

Successful Software Management: 14 Lessons Learned[©]

Johanna Rothman
Rothman Consulting Group, Inc.

Successful managers realize that they need to balance the needs of the business, the employees, and the work environment to be effective. In this article, the author summarizes her experiences in determining the work to accomplish and planning it, managing successful relationships with the group, and managing reactions to typical management mistakes.

Shortly after becoming a manager, I dragged myself home from work, flopped on the couch, and said to my husband, “This management stuff is hard. Nothing I learned in school prepared me for this people stuff. And that *management training*, that was just form-filling-out nonsense. The soft skills – dealing with people – are the hardest.” My husband chuckled and commiserated.

If you are like me, and you started your professional career as a technical person, this *management stuff* is difficult to do. Not the forms, although the forms can be irritating, but the difficult part is knowing how to deal with people, and completing the work your organization expects of you. I have now had more than 15 years of management experience, and have learned a number of lessons about managing people.

Define the Manager's Role

When you become a manager, your role is to organize purposefully [1]. For me, that means creating an environment where people can perform their best work. As a software manager, that means I work to create business value by balancing the needs of the business, the employees, and the environment. There is no *one right way* to do this; every organization is different. However, the following lessons have served me well in numerous organizations.

1. Know What They Pay You to Do

I have been a manager of developers, testers, and support staff. You would think it would be easy to know what the company paid me to do. However, my mission as a test manager – to report on the state of the software – is sometimes different from what the organizations desire: to find the Big Bad Bugs before the customer does, or bless this software. Even my mission as a development manager – develop the team members as much as the software – was different from what another organization desired: create software just good enough that we can be

bought out.

My mission does not have to be the same as yours, and you may modify your mission as your organization changes. However, delivering on your mission as a manager is what your organization pays you to do. What is important is to notice when your title, your mission, and what the company pays you to do are not synchronized.

One quality assurance (QA) manager said it this way, “My management only wants to me to manage the testing, not raise risks, look for process improvement

*“When you align
yourself with your
manager's priorities, you
do the work they pay
you to do.”*

opportunities, or even gather and report on what I think are standard metrics. My manager and I are both frustrated. Focusing on just the testing is wrong.” This QA manager has at least one alternative – change his title so that he and the organization both know that he is not attempting to perform organization-wide process improvement, to clarify expectations in the organization.

Doing what the organization pays you to do, and not doing what they do not pay you to do makes a huge difference in how successfully you and your group can accomplish your mission. Make sure you clarify your mission at your organization so you can create to-do and not-to-do lists. These lists help you plan the work – for you and your group.

One development manager who temporarily took over installations from the tech support people realized that he no longer had a development team, but an installation support team. The development manager put installations on his not-to-do list and developed a plan to move

installations back to tech support.

When you align yourself with your manager's priorities, you do the work they pay you to do.

2. Plan the Work: Portfolio Management

It is easy to be reactive at work and feel buffeted by the requested changes of your group. It is harder and necessary to be proactive and plan your group's work, even if that work changes every week. For me, planning includes these activities: identifying the project portfolio (i.e., new work, ongoing work, periodic work, *ad hoc* work), developing strategies for managing the work for each project, and knowing what done means for each project. One of the questions I like to ask is, “How little can we do?” I do not want to shortchange any project, so by asking about the minimum requirements, I can accommodate more projects successfully.

Part of planning the work is assigning the people to projects. I assign people to one important project then allow them to take on little bits and pieces of less important work when they need a break or are stuck on the important project. I avoid context switching (moving from one unrelated task to another) as much as possible.

3. Accept Only One No. 1 Priority at a Time

I have worked for many managers who demanded that my staff and I work on several top-priority projects simultaneously.

Senior managers perform different work than first-line and middle managers. It is not possible for senior managers to work on more than one top-priority task at a time. However, because they tend to have more wait states in their work, these senior managers are under the illusion that they are working on several top-priority projects at the same time.

Middle and first-line managers can only work on one No. 1 priority task at a time. However, sometimes we confuse urgency and importance [2]. At one

organization, I would arrive at work in the morning, check my voice mail, and respond to all message requests. That took until noon. Again after lunch, I would check my e-mail and voice mail and run around responding to those urgent requests. After a week of this, I realized I was not performing any of the important work such as planning for the group and lab, reviewing critical development plans, or planning my hiring strategy. I also realized that although people marked their e-mails and voice mails *high priority*, they did not utilize the information I had given them at the time I responded.

I stopped responding immediately to urgent requests and re-planned my days. While I still checked voice mail and e-mail, I tended to ask more questions about the deadlines for requests. Prioritizing requests helped me manage my management time. I still had the problem of too many high priority projects coming into my group, so I asked my manager these questions:

- If you could have one project first, which one would it be?
- What are the consequences if we release any of these projects late?

We talked and negotiated which projects had to be completed when and why. When I understood the trade-offs between projects, I was able to manage the work coming into my group.

4. Commit to Projects After Checking With Your Staff

Business needs change. Sometimes your manager will grab you in the hall and say, “Hey, can you do this project now, and finish it in two months?” Or, a senior management planning committee will call you into its meeting and say, “We need this project now. Can you commit to it?”

It is very tempting to say yes. However, saying yes is exactly the wrong thing to do. You can say, “Let me check to see if my previous estimate is still accurate, and I’ll get back to you before 5 p.m. today.”

If you say yes, you are training your senior management to ask you for answers when you do not know the answers. You have also committed your staff to a project that may not be within the scope you originally estimated.

5. Hire the Best People for the Job

Especially if you manage many projects, your greatest leverage point is in hiring appropriate staff. Too often, we hire people who have similar technical skills and

personalities as the people already in our groups. Hiring people who are *just like the ones we have now* does not always gain the best people for the job.

When you hire people your staff thinks are great, you increase morale in the group, and you increase your group’s capacity over time. I recommend you develop a hiring strategy that identifies the technical and soft skills you are looking for, and that you choose a variety of techniques for interviewing.

I have found auditions [3, 4, 5] to be an essential technique for interviewing technical staff. I normally create 30- to 45-minute auditions to see how a person works in a particular setting. Auditions help candidates show what they can do. If you organize a congruent audition, you do not trip people up on esoteric ideas or jargon; you create a simplified situation that

“Hiring people who are just like the ones we have now does not always gain the best people for the job.”

the candidate could encounter at work. Watching the candidate, or having the candidate explain their answers/results is a powerful interview technique.

You can create auditions for any position, including project managers, developers, testers, writers, support staff, analysts, systems engineers, product managers, program managers, and people managers. Define the behaviors you require in a position, and then create an audition using your products or open source products to see the person at work. Create auditions that are 30 minutes long to start. If you are having trouble deciding between multiple candidates, define another audition that is one hour long and invite the candidates back to see how they manage that audition. Auditions show you how the person works at work – priceless information.

I also recommend behavior-description interview questions [5, 6] to understand how a candidate has performed in previous jobs. Behavior-description questions are open-ended and ask the candidate to tell you the story of previous work. For example, to understand how a project manager deals with a project team who has not yet met a schedule, ask this

closed question: “Have you ever managed a project where the team had trouble meeting the schedule?” If the answer is no, you can decide if the project manager has enough experience to manage your team. If the answer is yes, ask the open-ended behavior-description question, “What did you do? What actions did you take on that project to help the project team meet the schedule?” The answers you hear will help you assess that candidate’s ability to work in your organization.

6. Preserve Good Teams

Part of my hiring strategy is to hire people who fit into my already-existing team, but sometimes you inherit teams or a project has completed and a team is ready to move on. When a team is successful, I try to keep them together so they can continue working well together. I may bring more people into the team, one at a time, especially if the team has been highly productive. But I do not scatter the productive team and hope they will form more productive teams. That just reduces their productivity.

Teams can overcome bad management and bad processes, but they cannot overcome a team un-jeller. A team un-jeller is the person who walks into the lunchroom, and suddenly everyone else leaves. Or, the un-jeller creates an argument out of every conversation. If you have a team un-jeller, find another place for that person to work, preferably for your competitor.

7. Avoid Micromanaging or Inflicting Help

Many of us were software developers, testers, analysts, or some other technical role before we became managers. When we were technical contributors, we knew how to perform the technical jobs. However, once you have been a manager for a while, you probably will not know precisely how to perform the employee’s job.

I once had a boss who liked to creep into my office, stare over my shoulder, and say, “On line 16, shouldn’t that be a ...” By the time he reached “16,” I had jumped out of my chair, become flustered, and lost my concentration. Micromanagement neither gets the job done faster, nor does inflicting advice or help.

On the other hand, you and the employee both need to know that the employee is progressing. I ask my staff to decide when they have been stuck for too long (time-box the work). Some tasks

require weeks of study, but most tasks require days or hours. If the employee spends more than the agreed-upon time on the task, their job is to ask for help. As the manager, your job is to find them help, not necessarily inflict your help.

8. Treat People Individually and With Respect

Buckingham and Coffman [7] claim that each employee's relationship with his or her manager is key to that employee's success and long-term happiness in the organization. That means we need to treat people fairly, but uniquely, so that we build and maintain the best possible relationships with each employee.

Everyone has his or her own preferences, especially in their communication patterns, and how they organize their thoughts about work. Some people prefer e-mail communications; some prefer in-person discussions. Some people want to understand all the reasons behind your requests, and others will take the request at face value. Some people need to gather data to make decisions; others will develop a model about the situation and make a decision based on that model.

It does not matter if people work top-down or bottom-up, or if they want to talk in person or by e-mail. What matters is that you, within reason, accommodate everyone's uniqueness.

I once managed two very talented developers who shared a large office. Begrudgingly, they allowed me to have 20-minute one-on-ones with each of them every two weeks. In between, if I wanted to talk to either of them, I had to e-mail them first – dropping in was not allowed. I treated them differently than the other people in my group, but fairly, considering their preferences.

They frequently worked on the same software. They never spoke to each other aloud, they only communicated via e-mail even though they shared an office. Because they were so successful at their work together, and even mentored others in the organization by e-mail, their communications preferences were a bit odd but acceptable. If I had tried to change them to meet my needs and work with them the same way I worked with the other people, none of us would have been happy.

9. Meet Weekly With Each Person

Even if you have hired stars, you still need to know each person's progress on their tasks, and how the project as a whole is

progressing. I use one-on-ones weekly to meet with each person. We discuss the employee's progress on his or her tasks. Sometimes, tasks are amorphous and it is difficult to know when to stop or if the employee needs help. I ask each employee to show me visible progress on each task: drafts of plans, multiple designs, prototype test results, anything that shows me the employee is making progress and is not stuck. If the employee needs help completing the task, we discuss what kinds of help are appropriate.

I receive many benefits from these weekly meetings. I learn what everyone is doing and can track it in my notebook. It is easy to write up useful performance evaluations, including examples of successful and not so successful actions the employee has taken over the year. And, because we meet weekly, I can give feed-

“Buckingham and Coffman [7] claim that each employee's relationship with his or her manager is key to that employee's success and long-term happiness in the organization.”

back then, not when we make time. I also reduce the number of staff interruptions because everyone knows they can ask me non-urgent questions during the one-on-one. I can perform weekly career development and learn if my staff has personal issues affecting their ability to do their jobs.

If I am managing more than eight people, I meet biweekly with more senior staff because they need less direct supervision.

Some of you are probably thinking you do not have time to meet with everyone once a week. However, if you do not set up specific times to meet with everyone, you tend to either not know what people are doing, or you are interrupted frequently by your staff with questions.

10. Plan Training Time Each Week

Technical work is constantly changing; most of the technical people I know enjoy

learning new things. If you have a budget for formal training, that is great. Even if you do not have a budget, plan weekly training time in the form of brown-bag lunches, presentations from other groups in your organization, an internal user-group meeting of one of your tools, or presentations from people in your group about their successes or difficulties.

I use the weekly group meeting as a time to deliver the training. When I managed development groups, I organized this internal training, including technical leads of other sub-projects to explain their architecture and application programming interface (API) to other groups, testers to explain patterns of defects they found, different techniques for peer review, or discussion of a particularly interesting article in one of the technical magazines someone had read.

11. Fire People Who Cannot Perform the Work

Even when you meet regularly with your staff and encourage them to acquire help when they need it, some people in your group may not be able to perform at the level you require. First, make sure you have been specific and have given feedback to the employee with examples of inadequate behavior. If the employee understands the lack of performance, you can choose whether to coach the person or perform a get-well plan, or in radical circumstances, escort the employee out the door.

Retaining non-productive employees has direct and indirect costs. The direct costs are easier to define: You are paying a salary and benefits and not receiving the expected work. The indirect costs are much subtler and more damaging.

When you continue employing an inadequate employee, the morale of the entire workgroup declines. If morale declines enough, your best people will leave. Not only do you have someone in your group who is not successful, that person has driven away the people who are the most successful.

In addition to low morale, you and your group accomplish less than you expected. You are not just accomplishing less because of the one employee who cannot work at the level you require; that person probably has to hand off work to others in the group, and those other people will be delayed by the inadequate work.

I once inherited a group where the previous management had *spared* an employee from layoffs because he was

having personal problems. Those personal problems affected his work – he did not always come to work, he was late on every deliverable, and he was unable to perform most of his work. In my one-on-ones with the employee, I gave him examples of his work and asked if he was able to continue to work. He said yes. (If he had said no, we would have put him on short-term or long-term disability.) We chose to perform a get-well plan, which the employee stopped after a week. After the employee left, the morale in the group jumped dramatically and we were able to accomplish more work.

12. Emphasize Results, Not Time

I have worked for senior managers who rewarded individuals based on their work hours, i.e., those who started early and stayed late. Unfortunately, these managers had no ability to understand the results the long-working employees imposed on the rest of the organization: buggy code, inadequate designs, and tests that did not find obvious problems. When people work long hours, their productivity decreases, not increases [8]. In “Slack” [3], Tom DeMarco says, “Extended overtime is a productivity-reduction technique.” The longer people stay at work, the less work they do. Instead, they perform the life activities they are not performing outside of work.

Make it possible for people to only work 40 hours a week. The less overtime people put in, the better their work will be.

If people tell you they are working long hours because they cannot accomplish anything in their regular work weeks, ask them where they spend their time. Look for patterns such as multi-tasking, or meetings that do not have any productive output. Use your management power to discover and remove the obstacles preventing people from working a 40-hour week.

13. Admit Your Mistakes

Sometimes, those obstacles to people completing their work successfully in 40 hours arise from your management mistakes. It is difficult, and sometimes embarrassing to have to admit you have made a mistake. In my experience, when I admitted mistakes to my staff, they have respected me more for it.

14. Recognize and Reward Good Work

Money is not an adequate reward for many technical people. If people think

they are paid fairly, then more money is not reward enough. Recognition of good work and the opportunity to perform meaningful work [9] is much more important. Lack of money can be a demotivator, but only money is not sufficient when recognizing good work.

Kohn says, “[Rewards] motivate people to get rewards.” If your organization has trained employees to expect money as a reward, this appreciation technique may seem small. Try it anyway.

When I use appreciation as a recognition technique I say, “I appreciate you, Jim, for your work on the blatz module and API definition. Your work made it possible for Joe to write great tests and for me to predict the project’s progress.” Appreciation between peers could mean even more than money from you. When you appreciate a person for good work and you explain what the work meant to you, you are motivating the person to continue performing similar work.

In addition, consider time off, group activities, movie tickets, or funny awards such as *best recursion of the week* as recognition techniques.

The most important part of a reward is to make sure it is congruent with each person’s performance. Your staff knows who is performing well and who is coasting. If you recognize and reward evenly, you are not differentiating between outstanding performance and adequate performance. Make sure you reward a person’s entire contribution (the entire work product, including how good the work product is, the timeliness of the deliverable, the person’s ability to work with others, and whatever else is important to you), not just the size or quality of the work.

Summary

Managers exist to help people do their best work to serve the business of the organization. Technical people can make great managers as long as they understand people and want to succeed at working with them. Many successful technical managers took the time to learn about management, putting as much effort (if not more) than the effort they took to learn the necessary technical background for the technical jobs. Managers do not have to be perfect; they have to be good enough to create a working environment for their employees to deliver great work. ♦

Acknowledgements

I thank Dwayne Phillips and the CROSSTALK reviewers for their input on this article.

References

1. Magretta, Joan. What Management Is: How It Works and Why It Is Everyone’s Business. New York: The Free Press, 2002.
2. Covey, Stephen R. The Seven Habits of Highly Effective People. New York: Simon & Schuster, 1989.
3. DeMarco, Tom. Slack. New York: Broadway Books, 2001.
4. Weinberg, Gerald M. “Congruent Interviewing by Audition.” Amplifying Your Effectiveness: Collected Essays. New York: Dorset House, 2000.
5. Rothman, Johanna. Hiring Technical People. New York: Dorset House, 2003.
6. Janz, Tom, et al. Behavior Description Interviewing. Englewood Cliffs, NJ: Prentice Hall, 1986.
7. Buckingham, Marcus, and Curt Coffman. First, Break All the Rules: What the World’s Greatest Managers Do Differently. New York: Simon & Schuster, 1999.
8. DeMarco, Tom, and Tim Lister. Peopleware: Productive Projects and Teams. 2nd ed. New York: Dorset House, 1999.
9. Kohn, Alfie. Punished by Rewards. New York: Houghton-Mifflin, 1993.

About the Author



Johanna Rothman consults on managing high technology product development, which helps managers, teams, and organizations become more effective. Rothman uses pragmatic techniques for managing people, projects, and risks to create successful teams and projects. A frequent speaker and author on managing high technology product development, she has written numerous articles and is now a columnist for Software Development, Computerworld.com, and StickyMinds.com. Rothman served as the program chair for the Software Management conference and is the author of “Hiring Technical People.”

Rothman Consulting Group, Inc.
 38 Bonad Road
 Arlington, MA 02476
 Phone: (781) 641-4046
 Fax: (781) 641-2764
 E-mail: jr@jrothman.com