ELSEVIER

Information and Software Technology 41 (1999) 1021–1026

INFORMATION AND SOFTWARE TECHNOLOGY

www.elsevier.nl/locate/infsof

In the 25 years since The Mythical Man-Month what have we learned about project management?

J.M. Verner*, S.P. Overmyer, K.W. McCain

College of Information Science and Technology, Drexel University, Philadelphia, PA 19104, USA

Abstract

This paper discusses Brooks' *The Mythical Man-Month*, a landmark work in the software project management field, and compares the software project management advice given there with practices employed some 25 years later. To find out the state of today's practice 20 experienced software developers were interviewed regarding their impressions of factors leading to success or failure of software development projects. Their observations are compared with the points raised by Brooks in his seminal work. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Project management; Software development project; Capability maturity model

1. Introduction

The Mythical Man-Month [1], a landmark work, and a cornerstone for project management was published in 1975. As evidence of its importance to the software engineering community, we examined the paper's citation history. It was first cited in 1976 (six times), and since then has averaged 23 citations per year, to 1998, when it was cited 33 times (see Fig. 1).

Not only do authors from information technology cite this book but authors in many other very diverse fields (including business, management, ergonomics, engineering, operations research, law, planning and development, psychology, economics, social issues, chemistry, oncology, physics, anatomy, physiology, energy and fuels, etc.), also reference it. In all we counted 50 different disciplines referring to this landmark work in the past 25 years. While investigating citations for this paper we were so intrigued by the many diverse kinds of research citing Brooks' book that we intend to further investigate why researchers in these other disciplines cite it so frequently.[1] Authors referencing the work come from 28 different countries including many whose native language is not English (USA authors head the list and are followed by authors from the UK, Canada, Australia, Germany, the Netherlands and Finland). Many citations are by authors from academic institutions such as the University of Maryland, MIT, Naval Postgraduate School, Syracuse University, Virginia Polytechnic, University of Victoria, Canada, and the University of NSW, Australia, though authors from over 30 institutions cite the book at least twice.

2. Brooks' assessment

Brooks' work raised so much interest because it was the first major publication to deal with the difficulties of managing large software development projects. It describes many of the pitfalls project managers face when trying to control such projects. Brooks discusses the fundamental problems of project management and suggests that more projects have gone awry for lack of calendar time than from all other causes combined. He notes in particular that project management problems stem from the following:

1. Our techniques of estimating are poorly developed and reflect an unstated assumption that all will go well.
2. Our estimating techniques confuse effort with progress hiding the assumption that men and months are interchangeable.
3. We are uncertain of our estimates and software managers do not stubbornly support them. We need to develop and

* Corresponding author.

E-mail address: june.verner@cis.drexel.edu (J.M. Verner)

[1] The citation data were gathered by searching the combined online files for *Science Citation Index* (SCISEARCH) and *Social Science Citation Index* (SOCIAL SCISEARCh) for all articles with at least one citation to any edition of *The Mythical Man-Month* in the period 1974–summer 1999. We are currently conducting a detailed citation content analysis of the 530 + source articles with an eye to exploring the relative impact of the many sigificant concepts for which Brooks' work is recognized across disciplines and across time.

0950-5849/99/$ - see front matter © 1999 Elsevier Science B.V. All rights reserved.
PII: S0950-5849(99)00077-4

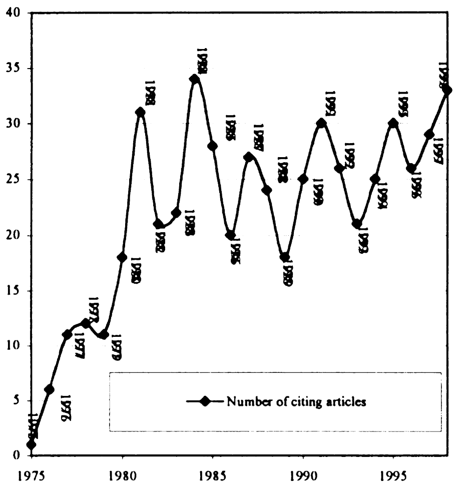

Fig. 1. The Mythical Man-Month number of citations by year.

publicize productivity figures and stiffen our backbones to defend our estimates.
4. Schedule progress is poorly monitored and techniques used in other disciplines are considered radical here.
5. When slippage is recognized the response is to add manpower. This makes the problem worse because adding manpower to a late software project makes it later (Brooks' law).

Brooks also included many cautions for project managers including:

1. Large individual differences exist between high and low performers, often by an order of magnitude.
2. Development team organization may make all the difference.
3. The project manager must have a written plan.
4. Written specifications are necessary but not sufficient.
5. A vertical division of labor will result in radically simplified communication and improved conceptual integrity.
6. Change is inevitable, making change management and planning imperative.

Since 1975 there have been many attempts to improve our software project management and software engineering practices and many useful books and articles have appeared. There have also been significant advances in software tools and techniques including the introduction of new requirements engineering methods, programming languages, process improvement guidelines, and CASE tools, etc. We do not intend to go into any detailed discussion here but note that Brooks published another much cited paper *No Silver Bullet* (NSB) [2] that discusses the progress made to that date. He also published an anniversary edition of *The Mythical Man-Month* in 1995 that includes the NSB paper plus three new chapters [3].

But even with all this progress we still read in books and journals about software development failures. Indeed, a 4 h workshop session was devoted to the discussion of software failures at a recent conference (STEP99) [4]. Since unsuccessful projects are likely to get rather less publicity than those that are successful, we sought information from those closest to the problem.

3. State-of-the-practice

3.1. Introduction

We were interested in discovering the current state-of-the-practice of software development in the average business organization. What do software practitioners see as the major factors that lead to the success of the projects they work on and what do they consider are factors that lead to unsuccessful projects in their organizations? To answer this question we have had structured discussions with 20 senior software development professionals from a number of different organizations in United States regarding their software development practices. These interviews were conducted independently of any discussion of Brooks' work.

Our respondents work in organizations that range from levels 1 to 4 on the SEI capability maturity model (CMM) scale with from 5 to 500 development personnel involved in each of the 34 developments summarized below. About half the projects were viewed as successful by the developers involved, although one developer did comment that a project she worked on was considered successful by management but was one of the most unpleasant projects she had ever worked on.

We have organized our discussion under the following headings: (1) management support, (2) customers and users, (3) requirements and specifications, (4) project manager, (5) estimation and schedule, (6) development process and methodology (including risk management and planning, monitoring and control), and (7) staffing. Within each of the headings, we review the factors that firstly lead to project success and then how that same factor may contribute to the failure of a project. We conclude with a discussion on what we think we, as a community, have learned since Brooks' landmark paper in 1975.

3.2. Management support

For the 20 successful projects, about half our respondents mentioned that high-level management support was a key factor in their project's success. Comments included:

- we had good support from senior management in all areas that was more than just lip service;
- we had support from the division head all the way down to the users;
- senior management held the team to milestones that they closely monitored;
- senior management made sure that the project was not held up by things the developers could not control;

- there was never a question about what needed to be done and the evaluation criteria for success; and
- project focus was on measurable business results.

Almost all of the failed projects were affected by the lack of higher level management support and/or support structures. In many cases the project manager was completely left out of the decision making loop, in other cases decisions were made with no input from the project manager; or the project manger was not given the authority to make personnel decisions. Three of the projects ran into serious difficulty because of the use of consultants. In one case, the project manager had no authority over the consultants who subsequently left without debriefing the rest of the team. In another case, there were too many consultants from too many firms with no one in control of them. Several respondents mentioned that a contributing factor to the failure of their project was the lack of a champion, while in other cases politics were the major factor leading to project failure. High-level management changes led to the failure of another project while infrastructure changes part way through project impacted badly on yet another project.

3.3. Customers and users

When describing successful projects, there were few comments by our respondents regarding customers and users. Only three respondents commented on the effects of the users on projects; their comments included:

- we worked closely with business consultants throughout; and
- the entire user-base had the opportunity of participating in acceptance testing and volunteers from the various organizations helped us test the product and gave us ample feedback which was incorporated into the system.

In contrast, problems with customers and users affected nearly 50% of our failed projects. Factors that were mentioned include:

- too little involvement with the user community;
- customers had no confidence in the project manager and tried to avoid dealing with him;
- the only customers who were interested in the project left the company;
- the project ran into trouble because of user–staff turnover;
- there were just too many managers and customers to deal with;
- late business partners affected the project badly; and
- there was serious friction between user groups.

3.4. Requirements and specifications

Good requirements gathering was explicitly mentioned by nearly 50% of our respondents when discussing successful projects; they made comments such as:

- the requirements and design were sound; and
- the requirements were clearly defined.

To get good requirements they mentioned working directly with users and management with prototypes and/or iterative refinement.

For the failed projects poor requirements were problems in 40% of the projects. Our respondents made comments such as:

- it was a huge project with vague requirements;
- there was no clear vision of the final deliverable; and
- the project was doomed because of poor requirements.

Poor requirements were due mainly to inadequate requirements gathering, lack of user input because the customer would not make the time available, and misunderstandings by the customer of what the specifications really meant.

3.5. Project manager

For the successful projects, respondents did not often comment on the project manager. When they did, they mentioned the presence of a knowledgeable and experienced project manager, or a project manager who was experienced in the applications area and who had upper level management support.

Over half of the unsuccessful projects had project manager problems. There was criticism of the project manager, the lack of a project manager, and with changes in the project manager midstream. Most discussion centered around a project manager who:

- had no experience;
- spent insufficient time on planning the project;
- was unable to provide an integrated project plan;
- was without people skills;
- could not communicate with staff;
- did not set priorities;
- was unable (or chose not to) control the project; or
- was obsessed with micro managing everything.

Other project managers played favorites, did not understand the customers' problems, showed no appreciation for staff working up to eighty hours and seven days per week, took vacations during heavy work periods while denying vacations to subordinates and labeled any staff member who complained as a troublemaker.

3.6. Estimation and schedule

When asked about effort and schedule estimation for the successful projects the following comments were made:

- project estimates were accurate;
- sufficient time was allotted to properly execute all tasks despite a somewhat aggressive delivery date;
- scope of the initial delivery was reasonably agreed upon by the business managers despite frequently shifting business aims; and

- developers were able to estimate their own effort and develop their own timetable.

One comment that was particularly interesting for a successful project was that though the project began with an unrealistic dictated date the project manager had the delivery date adjusted.

In contrast there were many more comments made regarding estimation and scheduling for the failed projects. Nearly 50% of the failed projects had estimation and schedule problems. Our respondents commented in particular on poor estimates that were made by management with no input from the project manager or any other member of the team. This included dictated dates that were decided before the specifications were complete. They also described estimates that were inflated fivefold by a manger who could not bear to deliver the project late, overly optimistic delivery dates, aggressive schedules with many "extras" forced on to the team and overlapping phases.

3.7. Development process and methodology

When discussing the development process itself, our respondents described many excellent features of the processes employed for successful projects. They particularly mentioned factors such as good risk management and monitoring, frequent project meetings and status reports, and the use of appropriate lifecycle models. In fact, for the successful projects there was more emphasis by respondents on the development process and allied factors than any other factor that leads to a successful project.

With over 75% of the failed projects having problems with the development process itself it is interesting to contrast the respondents' comments on the factors they felt were instrumental in the failure of their projects. The major problems discussed were in two main areas, the lifecycle model used and, monitoring and control of the process. In addition two of the failed projects had other problems. One was described as being a research oriented development and another as having changed tools in the middle of the project.

3.7.1. Lifecycle methodology

There was much discussion of the methodology used for the successful projects. All the respondents mentioned choice of the right methodology as being a relevant factor in the successful projects. Four respondents mentioned the use of prototyping for their successful projects, one a phased approach while the remainder of the successful projects were developed with the use of a waterfall approach.

Three respondents mentioned that their failed project had no methodology at all. These projects were developed using basically a code-and-fix approach. Other respondents mentioned that the wrong methodology (waterfall) was used or that the methodology chosen was not properly applied. One respondent suggested that if only an incremental approach had been used that the project probably could have been successful.

3.7.2. Risk management

Several of the respondents discussed risk management and monitoring as a factor in the successful projects. They made comments such as:

- potential risks were identified at the outset and incorporated into the project plan;
- issues were raised as they occurred and were dealt with immediately so they could not become a problem; and
- expectations and risks were managed.

Interestingly not one of the respondents mentioned risk assessment, or the lack of it, when discussing failed projects.

3.7.3. Planning, monitoring and control

This factor was singled out for over 75% of the successful projects. In particular, respondents mentioned:

- the use of project management tools to identify critical path components;
- planning then replanning when necessary;
- properly set up controls for requirements and/or scope changes;
- management of user expectations; and
- good project tracking with immediate identification of slippage with appropriate and timely action taken.

When discussing successful projects respondents frequently described the reporting systems that were set up both for the project team members and for management. Two-thirds of the respondents mentioned communication as being a key factor in their project success, while several respondents suggested that effective mechanisms for reporting to management and customers were major factors in the success of their project. Some of the notable factors respondents related to successful projects were:

- effective communication;
- quick daily progress meetings;
- circulation of status reports to the entire team;
- weekly status meetings;
- minimal preparation time for meetings;
- simple reporting that did not take up a lot of time;
- informal meetings with an agenda; and
- well-organized and short meetings.

For the failed projects respondents mentioned three projects that suffered from insufficient planning or had no plan at all, and one project where there was abandonment of planning under pressure. Respondents also mentioned that many of the failed projects had no change control system set up, that feature and scope creep were big problems, changes were not documented, and milestones that were not met with no subsequent action taken.

Project Success	Project Failure
High-Level Management Support	No High-Level Management Support
Involved Stakeholders	Uninvolved Stakeholders
Negotiated, well-defined requirements	Non-negotiated, vague requirements
Experienced Project Manager	Inexperienced Project Manager
Accurate developer-driven estimates	Inaccurate, management-driven estimates
Appropriate lifecycle models	Inappropriate lifecycle models
Managed risk	Unmanaged risk
High Intra-Team Communication	No Intra-Team Communication
Low Staff Turnover	High Staff Turnover

Fig. 2. Project success and failure factors.

3.8. Staffing

Our respondents singled out staff turnover as being a major contributor to both the success and failure of projects. In the successful projects there was either no staff turnover, or very little staff turnover. Our respondents noted that staff were not pulled off the project to work on other projects and that key individuals stayed with the project all the way through. Every unsuccessful project, on the other hand, had serious staffing problems. Respondents mentioned many projects with too many resignations resulting in poor staff continuity and low morale for the remaining staff. They also described several instances where there was too much dependency on a single person who resigned in the middle of the project.

4. Discussion and conclusions

4.1. Where are we now?

We have had, in theory, over 25 years in which to improve our ability to manage large software development efforts according to the principles that Brooks and others have observed and reported. There have been many other published accounts of the software engineering process, how it should proceed, how it should be measured, and how it should be managed (e.g. Refs. [5–9]). Regardless, we still experience significant problems with software development projects, some of which are avoidable, and some of which, in our opinion, are not. For example, project “scope creep”, resulting in cost and schedule discrepancies, can likely be attributed to a poor requirements process, and is avoidable, or to poor change control mechanisms, which is also avoidable. On the other hand, the simultaneous departure of several key personnel will result in setbacks, which while manageable, are probably unavoidable although serious consideration by upper level management to staff retention can help.

The results of our interviews suggest that, in general, both successful and unsuccessful projects today have some important variable characteristics, many of which appear to be under management control. From these observations, a series of nearly bipolar descriptor pairs emerge along which the success or failure of modern software development effort can be forecast, albeit with a low-level of precision. These pairs are shown in Fig. 2. These descriptors represent the critical success or failure factors as reflected by our respondents, based upon their experience. Of the factors pointed out by our respondents, Brooks mentioned several as problems in 1975. The notable ones are estimation, planning, communication/organization, risk/change management, and specification issues. Our respondents indicated that successful projects addressed these problems while unsuccessful projects were unsuccessful because of a lack of attention to these issues. In other words, projects heeding Brooks’ advice were more likely to be successful.

There are a number of problem areas mentioned by our respondents that Brooks did not directly address in his earlier work. Many of these problems are partially or completely beyond the control of the project manager, although all of these problems are such that strategies should be developed to mitigate the risk of project failure as a result of emergence of any one of them. Some examples of these are: (1) stakeholder involvement, (2) upper management support, (3) staff turnover, and (4) the project manager’s experience.

Our respondents did not mention Brooks’ law regarding specific project management response to schedule slips though it is likely that project managers still add manpower in an attempt to alleviate schedule pressure. No mention was made of assessing specific productivity of individual performers to address Brooks’ caution about discrepancies in productivity between high and low performers.

4.2. Conclusions

As reflected in the comments of our respondents, some software organizations have heeded the advice of Brooks and others, and have significantly improved their ability to successfully manage complex software development projects, while some have not. With the change in the nature of computing since 1975, have come many new problems related to end-users, their level of technical sophistication, and the high level of complexity of the development environment. As we face the challenges of managing COTS, object-oriented and yet-to-be-conceived developments, and of measuring process and productivity in these new paradigms, it appears that the same issues will likely resurface. Many of these issues are those that can be found in any project management environment and are the same nine factors listed in Fig. 2.

The idea that so many project managers are apparently unprepared to deal with these issues calls into question their preparation for management. Until software project management is viewed as both a technical and managerial discipline requiring a formal graduate-level education and requisite experience for its practitioners, we are likely to continue to encounter many of the same problems that Fred Brooks warned us about so many years ago. However, other problems have emerged, and until senior business managers and customers/users are made aware of their roles in a software development project we will still have project failures due to factors completely beyond the project manager's control.

References

[1] F.P. Brooks Jr., The Mythical Man-Month, Essays on Software Engineering, Addison-Wesley, Reading, MA, 1975.

[2] F.P. Brooks Jr., No silver bullet—essences and accidents of software engineering, Computer 20 (4) (1987) 10–19.

[3] F.P. Brooks Jr., Anniversary Edition of The Mythical Man-Month, Essays on Software Engineering, Addison-Wesley, Reading, MA, 1995.

[4] STEP99, Software Technology and Engineering Practice, Pittsburgh, August 1999.

[5] S. McConnell, Rapid Development, Microsoft Press, 1998.

[6] A.P. Sage, J.D. Palmer, Software Systems Engineering, Wiley/Interscience, New York, 1990.

[7] B.W. Boehm, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[8] S.J. Andriole, Managing Systems Requirements: Methods, Tools, and Cases, McGraw-Hill, New York, 1996.

[9] N.E. Fenton, S.L. Pfleeger, Software Metrics: A Rigorous and Practical Approach, Thomson Computer Press, UK, 1996.