Readings

- "You Have to Start Meeting Like This!" Article in Fast Company by Gina Imperato
- "Making Meetings Work For Everybody" Chapter 8 in Exploring Requirements -- Quality Before Design by D. Gause and G. Weinberg
- "IEEE Software, Building the Community of Leading Software Practitioners -- Teaching Teamwork" by Thomas B. Hilburn
You Have to Start Meeting Like This!

We work -- therefore we meet. But why do so few of our meetings meet our expectations? Michael Begeman may be the world's foremost expert on the business world's most universal ritual. Here's his short course on running meetings that will work for you.

From: Issue 23 | March 1999 | Page 204 | By: Gina Imperato

Michael Begeman is a leading authority on one of the business world's most universal rituals: the meeting. An anthropologist and computer scientist by training, he serves as manager of the 3M Meeting Network, a loose-knit collection of meeting experts that's been assembled by 3M, the innovation-obsessed manufacturing giant headquartered in Minneapolis.

But Begeman, 41, is much more than a meeting planner and facilitator. He spent four years as a member of the technical staff at Intel. He spent six years as a research manager at MCC, a high-tech research consortium based in Austin, Texas. He has run his own consulting firm. In short, he knows as much about how business works as he does about how meetings work.

So what's the most effective meeting that Begeman has seen lately? He says that it didn't take place in a high-rise office building or at a cutting-edge chip factory. In fact, it took place in a tepee -- in a scene from Dances with Wolves (1990), the Oscar-winning film featuring Kevin Costner. The scene takes place after a group of Native Americans discover Costner not far from their camp. Between 20 and 30 members of the tribe gather around for a meeting. There's one big question on their agenda: What should they do with this mysterious white man -- kill him to send a message to others who might follow, or leave him alone to signal their willingness to reason with such newcomers?

What follows, claims Begeman, is a clinic in good meeting behavior. "People actually listen to one another," he marvels. "There are some genuine disagreements, but everyone recognizes merit in everyone else's position and tries to incorporate it into his thinking. The chief spends most of his time listening. When the time comes to make a decision, he says something like 'It's hard to know what to do. We should talk about this some more. That's all I have to say.' And the meeting ends! He is honest enough to admit that he's not ready to make a decision."

How does Begeman compare that powwow with what takes place inside most conference rooms today? "Do you want to know the truth?" he asks. "Here's my mental image of what happens at most business meetings: You could take the people out and replace them with radios blaring at each other, and you would not have changed very much. That's what most meetings are like. People wait for the person who's speaking to take a breath, so they can jump into the empty space and talk. The quality of communication in most meetings is roughly comparable to the quality of the arguments that you used to have with your 10-year-old brother."

Begeman's mission is to change all that. The monthly email newsletter published by the 3M Meeting
Network goes out to thousands of subscribers. The group's Web site offers a collection of useful tools and techniques, of valuable hardware and software. "There is a 'science' of meetings that's available to people now," he says. "We have the knowledge we need to make meetings better. But most people haven't learned it or don't bother to use it. And then they wonder why their meetings just stumble along."

In an interview with Fast Company, Begeman offers a short course on how to make your meetings work.

Meetings Are Work -- And Great Meetings Take Lots of Work

Great meetings don't just happen -- they're designed. Producing a great meeting is a lot like producing a great product. You don't just build it. You think about it, plan it, and design it: What people and processes do you need to make it successful? But first you have to create agreement among people that meetings are work -- they are not an empty ritual to be suffered through before getting "back to the office." Meetings are events in which real work takes place.

That's a big mind flip. All primates -- monkeys, apes, humans -- are social creatures. When you're out in the wild, studying nonhuman primates, one of the things you appreciate is just how social they are. They hang out together, they play together, they groom each other. You very rarely see solitary behavior. But if you walk into a typical company, what you see are rows and rows of cubicles. We've taken these wonderfully social creatures -- human primates -- and we've isolated them. And then we've asked them to be productive in that environment.

Now, as more and more of what people do takes place in teams, meetings become the setting in which most of the really important work gets done. I see this everyday in my own work and life. I do almost all of my work with a team of people -- some from inside 3M, some from outside the company. If I spend most of the day sitting in my office, instead of interacting with people, a warning bell goes off in my head: I'm not getting my job done.

So many people complain to me, "I wish I didn't spend so much time in meetings." To which I say, "Resistance is futile!" The simple fact is, some of our peak experiences as people take place in work groups. Most people have attended at least a few meetings in which there's been a real breakthrough: People are facing a problem, banging heads, not making very much headway -- and then a kind of magic overtakes them. A wind comes along, it blows away the clouds, and you can just feel the energy in the room. It's possible to have more experiences like that -- if you design your meetings with the same care that you use to design your products.

Different Meetings Need Different Conversations

One of my main roles is to create useful linguistic distinctions for people. Organizations call meetings for lots of different reasons. And it turns out that different kinds of meetings require different kinds of conversations. If you're not clear about the kind of conversation that you should be having, then your meeting probably won't achieve a clear outcome.

For example, some meetings are built around a "conversation for possibility." The group acknowledges that it has come together to generate ideas, not to make decisions. The goal is to maximize creativity. Other meetings are built around a "conversation for opportunity." The goal is not to reach a final decision but to narrow down a field of ideas or options. You gather lots of information; you do some analysis; people take positions. Finally, there are meetings that are built around a "conversation for action." The goal is to decide, to commit: "We want to leave this room with our three investment
priorities for 2000."

Unless everyone understands these distinctions, you run into certain familiar problems. You convene a brainstorming session (a "conversation for possibility"), and people are afraid to speak up because someone might shoot down their idea -- or worse, someone might say, "Let's do it." Or you convene a budgeting session (a "conversation for action"), and someone loops back to an idea that was rejected earlier -- which drives everyone else crazy. If you call a meeting, make it clear to people what kind of conversation they're going to have, and then impose a certain amount of discipline on them. Remember: Meetings don't go off topic. People do.

**Always Play by the Rules (of Engagement)**

Most participants come to a meeting with clear expectations about how other people should act. And if the meeting lives up to such expectations, the participants will feel like they've had a really good experience. If the meeting violates those expectations, then people will become upset or withdrawn. So the key is to translate implicit expectations into explicit agreements -- into what I call "rules of engagement." Do people feel strongly about starting and ending on time? Then make an explicit commitment to doing that. Are people concerned that a meeting doesn't have a clear enough objective? Then make an explicit promise: "If we can't agree on a clear objective within the first 10 minutes, then the meeting is over. We'll schedule another meeting when the objective becomes clear."

You can even create rules of engagement about individual behavior. For example: Before anyone makes a point, that person has to find merit in the point made by the previous speaker. Or, the senior people in the meeting can speak only after the junior people have had a chance to express themselves.

It's a pretty simple idea, really. All you are trying to do is to make the invisible visible, to make the automatic deliberate. These rules of engagement take the bad behaviors that groups stumble into, shine a light on those behaviors, and then address basic questions: How can we change all of this? How do we want to act? Such rules of engagement give people a chance to design how they treat one another in meetings.

One last point about rules of engagement: You should be clear that not all successful meetings end with a decision -- which goes back to why I love that scene in Dances with Wolves. Decisions are the Valium of meetings. They offer relief from the tension of what lies ahead, from the uncertainty of the world. They tend to create an illusion of progress: "We've finally made a decision. Now we don't have to worry about that issue anymore." Often it takes courage for a group to end a meeting without making a decision.

**Small Talk is a Big Deal**

There is a legitimate social component to meetings. Sure, we'd all rather be efficient than sloppy in our work. Sure, we'd all rather spend our time on "real work" than on "idle chitchat." But you should never overlook the social side of work rituals -- even in meetings that are "all business." In many of the meetings that I run -- especially in meetings that take place early in the day -- I schedule 5 or 10 minutes of open time, just to encourage people to relate to one another. If you plan for such time, if you put it on your agenda, then you won't feel as if you're not doing what you ought to be doing. Instead, you can enjoy going around the room and asking people what they did last night, or over the weekend.

For some meetings, I book a certain amount of time at the beginning to ask, "Is there anything that
people need to say in order to be 'present' at this meeting?" Remember, just because people walk into a conference room doesn't mean that their mind is on your meeting. They may be thinking about an argument that they just had with a colleague, or about a computer glitch that they've been struggling with all day. If you let people express their frustrations before you get down to business, you allow them to clear their mind and to focus on your meeting.

Want Serious Meetings? Hand Out Toys!

There is much more to people -- even serious businesspeople -- than what's above the neck. We are not just intellects that come together to interact with other intellects. The more you involve the whole person in your meetings, the more people will learn, and the more of that learning they will retain. If you want people to work together effectively, let them play together.

That's why I think there is so much value in having kinetic stuff in meeting rooms: squeeze balls, Slinkies, little gizmos that you turn over and play with. Every so often, just go into a toy store, blow $20 on junk, and put all of it in your conference room. Toys are a great stress reliever -- and a great creativity enhancer. I've found that when people have something to play with, when they can get more of their body involved in what they're doing, they become more creative.

I'm famous around here for my bag of meeting toys. It comes in handy. Last summer, for example, I was working with a group of senior executives. The first thing I did when I started off the meeting was to give everybody two toys: a Meeting Network mouse pad and a Meeting Network squeeze ball. The executives played with this stuff throughout the meeting. It was great: One person would say something that another person didn't like, and the second person would throw a ball across the table. Everyone at the meeting had lots of fun.

And these were senior executives, by the way -- people who are not given to playing at work. A week later, I was in the same room, sitting in as an observer for someone who was presenting to the same group. The executives came in and sat around their table, and as the meeting was about to start, one guy said, "Wait a minute. We can't start yet." Then he ran out -- and came back a few minutes later with his squeeze ball!

Even Good Meetings Can Get Better

If you're serious about improving the quality of your meetings, then you should borrow an idea from the quality people: continuous improvement. Set aside five minutes at the end of every meeting you hold -- make it a discipline for your team or your company -- and ask some simple questions: What did we do in this meeting that really worked well? What happened that we never want to repeat? Are there bad habits that we seem to keep falling into?

Write down people's answers, keep a running record of their comments, and then see how well the entire group improves over time. A written record can also be a great source of ideas for future rules of engagement. It can tell you not just how to behave, but why people believe it's important to behave that way.

But don't overdo this. The best medicine in the world can make you sick if you take too much of it. If you become too intent on improving meetings, you're likely to become the most dreaded person in your department: "Oh no, Joe's in this meeting. What's he gonna come up with this time?" So, please, use these ideas and practices, but use them wisely.
Meeting Minutes

- One classic meeting dilemma is deciding how much to record. Michael Begeman's proposal: Don't worry too much about taking detailed minutes -- that is, exhaustive notes about who said what. Focus instead on three categories of information: decisions reached, action items that people need to follow up on, and open issues. "The record of all this becomes input for future meetings," says Begeman. "Plus, encouraging people to use these categories will sharpen the quality of their participation."

- Actions speak louder than rules. Leaders send nonverbal as well as verbal messages. So it's quite possible, says Michael Begeman, for your words to abide by the "rules of engagement" for a meeting, while your informal actions don't. If you're leading a meeting and people expect you to move the group toward a decision, then act accordingly. Sit at the head of the table to signal, "I'm in charge." Stand while others are sitting to signal, "I have the floor." If participants expect a collaborative meeting, ask one of your team members to run the meeting -- to signal, "I want to share leadership." Or to signal, "I'm with you," sit on one side of the table. All of this may sound obvious, but it's amazing how small, nonverbal behaviors can undermine -- or promote -- what you are trying to accomplish.

Associate Editor Gina Imperato (gimperato@fastcompany.com) has learned to love meetings. You can contact Michael Begeman by email (mlbegeman@mmm.com) or learn more about the 3M Meeting Network on the Web (www.3m.com/meetingnetwork).

* 

Copyright © 2006 Mansueto Ventures LLC. All rights reserved.
Fast Company, 375 Lexington Avenue, New York, NY 10017
There are two major types of adventure stories about exploration. In the first, the lone explorer faces the elements, showing ingenuity and problem-solving ability at every new crisis. In the second, a group of people face the elements together. They benefit from their multiple talents and points of view, and they pay for these benefits by the troubles they have in getting along together.

The lone explorer is the ideal image that many of us carry forward from our school days, when working cooperatively was called “cheating.” Many organizations reinforce this negative image of cooperative work, encouraging “competition” among employees by such devices as individual achievement awards.

In any sizable development project, however, the idea of doing it alone is more of a fantasy than any Indiana Jones adventure. In order to obtain the capacity and diversity that a team can provide, we must give up our heroic fantasies and pay the costs of working with other people. In no place are these interpersonal costs more evident than in meetings. In this chapter, we'll explore the use of meetings as tools: how to make them more productive and less vexatious.

8.1 Meetings: Tools We Can't Live With, or Without

Meetings can be terrible. At some moment in our lives, every one of us has sworn we would never attend another meeting. We complain that they're too long and boring, they stray from the subject, they're painful, and they have no purpose. But then we always relent, because we can't live without meetings. Meetings are tools—social tools—and without them we could develop only the simplest products.

Before discussing meetings in general, though, let's look at a terrible meeting in painful detail to see what can go wrong.

8.1.1 A terrible, but typical, meeting

(Jack and Zara arrive at Room 1470B at 9:05 a.m. and find it totally dark. Jack turns on the light and looks at his watch.)
MAKING MEETINGS WORK FOR EVERYBODY

Jack: Where is everybody? The meeting was supposed to start five minutes ago.

Zara: More important, where's the coffee? And Danish? I had to skip breakfast to make this meeting. I'll go see if I can find out what happened to it.

Jack: Wait, don't go.

(Jack sits down and starts going through his mail. At 9:12 Sid arrives, carrying a brown bag and a pile of mail. He takes out a cup of coffee and a large jelly donut, and sits down to start eating and reading without saying anything to Jack. At 9:17 Martha and Mary Jo arrive, also with coffee.)

Martha: Hello Jack! Hi Sid! Sorry we're late. Big line at the coffee trolley. Where is everybody?

Jack: Zara went to find out where the coffee and Danish are. Beats me where Ned and Retha are... I didn't know you were coming to this meeting, Mary Jo.

Mary Jo: Well, I didn't either, but I met Martha in the coffee line and she said she thought I might be interested.

Jack: Did you interview any users?

Mary Jo: Well, no, since I didn't know I was coming.

Martha: That's okay, Mary Jo. I didn't get any interviews done either.

Jack: (looks at his watch and grimaces) It's 9:25, and I've got another meeting at 10:00. We'd better get started. The others can catch up when they get here. Sid, could you put away your mail?

Sid: Grnngh.

Jack: What?

(Sid groans and closes his folder with a loud slap on the table. Jack goes to the whiteboard and picks up a green marker. He starts to write but the marker is dry. He tries the only other marker, a red one, and it's dry, too.)

Jack: Drat! Does anybody have a marker pen that works?

Martha: I know where some are. I'll go get them. (She stands up and heads out the door, even as she's still talking.)

Jack: Wait! Oh, all right, I suppose we can start without her. Sid, will you take notes?

Grnngh.

Mary Jo: Is there an agenda?
Jack: Ned was responsible for that, but he's not here. Anyway, the meeting is supposed to determine what our users think of the requirements document so far. Hey, here's Ned, at last!

(Ned and Zara come in, Ned carrying a large coffee dispenser and Zara carrying a round aluminum tray of Danish covered with plastic wrap. Jack looks conspicuously at his watch.)

Ned: I know. We're going to have to do something about the cafeteria. If Zara didn't go down there, they would have just let this coffee sit through our whole meeting. Fortunately, I met her as she was coming out, or she'd have had to make two trips.
Zara: Thanks, Ned. I appreciate it.

(Zara struggles to get the plastic wrap off the tray. Sid puts down his mail and comes over to help her. After a few minutes, they finally get one corner open, and Sid takes two cheese Danish and goes back to his seat to resume reading his mail.)

Zara: Come on, everybody! Get 'em while they're hot!
Jack: Could we get started? We're trying to make an agenda.
Ned: Why? I've got the agenda.
Jack: (angrily) Right! And you weren't here!
Ned: So, do you want to fight about it? I'm here now, aren't I?
Jack: (cooling down) No, let's get on with it. I was just telling Mary Jo that the agenda was to report on interviews with our users, about their level of satisfaction with the requirements so far.
Ned: Well, that's your idea, but the real purpose of this meeting is to find out if we've managed to reduce the ambiguity in the requirements, before we give it to the users.
Zara: You mean I wasn't supposed to give them the requirements document yet?
Ned: You silly fool! You mean you let them see that piece of junk? Can't you do anything right?
Zara: I thought that was the agenda. Didn't you, Sid?
Sid: Ggrnmhgn.
Zara: See!
Jack: Let's stop fighting, and get on with the meeting. So, we have two items on the agenda—user satisfaction and measuring ambiguity. Anything else?
MAKING MEETINGS WORK FOR EVERYBODY

Mary Jo: If there's time, I'd like to report on the status of the United Way drive. It won't take more than five minutes, or maybe ten.

Jack: (tapping his watch) We've only got twenty minutes. And I don't even have a good marker pen yet!

Mary Jo: I'm sorry, but I'm not responsible for the marker pen. Don't you think that United Way is a good cause? Think of all those poor children who get lunch supplements, and the old people who need someone to wheel them around, and take them out once in a while. Just because you're one of the privileged few, you don't have to be arrogant!

Jack: All right! All right! Take your ten minutes. Maybe everyone will be here by the time you finish.

(Mary Jo proceeds to report on the progress of the United Way drive. After seven minutes, Martha shows up with a handful of markers. After twelve minutes, Jack starts tapping on the table. After fifteen minutes, Sid packs up his mail, stands up, takes two more Danish, muttering something about "no more cheese," and leaves. At 10:00, the door opens and Millie sticks her head in.)

Millie: Are you guys almost finished? We have this room for a meeting at ten.

Jack: (sighing) Sure, we're finished. Come on in.

Millie: Thanks. Oh, boy, can we have what's left of your coffee and rolls?

Jack: Sure, help yourself.

(Everyone remaining gets up to leave.)

Millie: Hey, Martha, don't take those marker pens with you.

Martha: That's okay. There's a red one and a green one there on the board.

(Martha leaves, and Millie sits down with a raspberry Danish. She looks at her watch.)

Reader: As you read through this chapter, look back at the meeting of Jack, Zara, and the rest to see how many principles of productive meetings were violated, and what they might have done about them.

8.1.2 Meetings as measurements

Meetings like this are supposed to be tools for accomplishing work, but they often seem more like melodramas, displaying the entire range of human emotions. Meetings are social tools. Like other tools, they come in all shapes and sizes, depend-
EXPLORING REQUIREMENTS

...ing on the job they are designed to accomplish. One job that every meeting can accomplish is to measure the health of the requirements process.

If your meetings are terrible, then your process is sick.

Noticing what's wrong with meetings will help diagnose just what's wrong with the entire process, and indicate what to prescribe for the malady.

8.2 Participation and Safety

Some of the most common things wrong with meetings are when people stomp but, fail to attend in the first place, or are there in body but don't seem to contribute. These actions are not just an annoyance, they are symptoms of an unsafe climate.

These symptoms indicate people are not participating fully in the requirements process. In order to get full participation, meetings must be made safe. How is this achieved?

People are different, and what will make one person feel safe can make another queasy. So to make the situation as comfortable as possible for everyone involved, negotiate an agreement on ground rules before dealing with the content of the meeting. As participants negotiate this agreement, specific reasons that people are not participating will become evident.

8.2.1 Establishing an interruption policy

For instance, many of us are in the habit of interrupting others, yet being interrupted makes some participants insecure about speaking up. Most interrupters, we find, agree that they don't want to interrupt, but they don't want to be interrupted, either. So we are usually able to get agreement up front that interruption isn't okay in our meeting. Also, nobody will object that this rule must apply to everyone. With that agreement in hand, the meeting facilitator can readily and politely silence interrupters.

8.2.2 Setting time limits

Agreement on time limits has a similar effect and, in fact, limiting individual speakers may be a precondition to an agreement on interruptions. Interrupters often worry that someone will simply keep the floor forever. If everyone agrees in advance that the facilitator can enforce, say, a two-minute time limit on each speaker, then the environment becomes safer for the interrupters, who can then more easily make it safer for everyone else.
8.2.3 Outlawing personal attacks and put-downs

Another important agreement that's easy to get up front is a prohibition on personal attacks or put-downs. Nobody will ever object to this rule in the calm beginning of a meeting, but many will violate it in the heat of events. Prior consensus on this rule gives the facilitator the group's permission to intervene in any personal attack situation and resolve it before it grows into a knock-down-drag-out battle (see Figure 8-1).

Figure 8-1. Agreeing in advance to refrain from personal attacks and put-downs will make it possible to resolve attack situations before they escalate.

8.2.4 Reducing pressure

Although some people thrive under pressure, a boiler-room type atmosphere shuts down many people altogether. Hence, every meeting needs devices for staying out of emergency mode. One important device is the time-out, when participants agree in advance that anyone can call one-minute or five-minute time-outs at any point without explaining why. They may need to get more information, to have time to think, to call home to get some crisis off their mind, or simply to use the toilet.
Agreeing that each participant is entitled to a reasonable number of unexplained personal time-outs, helps prevent meetings from being high-pressure events.

8.2.5 Allowing time to finish, yet finishing on time

Another pressure-reducing technique is to agree to allow all the time needed to get the work done right—but not necessarily today. In other words, stick to agreed time limits, but schedule a continuation of the meeting if business isn’t finished.

One way to determine if there’s more business is simply to ask each person, “Has every one of your ideas been handled to your satisfaction?” As one of our clients put it, “I like to be asked that question. I don’t have to get my way every time, but when I feel I haven’t been given a fair hearing, you can be sure I’ll bring up my idea in other meetings until I feel I’ve been heard.”

8.2.6 Handling related issues

What if their issues aren’t really part of the business of this meeting? For those cases, use a related issues list. As the meeting begins, obtain agreement that issues irrelevant to this meeting will be posted on a large blank sheet of paper in full sight and labeled, “Related Issues.”

When an issue comes up that seems out of order, the facilitator asks, “Is this meeting the proper place to handle your issue?” This question elevates the discussion from the details of the particular point and onto the level of the point’s relevance at a specific time and place.

Participants will feel safe and satisfied if they know that their idea has been written down—and that the issues list is not just a fancy garbage can. To achieve this goal, assign each related issue to a person responsible for getting it to the proper people as soon as the meeting terminates.

8.2.7 Amending the rules

Whatever your meeting rules, always get agreement up front so the first transgressors won’t think they’re being picked on. Unfortunately, something always comes up that you hadn’t anticipated. When that happens, handle the case fairly, then stop the process to get agreement on amendments to the initial rules.

For this reason, get advance agreement to amend the rules when something unexpected comes up. It also helps to agree in advance not to use the amendment process as a political tool, for one side to dominate another.

8.3 Making It Safe Not to Attend a Meeting

Meetings should be as small as possible—but no smaller. One of the common complaints about meetings is that too many people attend—people who really have little
or no interest in the business being conducted. This might seem to be the simplest of all meeting problems to solve. After all, why in the world would people attend a meeting in which they have little or no interest?

The answer is that most will tolerate a great deal of boredom in order to buy safety. People don’t want to miss a meeting that might be important to their work, so they go to a meeting where they probably don’t belong just in case something happens that might be important to them.

In other words, the presence of unnecessary people in meetings is a measure of uncertainty over what’s going on in the requirements process. In order to make it safe not to attend a meeting, use techniques to remove the uncertainty about what will actually be covered. Let’s look at four such techniques.

8.3.1 Publishing an agenda and sticking to it

The first safety technique is to publish an agenda well in advance and not to deviate from it, no matter how tempting. By deviating, you may handle the side issue well, but doing so sends a dangerous message to all those excluded from the meeting that they should have attended to protect their interests (Figure 8-2). Next time, they’ll ignore the agenda and show up, just to be safe.

8.3.2 Staying out of emergency mode

If you don’t want to punish people who believe the agenda, you need a way of handling emergency issues that doesn’t hurt people who don’t attend the meeting.

The first principle, of course, is to stay out of emergency mode. Rarely in requirements work is there a true emergency issue that can’t be postponed until it can be put on a published agenda. If you habitually find yourself yielding to the emergency argument, then perhaps you should pay attention to the real cause of this symptom. A project that’s in emergency mode during requirements work will be in death mode when it comes time to deliver a product.

8.3.3 Handling people who don’t belong

Even when you publish agendas and stick to them, there still will be unnecessary people in attendance. They may not have received a proper hearing in other meetings, so they attend whatever meeting is available, hoping to find a willing audience. To prevent this problem, be sure that participants don’t leave meetings unheard (see Section 8.2.5).

If surplus people show up anyway, don’t pretend they’re not there and hope they won’t interfere. Confront them immediately and if possible unobtrusively. The
Figure 8-2: Posting an accurate agenda in advance—and sticking to it—will help keep out people who don't belong in the meeting.

easiest time for a surplus person to leave and still save face is before the meeting officially starts. Once people have been there a while, it's hard for them to admit they were wrong. Instead, they are likely to start participating to "prove" they were right to be there in the first place.
8.3.4 Including the right people

Another reason people show up at meetings where they obviously don't belong is that they weren't invited to meetings where they do belong. The first rule of successful meetings is to be sure the right people are involved, as we discussed in Chapter 7.

8.4 Designing the Meeting You Need

Applying these general rules to every type of meeting assures a safe and healthy project culture. However, even with this culture of safety as a background, you'll still have difficulty with meetings unless you learn to tailor each meeting to the job at hand. Designing each specific type of meeting so it has its own appropriate climate will assure not just safety, but also productivity.

For example, there are meetings to disseminate information, to gather information, to raise everybody's spirit, to increase the number of ideas, and to decrease the number of ideas—each with its own structure, pace, and rhythm. A meeting that tries to do two different types of job winds up doing neither job well. Even if the structure is right, the meeting will be larger if it covers two distinct topics, as illustrated in Figure 8.3.

The first step in tailoring a meeting is to decide on its single purpose. If you need a pep rally and a technical review, for example, make two meetings. If they are scheduled one right after the other with the same people in attendance, take a break and change rooms. At the very least, change seats.

The final rule of successful meetings is the same as the Boy Scout Motto: Be prepared. Ninety-five percent of all meetings that fail could have been predicted failures before they ever started because of inadequate preparation. A checklist to take care of all the administrative matters, such as having fresh marker pens, to ensure every checklist makes sense, start it with the number one preparation question:

Do you have the appropriate design for this type of meeting?

Helpful Hints and Variations

As a guide to the design of meetings of all sorts, we have used *How to Make Meetings Work* with great success. It's an inexpensive paperback, and we often provide a personal copy to every participant in a project. It's always paid to itself in one meeting.

Figure B-3. Keeping meetings to one topic helps keep meetings small, short, and relevant to all in attendance.

- If your project seems to hold too many meetings, that may be a symptom of overstaffing. Alternatively, it may mean that the project organization is not broken down along natural lines of cleavage, so that nobody can do anything without affecting everyone else.
- Within any meeting, you can learn enormous amounts about various aspects of the project by observing individual behavior. We'll consider this use of meetings in Chapter 13 on tools for facilitating meetings.
Rescheduled meetings promote a ripple effect, leading to more rescheduled meetings, which in turn lead to more rescheduled meetings, and so on. If meetings are frequently rescheduled, or canceled, this may be a symptom of a lack of planning, or overload, or simply a project completely out of control.

8.6 Summary

Why?
Because of their central role in exploring requirements, meetings must be considered like any other tool: Design them, select them appropriately, train everyone in their use, and practice, practice, practice. In particular, use them as devices for measuring the health of the project environment.

When?
Meetings are used all the time. They may be formal, scheduled meetings, or informal meetings held in the corridor.

How?
Keep in mind the following:

1. Create a culture of safety for all participants.
2. Keep each meeting as small as possible, but no smaller.
3. Limit each meeting to a single type, and design that type to the job at hand.
4. Be prepared.
5. Use skilled facilitators, a subject to be discussed in Chapter 13.

Who?
Meetings involve everyone. Specific meetings may include fifty people, or two. We never know when we might find ourselves in a meeting, so we must always be prepared. Choosing the right people to be in the meeting, and those to exclude, one of the most important parts of being prepared.
Teaching Teamwork (http://computer.org/software/homepage/2002/05hil)

Thomas B. Hilburn, Embry-Riddle Aeronautical University
Watts S. Humphrey, Software Engineering Institute, Carnegie Mellon University

The software industry needs engineers who know how to produce quality products on schedule. Because computer science programs do not typically teach engineering concepts or practices, students often start their professional careers with little understanding or appreciation of the discipline needed to build quality products or the methods needed to keep projects within cost and schedule constraints. As a result, industrial software groups generally work without plans and have serious schedule and quality problems.

To meet industry's needs, many computer science programs now offer software project courses. Course methods vary, however, and there is little agreement as to what the courses should cover. Most academic programs have at least one software engineering overview course and a few offer several courses on the subject.

At Embry-Riddle Aeronautical University, students learn the Personal Software Process (PSP) in their first year, and the Introductory Team Software Process (TSPi) in their second and third years. The PSP uses a phased development paradigm to teach students fundamental engineering practices. The TSPi course teaches teamworking methods and exposes students to the practical issues and problems of team-based product development. Many other schools are taking an approach similar to ERAU's, and some schools have even instituted software engineering degree programs. This article reports on experiences with the TSPi course at ERAU and at several other institutions.

Software project courses

In the latest draft of volume II of Computing Curriculum 2001 (CC2001), the CC2001 Task Force states that to help students acquire professional skills as undergraduates, all computer science programs should include:

- Early opportunities for teamwork
- A complex project (usually undertaken in the senior year) designed and implemented by small student teams

Software project courses range from first-year two- or three-student teams working on brief (few-week) projects to one-year senior-level courses in which student teams work for real or quasi-real customers. At some universities (Carnegie Mellon, Georgia Tech, and the Milwaukee School of Engineering, for example), students join ongoing laboratory- or studio-based software projects that might involve both development and maintenance.

One hotly debated issue is the degree to which project courses should incorporate real-world experiences. One side advocates exposing students to real projects with vague and unstable requirements and unrealistic (or unmanageable) schedule constraints—a “sink or swim” strategy. The other side promotes developing toy projects with stable requirements and realistic schedules in a familiar development environment. Most project courses fit somewhere between these two extremes.
Designing a project course

The academic environment places some special constraints on project courses. First, you must restrict the course to a single term or plan, manage, and coordinate it over several terms. The dynamics of typical student populations make it difficult to maintain stable team membership for more than a single term. Moreover, academic environments can rarely find users to test multiple versions of the completed products.

The second problem concerns team formation. Computing faculty rarely have training or experience in creating effective software teams. Forming and building teams involves

- Selecting team members
- Assigning team roles
- Building cohesive units
- Assessing progress
- Providing meaningful advice and guidance

Third is the problem of project oversight and guidance. A software project course should not focus solely on producing working computer programs. It must teach the proper use of accepted engineering practices, such as project estimation and planning, requirements analysis and specification, and high-level and detailed design. These courses must also address practical project issues—for example, task scheduling and tracking, quality measurement and management, and phased development and control.

Unfortunately, the typical undergraduate computer science curriculum does not include these practices, and few faculty can provide practical and specific guidance in these areas. Without explicit guidance, most student projects become lessons in how not to develop software. Indeed, most of today's software developers learned to program by learning a programming language, with little or no guidance on disciplined methods or quality practices.

The Introductory Team Software Process

To address these industrial and academic problems, the Software Engineering Institute has developed a family of process improvement methods for individuals, teams, and organizations. The PSP helps students and professional software engineers organize and plan their work, track their progress, manage software quality, and analyze and improve their performance. It provides the necessary foundation for subsequent team-working courses. More than 30 institutions now offer introductory and graduate-level PSP courses.

Once trained in the PSP, students take a software project course using the TSPI, which is an academic version of the Team Software Process that industrial software teams use. TSPI support materials include a textbook, an instructor's guide, a support tool, and all the scripts, forms, standards, and methods needed to develop quality software products.

The TSPI process divides a software development project into development cycles, with the team producing part of the product in each cycle. Figure 1 shows the TSPI processes and their cyclic structure. Depending on the course constraints, students can complete two or three cycles in a one-semester course. In the final cycle, students integrate and test the finished system.
Process description

The TSPi starts with team building. During project launch, students form teams (of four to six students), establish team structure, and produce a project plan. This provides the essential foundation for a successful project. Teams then set measurable goals and objectives. For example, a team might state a quality goal as

- **Team goal**—Produce a quality product.
- **Measure 1**—More than 80 percent of the defects will be found before the first compile.
- **Measure 2**—No defects will be found during system test.
- **Measure 3**—At project completion, all product requirements will be correctly implemented.

One of the most important factors in effective team building is defining clear roles for each team member. Table 1 describes the TSPi roles. The process details the responsibilities and activities for each role during each phase. Thus, at the beginning of the project, students understand their roles and know what is expected each week.

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team leader</td>
<td>Leads the team and ensures that engineers respect their process data and complete their work as planned</td>
</tr>
<tr>
<td>Development manager</td>
<td>Leads and guides the team in product design, development, and testing</td>
</tr>
<tr>
<td>Planning manager</td>
<td>Supports and guides the team in planning and tracking its work</td>
</tr>
<tr>
<td>Quality/Process manager</td>
<td>Supports the team in defining the process needs and establishing and managing the quality plan</td>
</tr>
<tr>
<td>Support manager</td>
<td>Supports the team in determining, obtaining, and managing the tools needed to meet its technology and administrative support needs</td>
</tr>
</tbody>
</table>

The TSPi process covers all of the essential elements for effective quality management—for example, test planning during the requirements and design phases and structured test plan reviews. Each team formally inspects the requirements and design specifications and holds both personal code reviews and peer code inspections of each product unit.

Students receive review and inspection process documentation that includes process scripts, guidelines for developing review checklists, and forms for recording and reporting inspection data. Because all TSPi artifacts receive quality reviews and inspections, student teams generally produce quality products.

During the planning cycle, teams produce a comprehensive plan that includes

- A list of the products to be produced and their estimated sizes
- A list of tasks to be completed and the team member responsible for each task
- Estimated effort to complete each task
- A week-by-week schedule that identifies the tasks to be completed and the available student work hours
- A quality plan that estimates defects to be injected and removed in each development phase
- A template summarizing the product’s estimated and actual size, effort, and defect data

TSPi teams develop and track their plans with a spreadsheet tool developed by the SEI. This tool supplies a host of metrics with which faculty and students can track and assess process fidelity, product quality, plan accuracy, and project status.

Faculty and student feedback

More than a dozen university computing programs have used TSPi in software project courses. Several schools, including Auburn University, Carnegie Mellon University, and the University of South Carolina, have shared with us their experiences using the TSPi in both undergraduate and graduate courses. In most cases, instructors presented TSPi material in a laboratory setting using a combination of lecture and informal coaching.
While course results varied considerably, all faculty felt that the TSPI course was worthwhile. Many also found that they could adapt the material to the level, background, and sophistication of their students. Not surprisingly, instructors who had used the TSPI several times reported greater success and satisfaction than those who taught it only once.

Early problems resulted from defects in the initial version of the TSPI tool as well as its size—it was too big to easily distribute for team use. Also important was students' PSP preparation. The better their preparation, the more likely they were to appreciate and properly use the TSPI process. The following quotes are faculty reactions to the TSPI process and course.

- "TSPI gives very good material and insights about how to help students organize as a team: role definition, visible commit-ments, meeting organization, schedule, and so on."
- "I think the TSPI is a very good vehicle for teaching a project course once the students have PSP knowledge... it provides a lot of good, usable structure that students can pick up, try out, and take away from the course to their jobs."
- "I will never teach a team project course without TSP again!"

In general, students were most positive when they studied PSP in their first year and continued to use it in subsequent courses. When instructors introduced PSP and TSPI late in the curriculum, more students objected to the required discipline. Also, students who are already accomplished programmers and have never experienced the industrial quality and management problems the PSP and TSP address, can be somewhat negative, especially at the beginning of the TSPI experience.

The most common complaints about PSP and TSP courses concern data gathering, planning, tracking, and quality management. Although these disciplines are not easy to learn or practice, the best industrial work requires them. Students who do not learn these planning and quality management skills through their coursework will rarely learn them on the job.

Data analysis

The computer science program at ERAU introduces the TSPI in a second- or third-year undergraduate team project course focusing on software engineering fundamentals. Prerequisites are programming experience in an object-oriented language (such as Ada, C++, and Java) and PSP experience.

The TSPI course aims to expose students to software team project issues, principles, methods, and technology. Because about one-third of the course time is devoted to software engineering lectures and discussions, there is only time for a two-cycle process. Each team follows the TSPI process but adds a customer review at the end of the first cycle and a more formal customer acceptance test at the end of the second, with the instructor acting as the customer.

For consistency, we took the TSPI data in the following analyses from the ERAU courses. Since Fall 1998, ERAU has offered the TSPI course each semester. Through Spring 2001, 42 teams had completed projects using the process. Each team collected data on task completion time, defects found in review, compilation, testing, and size of resulting artifacts. They entered these data into the TSPI tool to produce metrics for tracking progress, planning support, and assessing product quality and team performance.

In analyzing data from the 42 ERAU teams, we found several errors and omissions. Also, data from the course's first year were organized differently from other years or were not available. For this article, we identified 18 teams that reasonably represent the work you can expect when students follow the TSPI process properly. Table 2 summarizes the data from these 18 teams. While they produced slightly more code in cycle 1 than in cycle 2, the cycle-2 code was generally more complex. We attribute the greater productivity in cycle 2 to the teams'
better understanding of the TSPI and increased efficiency.

Defect density did not significantly improve from cycle 1 to cycle 2, but test defects increased. This is partly due to increased code complexity but also likely indicates that teams discovered defects in the second cycle that they had missed in the first. Most importantly, the teams had not yet developed effective precompile reviews, as evidenced by cycle 2's average code-review rate of 309 lines per hour—well above the recommended maximum rate of 200 lines per hour. This area requires improvement.

Teams estimated effort quite accurately for cycle 1 but overestimated effort for cycle 2. This occurred because the teams based their cycle-2 plans on cycle-1 data, but their average productivity improved 66 percent from cycle 1 to cycle 2. In a three-cycle course, the students would have recognized and corrected the problem. Figure 2 shows the effort distribution over the development phases.

The 15.6 percent of effort devoted to requirements is appropriate for the problem size and complexity, but the 13 percent spent on design is much too low and is probably one reason for the high test defect density. The course introduces a simple object-oriented design methodology, but students do not get intense design experience until a follow-up junior-level software analysis and design course.

Figure 3 shows the average defects injected and removed in each project phase. As shown, teams found and removed about 80 percent of the defects before testing began. This is remarkable for sophomore and junior student teams.

Student assessment

After each cycle, students completed an anonymous survey. Students, especially those who had been involved in previous team projects, were generally positive about their experiences. About 75 percent of the students were positive about TSPI, and more than 90 percent felt that working on a team project was a worthwhile-learning experience.

The survey also included questions about what students liked most and least about the course.
Overwhelmingly, students were most positive about having clearly defined roles and a process that describes the tasks for each project phase. The biggest complaint was the amount of paperwork involved. Although most students grudgingly admitted that documentation, data collection, and data analysis were important, they did not enjoy it.

Course suggestions

Designing and implementing a software project course can be challenging and demanding. Faculty who have taught such courses will tell you that a successful teacher must work hard, properly plan and prepare, track team progress, and assess both the team members’ work and their products. To improve the likelihood of a successful effort, you should follow several guidelines.

First, clearly identify course goals. Incomplete or fuzzy goals lead to incomplete or fuzzy results. Second, if the course is time-restricted or will represent students’ first team project experience, use a modest and well-defined problem. Real-world, complex problems can doom a team to frustration and failure.

Third, use a defined team process for the project work. We believe the TSPi includes all the elements needed to build, guide, and support effective teams. However, whatever process you use, be sure it includes

- A detailed written description of the project and the process to be used. This should include support for planning, tracking, and configuration management; procedures and standards for requirements, design, implementation, and testing; and methods and procedures for inspections and reviews.
- A description of team roles and associated responsibilities.
- A list of data teams should collect for use in tracking and assessing performance and product quality.
- A cyclic development approach. We believe incremental development is not only good engineering practice but is a pedagogically sound method of teaching and learning.

Fourth, enforce process discipline (phased development, data collection, documentation standards, reviews and inspections, and so on). After you have decided what activities and deliverables are important, make sure your students follow the process. Students do not easily accept or understand the need for process discipline. Most students (and many software engineers), for example, do not enjoy documenting project work, but few will debate the need for it. As teachers, we must motivate and help our students understand the need for discipline. Easily acquiescing to complaints and criticism can cause teams to lose confidence in the process and produce a chaotic "process du jour" work environment.

Finally, the course instructor should move from the lecturer mode to a coaching mode. Coaches not only explain methods and procedures for carrying out tasks, they also track and observe team and individual performance and help team members improve. Moreover, they do not throw teams into the big game (an industrial-strength project) before they have learned and practiced the basics (with well-defined, modest projects).

Before initiating a TSPi-based course, you should confer with other faculty who have taught the process. We recommend that you attend the TSPi faculty workshop offered each summer. We have also found that the PSP and TSPi are most effective when integrated into an entire software curriculum rather than treated as isolated technologies.

At Embry-Riddle, we have had excellent success using TSPi in introductory software engineering and senior design courses. Of the 42 TSPi teams in the introductory course, those that carefully followed the process had the best performance and produced the best products. However, all teams finished on time with working products. This is in marked contrast to our previous experiences with team projects where many were late or did not produce a working product. Thus, "teaching teamwork" can make a significant difference.

Acknowledgments
We would like to thank the following faculty who shared with us their TSPi experiences: Bob Cannon (University of South Carolina), Rubby Casallas (Rochester Institute of Technology), Jorge Diaz-Herrera (Southern Polytechnic University), Greg Hislop (Drexel University), Keith Olson (Montana Tech), Mark Sebern (Milwaukee School of Engineering), Jim Tomayko (Carnegie Mellon University), and David Umphress (Auburn University).

Several reviewers kindly assisted us in preparing this article. We very much appreciate the helpful comments of Julia Mullaney, Bob Musson, and Bill Peterson. We also thank the IEEE Computer Society reviewers for their insightful comments and suggestions. Our thanks also go to the IEEE Computer Society editorial staff for their professional assistance in producing the finished article.

The Personal Software Process, PSP, Team Software Process, and TSP are service marks of Carnegie Mellon University.

References


