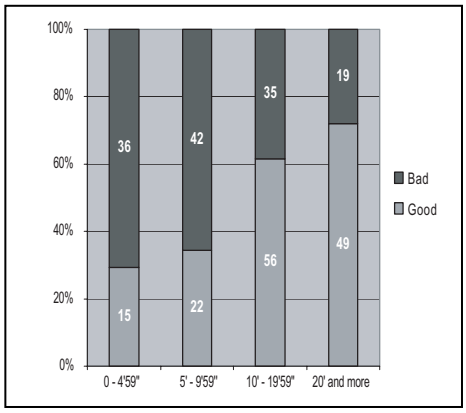


Figure 4: *Distribution of Result Quality Versus Number of Roles*

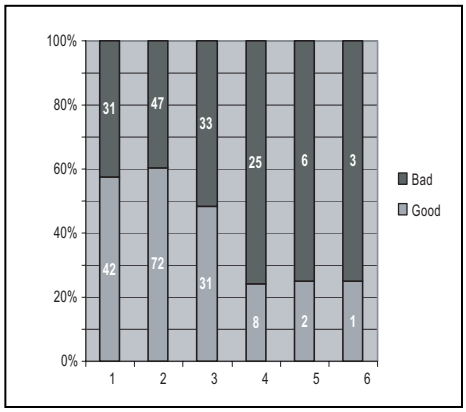


Figure 5: *Distribution of Result Quality Versus Maximum Hierarchical Distance*

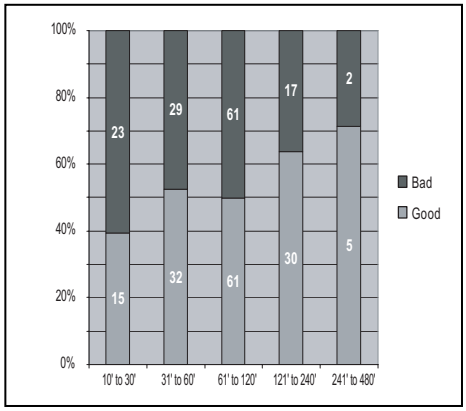
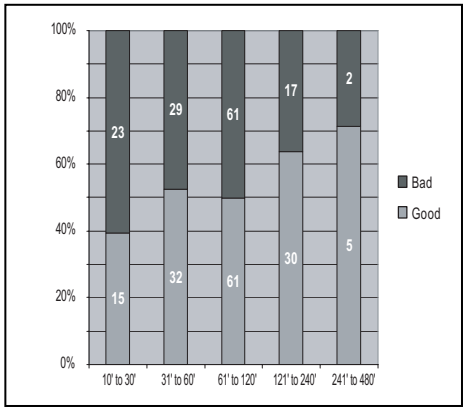


Figure 6: *Distribution of Result Quality Versus Meeting Duration*



Setting and Design of the Study

This study was performed within a goal/question/metric (GQM) [6, 7] measurement-based process improvement program at a telecommunications company's software engineering department, which comprises approximately 50 people engineering control software for phone systems. The time frame was one and a half-years in which 315 communications (i.e., meetings and technical discussions) were registered[4].

Every communication, i.e., meeting, that lasted longer than 10 minutes had to be registered by two data collection sheets. The first one had to be filled in once and was for characterizing the past situation. This was achieved by capturing the type of the communication, the roles involved, the duration, the number of participants, and whether there was moderation or leadership in a communication or not. The second one had to be filled in by every participant measuring the individual perception of the outcome of the situation by the quality of the results of the communication, i.e., the relevance of the communication for the individual. Moreover, the role of every participant was captured.

The goal of the study was to characterize the communication result quality from the perspective of the people working in the particular department. Thus, the variable under study was the perceived quality of results of a communication (Q). The GQM questions were aiming at the relationship between potential factors influencing Q and Q itself, e.g., "How does the number of participants effect the result quality of communications?" "How does the duration of a communication effect the results quality?" More generically spoken, "How does factor X influence Q?" From potentially Q-influencing factors, a set of variables (metrics) was defined, as listed in the metric description table below. Q was measured on a six-point semantic differential scale ranging from satisfactory to unsatisfactory. For further analyses, Q_b was derived from Q by defining the results of a single communication as *bad* if Q was on or above the median of all communications. Overall, the median for Q was *5*, meaning that a total of 50 percent of all meetings was rated 1 through 4, whereas another 50 percent of the meetings was rated *5* or *6* (cf. Q and Q_b in the table below). Thus Q_b can have two values – 0 (bad) and 1 (good). Each participant was asked the following questions to capture his or her perception of the meeting[5]:

I perceive the relevance of the meeting for myself to be …

relevant ❍ ❍ ❍ ❍ ❍ ❍ **non-relevant**

I perceive the quality of the results to be …

good ❍ ❍ ❍ ❍ ❍ ❍ **bad**

I perceive the atmosphere during the meeting to be …

relaxed ❍ ❍ ❍ ❍ ❍ ❍ **tense**

Metric	Description	Scale	Range
TYPE	Type of communication	nominal	TYPE∈{Technical Discussion, Project Meeting, Review, Group Meeting, Kick-Off Meeting, Education, Project Planning Meeting}
LEAD	Type of moderation	nominal	LEAD ∈ {Not Moderated, Moderated}
NOP	Number of participants	absolute	NOP ∈ {2,...,50}
NRI	Number of roles involved	absolute	NRI ∈ {1,...,6}
DUR	Duration of the communication	ratio	DUR ∈ [10,∞)
ATP	Average time per participant	ratio	ATP ∈ (0, ∞)
XHD	Maximum hierarchical distance	absolute	XHD ∈ {1,...,3}
Q	Perceived quality of the results	ordinal	Q ∈ {1,2,3,4,5,6} (1 is worst, 6 is best)
Q_b	Binary quality measure	nominal	Q ∈ {0,1} (0 is lower than Median of Q, 1 is greater than or equal Median of Q, i.e., *bad* or *good* quality)

longer meeting then to compensate on speaking time. Second, shorter meetings are often not planned, meaning that for at least one of the participants it is perceived as an interruption (with the related negative perception) [8]. Finally, one can achieve much better results in a one-day workshop than in eight one-hour meetings [9].

Taking a closer look at the correlation between meeting type and its duration, it is evident there are certain types of meetings where you can say, "The longer the meeting the better it was perceived." Whereas there are also meetings where you can say, "Keep them short to get good results from motivated people." Detailed analysis of meeting types shows the following general visible trends:

- The longer a technical discussion lasts, the better the results are perceived.
- The longer a kick-off meeting lasts, the worse the quality of the results is perceived.
- There is not a clear general trend visible for review meetings.
- For project meetings, no clear trend was identified in the behavior of quality vs. duration; there must be other influencing factors out of the reach of this study.
- The project planning meetings, which are largely perceived to be *bad*, show a switch to a perception of *good* when lasting for at least four hours.
- For group meetings, which are perceived almost the same as the project planning meetings, you can state that a longer group meeting, e.g., of at least two hours, is more likely to be perceived *good* than a shorter one.
- Finally, the analysis shows that seminars and trainings seem to last long and have a comparatively high quality.

Conclusions

Software engineering is more than just writing code. It is a multi-disciplinary job, largely depending on peoples' social skills and social interaction. We emphasize the need for more thorough research on how software engineers interact, and how this interaction can be made more effective.

We showed some factors that influence the participants' perception after attending a software-engineering meeting. Especially the number of participants (maximum eight) and the average speaking time (minimum 15 minutes) are factors that can be controlled in practice in order to steer toward successful meetings. Much research has been performed in the social sciences on human interaction and communication. For example, social science research showed that five to seven persons in a meeting appears an optimum, when more than 12 persons are present a chairman/moderator is necessary, and above 30 participants no dialogues are possible in meetings [10].

Though our findings may not be valid for all kinds of organizations, we are confident that several of the detected success factors are applicable to other organizations as well. These factors can be worked into the following guidelines:

- Keep the number of meeting participants as low as responsibly possible to foster an effective exchange of information.
- With the lowest possible number of participants, plan for sufficient time to give each the possibility to have a valuable contribution to the meeting; "being

effective is much more important than being ultimately efficient" [11].

- Make a clear agenda and improve the meeting process (e.g., using creativity techniques) of poorly performing meetings, i.e., project planning meetings, group meetings.
- Carefully consider the required hierarchical distance of meeting participants; the measurements indicate a drop in perceived meeting quality when maximum hierarchical distance in a meeting is high.
- A lower number of different roles makes a meeting easier to conduct since people speak the *same language*. The chance for stumbling into contradicting interests of different stakeholders is kept low as well. Low variety of roles keeps – on an average – the hierarchical distance low as well.

It is up to the readers to determine whether these guidelines help them make meetings more successful. We hope this article contributes to increasing the effectiveness of software engineering by uncovering some success factors for software engineering meetings that *make meetings work* and by emphasizing that software engineering highly is a people discipline.◆

Acknowledgements

The authors would like to thank Professor Dr. Lionel Briand, Ralf Kalmar, Dr. Martin Verlage, and Kerstin Lünenbürger for their considerable contributions to the research presented in this article. Furthermore, the authors would like to thank Professor Dr. Shari L. Pfleeger, Dr. Robert Glass, and Professor Dr. Dieter Rombach for their valuable comments to earlier versions of this paper.

References

1. Weinberg, G.M. The Psychology of Computer Programming. Van Nostrand Reinhold Computer, 1971.
2. DeMarco, T., and T. Lister. Peopleware: Productive Projects and Teams. New York: Dorset House Publishing, 1987.
3. McCue, G.M. "IBM's Santa Teresa Laboratory – Architectural Design for Program Development." IBM Systems Journal 17.1 (1978): 4-25.
4. Perry, D.E., N.A. Staudemayer, and L.G. Votta. "People, Organizations, and Process Improvement." IEEE Software July 1994: 36-45.
5. Briand, L.C., R. Kempkens, M. Ochs, M. Verlage, and K. Lünenbürger. Modeling the Factors Driving the Quality of Meetings in the Software Development Process. Proc. of the 10th ESCOM Conference, Apr. 27-29, Maastricht, Netherlands. Shaker, 1999: 17-26.
6. Basili, V.R., C. Caldiera, H.D. Rombach, and R.v. Solingen. "Goal/Question/Metric Paradigm." Encyclopedia of Software Engineering 2nd Ed. Vol. 1. J.J. Marciniak, ed., New York: John Wiley & Sons, 2002.
7. van Solingen, R., and E. Berghout. The Goal/Question/Metric Method. London: McGraw-Hill, 1999.
8. van Solingen, R., E. Berghout, and F. van Latum. "Interrupts: Just a Minute Never Is." IEEE Software Sept./Oct. 1998: 97-103.
9. Arthur, L.J. "Quantum Improvements In Software System Quality." Communications of the ACM 40.6 (June 1997): 46-52.
10. Mitchell, T.R., and J.R. Larson. People in Organizations: An Introduction to Organizational Behavior. McGraw-Hill, 1978.
11. DeMarco, T. Slack: Getting Past Burnout, Busywork, and the Myth of Total Efficiency. Broadway Books, 2001.

Notes

1. At the time of this research, van Solingen was head of the Quality and Process Engineering Department at Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.
2. Meeting success was measured on a six-point semantic differential scale capturing the participants' perception of the quality of the results of the meeting.
3. The project, which is referred here, was a mixed maintenance and development project for telecommunications software. Overall, there were approximately 50 people involved in the project, which lasted for approximately 18 months.
4. This set of 315 communications enlarges the study in [5], which relied on ~190 meetings.
5. Only results of the question on quality of meeting results are referred to in this study.

About the Authors

Michael Ochs is a business area manager at the Fraunhofer Institute for Experimental Software Engineering. He is responsible for the Banking and Insurance industry. His interests focus on cost/benefit analysis of information technology and software investments, software engineering decision support, and commercial off-the-shelf-based software development. Ochs is an experienced project manager in various domains of industry and publicly funded research. He holds a diploma in mathematics, computer science and economics from the Technical University of Kaiserslautern, Germany.

Fraunhofer Institute for Experimental Software Engineering
Sauerwiesen 6
D-67661
Kaiserslautern, Germany
Phone: +49 6301 707 251
Fax: +49 6301 707 202
E-mail: michael.ochs@iese.fraunhofer.de

Rini van Solingen is a principal consultant at LogicaCMG and a professor in quality management and quality engineering at Drenthe University in The Netherlands. Van Solingen has been a senior quality engineer at Schlumberger RPS and head of the Quality and Process Engineering department at the Fraunhofer Institute for Experimental Software Engineering in Kaiserslautern, Germany. He has published more than 100 titles in journals, conference proceedings, and books, and is author of "The Goal/Question/Metric Method."

LogicaCMG Technical Software Engineering
P.O.Box 8566
NL-3009-AN
Rotterdam, Netherlands
Phone: +31 10 2537 556
Fax: +31 10 2537 033
E-mail: rini.van.solingen@logicacmg.com

Verification and Validation People Can Be More Than Technical Advisors

George Jackelen
Jackelen Consulting Services

A verification and validation (V&V) organization (whether independent of developers or not) is normally described in technical terms, e.g., how V&V can verify that a developer's processes are technically sound. Some companies and government agencies (federal and state) are recognizing the benefits V&V can extend to non-technical areas such as project management, resource management, finance, and scheduling.

Several books [1, 2, 3], articles [4, 5, 6], and standards [7, 8, 9] address verification and validation (V&V). To some, the V&V descriptions are too narrow and restrictive of the V&V abilities. As shown by the above, publications usually relate to technical areas of V&V and rarely address non-technical V&V, e.g., evaluation of project resources and schedules.

However, a customer (i.e., who V&V reports to, not a product customer/client) could expand V&V's normal scope of work. For example, V&V could assist a customer in identifying non-technical risks such as, "Are a developer's change request schedule and cost estimates reasonable?"

In some cases an organization may not have a fully qualified Project Management Office (PMO) – a group of people dedicated to help a project manager oversee a project by handling the project's schedule, budget, deliveries, etc. In this situation, V&V can be used to coach the PMO or provide limited direct support. For some projects, this arrangement can work as long as everyone understands that V&V can only make recommendations and not provide direct or contractual direction. This point must be remembered while reading the rest of this article.

Several times a reader may say, "This is not V&V work." That is yesterday's mode of thinking. If V&V can become involved with contractual work, requirements identification, etc., even prior to contact award, the benefits could pay the V&V cost by preventing problems, reducing risks, and having an improved contract, statement of work, etc. This does not mean V&V can identify all the requirements, etc.: It can help, as a normal V&V function, an organization ensure the requirements are testable, real, etc. As needed, a project manager may assign some of the V&V efforts mentioned in this article to other organizations for efficiency or to minimize duplication of effort.

The rest of this article provides a quick overview of V&V and its relationship with other groups, and then addresses how V&V can help in non-technical project areas. The following additional notes pertain to this article:

Note 1: For this article, unless needed for clarity, V&V is used as a convenience, even if a reader prefers use of the term Independent V&V (IV&V). In addition, software V&V is assumed even though many of the points apply to non-software V&V. In either case, this article assumes V&V is a group of people rather than a pure process. Except for Note 2, how V&V fits within an organization is ignored.

> *"In a well run, quality-oriented organization, V&V is not a separate group but an integral part of an organization's operation."*

Note 2: In a well run, quality-oriented organization, V&V is not a separate group but an integral part of an organization's operation. Where this does not exist, because the risks are very high or there is a requirement for an independent group (e.g., safety within nuclear power plants), a unique group can be formed or contracted to perform what is known as IV&V. Reference [6] provides more detailed explanation on the technical differences between V&V and IV&V, e.g., amount of financial and managerial independence.

Note 3: The following applies to this article:

- Verification is the process of ensuring the outcome of each lifecycle phase/activity satisfies the requirements of that phase/activity and can support the next phase/activity.
- Validation is the process of ensuring a set of requirements is satisfied, i.e., the product satisfies the requirements.

V&V Overview

As indicated in [7, 9], V&V is totally independent if it has no financial, technical, or management link to a developer. As illustrated in Figure 1, V&V can be viewed as a systems engineering process employing a rigorous methodology for evaluating the correctness and quality of a product throughout a life cycle [10]. Some groups believe V&V should be restricted to testing, to duplicating a developer's efforts using a different approach or set of tools, or to only verifying/validating a developer's technical approach to produce a useful product for a client. Some groups believe V&V should concentrate on high-risk areas by performing one or more of the following:

- Defining risk levels in project-specific terms associated with cost, schedule, and performance.
- Identifying initial project risks.
- Analyzing project risks for impact and likelihood of becoming a problem.
- Identifying possible mitigating actions.
- Prioritizing risks based on return on effort.
- Implementing selected mitigation actions.
- Evaluating risk mitigation progress.
- Continually reassessing risks and actions.

Since formal V&V can be very costly, V&V is usually only involved with risky, costly, or very complex efforts. Much of the V&V cost is related to the experience level a client is paying for, especially if the development effort is very specialized, e.g., nuclear power or advanced space/aerospace technology. When not properly structured with other project groups, V&V can be costly due to duplication of effort, i.e., roles and responsibilities between groups are not well defined.

V&V and Quality Assurance

Many people believe V&V is not needed if

a project has a quality assurance (QA) group. QA and V&V play distinguishable and important roles in systems. Even if QA is independent of a developer, V&V and QA do not normally perform the same functions. As with V&V, QA works best when it is oriented toward preventing problems rather than finding problems.

While QA and V&V are concerned with quality, their perspectives are different. V&V usually focuses on ensuring the requirements are being met, the overall project is focused on the correct objectives, and risk is being managed by the PMO. QA on the other hand, is focused on the day-to-day aspects of a project and is used to determine if procedures are followed. For a project consisting of QA and V&V, QA may witness testing while V&V may measure the adequacy of a test and its associated processes, e.g., test procedures. As a result, QA and V&V can work together to ensure a system is built to meet its requirements, to meet the development standards, and to ensure that risks are minimized. [10]

QA consists of development experts to ensure standards and processes are followed, and to ensure other groups identify and resolve problems. As stated by Roger Fujii, "Following process does not necessarily ensure that the product correctly solves the user's problem" [11]. Normally, QA is involved in:

- Evaluations of processes.
- Audits of events to measure compliance with processes.
- Product quality – product satisfies the quality requirements.

If there is no V&V, QA activities may include:

- Testing.
- Test witnessing to ensure that test procedures are followed.
- Predicting system reliability.

Because V&V can capitalize on the existence of QA, V&V is normally involved in:

- Risk identification, i.e., part of the project risk management effort.
- Assessments, forecasts, and trend analysis to support management decision-making.
- Requirements analysis assessments.
- Traceability of requirements to design and testing, and traceability of design to code.
- Traceability of given/business requirements to derived requirements.
- Evaluation of technical issues.
- Technical evaluations of technical plans, approaches, or methodologies.
- Testing.
- Evaluation of how well a product satisfies the requirements.

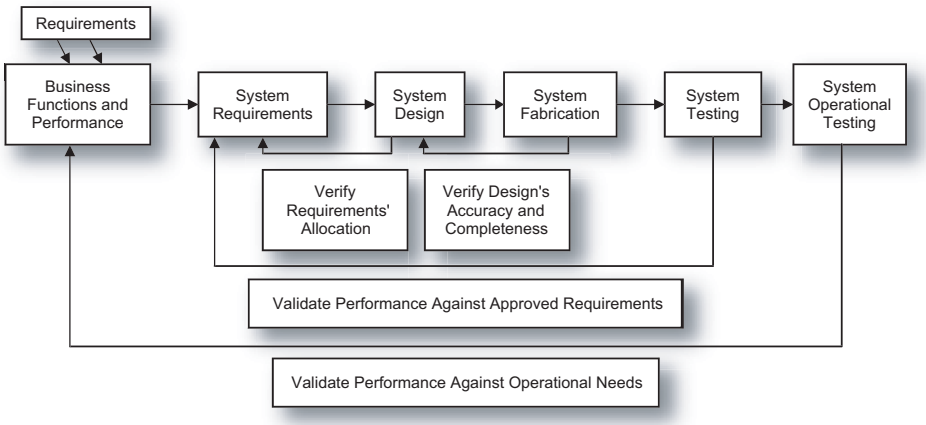


Figure 1: *Summary of V&V's Role in Development*

Some development organizations have a testing group, i.e., not part of development or QA. Other organizations use V&V as an independent testing group (in fact, testing may be the sole function of V&V). Some organizations use V&V to review a test group's test plans and test cases, or to develop test plans independently.

If an organization does not have its own QA, and it does not have confidence with a developer's QA, then it can combine product V&V with process V&V to address any certification needs. For instance, they can assess whether the processes, requirements, design, and tests satisfy the software, hardware, or system security needs.

V&V Non-Technical Activities

In addition to the technical activities, organizations can use V&V for non-technical areas to help manage a project. This includes the following:

- Evaluating a developer's project schedule and management plans.
- Evaluating a developer's product to determine if the contractual requirements are satisfied and then perhaps recommend if payment should be made.
- For an independent client organization, evaluating how well a client is performing on a project.
- In terms of determining a project's budget, schedule, and constraints, helping an organization to do the following:
  - Modify an existing system or build a new system.
  - Use state-of-the-art or state-of-the-practice technology.
  - Use an evolutionary or an incremental development model [10].
- Helping prepare proposal evaluation criteria.
- Evaluating proposals and making recommendations.
- Evaluating developer's or client's processes for impact on product development.
- Performing periodic evaluation of client's or developer's resources to help determine if changes are needed.
- Advising a client on the impact of proposed developer's or client's changes to its management, policies, requirements, or procedures.
- Assessing how well risks, problems, issues, action items, change requests, etc., are being identified and addressed.
- Improving client's acceptance of a product.

The following items are not thought of as being part of V&V. Thinking *outside of the box*, however, V&V experience can be used to prevent problems by having V&V help an organization with the following:

- Prepare a Request for Proposal (RFP).
- Prepare a client's development contract.
- Develop a client's organizational structure and select project personnel.

The above are examples of an organization using V&V to supplement its staff. In some situations, an organization may not have enough, or the right, people available to ensure an effective and efficient project. To prevent any adverse affect on the freedom, independence, and investigative curiosity of V&V, it must not become totally integrated into a client's organization. A *close working relationship* is the best way to describe the necessary interaction, e.g., daily contact, near-total exchange of technical information, and frequent informal meetings or briefings. According to Robert O. Lewis:

> One of the keys to a successful … verification and validation … program is how the [V&V] personnel interface with development personnel [1].

Since V&V may report directly to a client (usually via contract), this statement is also true of V&V's working relationship with other organizations.

Without written PMO permission, V&V may need to document and route its results, requests for data and documents, etc., to its customer and refrain from direct interaction or interface with other organizations. This is especially true if the developer and V&V are contractors. If the contracts permit and the inter-organizational relationships are good, direct interactions are preferred.

There have been attempts with V&V to use informal methods to advance a project. For instance, have developers verbally address V&V questions/concerns to ensure the V&V team understands a situation, or to have a minor concern resolved without making a documented, *federal* case out of a situation.

Sometimes this works, but eventually a developer or client's manager is concerned that this approach is bypassing his or her authority. Another reason to avoid an informal approach is that V&V may be a participant in a court case between a developer and a client. In this situation V&V needs documented evidence of what they identified as good points, problems, etc., and what V&V did to communicate and help resolve the situation. At the same time, V&V may have to show it did not provide any direction that violated a contract, or significantly interfered with the developers.

As an example of the above discussion about authority, V&V could participate in the evaluation (for technical feasibility, and reasonable number of hours and cost) of a developer's change requests based on contractual staffing rates, other documentation, and discussions with the developer. The proper authority could then take V&V's comments and recommendations, along with comments and recommendations from others, under advisement when making the final decision to accept or reject a change request.

There are situations where developers and clients try to restrict V&V's effectiveness by creating barriers, e.g., preventing timely access to people or data. As with QA, this can be overcome if there are clear sets of processes, roles, and responsibilities.

Based on [12], the following sections are examples of how V&V (e.g., without hiring another contractor) can assist project managers by providing independent assessments, recommendations, or comments.

Planning Processes

- Identifying contractual criteria to measure progress.
- Defining performance measurements.
- Identifying, assigning, and documenting specific activities needed to produce a product.
- Verifying the accuracy, consistency, or completeness of a published schedule.
- Identifying and documenting interactivity dependencies.
- Determining what physical resources (people, equipment, material) and what quantities of each should be used to perform project activities.

"The biggest technical payoff in using V&V can be to have V&V parallel each phase of the development effort. This provides a thorough requirements and design verification aimed at preventing otherwise costly errors, omissions, and inadequacies from ever reaching the coding, testing, or acceptance stages."

- Independently developing an approximation of the cost and time needed to complete project activities, e.g., change requests.
- Determining a project's communications and information needs.
- Identifying initial project risks.
- Preparing documents to support solicitation.

Executing Processes

- As needed, providing *out-of-the-box* thinking and evaluations to help determine project progress, effectiveness, and reporting.
- Responding to information requests and providing information in a timely manner.
- Helping ensure a developer's performance meets contractual requirements.

Controlling Processes

- Providing evaluation of a project's overall change (e.g., changes in scope, schedule, cost, performance, or quality) control process.
- Identifying risks and evaluating a project's risk management process.
- Collecting and disseminating performance information, forecasts, and status or progress reporting.
- Evaluating if change requests are within scope.
- Helping determine if a change will be beneficial.
- Identifying influencing factors that are, or could create, changes in schedule, cost, or performance.

Conclusions

The biggest technical payoff in using V&V can be to have V&V parallel each phase of the development effort. This provides a thorough requirements and design verification aimed at preventing otherwise costly errors, omissions, and inadequacies from ever reaching the coding, testing, or acceptance stages.

Customers also need to understand the capabilities of a good (or better yet, an outstanding) V&V. As with any support group (e.g., QA, configuration management, data library), a properly used V&V can provide resources to help a PMO to make timely, correct decisions.

For complex projects, an effective V&V is composed of people with experience; in-depth technical expertise; and strong communications, management and planning skills.

Key objectives of V&V are to:

- Assure a successful project.
- Improve management visibility into the project's processes and product usefulness.
- Avoid system failures of high consequence.

The following V&V activities complement these V&V objectives: [10]

- Provide a project manager/client with objective analysis of data to support project decision making by the following:
  - Identify project risks as early as possible.
  - Prioritize risks by impact and probability of occurrence.
  - Define mitigation actions and perform cost-benefit analysis.
  - Help a project manager perform trade-offs, e.g., manage limited resources.
  - Evaluate progress on resolution of risks and corrective actions.
- Provide quick assessment of pro-

posed changes and consequences.

- As requested, provide PMO-type assistance.

Thus, organizations should look at V&V as more than a group with only technical knowledge. This may mean expanding the next version of [7] to include a new definition of V&V by indicating where V&V can help projects with non-technical issues. Another suggestion is to have [7] provide a section on how V&V and QA are alike and yet different.

No matter how V&V is utilized, the best results can occur when a well-planned V&V effort is conducted throughout a full system development life cycle.◆

References

1. Lewis, Robert O. Independent Verification and Validation – A Life Cycle Engineering Process for Quality Software. New York: John Wiley & Sons, Inc., 1992.
2. Ratkin, Steven R. Software Verification and Validation: A Practitioner's Guide. Artech House, 1997.
3. Schulmeyer, Gordon G., and Garth R. MacKenzie. Verification and Validation of Modern Software-Intense Systems. Prentice Hall, 2000.
4. Stauffer, B.C., and R.U. Fujii. "Security Evaluation of Mission Critical Computer Systems." Compass '86 July 1986.
5. Walters, Dan F. "Writing an Effective IV&V Plan." CrossTalk Nov. 2000: 20-23
6. Wait, Patience. "Project Protection Comes at a Price." Washington Technology 28 Aug. 2001.
7. Institute of Electrical and Electronics Engineers. IEEE Standard for Software Verification and Validation. IEEE Std. 1012, 1998.
8. Goddard Space Flight Center. Software Independent Verification and Validation (IV&V) Management. NASA, Aug. 2000 <www.ivv.nasa.gov/about/tutorial/Handbook_for_Program_Management_v1.pdf>.
9. Wallace, Dolores R., Laura M. Ippolito, Barabra Cuthill. Reference Information for the Software Verification and Validation Process. NIST Special Publication 500-234. Gaithersburg, MD: National Institute of Standards and Technology, 29 Mar. 1996 <http//hissa.ncsl.nist.gov/HHRFdata/Artifacts/ITLdoc/234/val-proc.html>.
10. Stevens, Arthur "Ed." White Paper presented to the State of Colorado, Apr. 2001.
11. Fujii, Roger. "Software Assurance Philosophy – Software Certification: Merits of Product IV&V vs. Process IV&V." First International Software Assurance Certification Conference – ISACC '99, Dulles, Va., 1999.
12. Project Management Institute. A Guide to the Project Management Body of Knowledge. Newtown Square, PA: PMI, 2000.

Additional Reading

1. Project Management Institute. Government Extension to a Guide to the Project Management Body of Knowledge. 2nd ed. Newtown Square, PA: PMI, 2000.

About the Author

George Jackelen is an independent verification and validation analyst and owner of Jackelen Consulting Services. Prior to this, he was involved in systems and software analysis, computer programming and operations, project management, and computer contract development and monitoring. He has also been involved with and led the development of organizational, national, and international software standards. Jackelen has a Masters of Science in computer science from Texas A&M University and a bachelor's of science degree in math and physics from St. Cloud University.

Jackelen Consulting Services
9402 Troon Village DR
Lone Tree, CO 80124-3133
Phone: (303) 662-0843
E-mail: jackelen@earthlink.net

Letter to the Editor

Dear CrossTalk Editor,

I read your publication and consider it a valuable resource in my reference library. Thank you for the technical information each issue.

Your November 2003 issue refers to "DO-178B" as "Defense Order (DO)-178B." RTCA says that the DO is "Document Order."

Roger Souter
Federal Aviation Administration

CrossTalk
The Journal of Defense Software Engineering

Get Your Free Subscription

Fill out and send us this form.

OO-ALC/MASE
6022 Fir Ave
Bldg 1238
Hill AFB, UT 84056-5820
Fax: (801) 777-8069 DSN: 777-8069
Phone: (801) 775-5555 DSN: 775-5555

Or request online at www.stsc.hill.af.mil

Name:_____

Rank/Grade:_____

Position/Title:_____

Organization:_____

Address:_____

_____

Base/City:_____

State:_____Zip:_____

Phone:(____)_____

Fax:(____)_____

E-mail:_____

Check Box(es) To Request Back Issues:

Dec2002 ☐ Year of Eng. and Sci.
Jan2003 ☐ Back to Basics
Feb2003 ☐ Programming Languages
Mar2003 ☐ Quality in Software
Apr2003 ☐ The People Variable
May2003 ☐ Strategies and Tech.
June2003 ☐ Comm. & Mil. Apps. Meet
July2003 ☐ Top 5 Projects
Aug2003 ☐ Network-Centric Archt.
Sept2003 ☐ Defect Management
Oct2003 ☐ Information Sharing
Nov2003 ☐ Dev. of Real-Time SW
Dec2003 ☐ Management Basics
Jan2004 ☐ Info. from Sr. Leaders

To Request Back Issues on Topics Not Listed Above, Please Contact Karen Rasmussen at <karen.rasmussen@hill.af.mil>.

Information Assurance in Wireless Residential Networking Technology: IEEE and Bluetooth

Ambareen Siraj and Rayford B. Vaughn
Mississippi State University

Information Assurance (IA) and wireless communication remain major areas of information technology research today. IA is an increasing concern in both the government and commercial sectors. Wireless residential networking is among the wireless technology areas that have accelerated most rapidly over recent years. The Institute of Electrical and Electronics Engineers (IEEE) 802.11b and Bluetooth are two of the most commonly used wireless standards in residential network communication. Although targeted at the same wireless home/office network market, these two standards have different roles that are complementary rather than competing. Foreseeing the need for both technologies, recent efforts are ongoing to make the coexistence of both successful. This article focuses on the security issues of IEEE 802.11b and Bluetooth from the IA perspective and highlights their individual strengths and weaknesses from an IA point of view.

Information Assurance (IA) and wireless communication remain major areas of IT research today. IA is an increasing concern in both the government and commercial sectors. Wireless residential networking is among the wireless technology areas that have accelerated most rapidly over recent years. The Institute of Electrical and Electronics Engineers (IEEE) 802.11b and Bluetooth are two of the most commonly used wireless standards in residential network communication. Although targeted at the same wireless home/office network market, these two standards have different roles that are complementary rather than competing. Foreseeing the need for both technologies, efforts are ongoing to make the coexistence of both successful.

This article focuses on the security issues of the IEEE 802.11b and Bluetooth from the IA perspective and highlights their individual strengths and weaknesses from an IA point of view. It is not intended to be a comprehensive treatment of the subject, but rather a survey of assurance issues that should be of concern to users. There are certainly others beyond those presented here. This article also assumes that the reader has some knowledge of the two protocols discussed and does not include a tutorial.

IEEE 802.11b and Bluetooth

The IEEE 802.11b and Bluetooth are two of the most widely used emerging wireless protocols for over-the-air wireless information exchange targeted for the residential (home/office) market. The IEEE 802.11 is the Medium Access Control (MAC) and Physical Layer (PHY) protocol, developed for Wireless Local Area Networks (WLAN). The PHY layer protocol oversees the actual data transmission process over a communication channel, while the MAC layer oversees reliable transmission of data with tasks such as data frame formatting, frame flow control, error checking, channel allocation, etc. The IEEE 802.11b is a supplement to the original IEEE 802.11 standard extended to define the standard for wireless LAN products that operate at an Ethernet-like data rate. It is also known as the Wireless Fidelity standard.

> *"Today, researchers foresee the need for these two prevalent technologies to co-exist and already there are products in the market that support both of these technologies."*

Ericsson, Nokia, IBM, Intel, and Toshiba founded the Bluetooth Special Interest Group (SIG) in May 1998. Since then, many major companies in the telecommunication business have joined this Bluetooth SIG. Bluetooth is an ad-hoc networking technology that dynamically connects portable/handheld devices such as cell phones, personal digital assistants, laptops, and printers by means of variable network topologies. Bluetooth is primarily designed to replace cables in residential networks for intercommunication between computing/communication devices.

It is important to note that the IEEE 802.11b is designed to support MAC and PHY protocols for a WLAN, whereas Bluetooth is designed as a complete protocol for the Wireless Personal Area Network (WPAN). Bluetooth is based on the IEEE 802.11b, but it is basically a radio frequency specification for both voice and data transfer technology that has low latency, low power-synchronization, and short range. While Bluetooth supports voice communication and the IEEE 802.11b does not, Bluetooth does not support many of the features that a full-blown wireless LAN implementation such as the IEEE 802.11b does in order to be used for corporate local area networks.

The IEEE 802.11b does more than Bluetooth in terms of data rate (11 megabits per second [Mb/s] versus 721 kilobits per second [Kb/s]), range (100 meters versus 10 meters), power throughput (280 megawatts [mW] versus <4mW), and therefore costs more than Bluetooth. We point out here that the rates, ranges, and costs are changing so rapidly in this field that figures presented here might change prior to publication. Therefore, we will simply site the differences without further particulars.

Both the IEEE 802.11b and Bluetooth protocol operate at the data link layer with some common operational features but with varied utilities.

Information Assurance

IA is an increasing concern in both government and commercial sectors. IA represents a goal that guarantees all electronically held information would remain protected to a sufficient degree associated with a risk that one is willing to accept.

Due to space constraints, CROSSTALK *was not able to publish this article in its entirety. However, it can be viewed in this month's issue on our Web site at* <www.stsc.hill.af.mil/crosstalk> *along with back issues of* CROSSTALK.

Laws of Software Motion

The theme of this issue deals with consulting. As consultants, the authors know that consulting is all about leverage and motion. Of course, being card-carrying geeks (punch cards, that is), the authors immediately made the connection between motion and Newton. Newton, after all, formulated the "Laws of Motion."

Newton's laws govern the movement of the universe. Unfortunately, we've noted that software doesn't seem to follow Newton's Laws of Motion. In fact, software often seems not to follow any laws of motion at all. Thus, the authors have developed what we modestly call the "Cook-Leishman Laws of Software Motion."

These so-called laws can provide understanding and guidance to program managers, developers, and users. These laws help indicate the relationships between time, cost, effort, and program progress.

Law No. 1

Newton's First Law – An object in uniform non-accelerated motion (or at rest), will remain in the same state of motion unless an outside force acts upon it.

Cook-Leishman's First Law – Any software intensive program not given adequate force (motivation) will degrade and cease to progress.

Any program slowly making progress at a steady rate will eventually grind to a halt unless continual outside force is applied to it. Programs will not coast. It takes active program/project management and focused risk management to keep a program continually and successfully proceeding [1]. If you do not have an active risk management plan, you run the risk of having your program stall.

If by some rare chance a program quits making progress, it takes a lot of effort to get it moving again. In fact, lack of progress usually means that programs tend to degrade. Personnel are reassigned, budgets shrink, and out-of-sight, out-of-mind thinking takes over.

Law No. 2

Newton's Second Law – There is a relationship between force, mass, and acceleration that is specified as F = ma.

Cook-Leishman's Second Law – There is a relationship between the forces required to accomplish a software intensive program/project. This relationship correlates requirements (R), changes (C), external influences (X), and timing (T) (schedule), thus F = f(R,C,X,T) where f stands as function that relates R, C, X and T.

Often, in the real world, there is actually an inverse relationship between the software force F and R, C, X, and T. In large programs that are well staffed and well funded, relatively small efforts or influences can make a change; this is due to the fact that the small changes require relatively minor effort with respect to the entire program. On the other hand, in small programs, small efforts or influences might reflect major changes to the entire effort. In addition, in some programs, major efforts have to be made to implement relatively small changes.

Changes can occur based upon outside forces (changes) that seem to have no influence upon your program. With interservice programs, multiple users and multiple sources of requirements can have ripple effects in which insignificant events can have major impacts upon your program.

In the first law, risk management (and risk mitigation) is required. In the second law, configuration management is the major player [2]. Without a configuration management plan that puts you in proactive instead of reactive mode, changes will eventually stop forward progress. At this point, Law No. 1 will apply.

Law No. 3

Newton's Third Law – For every action, there is an equal and opposite reaction.

Cook-Leishman's Third Law – For every action, there are varying reactions. Some are small, and some might have exponentially greater force. You really don't know what the consequence will be. The same action at a different time might have totally different consequences.

Small changes early in the program have minor consequences. The same changes during design might cause relatively major rewriting of both code and design. During integration and systems testing, the changes might cause a catastrophic delay in final delivery. Timing is everything. Attempts to improve quality might have positive effects one time, but the same attempt might not help at a different time (or for a different organization). You have to keep the end product in focus, and remember that the process has to be tailored for each product, not vice versa. [3] To put Law No. 3 in a nutshell, "Timing is everything."

Are these laws really laws? Of course not! In reality they are, at best, general guidelines or hints. They are based on an examination of many software projects under varying constraints. Certainly, software development is far too complex to be summed up in a few so-called laws. After all, even Newton wasn't totally correct[1].

However, these so-called laws do help provide general guidelines to help you keep your program moving forward. Have a strong risk management and risk mitigation plan. Implement configuration management, and use risk management and configuration together to proactively predict upcoming changes, thus mitigating major cost and schedule impacts. Focus on the product and quality, and modify the process to accommodate the needs of your program.

To sum it up, you can help your program greatly by applying Cook-Leishman's Fourth Law.

Law No. 4

Cook-Leishman's Fourth Law – Read (and heed!) CROSSTALK regularly.

-Dr. David A. Cook
AEgis Technologies Group
and
-Theron Leishman
STSC/Northrop Grumman

References

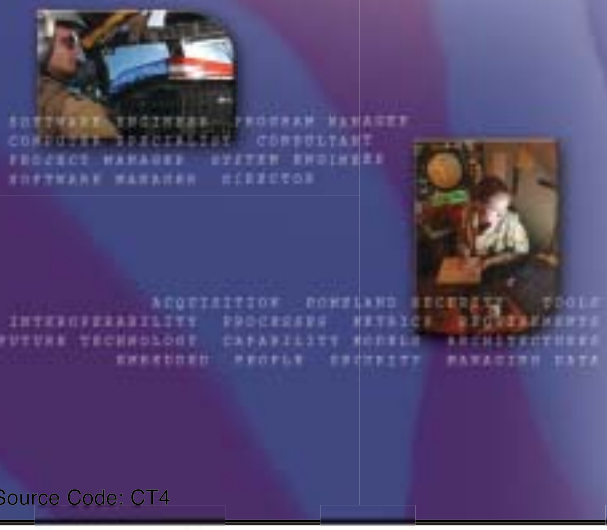
1. Leishman, Theron R., and Dr. David A. Cook. "Requirements Risk Can Drown Software Projects." CROSSTALK Apr. 2002: 4.
2. Leishman, Theron R., and Dr. David A. Cook. "But I Only Changed One Line of Code!" CROSSTALK Jan. 2003: 20.
3. Cook, Dr. David A. "Confusing Process and Product: Why the Quality Is Not There Yet!" CROSSTALK July 1999: 27.

Note

1. The authors can see the e-mail now. Yes, Newton was correct given the understanding of physics at the time. Quantum theory and Einstein have changed a few things. Newton's laws do not apply either at relativistic speeds (near the speed of light), or at the very small level, where quantum mechanics must be used. Newton's laws, of course, do apply to all generally observed behaviors.

STC Systems & Software Technology Conference

19 - 22 April 2004 • Salt Lake City, UT



Register today!

For registration, conference, and trade show information visit our Web site or call

www.stc-online.org

800-538-2663

Source Code: CT4

Co-sponsored by:

United States Army
United States Navy
United States Marine Corps
Department of Navy
United States Air Force
Defense Information Systems Agency
Utah State University Extension

Co-hosted by:

Ogden Air Logistics Center/CC
Air Force Software Technology Support Center

STSC

Published by the Software Technology Support Center (STSC)

CrossTalk / MASE
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056-5820

PRSRT STD
U.S. POSTAGE PAID
Albuquerque, NM
Permit 737