

;login:

THE MAGAZINE OF USENIX & SAGE

June 2001 • Volume 26 • Number 3

inside:

THE WORKPLACE

Easy Management: Giving Good Report,
or I Keep Doing Work, Why Do They
Keep Yelling at Me?

by Richard Threadgill

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

easy management

Giving Good Report, or I Keep Doing Work, Why Do They Keep Yelling at Me?

Lots of techies give really lousy progress reports and are really hard on their managers for no good reason. This is particularly a problem for sysadmin types, systems programmers, and other people who love math too much. I spent several hours with a coworker last week discussing “how not to be an employee of doom,” and these are my notes from that conversation.

First, an aside: these notes offer advice both for techies in general, who often have a poor model for the pressures on, and motivations of, their management, and for systems and math people in particular. Math people have two classes of problems: trivial problems, which merely require identifying an existing solution; and unsolved problems, which require thinking, hypothesis, and, potentially, experimentation. This often leads them to front-load their work, going through the list of their tasks and performing the “hard” tasks first because the others are “just work.”

Systems people have a strong tendency to suffer from “searchlight focus” as well, because it’s a really useful trait in a high-interrupt environment where you need to context-switch pretty completely. Unfortunately, it leads to some work habits which make your behavior (our behavior) really unpredictable. Management can tell how often they’re getting complained to about things you haven’t gotten done, and how often you’re reporting finishing tasks which they cared a lot about personally, but that’s about all they know.

This unpredictability makes writing job-requirement justifications basically impossible. And that’s bad because it means you get fewer raises and spend all of your time being overworked. It also means that development managers basically can’t deal with you in any constructive way, because your behavior is inexplicable and unpredictable.

Finally, this article is most intended for people who have a soft handle on how much time they spend on tasks, because they think about tasks from the perspective of difficulty rather than from the perspective of expected time-to-accomplish.

So, some rules for being easier to manage.

First and foremost, NEVER go radio silent. This is your manager’s worst nightmare: they don’t know what you’re doing, they can’t defend spending their resources on it, and they don’t know when you’ll finish. So, if you are about to embark upon a task which might cause you to go quiet for a while, discuss it with your manager first. Be prepared for them to direct you to attack a different problem first, so that they (and you) can build some capital to defend you while you’re silent. Think of this as giving your manager a good answer to the question “What has that employee done for you lately?” when they get asked by their peers and their management. This makes their life easier.

Give status early and often. This makes your manager’s life easier. Most of the rest of this document will talk about how you can order your work and reporting to make your work more predictable and thereby more visibly valuable.

Attempt to show consistent levels of output. This creates a perception of predictability and changes the conversation your boss has with management from “Has Dave gone silent again? Do we know what he’s working on this time?” to “How’s the really huge

by Richard Threadgill

Richard Threadgill is the cofounder of Ponte Communications. He is also a decade-long sysadmin and project manager.

<richardt@ponte.com>

When beginning a project, make a list of tasks. Then make a list of questions which must be answered to perform those tasks, including who needs to answer those questions.

project we asked Dave to deal with coming? Are we interrupting him with too many other little tasks?”

Learn to report when you're overloaded. It's much, much better for your manager to know that you are not going to be able to accomplish something at request time than to discover it at the expected time-of-completion.

Order your tasks so that you generate usable, partial, visible results often, thus enabling other people to get leverage from your work quickly, and making your manager's life easier. This hurtles headlong into the typical math-geek work ordering model, which tends to start with “Do the hard bits, because we don't know if those are possible, and that's the most important thing to learn before we get into this too deeply.” Unfortunately, this behavior gets interpreted by a lot of managers as “Dave just wants to do the fun bits and never the actual work.” So while it will make your teeth itch, gang, when you do your task breakdown, plan to do a bunch of the simple ones in parallel with the hard, thinking-about bits. I know this will sometimes mean you run down a rat hole, building trivial bits of an intractable task. But you'll be showing progress while you lose, which is vastly better than not-showing progress while you lose more quickly. Your manager will almost never get points for you finding out that a solution is intractable faster than you might have. Check – if that's actually your job, much of this document is not for you.

When beginning a project, make a list of tasks. Then make a list of questions which must be answered to perform those tasks, including who needs to answer those questions. Note which ones you have already answered. I know it sounds crazy, but work with me on this one. Now, in another document, note what those answers are that you already have. Do not spend time trying to determine new answers at this stage – either you have already got the answer or it should be listed as a “collect somehow” question. Send a copy to your manager, this makes their life easier, and forward selected portions of the list to each answerer. This gets answering your questions into their task queue. Each one of those “collect answer” questions should be treated as a task. Now begin performing tasks.

Make daily logs. Most of you get parts of many more tasks done every day than anyone actually realizes (including you). Don't expect to remember what you've been doing; write down completed subtasks as you work on things so that you can forward it at regular intervals. It's easier for your manager to throw away data (if you've organized it well for them) than it is for them to extract it from you if you can't remember things. BTW, this is one of the skills which makes admins really love a manager – the near-psychoic ability to figure out what their staff are actually working on, even though their staff aren't very communicative. As former admins themselves, they might be good at interpreting those reticent grunts you give out when you're sloggng through a lame name-service problem for the fourth day in a row, but you're still making progress and so aren't at the “just firebomb the vendor and get it over with” stage. But being able to perceive that requires a lot of domain experience on the part of your manager. And that much insight is really expensive to maintain, so try not to bet your career on your manager always being able to bring it to bear on your behalf.

If you are hit with inspiration, work on that task until you run out of steam. Take good notes while you're doing so. Then complete a trivial task.

Do not work on more than one complex task per day, unless you (1) have finished a complex task or (2) are inspired. Don't let a unit-of-time go by without finishing at least one task.

Try to make your list of tasks contain tasks of comparable amounts of temporal effort. Perform those tasks by strictly alternating trivial tasks and complex tasks within a unit-of-time (day/week/whatever).

Once per mega-unit-of-time, ask people who you need information from (see the task-listing task, above) to answer the questions you need them to answer. Getting information from someone is itself a (not always trivial) task. Do not attempt to complete getting information from more than one person per day; keep trying to get info from different people until *someone* gives you at least one answer, but stop when you've succeeded with one of them. If other people send you answers, that's gravy, but you don't want to go radio silent because you're spending days on end appearing to block while you're trying to extract information from other people. If someone tells you to go find the information in a named location, that should be construed as an answer for the purposes of this discussion, although it creates a "collect information from a known document" task. "I don't know" is not an answer, but changes your list of people to ask. If you get an answer or an "I don't know," write down the answer in your answers list.

Finally, let me reiterate the cardinal rule: Silence is bad. Management cannot differentiate between someone who's in over their head, someone who is malingering, someone who's trying to solve an intractable problem, and someone who is making progress on a hard design issue. You'll note that many of those options are bad. If you don't tell your manager what you're doing in a way that management can easily communicate to their peers, you're creating a lot of new work for your manager in two ways: first, by creating a need for them to defend you to their peers, and second, by making it actually difficult for them to do so. Good managers will review and evaluate their own focus and resource allocation continually. Making it easy for them to do so is good for both of you.

Silence is bad.