Some practical problems of software maintenance
J.W. Duncan

Department of Business Administration, Glasgow Caledonian University, Glasgow, UK

ABSTRACT

Many problems occur in dealing with the Maintenance of Software in most Organisations. These problems must be solved and managed in a practical fashion in order to satisfy the Information Technology requirements of Organisations which use a wide range of computers and data communication links. The problems are described and practical possible solutions proposed.

INTRODUCTION

The provision and supply of software within organisations is undertaken by two methods. Either by internal information technology staff writing the software or by purchasing an externally supplied package. The view will be taken in this paper that software can be considered as being externally supplied package(s) as it can be asserted that an internal information technology team is likely to be treated as independent cost centres within organisations, charging for their services in a competitive fashion. It may also be surmised that an external supplier of software produces a more complex management situation in terms of control; internal information technology staff can have greater pressures put on them to complete an internal project.

Following the installation and implementation of computer systems, a period of relative stability
occurs while the users and information technology staff supporting the computer system familiarise themselves with the facilities in place. This period is soon disturbed by software releases. Software releases are brought out by the supplier of the software for two reasons:

a. to correct reported or observed faults and deficiencies
b. to provide new facilities or features.

These features and/or corrections to faults and deficiencies are incorporated as a result of faults and/or enhancements being reported to the software supplier from its customers through a help desk, from user groups, or even by observations from the software supplier's programming and support staff.

The consequence for users of software is that their software maintenance staff will periodically receive a release update to implement, usually at a time when they have just achieved equilibrium from the last release. The decision then has to be made - should this release be implemented immediately, when it is not known whether it will work or not, or should the implementation be postponed until any shortcomings that may be present are discovered by other sites who implement it as soon as they receive it. Such other sites will directly report their findings to the software supplier or will do so through a support/help facility or User Group.

It is postulated by Mellor that Software cannot be said to be reliable and that its failure occurs very often due to commercial pressure by the suppliers to hit a 'window of opportunity' in the market. In a commercial environment, the buyer must be very careful.

A further problem may also exist that if an organisation decides to remain with a proven version of software (or with one that they will accept, however reluctantly, as being acceptable) then the software supplier may (will almost certainly) eventually withdraw support for that outdated version of the software. This is more likely to occur than the other situation where the supplier will charge
increasingly exorbitant maintenance charges to deter the user from continuing with the maintenance service.

Different problems arise in different areas of the organisation depending on whether Systems or Applications software is involved and also End User Computing poses special problems, all of which are covered in later sections of the paper.

THE USER VIEW OF SOFTWARE MAINTENANCE

Users of software may present an almost cynical view to software maintenance, tempered by past experience(s) of changes to software. These experiences may have been uncomfortable or even traumatic and will make them resistant to a repeat of that experience. Society in general is resistant to any change, but users of software now sometimes adapt the view "If it works, leave it alone". This attitude can lead to the situation where a system can be found not to work but appears to work to the user. The user is the customer and is always right! Software changes should not be forced on an unwilling user on the insistence of the software maintenance staff, but should be carried out because the user can see and, more importantly, agree that the maintenance is necessary or beneficial.

SYSTEM SOFTWARE

Many problems are created by the maintenance of operating systems and communications software.

Operating systems
Operating systems are awkward in their own right because they must be operative before all other software on computer systems can even start. They are so complex that many manufacturers offer maintenance services to customers to keep their operating system software up to date. This frees the organisation of the onerous responsibility of having an expensive, and sometimes irreplaceable (in the event of illness or termination of employment), systems programmer(s) to maintain their operating systems.
The maintenance of systems software is exacerbated in circumstances where computers in different departments may have different operating systems, or indeed this can apply to computers within one department. There may be valid reasons for this such as the need for a specific package which will only run on one hardware platform which is different to the standard hardware used by the organisation. This is acceptable as long as the two (or more) environments support communications software to allow file transfer and application enquiries between each other.

**Communications Software**

Communications Software - Organisations now rely on an assortment of networks which vary in complexity. This can range from a simple central network to a very complicated series of Local Area and Wide Area Networks linked together. Communications software is more specialised than other software, particularly where detailed mapped configurations of all equipment and processors require to be generated including addresses and specifications of all terminals, nodes, paths, ports, bridges, gateways, etc in the network require to be kept up to date to enable the efficient day to day operation. Support for communications software maintenance is also offered by manufacturers/suppliers.

**APPLICATIONS SOFTWARE**

**Legislative Changes**

Changes in Legislation are a driving force behind a large amount of software maintenance, particularly in the area of payroll (frequently referred to as "the never-ending story").

Legislation very often requires changes to applications software to be carried out in very short time-scales. Quick and efficient implementations to legislation changes is very important to most organisations. A January, 1994 survey indicating that 24% of businesses will have their 1994 Recruitment Plans affected by changes to Sick Pay Administration is typical of the effect that reasonably straightforward legislation is having on businesses. More
complex legislation such as the Poll Tax and, more recently, the Council Tax rely on computer systems completely for their operation. The business pressure of requiring to have the software available for a strict deadline may be the crucial factor that determines that a package be bought in rather than written internally. Problems may then arise if the supplier of the package is not as knowledgable about the legislation as was suggested at the time of purchase of the package. In some instances, suppliers have gone out of business before the failure of interpretation of legislation on their part is discovered. This is another problem the risk of which is minimised by a proper credit check on suppliers before purchasing their products.

Software Support
Applications software requires ongoing support from the supplier and sometimes from the internal Information Technology department. Some 'faults' discovered by users can be circumvented by those users possessing sufficient knowledge of the system finding an alternative method. This arises where user staff are well trained and enthusiastic about their system. This training is costly, is certainly required, but makes the staff involved almost indispensable and can create recruitment problems in attempting to find a suitably qualified replacement if staff leave.

Achieving a good relationship between users and the software suppliers can also have its own drawbacks. Users who grasp the software solution quickly request changes/enhancements after the installation of the software, sometimes even before. Users enthusiasm must always be encouraged, but this situation requires great tact and diplomacy to keep the user happy with his system whilst still seeming to be not co-operating with him. To give in and be in a situation of constantly implementing enhancements is counter-productive and is typified by Alice and the Red Chess Queen in "Alice in Wonderland" by Lewis Carroll. "You have to run very hard to stay in the same position!". Agreement with the users should be sought for error correction/enhancement implementation as regards priorities and needs. A
compromise between both extremes should be agreed and a timetable agreed (and held to).

END USER COMPUTING

The concept of End User Computing where users develop their own software, sometimes independently of the professional information technologists in the organisation, is causing problems in terms of efficient software maintenance. Many users install their own software on a departmental computer and require interfacing problems with other software to be resolved. In addition to interfacing problems, there may be a conflict with other existing software, network/communications software in particular. An imperfection in existing software will sometimes only be brought to light when co-existing lateral software is upgraded.

Another problem with End User Computing involves security. Users tend to view backups as something other people carry out or the process to be carried out just after a disk crash. The strict discipline of regular backups and rotating copies of backups to the fire safe of the Information Technology Department, preferably in a separate building, is very much to be encouraged.

Software Inventories and Piracy

With the proliferation of personal computers which can be purchased out of departmental budgets without recourse to corporate permission, it is difficult to maintain an accurate inventory of hardware within the organisation without even considering the myriad possibilities of user-purchased software. An environment is produced which is a real problem for organisations in terms of software piracy. Where users have access to computers and can load or modify software, they often see software being used by colleagues or friends and ask for a copy (or a loan of the release disk) to see what they can do with the package. Although often quite innocently ‘perpetrated’, this still amounts to software piracy and education is necessary to inform users of what is permissible. Some organisations set aside some
personal computers to allow staff to experiment with various legally purchased packages, be they business productive packages or even games. The concept of allowing games may seem anathema to a business organisation, but it has its benefits in that games software is isolated in one machine, thus decreasing the possibility of infection by computer viruses, and the facilities shown in games software can often give users ideas for more efficient use of available and possible computer facilities. Access is often restricted to staff break times to ensure that staff do not abuse this concession.

Information Technology Steering Group
A method of controlling the software (and data) proliferation that can result from End User Computing is to create an Information Technology Steering Group. Such a steering group contains representatives from every department or section, preferably at a reasonably senior management level. The purpose of the group is primarily to agree Information Technology Strategy, agree levels of spending and priorities on Information Technology from the corporate budget. This includes software purchases and development. Such a group would also meet regularly and discuss their own End User development so that duplication of effort and data within the organisation is minimised.

A good software development strategy within such a group is to identify a "Champion" group or department amongst the users which can be used as an example to others. The example of what software systems they have successfully implemented is demonstrated to others by the "Champions" and details of advantages and disadvantages of their efforts can be effectively communicated to the other departments. There is however the danger that these meetings can be counter-productive if effective control is not exercised by the chairman of the Group.

The involvement and participation of such users at external User Groups where meetings are regularly held to discuss problems, experiences and hear of particular disasters and/or failures, should be
encouraged. These User Groups can provide invaluable motivation and ideas. They can also, unfortunately, provide enthusiasm without the necessary underpinning of experience or of understanding the limitations that may be in the software/hardware combination being used. Participation by the Information Technology department is advisable at these User Groups, or at least a good feedback of the content of such meetings and their affect on the users.

MEASUREMENT OF QUALITY

There are problems in attempting to define, specify and/or measure quality in software. Manns and Coleman² highlight the problems involved in software quality definition as indicated by Garvin³ by including reliability, conformance with expectations, durability (length of use before the need for replacement), serviceability and even aesthetics. They indicate that the point where the quality of software receives approval is at acceptance. This should be after comprehensive acceptance testing by all of the user and information technology staffs to be involved in the system when it becomes operational. Unfortunately, some of the problems will not manifest themselves until the system has been operational for some time. Does the system carry out the functions for which it was obtained, and within the required time restraints? Again, this may not be capable of being determined satisfactorily until it is too late. If a fault/shortcoming is found, what are the criteria for measuring how serious it is? It is a matter of matching result(s) against the initial requirements. Thus the initial specification of requirements by the user is of great importance. If his specification is imprecise, it is likely that he may not be happy with unsatisfactory results. What other factors may be involved if the installed product does not meet expectations? There will be implications in terms of time, equipment and costs.

Hicks⁴ indicates that the greatest change in software maintenance in the 1990s will occur as organisations change from a software "product" orientation to a software "service" orientation. This is based on the
concept that service orientation has the customer as the driving force and his participation changing from general to detailed, with user involvement at all times, not just at sign-off stages. This will lead to software defect prevention rather than removal/repair of software faults.

A more pragmatic approach will be required in the development of software to stop users viewing software as being informal guesswork by experienced software engineers.

CONCLUSIONS

In a similar fashion to providing a computer-based solution to any problem, there is no answer to the problems of software maintenance which is 100% correct. There is a large number of solutions that can be made to work. Some practical guidelines to minimising the problems are:

a) The number of hardware platforms available to a user should be minimised. A variation should only be agreed to if the system is completely stand-alone with no possible interface ever with corporate data or any other departments.

b) Redundancy (possible duplication of facilities) in any system that is of critical importance to the organisation should be ensured. Some systems will only appear as critical at particular times, e.g. Payroll will be critical only if it is not processed in time to pay employees, but assumes a less critical position in importance between processing cycles. If such a system fails, either through a hardware or software fault, have an alternative method of processing possible, even if it is only a limited facility. The use of a manual system in the event of a power failure or some other reason is acceptable only for a short period. This applies also to communications systems - have standby (dial-up) communication links available, or alternate, preferably automatic, routing paths available.

c) It is not advisable to have any individual solely responsible for the maintenance of any system. It is better to have several staff involved to provide cover in the event of illness/staff replacement.
d) The implementation of a release of software should be delayed for a period of time to allow others to find glaring errors. This may not always be possible in the example of legislative changes such as new Income Tax or National Insurance changes in a Payroll System.

e) It is very beneficial to have users involved in, and participating in relevant Manufacturer/Supplier/Professional Body User Group(s).

REFERENCES


2 Manns, Tom and Coleman, Michael, (1988), Software Quality Assurance, Macmillan, pps1-4
