Successful Software Management 28-30 Nov 1978

A Report by Martin Jackson

## Introduction

This seminar was run by Infotech, and given by Ed Daly, who works for GTE (I think this is the General Telephone, Electrics). He has also worked for Bell Laboratories. The seminar did not stick rigidly to the topic of management, and indeed for the first 2 days hardly touched on management at all, and all the examples were based on telephone applications, and were thus not very relevant to the majority of the audience. However, on looking back over the seminar, it was quite useful, and a number of useful ideas have developed from it. These can be summarised under three broad headings, and I will cover each in turn. They are Software Design Methods; PERT; and Matrix Management.

## Software Design Methods

Improved programming techniques have been covered in past papers, but they were again mentioned at this seminar, and it cannot be emphasised too strongly that some kind of formal control is essential for improving programming. A list of the main parts to the technique will suffice.

- High level language
- Structured code
- Composite design (hierarchy of small segments)
- Parallel, top down, bottom up design – all optionally used
- Simple data structures and work areas (not tightly packed)
- Team approach to design and ego less programming
- IBM's structured walk thru's for reviewing detailed design and code
- Formal management controls for schedules memory size and resources

At GTE, the programmers work in a "Software Factory" and for projects are divided into teams. These typically consist of a Supervisor, a Chief Programmer, three senior programmers and three junior programmers.

The supervisor is the administrator, and may cover up to three projects. He is responsible for productivity, the quality of the work, he assigns work and trains personnel.

The chief programmer is the technical leader. He is the technical expert, performs research, designs the critical modules, controls interfaces, reviews designs and reads code. Supervisors and Chief

Programmers are at the same level, and can swap jobs.

One of the main points to come out of GTE's studies was that planning and design take 50% of a project's time, coding 15% and testing the remaining 35%. The most effective tool for trapping errors was the "eyeball test", or code reading. This is also by far the cheapest method and so this is an area where most of the time should be spent.

This is something we should do more of – far too often time is spent during testing on correcting errors which should have been trapped earlier. Programmers tend to assume their work is correct – particularly on small amendments, and would not dream of asking someone to waste their time checking it – because they cannot conceive of it being wrong. This is a very dangerous practice and I believe code reading should become a LOLA standard in all situations.

Notes

Retyped by Alan Cooper, May 2024 from the original faded handwritten sheets. These original sheets have been scanned to PDF. Long paragraphs have been split for readability and the list of techniques given bullet points.

This looks to be an uncompleted draft as there are no headings for PERT (Program Evaluation and Review Technique) and Matrix Management.