what are the alternatives? What should we measure? Or do you suggest not using any operational measurements?"

"Not at all," he is quick to deny. "Do you want to schedule another meeting to discuss it?"

We all take out our diaries.

Chapter 16

"So far, does it make sense?" I ask them.

I can see how Mark, Ruth and even Fred are ready to jump down the throat of anybody who does not agree. Luckily, there are no targets.

The extent to which these three have become zealots is surprising. Their turnaround happened last session, when I guided the class to develop the solution. Not that they were dragging their feet before, but now it's like they have seen the light; they behave as if all their future depends on implementing it.

Tuesday morning all three came to my office and stepped all over each other trying to convince me to come to Genemodem and talk to Mark's team.

"The solution is almost against human nature," Mark was quite desperate. "I don't know how to persuade people to strip away their safety."

"And group consensus is vital," Ruth added.

Fred continuously repeated, "If the team doesn't buy in, results will not follow."

They didn't have to press so hard; I would give my left arm for the opportunity to try our ideas on a real project.

Now it's Thursday, and for the last three hours I've been talking to the team assigned to the development of the A226 modem. I don't know a thing about modems, nevertheless, I have succeeded in convincing them that I do know a lot about the pitfalls of managing a project. It wasn't easy, but I was able to get a true consensus on the current situation. It's written on the board.

1. We are accustomed to believing that the only way to protect the whole is through protecting the completion date of each step.

As a result,

2. We pad each step with a lot of safety time.

3. We are suffering from three mechanisms which, when combined, waste most of the safety time: a: student syndrome, b: multi-tasking and c: delays accumulate and advances do not.

We also agreed, and that was much easier, on what does make sense: the five focusing steps are also written on the board.

The real challenge is still ahead. Can I convince them to adopt the logical derivative of all that's written on the board? Can I lead them to develop the solution?

I take a deep breath and dive in.

"So what is the constraint of a project? What should we choose as the equivalent to the bottleneck?"

No answer.

Since they are interested in the subject, such silence can mean only one thing. I asked a question that demands too big a leap. I'll have to slice it into smaller chunks.

"Okay. What I'm asking you now is to ignore the pile of problems that you currently face and imagine the following scenario. You developed an excellent product, the A226 has been released on time and marketing turned it into a big success. Where is the constraint of the company?"

"In production," one answers.

"For sure," another backs him up. "With our best products, production never succeeds in adequately supplying the initial market demand."

"So in our futuristic scenario, there probably will be a bottleneck in production. What is a bottleneck?" And I answer my own question, "a bottleneck is a resource with capacity that is not sufficient to produce the quantities that the market demands. In this way the bottleneck prevents the company from making more money."

They don't have a problem with that.

"Let's go back to the situation as it now stands. The A226 is in your court, in engineering. What now prevents Genemodem from making more money from A226?"

"We haven't finished developing it yet."

"Exactly," I say. "So, in engineering, the desired performance is not quantity but? . . ."

"Finishing the development on time." They don't have any problem answering.

Mark is not entirely happy. "Or before time," he must interject.

We are in the team's room; there must be a PERT chart of their project somewhere. I locate it hanging on the opposite wall. It's big and colorful. I cross the room and stand beside it.

"Look on this chart," I ask them. "It represents all that has to be done to develop your modem. What dictates the lead time from start to finish of this project?"

"The critical path," they immediately answer.

"So what is the constraint of a project? What should we choose as the equivalent to the bottleneck?" I repeat.

"The critical path."

It is as simple as that. Why did it take me a whole week of floundering until I found the answer? Probably because the obvious is sometimes the last thing we see.

"Fine," I say. "We identified the constraint. What do we have to do to exploit it?"

"Don't waste it."

I fooled myself long enough on answers that look good but are meaningless. What's the meaning of "don't waste the critical path?"

It doesn't have any meaning until you express it in different words, and that's what I have to squeeze out of them. I had a hard time doing just that with my class; now I'm much better prepared.

"Don't waste what?" I ask.

We go through the expected questions and answers. The answers move from "critical path" to "time" and then, when we once again clarify that the time is based on estimates and pressures, we finally arrive at: "Don't waste the time allotted for the critical path." Any waste here will delay the project.

Semantics? Maybe. But semantics are sometimes crucial.

Now I ask the big question. "How do we, currently, waste the time allotted for the critical path?"

Based on the last three hours, they now have a lot of answers. Too many answers. But they seem to go out of their way to avoid the obvious one. Nobody mentions the core problem, the fact that we pad each step with a lot of safety time.

Maybe it's because of their fear of letting go of their precious local safeties, maybe they just don't see it. I don't know. I only know that I have to spend quite a lot of time relating each answer back to what is written on the board. Pointing out, over and over again, that it won't help us to deal with symptoms, we must deal with the core problem. We must bite the bullet and deal with our tendency to pad each step with a lot of safety time.

Maybe it was so difficult because of the implied action: "So, you want us to reduce the time to one-third?"

"I don't want you to do a thing," I clarify. "I just point out the unavoidable conclusions stemming from what you said. Do you agree that we shouldn't protect each individual step with safety time?"

"Yes."

"Do you agree that each step has a minimum two hundred percent safety?"

After more beating around the bush, the answer is still, "Yes."

"One plus one is two."

I had to repeat it, in one form or another, at least five times. At last someone asks, "But are we going to put in some safety?"

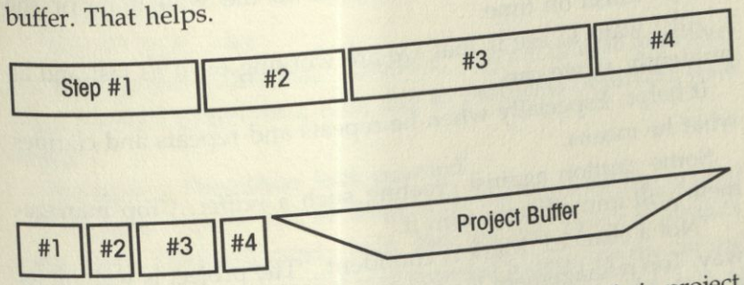
"Of course," I say. "Murphy does exist. But we are going to put the safety where it's going to help us the most. We are going to put it so it will protect the constraint. What is our constraint?"

"The critical path."

"So we have to protect the completion date of the critical path? Correct?"

"Yes."

"We put all the safety at the end of the critical path. Stripping the time estimates of each step frees up sufficient time to create a 'project buffer.' " I draw two pictures to clarify what I just said. The original critical path and the critical path with the project buffer. That helps.



They start to translate it into what it means for their project. The A226 has to be ready six months from now. There are many steps on the critical path that are not done yet. Their current estimates lead them to believe that they will be late by two months, which in their environment is verging on a crime. They already talked about saving time by compromising on some of the performance of the modem, but Mark hasn't allowed it yet.

"If we take the current estimates for the remaining steps," I

help them, "it adds up to eight months. If we do what we just said, we can create a project buffer of more than five months!"

Nobody likes it. Not Mark, who booms that a five-month buffer is much too much. And not his people, who categorically state that anyone who thinks they can finish their individual tasks in one third of the time must be out of their mind.

For a while, it's a zoo.

Mark has to use his voice at full volume to quiet them down.

"With these trimmed estimates," he tries to calm them down, "I understand that for each individual step, the chance of finishing is only fifty percent."

There is rapid response. "Fifty percent, ha!" "Less than ten percent." "Not a chance."

It's one thing to agree that theoretically there is more than two hundred percent safety. It's another thing to commit to a trimmed estimate. Inertia.

"I'm not going . . ." Mark's voice booms above everybody else. "I'm not going to put anyone to the wall if he or she doesn't finish on time.

All I want to see is that we are working on it as fast and as prudently as we can."

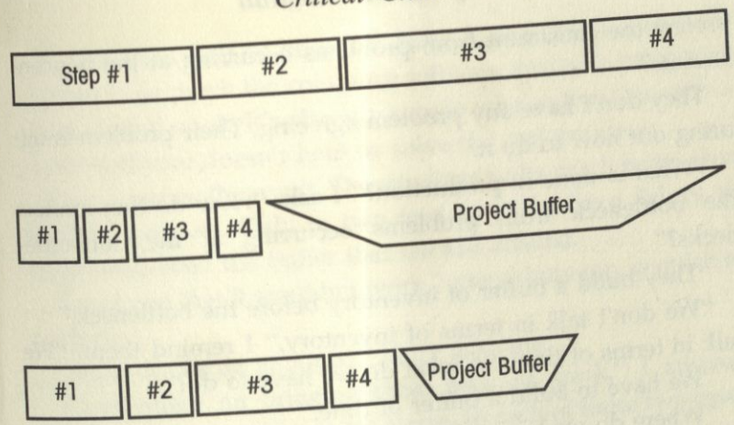
It helps. Especially when he repeats and repeats and clarifies what he means.

Some caution against creating such a buffer. "Top management will immediately trim it."

"Not a chance," Mark is confident. "The project is well on its way. Top management is not going to mess with it now. We just have to deliver on, or before, the original promised date."

At last they settle on the following. The time allotted for each step will only be cut by one-half (Mark squeezes from them some lip service that they will try to beat these times). On the other end, the project buffer will not be equal to what they trimmed. It will be set to only half of it. Mark is adamant that two months is more than enough. I suspect that he insisted on it in order to put the project back on the promised date.

I add their version to the diagrams on the board.



When all this is settled, Mark passes the baton back to me.

"Exploit the constraint," I start. "Don't lose any time on the critical path. We really can't do a good job of exploiting the constraint until we do the next step, until we subordinate everything else to it."

"Why?" Ruth asks.

"Without subordination," I answer, "we are unable to protect the constraint from losing time due to problems occurring elsewhere."

She agrees. The others look puzzled.

I explain. "So far you've dealt with the steps of the critical path itself. That will surely help. But tell me, hasn't it already happened in this project that you suffered a delay on the critical path because of a problem that occurred outside the critical path, in one of the many feeding paths?"

They laugh and start to bombard me with examples. I don't understand their jargon, but I let them speak. It's important that they realize that most problems that impact the critical path do not occur on the critical path itself. That's the only way they'll realize that subordination is not a nicety, it's a must.

When they somewhat run out of steam, I ask, "Do you agree that we must do something about it? That somehow we must

protect the constraint from problems occurring at the nonconstraints?"

They don't have any problem agreeing. Their problem is figuring out how to do it.

"What is done in production?" I ask. "How do they protect the bottleneck from problems occurring at the nonbottlenecks?"

"They build a buffer of inventory before the bottleneck."

"We don't talk in terms of inventory," I remind them. "We talk in terms of time. So what do we have to do?"

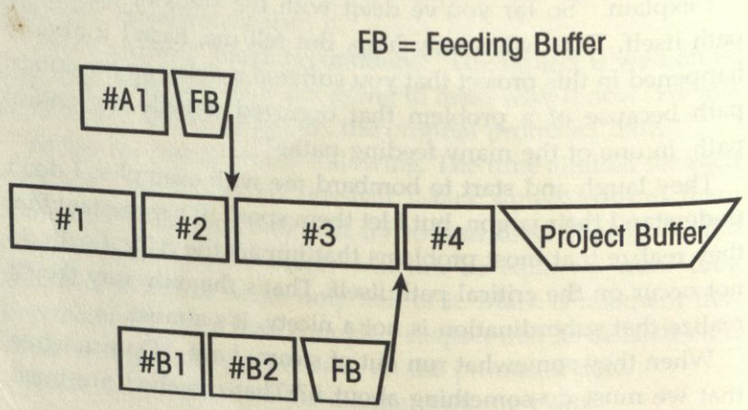
"We have to build a buffer of time."

"Where do we have to build these time buffers?" I go to the PERT chart hanging on the wall and put a ruler on the critical path. "What is the meaning of 'before the bottleneck' in our environment?"

It doesn't take them long to conclude that we must insert a time buffer at the points where a feeding path merges with the critical path.

"Where are we going to take the time from?"

By now they have the formula. For each feeding path they decide to cut the original time estimates of the steps in half and use half of the trimmed lead time as a 'feeding buffer.'



In less than half-an-hour we have a new PERT chart. It's amazing how much the computer software available today simplifies clerical work. It's also amazing to what extent this sophisticated software doesn't help us solve the real problems.

They examine the result. The situation looks much better than one might expect. Only in two feeding paths have delays already swallowed the buffer that we just created.

"I told you that it wouldn't work," one skinny guy is quick to conclude.

"What do we do about it?" Mark asks me.

"Concentrate on bringing them back on track," I answer. "But I don't see any reason for alarm. In each of these two cases the delay is about two weeks. Don't forget, if you really can't get it back on track, you still have a project buffer of two months."

That's an angle they hadn't thought about yet. The 'feeding buffer' protects the critical path from delays occurring in the corresponding noncritical paths. But when the problem causes a delay bigger than the feeding buffer, the project completion date is still protected by the 'project buffer.'

They like it. My attention is now on something else. On each of the two paths that are very late, the step that they are working on now is marked in red. This red color probably indicates top priority. My problem is that there are many other steps that are red.

I point to one red step where, according to their numbers, the buffer it feeds is still untouched. "What is the urgency in finishing this step?" I ask.

Nobody answers. Mark moves closer to examine what is written on the red step. He turns to one of his people. "Why the urgency?"

"I don't know," this person answers, and points to our skinny guy.

"You see the next step?" He has a matching squeaky voice. "My people are supposed to do it."

"So?"

"And they can't start before that step is finished."

I still don't get it.

Mark doesn't either.

"You know that they finished everything else," comes the squeaky explanation.

They are all over him. The efficiency syndrome is alive and kicking, not just in production. I wonder how many of their 'emergencies' are such false alarms. They probably wonder the same because they check every red dot on their chart. In the end, only four remain.

It's much better, but we still haven't finished.

"There is something else that might delay the critical path," I remind them. "Sometimes everything is ready for a step on the critical path except for the appropriate resource, which is still busy doing something else."

We discuss how to prevent such delays. They invent the resource buffer.

That's a concept I haven't yet covered in class, and their debate teaches me a lot about the practical aspects of exactly how to implement it. But I can't stay any longer. It's our theater night, and I'm not going to let Judith down.

I leave. They are deep in the details.

Chapter 17

"It's much easier to implement than we hoped," Mark says, concluding his presentation to the class.

"Any results?" Brian asks.

Mark fidgets. "It's only four weeks since we learned in class what to do, and it's three weeks since we actually implemented it. Now, you know that in a development project of two years . . ."

"Three weeks is nothing," Brian completes Mark's sentence. "I know. Still, can you see any tangible results?"

"What do you mean by tangible results?" Ruth is slightly snappy. "Do you expect that in three weeks we'll complete the project? I hope not. But then, what else would you call tangible?"

"Hey, I'm not criticizing," Brian defends himself. "I think that what you've done is terrific. I just wonder if you have any hard evidence of real progress. That's all."

Fred puts his hand on Ruth's arm, and to Brian, he says. "There are some numbers. But first I have to explain something.