Chapter 6.

THE PROGRAMMING PHASE

Part II

Summary

5 The Manager's Job

- The manager's job is the promotion of excellence.
    - Simply getting a task done, meeting deadlines, living within budgets, rewarding workers fairly, pleasing the customer, maintaining personal and company integrity, etc. — these are essential, but not enough.
- The manager should always:
    - Be looking for ways to provide an excellent product.
    - At the same time ensure the personal satisfaction and career growth of his people.
    - Those things go hand-in-hand;
        - An excellent product will enhance the careers of its makers.
        - Growing and satisfied people will produce a better product.
- There are a great many differing and sometimes conflicting views on how to go about managing.
    - **Townsend** describes the manager as the one who "*carries water for his people so they can get on with the job.*"
    - There are excellent managers who agree with this view.
    - There are also some managers who hold the opposite, a sort of Ptolemaic view, "*The universe exists to serve them!*"
- "Carrying the water" is important, but there's more:
    - To be really effective, a **manager** must be respected, and to be respected he must lead.
    - It means knowing enough and knowing how to find and use competent technical advice to be able to set technical direction for the organization.
    - It's a tough job, but necessary to the pursuit of excellence.
- The following sections describe the functions of a **manager** in a **programming organization**.
    - In many ways these functions are the same for a **first-level manager** as they are for an **upper-level manager.**
    - Significant differences between the first-level and the upper-level management job are summarized in the last section.

5.1 Technical Leadership

- The manager need not be the best technical person in the organization in order to establish its technical direction.
    - What he does need, besides the obvious attributes, is:
        - (1) An intense desire for his people to be in tune with the business' technology.
        - (2) And for the managers to understand the latest management methods.

- In the programming business there has been an increasing tendency to bind more closely together management and programming techniques.
- Programming management has perhaps become somewhat more specialized.
    - One theory of management says that a good manager can handle either a meat market or a computer project equally well, and no doubt that's true for a gifted handful of individuals.
    - For many others, though, in ours days this doesn't work. That's our opinion too.
- In fact, **managing software projects** it's a difficult problem because the project manager needs both **technical tools** and **management tools**. In every case:
    - They assist technical people in the execution of their jobs, from analysis through designing, coding, and testing their products.
    - But at the same time these techniques are helping technical leaders and managers to control the development process.
- What is really **important**:
    - (1) The traditional manager's manual is not enough; get to know thoroughly these technical/managerial tools and use them to communicate with your technical people.
    - (2) Given the very important focus and control point offered by a Development Support Library (or something similar), there is much more opportunity for anyone on the project to understand system status.
    - (3) Make sure that you always allow time and money in your budget — insist on it — for all project members to continue their career education.
    - (4) Encourage your team to take the time to learn about their business without having to feel guilty because they're not "producing."
- Where technical leadership is concerned, usually appears the tendency to feel no longer competent to dig into technical matters once you become a manager.
    - But the tools are there for you to understand and contribute if:
        - (1) You will learn the tools.
        - (2) You make sure that your technical people understand that you want to understand and you expect them to communicate with you in language you can understand.

5.2 Planning and Controlling

- Planning and controlling activities are at the core of every management job.
    - **Planning** means laying out what you want to happen.
    - **Controlling** means making sure it does happen.
- Planning and controlling are what this entire course is about.

5.3 Communicating

- The problem most people have in communicating is not how well they **speak**, but how well they **listen**.
- Some people are poor at oral communication but compensate for it by writing things down.
    - After a meeting, for example, this type of person jots down what he thinks was said or what decisions were reached at the meeting. That's a good technique.
    - There are meetings every day in any organization from which people come away with completely different impressions of what was said or decided.
- A manager ought to communicate his own **management philosophy** to the people working with him. It's a difficult assignment, but people must know.
- It's not necessary that you (as PM) call a meeting to make a speech entitled "My Management Philosophy," but you can call frequent project meetings and take advantage of any reasonable opening to talk about how you see things.
    - Practically anything is fair game. That means discussions about:
        - (1) How you expect to organize.
        - (2) What kinds of responsibility you expect people to assume.
        - (3) How much latitude an individual has.
        - (4) What you think about taking time to read periodicals and newspapers at work.
        - (5) Your views on tardiness.
        - (6) Your view on listening music, or on WEB navigation.
        - (7) What political problems you foresee for the project, and so on.
- If you can casually discuss these topics, you'll let people know the general direction of your thinking, and you'll have a group of people who better understand the job.
- But don't forget to listen.
    - Don't become so spellbound by your own words that you don't solicit and listen to opposing viewpoints from your people.
- If the people at your project meetings eventually speak up and question your remarks and your way of handling things, you've succeeded.
    - You've replaced monologue with dialog.
    - Listen and follow up with brisk action.
- A **final** point on communicating:
    - Establish basic definitions for your project and insist that they be used consistently.

5.4 Ensuring Work Conditions and Tools

- Whatever the product you are building, it's your people who put it together.
- (1) Part Project Manager job is to provide them with the **environment** and the **tools** they need to perform.

- You as PM must maximize their chances of success.

- (2) One thing you can do is set up the **best physical facilities** you can afford.
  - Maybe programmers don't need carpeted offices, but they need quiet and privacy.
  - The programming process can tolerate neither a noisy factory environment nor constant interruptions.
    - If we could perhaps hang a dollar sign on each distraction or disruption a programmer experiences in the course of a day, we would quickly invest in some remedies.
    - Consider the effect of interrupting a programmer in the middle of a complex piece of code.
      - Later he not only has to backtrack to pick up the thread of what he was doing, but he may easily forget some part of what he originally had in mind.
      - Result: a bug.
      - The bug leads to a loss of his own time during module test; it consumes extra computer time; it may show up during higher-level tests when it will cause a disproportionate loss of people time and computer time.
    - The other problem is the well known desire of the programmers to listen music while they are working.
      - Usually the problem can be solved by providing the working stations with **personal listening equipments**.
      - This attitude can solve a lot of internal problems.
- (3) Another area in which you can help the entire programming process is in providing the **best possible developing and test environment.**
- (4) Speaking of communication, try your best to establish an **environment** in which **people speak out** and **tell you when things are not right.**
  - PM should create a specific framework for this activity.
    - One way to do this is to be on the alert for the first comment that sounds like a valid complaint or criticism.
    - Pounce on it.
    - Fix whatever was being criticized.
    - Make sure that everyone knows you acted positively because of someone's criticism.
  - PM must react to complaints and criticism in a positive manner.
    - Conversely, if you turn off the first criticism, you'll never hear another, constructive or otherwise.
  - **Metzger's** experience: *"Some time ago I attended a project meeting in which the boss discussed status, levied a few new ground rules, asked for questions, and adjourned the meeting when there were no questions.*

Immediately afterward, a small knot of programmers gathered near my office and hanged management in absentia. I listened in and then asked the most vocal member of the group why he hadn't spoken up in the meeting. He answered that management never listens, anyway, so why bother. How many times this happens every day is anybody's guess, but there's only one person who can prevent it: you, the manager. You set the tone."

- (5) The manager should also act as a **buffer** by taking steps to **get needed information** to his people and to **screen the trivia**.
- (6) PM must act as a **filter** and **distributor** of information for his people.
    - Any large organization is plagued by too much paper going to too many of the wrong people.
    - This often happens because a manager bucks everything on to his other people with a "read and pass on" note.
    - Either PM is afraid to withhold information that staff members may need or he is afraid to miss something that one of them may pick up.
    - No matter how safe it may make you feel, you simply cannot have everyone read everything.
- (7) Be sure that in your project planning specific thought is given to search out **the most appropriate tools** for both management and programming.
- (8) It takes enlightened management to **encourage** people to read the literature, snoop around, and do the digging required to come up with new tools and new ways of doing a job.
- Somme **practical pieces of advice**:
    - (1) Have direct contact with your people rather than send them written papers or mails.
    - (2) Manifest tact and diplomacy in errors emphasizing.
    - (3) Pay attention to the sense of measure.
        - Don't overload capable persons (programmers, managers, staff).
    - (4) Establish a general silent period in your organization (9 a.m to1 p.m for example), in which interruptions are prohibited.
    - (5) Establish a weekly/monthly general fixed time template for different meetings and discussions.
        - The ordinary meetings should be planed based on this frame.
        - Of course there are exceptions.

5.5 Assigning the Work

- Assigning the work is one of the more important activities, probably at the top of the list.
    - **Metzger's experience**: *"I once witnessed a large proposal effort whose objective was to win a huge programming job under contract to the federal government. The proposal itself involved more than fifty people. No one was*

appointed proposal manager. People were loosely assigned to jobs. Some areas of work were covered by three or four people each of whom thought that he or she alone should be doing that job. Other areas were not covered at all. The manager who had the power to correct all this had a reputation for not making specific assignments. He waited, often in vain, for a hero to step forth and volunteer. Things just don't work that way. The person in a position to direct must direct."

- There are two manners for assigning the work:
  - (1) Persons assignment.
  - (2) Domains assignment.

5.5.1 Persons Assignment

- The solution is simple.
- First, **assign a specific person for any job.**
  - The idea that a busy executive can be acting manager of a major project in his spare time is ridiculous.
  - Better to appoint a slightly less qualified person to the job full time than to assign the job to an acting manager who can't devote enough time to do the job justice.
- Now suppose that **you** have been assigned as manager of a project or proposal effort.
  - (1) You must first gain an understanding of the job.
  - (2) Then make an attempt at breaking up the job and assigning pieces of it to individuals. But this is not enough.
  - (3) You must write down a description of each person's job. No exception! **Write it down!**
  - (4) Then give a copy of all assignments to everyone.
    - The first thing that will happen is that half your crew will come storming in to complain that someone else's job assignment bites into their territory.
    - If you're lucky, someone will come in and point out that nobody's assignment covers area X.
  - (5) After you've had a couple days of complaints, and people have chewed on the assignments enough, set up a meeting to talk over the problems that have been brought you.
  - (6) Then rewrite the assignments, pass them out again, and wait the second round of blasts.
  - (7) It may take a couple of repetitions, some of the meetings may be uncomfortable, but soon you'll have job descriptions that don't overlap and do cover what needs to be done.

5.5.2 Domains Assignment

- An **alternate approach** is:
  - (1) To assign work by topic only, and let each one write his own work description.
  - (2) Then you alter them in whatever way you see fit, and pass them out.

5.5.3 Work Assignment Objectives

- It doesn't matter which approach you take, as long as the following **objectives** are touched:
  - (1) All tasks are fully covered.
  - (2) Everyone knows exactly what his job is.
  - (3) Everyone knows exactly what others' jobs are.

5.6 Working Hours

- There are different techniques for allocate and control the working hours.
- The techniques presented above can be combined in dependence of:
  - The specific of the company.
  - The specific of the employees.
  - The specific of the project.

5.6.1 Normal Working Hours Allocation

- Working Hours are allocated to tasks, based on:
  - (1) A priori estimation of the tasks size.
  - (2) A pre-established schedule.
- Every programmer, manager or other implied people must have an **evidence** of the personal working time.
- First and second level managers must have the evidence of the working time for theirs teams as basis for control.
- Project Manager must to have an image of the total working time spent on project development as basis for control.

5.6.2 Supplementary Working Hours

- If you're relatively new to the computer programming business, perhaps you have not yet taken part in panic projects where everything was late and management had to resort to that ultimate remedy: scheduled overtime.
- It's hard to demonstrate that there is a much worse **waste of resources** than this.
  - There is a natural temptation to think that by working a given set of people 25% more hours a week, 25% more work will get done;

        - Some managers even think that crash overtime efforts help to bring the troops together and that develops a strong feeling of camaraderie.
    - But **experience** proves otherwise:
        - First, 25% more hours can easily produce 10% less work, or at least less usable work.
        - Given more hours to work each day for an extended period of weeks or months, most people will simply settle into a new routine, pacing themselves more slowly.
            - Their work may get sloppier.
        - The first week, of course, some extra work may indeed get done, and perhaps even the second and the third, but after that it may be a losing effort.
            - People unconsciously slow their efforts to fill the scheduled time.
        - Even extra work accomplished during that first week or two will not be nearly in proportion to its cost — either the dollar cost or the cost in morale as private and family lives are inevitably intruded upon.
        - And as for that feeling of camaraderie, it soon evolves into negativism: damn the customer for insisting on that delivery deadline; damn management for agreeing to it; damn the computer time; damn everything!
- In fact we're discussing scheduled overtime. It simply does not work well.
- The only real chance for overtime to work well is when it's done **voluntarily** and over **short time spans**.

5.6.3 Flexy-Time Technique

- There is something else to consider about working hours.
- Various companies have experimented with the idea of letting employees set their own hours.
    - (1) Generally, the total time to be worked each day is set, but starting and ending times are allowed to float.
    - (2) This can be further liberalized by setting a number of hours for, say, an entire week, and leaving the specific days or hours up to the individual.
- This is **Flexy-Time Technique**.
- Perhaps those latter notions are too liberal for now, but the idea of the floating workday is not.
    - It's been tried and it's been shown it can work.
    - It has an obvious **advantage:**
        - Catering to the individual needs and inclinations and body clocks of each employee, which should lead to improved satisfaction with one's job.
    - There are also obvious **disadvantages**:

        - For example, how can you call a department meeting when you never know who will be around at any given hour?
        - And how about the frustrations of programmer A which needs to talk to programmer B, but A's chosen hours are early in the day and B's are late?
- An obvious way to experiment with the set-your-own-hours idea is to start small, see how well it works, and adjust as you go along.
    - (1) Begin by limiting the range over which the individuals' hours might float.
    - (2) Set aside a part of each day, say 1:00 P.M. to 4:00 P.M., when **everyone** is expected to be on the job. Correlate this period with general fixed time template.
        - That will allow some to work from 7:00 A.M. to 4:00 P.M., others from 1:00 P.M. to 9:00 P.M., still others from 10:00 A.M. to 7:00 P.M.
- Sound like a management headache? You must try at least!

5.6.4 Dead-Line Technique

- Carried a step further, hours might be eliminated as a measure of work or worth.
- The means of measurement or control might be that the employees complete a given task by some predetermined date.
    - This is **dead-line technique**.

5.7 Adding More People

- Just as resorting to overtime is generally wasteful, neither is it helpful as a rule to load the project with more people to try to bail out of problems which are the result of poor planning in the first place.
- Beware of holding to impossible deadlines in the mistaken belief that what you lack in calendar time you can make up with bodies. It just does not work.
    - **Brooks** claims to be oversimplifying outrageously when he postulates **Brooks' Law**: *"Adding manpower to a late software project makes it later. The bearing of a child,"* says Brooks, *"takes nine months, no matter how many women are assigned."*
- There are two reasonable alternatives to adding more people when the project falls behind schedule:
    - (1) **Reschedule**. This, of course, will make the customer scream, and that will make your management scream.
    - (2) Arrange to **deliver interim, incomplete versions of your system**.
        - If the first delivery can be made at the time you were supposed to deliver the final product, that timing should help; only the final version, then, need be rescheduled.

- If you become involved in rescheduling and delivering interim versions of the system, be as sure as you can that this time you can make the deliveries as promised.
    - There will be a terribly compelling urge to deliver as soon as possible, and it's probable that you'll offer new dates that are again too optimistic.
    - Best to make your stand now.
        - Admit to any bad planning or execution you've been responsible for so far.
        - **But** make sure you don't repeat those mistakes.
- Screaming customers are a nightmare, as are screaming bosses.
    - Don't let their screams wear you down when you feel you're right; otherwise, you'll hear them scream again in a few months.

5.8 Reporting Technical Status

- Anyone who is doing a job for anyone else must somehow communicate how things are going.
- Usually, there is a need for both:
    - (1) **Written reports**.
    - (2) **Oral reviews**.
- Let's consider each in turn.

5.8.1 Written Reports

- **Reporting the status** of anything presumes that there is some **baseline plan** to **report** against.
- **General considerations**:
    - (1) The first requirement for **effective status reporting** is that there must be a plan and milestones against which to measure progress.
    - (2) A second consideration is that all reports should be tailored to fit the management level for which they are intended.
        - The reports should be tailored both in terms of content and frequency.
- A **reasonable scheme** for a project might work like this:
    - (1) **Individuals**, for example, programmers, report biweekly to their first-level managers on the status of all tasks, or work packages, to which they have been assigned.
    - (2) The **first-level manager**, after receiving inputs from his people, reports to the second-level manager the status of selected tasks;
        - That is, he reports on those tasks shown as milestones on his work assignment bar chart.
        - A first-level manager responsible for, say, fifteen tasks, might report to a second-level manager only four or five.

- (3) The **second-level manager** reports to Project Manager the status of milestone tasks on his charts.
  - Effectively, what **PM** receives is **not** a net summation of all the individual pieces of work on the project.
  - Rather than pass on all possible status data, each manager acts as a filter and lets through only that which is important.
- (4) In turn, **PM** pass on a **condensed report** to his management and to the customer (see Fig. 6.5.8.1.a.).

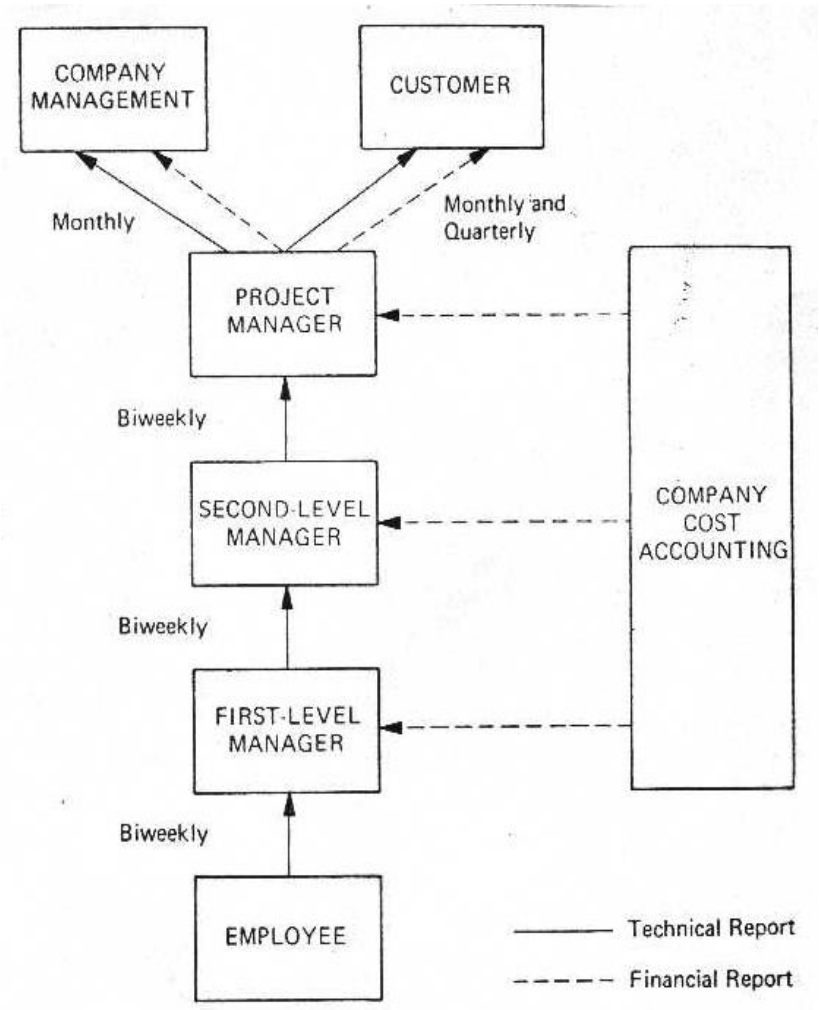

Fig. 6.5.8.1.a. Status Reporting

- All the reports in this chain should follow similar, compatible formats.
- That way, if PM occasionally calls for more data from a manager, he can give him the more detailed reports gotten from his subordinates, and the detailed reports will make sense to him.
- **Guidelines to write a report**:

- (1) In the section of a report in which you discuss problems, state the problems, but make them appropriate to the level of the reader.
- (2) State your problems according to their priorities.
  - That is, make sure the reader of the report knows which problems you consider most important.
- (3) Always try to indicate potential solutions and the status of attempts at finding solutions.
  - Give the reader the warm feeling that you're working on the problem (if you are).
- (4) A basic principle of our technology is laying out all the work on the project in discrete chunks.
  - Report on those discrete chunks in your status reports.
- (5) Don't let anyone get by with qualitative mealy-mouthing or meaningless "percent complete" reports.
- (6) Insist that your people report to you in quantitative terms.
  - Metzger's **experience**. *I recall one project many years ago in which we reported partly by filling in a bar chart. When a bar became entirely black, that job was theoretically done. Unfortunately, we ran out of bars before the job was finished. We found that filling in the bars told us nothing. They made depressing wall decorations in a status-control room.*

5.8.2 Oral Reviews

- Oral Reviews were discussed in the previous chapter.
  - Here, however, we'd like to emphasize the usefulness of **oral reviews** as part of the PM activity.
- **Project reviews** have many advantages**:**
  - (1) They help give people a sense of belonging to something besides their two-by-four cubicles.
  - (2) They help people to understand the part their work plays in the total job.
  - (3) They provide a break from routine.
  - (4) They provide a forum for uncovering problems and broaching better solutions.
- You can help your project immensely if you set up these reviews in order to give people an opportunity to be heard.
  - You might even come to be looked upon as a human being rather than that recluse who sits in the corner office with the door closed.

5.9 Reporting Financial Status

- **Financial reports** are often generated by some cost-accounting function separate from the management of the project.

    - These reports may be sent only to the project manager, or to each of his subordinate managers.
- Whatever the case, PM should insure during planning that the tasks, or work packages, to be used as the basis for financial reporting are the same tasks used for technical reporting.
- PM should always be able to equate technical and financial reports without need for a massive conversion exercise to find out how the two reports relate.

5.10 Training

- Every manager has responsibility for training his people.
    - This is **totally** apart from any training that must be done under the contract.
- The manager must seek to raise the level of competence and understanding of every individual in the organization.
    - Otherwise, the organization stagnates.
- When **times are tough** and an organization's overhead costs must be trimmed, education is one of the first expenses cut. It's considered a luxury.
    - Classes are cancelled because they add to overhead, and yet the same people who would have attended those classes now sit around, still charging their time to some form of overhead, doing little or nothing.
    - Get those idle people into education programs.
        - Of course, if you work for a company that gets rid of people as soon as they become idle, the argument is academic.
    - There are several kinds of training a manager ought to provide and there are a number of ways of providing it.

5.10.1 Technical Staff Training

- Let's start with your technical people, for example, the **programmers**.
- The **programmers** should be trained in, or at least exposed to, programming languages and computers other than those of concern on their current projects.
    - How else will they be able to grow?
    - How can they offer you alternative technical solutions if all they know is machine x and language y?
    - These people should be given the time and the encouragement to attend formal classes, subscribe to and read technical literature, and hobnob with their counterparts on other projects.
        - The latter is often hard for a manager to swallow. It's like encouraging coffee breaks. But what a payoff! What so often results from a bull session is either a new technical idea or a pitfall to avoid.
- Your **newer programmers** need special attention until you have a feel for the competence of each.

    - Managers, especially green managers, often tend to accept their new recruits as experienced, professional people without demanding any proof.
        - It's not always true! You may have surprises!
    - How do you solve this problem and set them straight?
        - By making code walk-throughs and code reading routine procedures and peer programming techniques for every programmer on the project.
        - The work of **Mills** and others has shown this approach to be extremely valuable, both as a quality control technique and as a training mechanism.
- An important **kind of training** for all your people (and you) is **rotation** through jobs other than their normal ones.
    - This can often be done without an enormous investment in time, and the payoff is handsome.
        - This can happen inside the same project or in time during different projects.
    - Rotation, or cross-training, is helpful to your non-technical people, too.
        - How about a basic course in the fundamentals of system analysis, design, and programming for your secretaries and typists.
        - They would enjoy knowing what all those scribbles are about, and would do a better job of transcribing them.
    - There is no one on the project who would not benefit from at least some exposure to what goes on outside his own job and this will in turn benefit the project.
- A word of **caution**: Federal and some state labor laws severely restrict the types and duration of rotations you may legally use.
    - You may not, for example, freely interchange "professional" and "clerical" people.
    - Ask your company's lawyers what's allowable in your area.
    - Don't ask them if you can do it; it's easy to say no.
        - Tell them you're going to, and ask how to do it legally.

5.10.2 Managers' Training

- Then there are your managers.
    - (1) They need day-to-day assistance from you.
    - (2) They also need to attend management classes and seminars.
    - (3) They need technical updating classes to help stave off obsolescence.
    - (4) The managers working for you need to see in you a good example of how to manage.

- If you plan your activity poorly (and rely on excessive overtime to get your own work done), you're providing, by example, training in sloppy management.
- Perhaps most important of all, you must **bring together** the entire project often enough to update everyone on **status** and **plans**.
- It's impossible to overstate the benefits the project will receive from simply having everyone understand what is going on and where each fits in.

5.11 Appraising and Counseling

- (1) As manager you have an enormous amount of influence on the lives of your people.
- (2) How well you pay a person is important, of course, but just as important is your ability to help each person on your project to find a fulfilling job.
- (3) Many things about the project may be bad, but if you can just carve out a piece of work that appeals to an individual and is within his capabilities, other problems fade.
    - **Nietzsche** said, *"He who has a **why** to live for, can bear almost any **how**."*
- (4) You and your employees must agree on a suitable task and a schedule for doing it.
    - Again, the simple act of writing down a description of people's assignments and getting their agreement to it can be extremely helpful.
- (5) Often you can let people set their own schedules.
    - You'll be surprised how hard they will work to meet a deadline they themselves fixed.
- (6) When one of your people goes off course or does a bad job, you've got to let him know it the best way you can.
    - This can be so difficult that some managers avoid doing it until they are forced to.
    - But then, things are usually at crisis stage.
- (7) A way of avoiding big problems is to subdivide the job sufficiently and have enough checkpoints so that missing one is a signal, not a catastrophe.
- (8) There are managers who tells you what you've done **wrong** but not what you've done **right**.
    - **Metzger's experience**. *"I remember one manager who was good at pointing out my mistakes, but when I finally asked him one day whether there was anything I had done for him with what he was satisfied, he was astonished. Of course he was satisfied. Didn't I realize that anything he didn't criticize was, by definition, okay? This man, who was a fine manager in most respects, simply did not understand that people need a good word now and then. Some people, in the absence of any good news, tend to fear the worst."*
- (9) Any manager's success should be gauged by:
    - How well he **encourages growth**.

    - How fairly **hard work** is rewarded.
    - How **incompetence** is dealt with.
- (10) A good manager will not hesitate to promote a subordinate to the manager's own level.
- (11) A good manager will not selfishly hide a key employee within the organization;
    - Instead, the manager will promote the employee and risk losing him or her to some other group.
- (12) A good manager will never transfer a "problem child" to someone else without full warning.
- (13) It is essential the manner in which the manager treats and awards the **well done work** and the **competence**.

5.12 Sanity Maintenance

- Many managers get into tons of trouble because they could not say no.
    - The word is negative, after all, and no eager young manager wants to sound negative.
        - Anything the boss (or the customer) wants must be done.
    - All the books on "positive thinking" and "thinking big" say you can do anything in this direction.
        - For **example**, you can strike the word "no" from your vocabulary.
            - It's a mistake!
        - There are many things that are impossible and many more that are unreasonable.
- The real positive thinker is the person who can sort things out and distinguish between the reasonable and the unreasonable.
    - There have been countless disasters in the programming business because someone allowed himself to be pressured into committing his efforts to something he really felt was impossible.
        - (1) Sometimes it's of the gentle variety.
            - Your manager is in a tough position and needs a certain commitment from you in order to save himself.
        - (2) Sometimes the pressure is applied by erosion.
            - You're asked to give in a little at a time, in easy installments, and when you finally realize what you've done, it's too late.
        - (3) Other times you are fast-talked and lulled into playing RAM (Repeat After Me).
            - The manager tells you what he wants to hear and your only job is to say it.
        - (4) And in still other cases, strong-arm methods are applied;

            - You are made to feel your future promotions or your very job is in jeopardy.
    - There are many situations in which these tactics are applied.
        - Perhaps what is at issue is an estimate that you have submitted; it's too high for the boss to swallow.
        - Or maybe you're being asked to accept some added work you don't feel you can handle.
        - IBM's **Bill Weimer** has been responsible for many innovative training courses for both management and technical people. He said:
            - *"We found that technical people, in general, were actually very good at estimating project requirements and schedules. The problem they had was defending their decisions; they needed to learn how to hold their ground."*
        - **Charles P. Lecht** has this to say about refusing the impossible:
            - *Equally responsible for the initiation of a project with predefined failure . . . is management that insists upon having fixed commitments from programming personnel prior to the latter understanding what the commitments are for.*
- Too frequently management does not realize that in asking the staff for the "impossible," the staff will feel the obligation to respond out of respect, fear, or misguided loyalty.
    - Saying "no" to the boss frequently requires courage, political and psychological wisdom, and business maturity that come with much experience.
- There is another way to help maintain **balance** in the complicated business of programming management:
    - (1) Force yourself periodically (perhaps once a day) to take a few steps back and look at the total job, not the details. Simplify.
    - (2) Try to get things in perspective.
    - (3) Get away from the job physically — no phone, no in-basket.
    - (4) List all the things that you have to do and then decide which items on the list are really important.
    - (5) If there's more there than you can do, pass some of it on to someone else.
    - (6) Look for those items that you could cross off the list and never do (there are always some of these).
    - (7) If you take frequent enough looks, you'll begin to look at the job more rationally.
    - (8) You'll have a constant awareness of how much you have to do, and, therefore, how much more you can take on.
- In **assigning priorities** to the tasks you have to do, the "people problems" should come first.
    - Don't let your people feel that they come second to anything.

        - If you lose their loyalty and respect, you're dead.

5.13 Management Levels

- In the preceding discussion we haven't made much distinction between the various levels of management.
    - The job is essentially the same, regardless of level.
- What really varies is the **ratio** of technical to non-technical involvement.
    - This ratio decreases as you go up in the management chain.
        - (1) A **first-level manager** is normally very directly involved in the technical work his people are doing.
        - (2) A **second-level manager**'s technical involvement is broader.
            - This position requires more time than that of the first-level manager on financial matters, proposals, planning, personnel matters, and the like.
        - (3) And so it goes until at some level the **manager** is concerned much more with general business decisions than with detailed technical decisions.
- A difficult problem arises from all this:
    - How does the upper-level manager have any feel for what's going on technically?
    - How does the manager fight technical obsolescence and have any confidence that things are going well?
- There are some partial answers to these questions.
    - (1) First, a manager should devote a significant portion of time to technical updating by reading specifications and hearing briefings by his subordinates.
        - But he must resist the urge to bit-fiddle and leave detailed technical work to those best qualified to do it.
    - (2) Second, the manager should set up technical checks and balances to help assure that the technical work is getting done properly.
        - One such check is the separate analysis and design organization discussed earlier.
        - Another is the separate test group also described.
        - And still another is the extensive use of structured walkthroughs or inspections to provide for detailed scrutiny of all the items developed on the project.
    - (3) A third way to keep afloat technically is to return periodically to technical work.
        - In some organizations switching between management and technical jobs is discouraged, but if you can accomplish it, it can do wonders for your technical competence and greatly enhance your confidence in being able to manage the next job.

- (4) Finally, the manager must read the literature in his field and attend classes whenever possible.
  - Often this will have to be done on one's own time.

Exercises #10

1. What is *structured programming*? What are their *goals* and *advantages*?

2. Describe the main characteristics of the *object oriented programming* (OOP), *object oriented design* (OOD) and *object oriented analysis* (OOA)? How are they *related*?

3. What kind of *organization modalities* do you know? Describe their main *features*, *advantages, disadvantages* and *application area*.

4. What is *Conventional Organization*? Draw the *Conventional Organization flowchart*. Detail each component of the structure.

5. Which are the *tasks* of the *Analysis and Design Group* during Programming Phase? Describe them.

6. What are *Structured Walk-Through* and *Inspections*? Describe the *main steps* of this activity.

7. Which are the *tasks* of the *Programming Group* during the Programming Phase? Describe them.

8. What kind of *integration* do you know? Describe their *advantages, disadvantages* and *implementation techniques*. What is the content of the *Integration Test Specification*?

9. Which are the *tasks* of the *Test Group* and *Staff Group* during the Programming Phase? Describe them.

10. What is *Chief Programmer Team organization*? Draw the *Chief Programmer Team flowchart*. How does it work?.

11. What is *Change Control*? Which are the *Baseline Documents* affected by *Change Control*? What are the *Change Control Procedures*? How they work?

12. What *programming tools* do you know? Describe some of them emphasizing their *role* in the Programming Phase for example *Test Executives* or *Project Library*.

13. Which are the *Project Manager's tasks* in the Programming Phase?

14. What kind of *work assigning modalities* do you know? Describe their main *characteristics*. What are the *objectives of assignment of the work*?

15. What *techniques* for *allocation and control of working hours* do you know? Describe them underlining their *advantages* and *disadvantages*.

16. What are the main modalities of *reporting technical status of the project*? Detail them.

17. What do you know about the *training*? What *types of training* do you know? Describe them.

18. Explain the importance and describe the content of the following attribution of the Project Manager: *apprising and counseling, sanity maintenance*.

19. Which are the main *management levels* and which are their main characteristics and differences?