Demanding higher productivity

Remember the human factor

Richard Veryard

First published in data processing vol 28 no 7 (September 1986) pp 351-355

Keywords

Data Processing, Software Techniques, Fourth-Generation Languages

Abstract

Productivity of data processing staff is a much-discussed, but little analysed, concept. This article discusses the consequences of using productivity enhancing tools (PETs), including fourth-generation languages, and considers future tools that might be worth developing.

Acknowledgements

I should like to thank my colleagues at Data Logic, and those on the BCS Information Systems Analysis and Design Working Party, especially Love Bhabuta, for discussing these ideas with me.

Biography (1986)

Richard Veryard is a lecturer/consultant.

Current Author Details (1999)

email: richard@veryard.com

web: http://www.veryard.com

One of the perceived problems of the DP industry is the shortage of technical skills. Assuming this to be a genuine problem, there are three possible ways of addressing it. One is to improve the input of trainees into the industry, which raises the contentious issue of the value of degrees in computing science. Another is to reduce the demand for DP, possibly by allowing the costs of software development to increase out of proportion. A third way is to make better use of existing skills, which may be wasted either by using highly-trained technical staff for inappropriate jobs, or by using their skills inefficiently.

So there is much interest in enhancing productivity. Various tools and techniques, loosely called 'fourth-generation', have been launched in recent years to address this problem. The best-known fourth-generation products are very-high-level languages for application development and end-user queries.

Using these tools and techniques may indeed enhance productivity. This article discusses what exactly productivity is, and what kind of typical improvements can be expected. It then discusses the wider socio-economic impact of using enhancing productivity in DP. Finally, some areas are identified where the development of new kinds of tools and techniques might bring new social and economic benefits.

Case Study: The US Navy

A naval story, although remote, will shed light on the process by which new techniques are introduced. The story is useful because it is always easier to see other people's problems from a safe distance than to solve one's own. We may then better understand the slowness with which new tools and techniques catch on in the computing industry, an industry which critics characterize as being dedicated to the computerization of other people's jobs, and the preservation of its own.

The following discussion is based on Morison, to whom readers are referred for further details and analysis.

In 1899, five ships of the US Navy, aiming at a lightship hulk at a range of 1600 yards, hit the target twice in 25 minutes of firing practice. Six years later, a single gunner with essentially the same equipment managed to hit a much smaller target at the same range 15 times in one minute. This 3000% improvement in accuracy was achieved by introducing a new technique of firing, called continuous-aim firing, which however, the Navy establishment at the turn of the century was extremely reluctant to adopt.

The introduction of continuous-aim firing into the US Navy was the work of one man, William Sims, then a junior lieutenant. He learnt the technique from a British naval officer, Percy Scott. Both men had difficulty persuading their superiors to take the idea seriously, despite the spectacular success shown on Scott's ship.

There were three reasons for this difficulty.

First, there was honest scepticism about the new technique; it was hard to believe that such enormous improvements were possible.

Second, the traditional techniques commanded much loyalty; in a changing world, people need to identify with familiar tools and techniques.

Third, the existing structure of the navy relied upon the current technology; relations of power and influence would be altered by the improved status of the gunner, thus disrupting the closed society which the navy was. Sims only succeeded by going over the heads of the admirals and appealing to President Roosevelt, who was not a navy man. He looked at the claims objectively

0011-684X/86/070351-05$03.00

and, convinced, forced the admirals to accept the change.

This story has parallels in DP. Many industry pundits, in particular James Martin, argue that the shortage of DP skills and the increasing demand for computer systems should force the industry to adopt new tools and techniques for systems development. However, this advice is not universally adopted. This is partly because the figures quoted by the suppliers of fourth-generation tools and techniques seem so exaggerated. It almost seems as if a complete management information system could be written by a 17-year-old trainee before lunch, which would otherwise have required 2000 man-years of effort.

The fourth-generation suppliers put off many of their potential customers by trying to outdo and outbid one another with such claims. Resistance on the part of DP managers is also partly because of a subconscious fear of no longer understanding what happens within the computer. With COBOL or BASIC or any procedural language, the DP manager at least understands what his staff do, and feels himself to be in control.

With the introduction of new tools and techniques, the whole department may be turned upside down. Programmers may need to be retrained as analysts, reporting structures and development standards will have to be revised, perhaps control will end up in the hands of the users. Perhaps many people will wait until change is forced upon them, by people who have no sympathy for their reluctance. It is one thing to wreak disruption upon the user department, it is quite another to be the victim of change.

Productivity

There are three variables of production:

- resources consumed
- time taken
- quality of the product

For simplicity, we shall here assume that a productivity enhancement improves one of these three variables while leaving the other two constant, i.e. production will be cheaper *or* quicker *or* better. (In reality, of course, the most common case is that all three change, and not all for the better.)

There are the following typical ways of increasing a worker's productivity at a task. Any of them may involve new machinery, or training to use existing machinery more efficiently.

First, the worker can use a lever/amplifier to process more material with the same effort and/or in the same time.

Second, the worker can delegate sub-tasks to an assistant or colleague (human, computer or beast of burden). This is 'division of labour'.

The worker can be given expert guidance for some or all sub-tasks, or with measuring tools to improve accuracy and reliability. (This is related to 'standardization', the use of interchangeable parts.)

The manager can restrain the worker from carrying out unnecessary sub-tasks, possibly including thinking. Gramsci discusses the example of a typesetter, who works faster if he or she does not understand the text.

The task can be reorganized into sub-tasks as a preliminary to any of the above. This is called 'rationalization'. Marx recognized that the scientific analysis of a productive activity into separate steps is a precursor to technical innovation and increases in productivity. [Rosenberg]

Finally, the manager can increase the motivation of the worker. This can involve threats, inducements (financial or otherwise) or making the

job more fulfilling. Motivation is often reduced by job fragmentation and loss of autonomy, although strict adherence to Taylorism [Taylor] tends to ignore such matters [Herzberg].

One approach to improving productivity is known as Group Technology (GT).

The essence of GT is to capitalize on similarities in recurring tasks in three ways:

- By performing similar activities together, thereby avoiding wasteful time in changing from one unrelated activity to the next.
- By standardizing closely related activities, thereby focusing only on distinct differences and avoiding unnecessary duplication of effort.
- By efficiently storing and retrieving information related to recurring problems, thereby reducing the search time for the information and eliminating the need to solve the problem again. [Hyer & Wemmerlov]

Thus productivity enhancing tools, in general, can include:

- performance tools, i.e. those acting as levers or amplifiers, or those that can perform entire sub-tasks with a minimum of effort from the worker
- communication tools, involved in the coordination or instruction of workers,
- control tools, i.e. those that guide and monitor the worker and help impose some form of discipline. For example, piped music in a factory, to relax the brain, would count as a control tool.

In addition, we can consider productivity improvements resulting from the replacement of one worker altogether, either by a cheaper worker (i.e. one with less native ability, experience, training or union solidarity) or by a robot. Note that factory automation may improve productivity per worker, but does not necessarily improve overall productivity. What it does is change the labour/capital ratio. And clearly, the productivity of the redundant worker him/herself is reduced to zero.

There are three reasons why the productivity increases from a given innovation are difficult to measure.

First, although it is easier to measure average productivity, we are often more interested in marginal productivity.

Second, there is a tendency for productivity in a stable environment to increase anyway. This is known as 'learning by doing'. [Arrow] Third, there is the difficulty of separating the effects of multiple innovations, possibly with some delay. [Rosenberg]

Productivity in DP

Fourth-generation tools: PETs and IPSEs

The DP industry is swamped with so-called fourth-generation tools and products. These are called productivity enhancing tools (PETs), although this should not imply that their sole purpose is to enhance productivity. They include query languages and application program generators, data dictionaries and workbenches, and a variety of other automated tools. Current research is concentrating on developing integrated sets of tools, or IPSEs. They are generally intended for systems development, but they can also be used to support system maintenance.

Impact on Productivity

Do these tools actually increase productivity, in what way and by how much? To answer such questions raises a number of difficulties.

The first problem is the measurement of productivity (referred to in previous sections), and the comparison between unlike situations producing different products. Previously,

production was measured by the number of lines of program source code, but this is not longer valid, if indeed it ever was. A recently proposed alternative is the function point measure [Drummond] but this appears arbitrary and limited in scope.

Second, whose productivity is of interest? Are we interested in the productivity of the computer programmer or of the end-user manager? Is productivity increased by producing each month a different format for the printed output? Can productivity be increased by reducing the functional power and complexity of an existing system?

The third difficulty is the variety of ways a tool affects the division of labour, between users and DP staff, between programmers and analysts, and between specialists and generalists. The same tool can be used in a number of different ways, even within the same organization. This will be discussed later.

Economic Impact

Productivity and salaries

What impact will the use of PETs have on the salaries of DP staff? Many people expect to be able to use more junior staff for certain tasks. The question that no one can yet convincingly answer is what the senior staff will do with their skills, and will these skills be of greater or lesser value.

'A rational and far-sighted capitalist will search for the innovation that has the best net effect on the profit rate, taking account of both the impact on productivity and the wage rate. In addition ... one might have to postulate a measure of solidarity with other capitalists to overcome the free-rider problem.' [Elster]

Productivity and microeconomics

Do the concepts of productivity or efficiency have any genuine meaning in a service industry, or in administration? In any case, even in a manufacturing environment, not all of the company's objectives can be expressed in terms of the quantity of product. Even profit cannot be regarded as a sole measure of a company's success.

It cannot be proved that the use of up-to-date or sophisticated information technology improves or even influences the success of a company. If there are causal links, they could well be in the opposite direction: successful managers have more money to spend on computers, and a greater inclination to do so. Furthermore, Paul Strassmann has produced statistics that appear to weaken, if not deny altogether, the causal links between IT investment and overall success.

Productivity and macroeconomics

If we broaden our scope from the economic success of a single company to the national or global economy, the situation becomes yet more complex. In the early days of computing, it was perhaps an acceptable simplification to regard the computer as a useful adjunct to normal wealth-creating enterprise. Thus the productive value of the computer could be measured in terms of its contribution to the production process. Even so, a computer in a bank would be several steps away from direct contribution to the production of physical wealth.

Nowadays, information technology is itself an important sector of the economy, and forms a significant part of the gross national product (GNP). It receives much attention from politicians, businessmen and journalists, who want the IT industry to create jobs and investment opportunities. Computer power is now a self-perpetuating commodity (like nuclear defence or literary criticism), whose value in the marketplace has become detached from its productive usefulness.

Social Impact

Bottlenecks and efficiency

Macfarlane suggests that the key to productivity is not how advanced a production aid is, but whether the production process has been optimized to benefit from its most effective use.

In a process network, if you remove the critical path, there will be a new critical path somewhere else. Similarly, if you remove a bottleneck from a complex process, you will only create a bottleneck somewhere else. [Godfrey & Struthers] In a complex environment, the bottleneck is usually associated with the processor rather than the process, with individuals or sections rather than specific tasks.

The traditional bottleneck in computer systems development has been the systems and programming section. If fourth-generation languages and other productivity aids remove the bottleneck from this part of the process, it shifts to the business analysis section. And if analyst workbenches and business modelling tools remove the bottleneck from this part of the process, it shifts to the users. Many companies have already reported that they are producing computer systems faster than their users can absorb.

The bottleneck is then the amount of change the user department can stand. In the past, they have always demanded immediate development and implementation of new computer systems, knowing that they will have to wait months anyway. Nowadays, there is a real chance of delivery within weeks or even days.

In truth, the user usually *needs* a period of anticipation before getting a new system, or even a substantial change to an existing system. This waiting period serves a similar psychological function to that of human pregnancy. Prospective parents need many months in order to adjust to the fact that their formerly free and easy lifestyle is to be disrupted and constrained.

Division of labour

The following trends have been observed, in different situations.

- Some systems analysis, design and even programming tasks are taken over by the end users.
- Some analysis and design tasks are taken over by the programmers.
- All programming tasks are taken over by the analysts, allowing all existing programmers to be promoted to analyst.
- There are fewer programmers per analyst, allowing some existing programmers to be promoted to analyst.

> *The word 'promotion' does not imply that programming is an intrinsically inferior job to analysis, but merely that it has lower status within the career structure of most data processing departments.*

However, more radical changes can be expected in the longer term, affecting the management, struture and skills profile of the typical DP department. There is a trend towards increasing specialization, which is not wholly caused by the use of PETs, but is reinforced by it.

Specialists and Generalists

In theory, generalists are of more value than specialists:

'Wherever demand for products is of an uncertain or variable nature, it is an economy in the long run to use non-specialized machines: this decreases the burden of wasted effort and idle machinery. What is true of machines is equally true of the worker: instead of a high degree of specialized skill, an all-round competence is better preparation for breaking through stale routine and for facing emergencies.' [Mumford]

But this is a long term consideration. In the short term, there are two contrary trends: one towards deskilling of labour and one towards increased skill specialization. This was recognized by Marx. [Elster]

There is a growing number of technical specialisms within DP. Systems development is carried out no longer by monolithic project teams of analysts and programmers, who are generalists at different stages of the same career path, but increasingly by interdisciplinary teams. Such specialist tasks as database design, man/machine interfacing, networking, knowledge engineering, etc., are carried out by local experts, perhaps shared part-time between a number of projects. The PETs and IPSEs are primarily intended for use by the generalists; if the productivity of the generalists increases more than that of the specalists, we will need proportionately fewer generalists in future. The skills of the generalist will be required mainly for coordinating the efforts of a growing variety of specialists; this will require an ability to cope with complexity, but this has always been important for the systems analyst.

Conclusion

In many ways DP is more like hairdressing than factory mass-production. Hairdressing is a typical services industry, although there is a product – the haircut.

Sometimes the efficiency of the hairdresser can be measured, e.g. an army barber may be paid to produce fast, not fashionable, haircuts. Usually this is not the case. Even when a hairdresser performs what appear to be unnecessary tasks may not be unnecessary to him/her. Eliminating these tasks may have no effect, or worse might reduce the quality of the haircut. The time to fetch tools may have been essential thinking time.

The concept of productivity assumes a production system, i.e. a systems development methodology itself regarded as a system, with measurable input and output, fixed boundaries and fixed objectives. Each of these assumptions is dubious for business computing; attacks from the 'soft' systems community are particularly telling. [Miles] The obsession with programmer productivity may indeed obscure and delay the broader potential benefits from automated tools and software 'factories'. Improving communication, coordination and control is more important than removing wastage and drudgery from computer systems development.

'The most advanced and powerful programming tool you could develop would be something that gives a programmer the skills and courage to share work with colleagues, but we're not working on such tools.' [Weinberg]

Tools are perhaps required to improve the end-users' perception of the software engineering being done on their behalf, and to enhance their active participation in system design. Tools are perhaps required to optimize the level of automation, so that the marginal benefit of a more powerful computer system can be balanced against the additional cost. Tools are perhaps required to decentralize decision-making and planning, to enhance job satisfaction of DP staff at all levels. 'Water wings' tools will certainly be required, to support novices in new skills.

The use of these tools must be optional. Mumford argues that it is generally beneficial to retain some 'craft' production alongside automated 'factory' production, as a source of 'education, recreation and experiment' and 'as a means to further insight, discovery and invention'.

The quest for automation is always intellectually exciting, because it is an abstraction from normal practice. PETs and IPSEs are inevitable, and probably beneficial. But it is a mistake to concentrate on enhancing productivity, and to forget human and

organizational needs. Time and money are means, not ends. There is an opportunity to bring fulfilment into many people's lives, by increasing their understanding and control of the systems they belong to. This opportunity must not be missed.

References

Kenneth **Arrow**, 'The economic implications of learning by doing' Rev Econ Studies (June 1962)

Love **Bhabuta** & Richard **Veryard**, 'An assessment of fourth-generation environments'. in D. Maitland, S. Holloway & L. Bhabuta (eds), Fourth-Generation Languages and Application Generators (Technical Press 1986).

S. **Drummond**, 'Measuring applications development performance' Datamation (Feb 1985)

Jon **Elster**, Making Sense of Marx (Cambridge University Press, 1985)

P. **Godfrey** & K. **Struthers**, 'Material flow is key to productivity' Chart Mech Eng Vol 32/5 (May 1985)

Anton **Gramsci**, Selections from the Prison Notebooks (Lawrence & Wishart, London 1971).

F. **Herzberg**, 'One more time: how do you motivate employees?' Harvard Business Review No 46 (1968)

N. **Hyer** & U. **Wemmerlov**, 'Group Technology and Productivity' Harvard Business Review (May/June 1984)

T. **Kizilos**, 'Kratylus automates his urnworks' Harvard Business Review (May/June 1984)

R. **Klein**, 'Does automation necessarily mean an increase in productivity?' J Syst Man (July 1984)

James **Macfarlane**, 'The key to productivity' (BCS Database Group ISAD working paper, 1985)

R.K. **Miles**, 'Computer Systems Analysis: The Constraint of the Hard Systems Paradigm' J Applied Systems Analysis No 12 (1985)

E.E. **Morison**, Men, Machines and Modern Times (MIT Press, 1966).

Lewis **Mumford**, Technics and Civilization (Routledge, 1934).

N. **Rosenberg**, Inside the black box - technology and economics (Cambridge, 1982).

Paul **Strassmann**, Information Payoff (Collier-Macmillan, 1985).

F.W. **Taylor**, Principles of Scientific Management (Harper & Row, 1947).

J. **Weinberg**, 'Visit to Vienna ...' Datalink (July 1, 1985).

0011-684X/86/070351-05$03.00

Retrospective Notes (1999)

In the mid 1980s, I wrote a batch of articles, covering a range of related topics. Some were published in newspapers and magazines (The Times, The Financial Times, New Society), some in computer journals and magazines (Computer Weekly, Data Processing, Information and Software Technology), and some in system journals (Journal of Applied Systems Analysis, Human Systems Management). I also wrote a number of book reviews, mostly for Information and Software Technology.

That body of work did two things. It established a set of themes that have remained important for me, and it also hinted at some further themes whose form and significance have only emerged more recently.

These articles are the work of a young man, with the brash and sometimes careless optimism of youth. Although I now find some of the analysis simplistic and naïve, and I would certainly try to express the positions and opinions of these articles with greater precision and care, I hope I have retained the spirit of them.

❋❋❋

When I sent this article to the publisher in May 1986, I was working for an independent software house. By the time the article was published, I had joined James Martin Associates as a consultant. I spent over ten years working for JMA and (after a merger in 1991) Texas Instruments Software, where all our products and services were intended primarily to enhance development productivity.

With this experience, I can hardly claim to be neutral about productivity. If I were to write another article on the subject today, I should have to examine my own record closely. Certainly there is much I could add now, both about the nature of productivity, and about the various ways people in organizations resist measures that are aimed at improving productivity.

Among other things, I now know a lot more about the use and abuse of function point analysis, as a somewhat flawed measure of software size. When divided by effort, a function point count appears to provide a crude measure of productivity – but it is the productivity of a specific subsystem rather than of the whole development system. In other words, it measures the efficiency with which a given design is implemented in software, but it doesn't measure the overall efficiency with which a given software design achieves a given business objective.

Although I was correct to say that the bottlenecks in systems development were merely shifted rather than eliminated, I was wrong in my prediction of the new location of the bottleneck, and wrong to blame the users (and especially wrong to insult and patronize them). As it turned out in practice, when much of the programming effort was eliminated by CASE tools, many of the bottlenecks remained within the IT domain, in such areas as systems testing, quality and above all implementation planning. (In recent years, this has also been evidenced by long timescales to implement packaged software, such as ERP packages.) And although much of the software temporarily became shelfware, there were many other reasons for this.

As a consultant, I have seen productivity initiatives fail in many organizations, for a range of reasons. In some, there has been too much status associated with headcount and budget – especially for project managers – for there to be any genuine enthusiasm for productivity enhancements that might reduce the size of projects and project teams. In others, there has been a vague hope that new tools would enhance productivity, but a complete lack of management responsibility or action to realise this expectation.

❋❋❋

Besides productivity itself, the other important theme running through the article is technical change. Technical change may be motivated by an intention to increase productivity, although there are of course many other motives. And as the implementation of IT systems can be regarded itself as a technical change – at least for the users – the article implies that we can talk about the productivity of technical change. I am now uncomfortable about this implication. There is a lot more to say about technical change, but it doesn't belong here.

❋❋❋

One of the reasons for writing the original article was to challenge simplistic notions of productivity, by indicating a diverse set of modes of productivity enhancement. I still think this purpose was valid, although I appear to have missed out many modes of productivity enhancement I should now consider worthy of note.

Throughout the article, the emphasis is on seeing productivity as a property of a system, rather than on an individual agent or artefact. In the discussions of the ISAD working group, James Macfarlane insisted on this point, and I remain convinced that he was right. Many of the issues raised in the article have to do with how the system is scoped. There are also issues in terms of the stakeholders of the system: productivity for whom? But despite a reference to soft systems thinking, I don't problematize the notion of system itself. If we take the system to be a socially constructed fact, rather than a hard physical fact, even the location of a bottleneck in a process flow becomes a social construction, perhaps even an illusion. (We often see this in practice – for example, where two departments conceptualize a situation differently, each may accuse the other of being the bottleneck.) But as the article hints, the bottleneck isn't the problem anyway – it is a symptom of something else.

❋❋❋

Although there is some irony in places, trying to distance myself implicitly from the belief that productivity is always a Good Thing, I did not directly challenge the value of productivity, which I now regret. Even from the employer's perspective, there is often an important trade-off between straight productivity and flexibility.

