Human Resource Management and Business Success in Small, Hi-Tech Firms: Some Lessons for Data Processing

P. S. LICKER

Faculty of Management, University of Calgary, 2500 University Drive, Calgary, Alberta T2N IN4, Canada and Visiting Lecturer, Cambridge University, UK and Nottingham University, UK

What does human resource management contribute to the success of small, high-technology firms? The results of our survey clearly indicate three human resource-related characteristics that increase the chances of success in small software firms. The lessons from this can be generalised to apply in high-tech departments within firms that deal on a product basis with internal customers. Although success is rarely guaranteed, these three characteristics contribute to a flexible planning framework to meet and counter challenges in fast-changing markets. Ten success tactics are detailed and translated to a data-processing context.

Received August 1988

1. IIUMAN RESOURCES AND HIGH TECHNOLOGY

We are all fascinated by success, and high-technology success is even more enthralling. High-technology products seem glitzier and the entrepreneurs more dynamic than more traditional successful industrial areas. While 'excellence' may lead to success outside the high-tech arena, success alone, without excellence, captures our imagination when computers or communication become involved – most laymen really don't have the capacity or background to evaluate excellence, but we certainly recognise success. Those successes are most noticeable in the mass-marketed software area of spreadsheets, word-processors, and database managers. In specialist markets, too, successes and successes-in-the-making are a source of intrigue, even more so because the products are aimed at highly trained users rather than the general public.

On the other hand, the concept of 'success' in data processing (DP) seems more remote; indeed, the focus of trade literature is on the failures of DP and ways to avoid them. Certainly the early efforts in automation of routine business systems encountered success, but as design and implementation has moved into more difficult areas and as users have begun building their own systems on microcomputers and local area networks, the focus on 'success' in DP has moved into more political areas including expectations of upper management,[1,4,7,8] the use of IS as a tool of the organisation,[9,14] and organisational power.[12,13,15] What most of these more sophisticated approaches to DP 'success' seem to miss is the flair and excitement of the experience of success. This emotional experience is now almost exclusively found in the successes of growing high-technology firms, particularly those that produce software for sale. And fostering the experience of success is in the province of human resource management rather than that of technology.

Most of the studies of success in high technology concentrate on technical or market factors, to the exclusion of human resource considerations. That is, the primary attribute of these companies that brings success seems to be the control of a product that appears at the right time, directed towards the right customers, acting in a marketing vacuum. These studies generally indicate that success would have been failure if attempted earlier, packaged differently, or in the face of later competition. An exception to this focus is one on the attributes of entrepreneurs in the software business.[17] This study shows that a multidisciplinary entrepreneurial team with software marketing experience is a better and more successful business group. History provides good examples of how relatively simple products (Lotus 1-2-3®, dBase II and III®, and WordStar®) captured the marketplace, built product identity and identification, and continue to be popular in the face of enormous competition. However, almost nothing has been written in the academic literature about the role of human resources management in fostering success.

This paper reports on survey research conducted in six high-technology organisations and three data-processing groups in England. We discovered that, to a great extent, success at a point in time derived from three general characteristics of the staff and supervision in areas of culture and skill management, market awareness, and product management.

Our small sample cuts across the range of high-tech firms from 'baby' firms employing seven persons through quite large ones employing several hundred in technical positions. Clearly six firms is not six hundred, but as case examples they provide us with a range of material for discussion and comparison. They share common traits despite the very real differences in their marketplaces, internal structures, and origins. It is these traits that we wish to focus on, because they form the basis for whatever success – and the success in several cases may only be temporary – the firms are experiencing. Most important, the traits are human resource characteristics (hiring practices, market awareness programmes, and supervision) only loosely related to either the technology or the personalities of the entrepreneurs.

These traits may also describe aspects of departments providing technological innovation within larger firms. Because of this, lessons learned from the study of human resource contributions to the success of small, high-technology firms may equally be applied to an intra-organizational context. In short, what we know about a commercial software developer can help us increase the value of the products of a data processing department.

2. THE FIRMS

We investigated six English high-technology firms, three in the Cambridge 'silicon fen' area and three in Nottingham. We also interviewed in three data-processing departments to see if ideas learned in the other six would apply. Business details on the firms appear in Table 1. As can be seen, the range of business type, size, age, and revenue is large, although most of the high-tech firms specialise in CAD/CAM or mapping software/hardware systems. As is common in the Cambridge area, many of the firms trace their financial, intellectual, and physical origins back to the University's laboratories. These Cambridge firms also seem to share the fate of American start-ups – acquisition by others: hardware suppliers, competitors, or others.

A firm was considered successful if (1) it has experienced at least 50% growth in staff over the past four years, (2) it is profitable, and (3) it brings out a new product that is accepted in the marketplace every year. Growth is the hallmark of a successful firm and the ability to attract and retain staff reflects employees' perceptions of that success. Profitability is necessary to fund expansion and innovation. Innovation, represented by continued growth in product diversity, demonstrates an underlying vitality of ideas and the ability to meet challenges (*a*) from the environment and (*b*) from staff wishing to express their creativity. As we shall see each of these characteristics contains a seed of decreased success as growth brings human resource challenges, profitability makes the firm ripe for takeover, and innovation builds a burden of maintenance.

In series of hour-and-a-half semi-structured interviews, we spoke with the heads of software development, managers, supervisors and project leaders in each firm. We asked questions about human resource management, product development, marketing, innovation, departmental organisation, planning, and interaction with others in the firm.

Among the software firms, one markets its own hardware, while several develop and sell software to be run on a variety of minicomputers. One not only sells software systems, but markets several data services. A very small firm has developed a single modelling product that runs on a micro-computer and not unexpectedly markets consulting services in the use of its product. One of these firms is just starting in business; it has not completed developing its first-release product, intended for marketing only in the U.S. Another firm has developed real-time animation systems for industrial design and markets extensively in the U.K.

These firms average 6 years of experience as independent concerns. Their sizes varied considerably – the mean of 250 is not characteristic of any firm. A rule of thumb of one clerical/administrative employee for each two technical employees seems to have held rather closely for all these firms. Annual revenues varied, too; the mean is 18 million pounds, inflated considerably by two very large firms. The income of one firm is effectively zero, since it has not begun selling its product yet. Including this firm, the average moves down to about £15 million. The average contribution per employee to revenue ranges widely between £27000 and £167000 per annum. The mean is £72800 per annum per employee excluding the firm not yet making sales; it is £60000 per annum including that firm. Employing the rule of thumb above, the mean contribution to revenue per technical employee would be about £90000 per annum. Although salary figures for the individual firms were not available, using a figure of about £20000 per employee gives about a 4-to-1 ratio; every pound invested in technical employees provides about £45 of revenue (out of which to pay the bills including the salaries of non-technical employees). A director of one of the more profitable firms stated that contributions of £25000 per annum per employee were the minimum acceptable and that £35000 was required for profitability, with £45000 considered by him an indicator of success.

The data-processing sites include a local government, a large utility and a smaller, private-sector site. Each of

Table 1. Data on Firms Contacted

Firm no.	Product(s)	Sales (£m pounds)	Employees	Age (yrs)	Firm type	No. of interviews	
	Software firms						
1	CAD/CAM	50	300	10	Division	3	
2	Modelling	0.75	21	4	Start-up	4	
3	GIS (hardware/ software/services)	4	150	10	Merged	7	
4	Design	1.5	23	4	Private sector	2	
5	Graphics	0*	7	1	Start-up	1	
6	Commercial	33.5	1000	7†	Spin-off	8	
	Means (sum)	18.0‡	250	6.0		(25)	
	Data-processing departments	Budget					% Revenue
1	DP services	6	120	20+	Government	7	1.0
2	DP services	1.1	14	20	Private sector	3	1.0
3	DP services	16	350	20	Utility	5	1.6
	Means (sum)	7.7	133.3	20		(15)	1.2
	Overall means (sum)	14.1	211	10.7		(40)	

* No revenue yet; product not yet marketed.
† As a DP department, it goes back 20 years.
‡ Does not include firm not yet making sales.

these departments provides traditional data processing services to others in their firms; none sells services outside, however. By coincidence, each data processing department is about 20 years of age and their budgets range around 1–1.5% of the firm's gross receipts, consistent with published results.[3] Employee 'productivity' is difficult to compute in real terms for departments which cannot show a profit, but the mean contribution per employee to the budgets of these departments is around £50000.

This large variety of products, organisations and sizes ensures that commonalities are not due to chance, unseen correlated characteristics other than the 'high-tech' business or data-processing realities. In fact, the initial hypotheses of the research concerned the effects of rapid growth on internal organisation and human resource practices. Clearly rapid growth is a phenomenon in itself and has attendent human resource implications.[10,11] But in these high-technology firms, growth was neither monotonic nor independent of other factors, such as changes in the marketplace, technology, or organisational plans. One firm has recently experienced 25% annual growth rates after two-and-a-half years without growth. Another has been relatively stable for over a year, showing little turnover or expansion – staffing and product line have remained static. One firm has doubled annually for three years. The smallest firm has grown the most (400% in two years) but further growth seems unlikely over the short run (and it has fewer than ten employees now).

In addition, the business situation of the firms differs considerably, too. The largest firm has been slowly spun off for seven years from its founding company for which it was the data processing department. The most dynamic of the firms was acquired early in its history by another firm which, while far from failure, was not very successful in its own field and therefore left the acquired firm alone. Now however, the owner firm itself has been bought by a large American computer manufacturer and changes are in the wind. Another of our firms merged with a firm of a different nature and this merger is going to have a profound effect on product lines, financing, and ultimately management. One firm, a microcomputer manufacturer, was bought out by another manufacturer when it experienced a cash problem brought on, ironically, by its own successful marketing. Finally, three of the firms have at least partial origins in a university laboratory and have university professors as directors because of their origins.

Thus, commonalities in human resource management that seem to lead to success come about less from size, growth, specific product, geographical location, organisation, business position, or most characteristics of past history. It is in the unique interplay of human resource management and high-technology innovation that such successful tactics seem to arise.

3. SUCCESS CHARACTERISTICS

Three characteristics are common across the six high-technology firms (and seem absent in the data-processing departments). These are the following.

1. *Culture.* The successful software houses actively recruit individuals who already fit into the culture there. Not only are they almost uniformly self-motivated and self-directed (consistent with Couger and Zawacki[2]), but they are excited by being around others of the same description.

2. *Market-awareness.* Key employees in the successful firms are either former users of the systems being developed or deal on a daily basis with users of these systems; in very few cases have individuals been brought in with exclusively 'programming' backgrounds.

3. *Product management.* Supervisors are seen uniformly as product managers. Their rewards are more or less directly tied to the performance of the products they manage; users know that these supervisors are responsible for the products as well as the people developing them.

Workers arrive on site already pre-attuned to the culture they will find in the organisation. They do not have to fight a bias against machines, data, systems, or systems people. They are recruited by individuals who had a hand in setting the company up and are not hired if they demonstrate values which are contrary to the firm. However – and this is quite important – the recruiting process is not arbitrary; the recruiter also shares many of the values of the individual applicants. Because both recruiter and applicant are product-aware (from their backgrounds as users) rather than merely technology-aware (i.e. programmers), their communication can be about the product, and confidence is more easily established.

An interesting and logically necessary departure from this principle was observed at one firm. It sells both software and hardware for the same purpose. There has been an increasingly noticeable gap between the cultures of the two development arms, culminating in a recent divisionalisation (taking place simultaneously with acquisition by a hardware vendor). Prior to this, supervisors from each group expressed the opinions that (1) the real 'business' now or in the future lay in their own products and (2) the managers in the other group not only failed to recognise this, but they also blocked attempts to move in the correct direction. Software supervisors saw their product as essential to both lines; hardware supervisors saw the development of machines as the true business of the firm – software was seen as an upstart and a diversion of resources. This latter sentiment was sufficient for the hardware group to hire its own programmers to produce firmware; the software group was actively pursuing products it claimed would compete with the hardware products on a functional basis!

In effect, firms, while searching out a variety of technical skills, are looking for cultural clones. And the clones, where not available, have to be built. Evidence comes from a variety of sources in our interviews. In one firm, the same individual hired all 200 technical people. In another, technical employees are first hired as contractors to the firm, which is itself a contractor to its customers – each employee is a consultant and each consultant has some time to spend on customer sites. Only one of the firms has any internal promotional literature. The largest, with over 1000 employees, has just started a company newsletter. This firm takes its employees almost exclusively from a government-sponsored training programme that 're-careers' individuals into computer-related jobs. The others hire computer scientists or engineers, as required, from the universities they have established special relationships with.

These hiring and induction practices should be contrasted with the governmental data-processing department which is undergoing reorganisation. While many of its employees have previous governmental computing experience, the feel of the firm is one of lack of cohesion, an inability to develop a coherent set of products, and general lack of direction at all levels. Where the DP department can be called 'successful', it has managed to keep the users at bay by keeping them relatively uninvolved and ignorant – a classical avoidance tactic. The utility DP department has its own training scheme; it is quite successful, and managers there expressed concern that salaries would not be sufficiently high to prevent other DP groups from raiding their 'school'. This group does not hire computer scientists. They are looking for more general skills, especially communication skills and the ability to work in groups as projects take on a more user-oriented outlook. This DP department has 1500 active, registered end users trained and supported through an information centre.

Also, the level of excitement in the firms that are successful is very high. Even the very largest, where working conditions are relatively unfavourable (inadequate space, poor lighting, lack of company facilities, relatively 'ancient' computers), morale is extremely high. Short deadlines contribute to the feeling of accomplishment rather than a feeling of being oppressed. Growth has been so high that physical movement from floor to floor and building to building is ingrained in the culture. In fact, every one of the high-tech firms has recently made a major physical move (in several cases to buildings designed for them) and is contemplating other moves in the near future. Garden[6] pointed out that excitement is a separate and independent 'stage' of arousal in software firms, one which draws performance from a worker beyond that obtained from 'motivation'. Even in relatively large groups excitement levels are evident and high, although clearly these levels of arousal could, in other circumstances (such as product failure) be labelled differently and give rise to an atmosphere of frustration and anger.

In most firms, too, the average age of employees is quite low. High-tech firms tend to hire younger people and those connected with universities draw upon recent graduates for their labour source. Managers and supervisors in these firms are also very young. Few are over 35, whereas the more mature data-processing departments are characterized by significantly older managers. Because small firms age as rapidly as their employees when turnover is slight, it is likely that the managerial culture of the original founders becomes the established culture at all levels. As firms mature, turnover increases and management's culture becomes increasingly isolated from the younger people who filter through staff and supervisory positions at lower levels. Thus small firms have less of an overt need to build culture, although obviously at some size and at some level of activity and turnover, the need will arise. This was apparent in all three data processing departments and in the largest (and oldest) of the high-tech firms; at these four sites, managers were significantly older than their employees and at the high-tech firm, this was a source of significant concern for future innovation. In each case, a promote-from-within policy had led to a management group with lengthy service records supervising relatively recently recruited staff and trying to build a 'culture' when it could not be purchased.

Most of the workers and all of the supervisors have a high level of awareness of the marketplace for their products. There are two ways this is achieved. The first comes from recruiting practices that stress hiring software developers who have experience in the application field. This generally implies bringing in engineers for the CAD/CAM companies, consultants for the small firm making modelling software, and mature individuals who are attuned to business needs for the large credit-bureau software firm. In no case is someone hired just because that person is a cracker-jack programmer. This is certainly not the strategy in the three data-processing departments. While many of the programmers and analysts have experience with specific types of user (for example, several of the employees in the county council DP shop have worked for several other local authorities), their experience as users is remarkably limited. In one of the DP shops, rotation to user groups has just begun on an individual basis as part of employee development initiatives – but curiously not as a way of building this sort of strength into the DP department as a matter of policy.

The second way of building market awareness is to put workers at all levels in close contact with actual or potential customers. Programmers deal with existing customers for specialised, single-customer software on a daily basis – programmers simply do not come to blows with users over technical matters or ego-related issues because (1) they are not merely programmers and (2) they know that their jobs depend on the software's correct specification and employment. In the data-processing departments, supervisors are far more careful in lubricating the interactions of programmers (and analyst-programmers) with internal customers.

The benefit of market awareness is that innovation is not usually led from an independent marketing group. Instead, in the majority of cases, new ideas arise from the technical ranks. When asked where new products come from, the middle managers we spoke with said universally 'From our technical people. Who else would know what works?' This significantly reduces new product development time and decreases the risk that new products will work incorrectly or inappropriately. In the case of the three larger companies, there are marketing specialists who sell existing software and services. Their task is to channel customers, ideas back to the technical people. But it is the technical people who make the important decisions. And, in the very largest firm, the marketing specialists are former technical managers.

Why should staff software work so well in these firms when this attitude has generally led DP departments and their customers to near warfare? Again, the reason lies in market awareness and the unique way in which the successful firms implement innovation. In the most successful of these firms, innovation means re-configuration of existing products to meet a newly determined market need discovered from customer comments. This implies that innovation seldom moves far from existing products. For example, one CAD/CAM firm discovered that its design software could be easily re-configured as a mapping workbench. And because it was already creating flow-design aids, a relatively simple reorganisation of software led to plant-design product – this may

well lead to an integrated plant-planning, design, and implementation product, too.

Rapid re-configuration of the product into technical clones with different front-ends has led, in many cases, to a plethora of look-alike products sold to different markets. This, in turn, has led to a situation in which multiple versions of essentially the same product have to be maintained and, in most cases, to the technical necessity of maintaining a library of drivers and device-dependent software to support the reconfiguration. None of the CAD/CAM software firms has developed documentation systems, however, to support this work. 'We'll have to get to it, sometime,' is the typical response, 'but we don't have time now.' Generally, 'specs' are kept in programmers' heads. Although most firms have developed a systems group to maintain the kernel of support software, reconfiguration has tended to depend on individual managers' initiatives, aided by the relative smallness of the firms and the familiarity managers have with the product and the marketplace.

The re-usability situation is in sharp contrast to that of the DP departments where there is little chance of re-using modules. This stems from two inhibitions: (1) most technical employees work for specific internal customers for long periods of time and have little opportunity formally or informally to share module information with others and (2) most technical environments are unstructured for re-usability even were individual initiatives begun. The latter reason underlies the lack of support for re-usability and rapid re-configuration. A COBOL or FORTRAN shop even with the kind of documentation needed to support maintenance is not set up to encourage generality in initial module generation; none of the shops even had committees to foster module libraries, module sharing, or an audit of existing modules. Pressures to meet deadlines and the lack of demonstrated advantage in these kinds of shops stand in the way of building re-usability into the everyday worklife of programmers in these DP shops.

A step in the right direction on the part of one of the DP departments was the creation of a 'Productivity Centre', charged with acquiring productivity-enhancing tools such as word processors and analysis tools. This centre is literally central to all project management and maintains voluminous documentation on outstanding projects. It is also planning to bring in a code generator and work towards integrating that with a relational database package. The productivity centre has been in existence for less than a year, and while it doesn't aim specifically at re-usability, its potential as a clearing house for modules at a project-initiation level is evident. However, currently all re-usability responsibility rests, as expected, with the data administrator. The productivity centre manager sees avoidence of duplication rather than re-use as his mandate.

In the high-tech firms, so long as innovation is narrowly restricted to re-configuration of existing software elements and the technical infrastructure for re-use is available, successful new product development is much less risky. In the few cases where innovation was not successful, one facet or more of this tactic was missing or functioned incorrectly. Failures are clearly attributable to either (1) market-driven innovation into new areas in which technical people had little user experience, (2) lack of ability to re-use existing software components because there was no database of such parts or because technical re-use was impossible, or (3) lack of communication among sub-departments, often organised into non-interacting divisions that made the development of re-usable software difficult and practically unthinkable. Curiously, the DP departments had much more formalised documentation systems (an essential element in re-usability), but seemingly lacked the time and motivation to put effort into formal technical or human communication-based systems for re-use; the high-tech firms had the systems already in place and seemingly have dispensed with most of the documentation. This apparent trade-off bears further investigation.

Finally, supervisors are product managers as well as managers of people. They know that their rewards are tied to successful implementation. Their daily interaction with users has convinced them that users' needs can be met with a re-configuration of existing software and staff skills. Their more catholic backgrounds outside the narrow realm of software development provide a proven confidence in their managerial skills – and the intuitions of their employees. Again, in most cases supervisors are not merely promoted programmers, but proven innovators who can hold a team together.

In contrast to the high-tech firms' managers – most of whom had come up with the firm through a short period of rapid growth and high challenge – the supervisors in the DP departments are almost all promoted programmers. Their managerial expertise has been acquired in the field during a period of relative stability; their lack of formal management training has not been held against them because, in part, their managerial challenges have been controlled for by the sheer bulk of technical work to be done. Stated another way, failure as a manager in a DP department requires a great deal higher profile than it would in a small high-tech firm to be noticed, since managerial failures in the small firms mean delayed or defective products and lost sales. While few of the high-tech supervisors and managers have managerial training, they showed promise when they were hired because they would be able to meet supervisory challenges. In one of the firms, all the supervisors are taking an Open University course in management as a job requirement, consuming 6–12 hours of their off-the-job time in addition to very high job demands. None of the managers in the DP departments is working on a managerial degree or diploma.

Most firms have had uneven growth patterns that at some points in their histories encouraged promotion of just about anyone with a few months' experience. But they insist that they now hire only individuals who can supervise later in their careers. Few hire supervisors directly, by the way; promotion from within is generally the rule as in DP departments except in times of extreme growth. One firm complained that almost everyone who could be promoted had already been promoted; but a combination of turnover and growth meant that fully 25% of their employees this year had not been in the firm last year.

4. THE DARK SIDE OF SUCCESS

There is a reverse side to success. Success may come in the form of too much of a good thing. Several of the firms we looked at are victims of too-rapid growth and too

many successes, without learning where they are weak from a few failures. On the other hand, clearly it is impossible to follow a winning strategy for ever. The example just cited is one – the firm has no one left with the proper experience to promote, yet the number of products which need managing is now more than the number of upper-level technical managers available.

In addition, one of the major contributors to success in these firms is success itself, in a positive feedback cycle that may create a false sense of invincibility. For example, a firm may innovate a first product which sells well and it may build in re-usability and re-configurability so that a number of related second-generation products can be produced. In many cases, this first product comes out of a university laboratory, guaranteeing the uniqueness of the product into a market which includes universities. Financial success is more or less guaranteed by the technical success of the products regardless of production strategy or human resource management tactic. That is, at an early stage it is irrelevant what managers at any level do so long as the product gets out, because almost everyone is willing to buy it.

However, customers do learn and may become more sophisticated than the existing employees, many of whom become comfortable within the firm and are promoted to product-shaping positions. Obviously, at some stage the firm cannot continue hiring outsiders with current user experience in sufficient proportion. Internal product awareness, regardless of company ploy, will be diminished. The volume of existing software to be maintained implies an increasing burden of old work of a less exciting nature competing with new work for existing resources. Making products obsolete to reduce this load may harm product identification among existing users.

And there is always the threat of a better product coming along, if only by chance. The firms we looked at have not planned future moves against competitors to protect market share. A firm examined in a related study[5] failed and has recently folded after acquisition because it missed a production deadline, but also because the competition caught up. Only the largest firm has a very sophisticated marketing department. It is clear that none of the firms we looked at would do well in the face of concerted effort by competitors. That they got there first with a very good product seems to have ensured that someone else will eventually take it away.

That has meant in almost all cases that acquisition or merger has been necessary. And this seems to be the fate of most software firms in general. When the initial steam runs out and really new third-generation products are not forthcoming, the existing customer base resembles a cash source to another firm with related products and the smaller firm is acquired. In three cases this has happened. The exceptions are the 'baby' firm, which is still developing its first product, the start-up modelling software firms and the very large spin-off which has only just left the fold of its initial founding company and is still innovating products at a rapid rate within its business area. The other firms have either merged with companies doing semi-related business or have been acquired by suppliers.

5. STRATEGIES FOR SUCCESS

These firms are implicitly pursuing ten strategies for success, based on the three principles mentioned earlier. These principles are the following.

(A) Managing the Culture and the Skill Pool

1. *Hiring workers who share the firm's values and those of its customers.* These workers will become more productive sooner with less effort and will require less direction. They will, in fact, give product direction based on solid intuitions about potential products.

2. *Rotating staff through many positions.* By rotating staff through products, into project leadership, librarianship, customer contact and contracting, each employee will obtain an understanding of what constraints the others work under and reduce inter-departmental conflict. It will also build communication channels to insure re-usability of experience and software.

3. *Watching out for growth problems and an increase in maintenance work.* Growth implies new software, a relative average de-skilling of the staff as new workers are hired, and strains on resources. Since training is expensive and time-consuming, it may be cheaper to hire skills. A development methodology that stresses layered upgrading of products, prototyping, information-hiding, and other re-usability techniques lowers the relative burden of maintenance work. Constant attention to the costs of maintenance and its implications will increase the chances that older software can help derive new software without the costs of maintaining the older versions alongside the new.

(B) Build market awareness

4. *Hiring customers on a continuing basis.* Customers for software tend to have user needs primarily in mind. Technical innovation can be handled by technical hotshots hired as needed, but much of the implementation team can be staffed by customers brought in on secondment or rotation.

5.. *Training software developers in customer applications.* Those developers who are not the ultimate customers have to develop a product sense. Short of rotating them to customers – feasible in some cases but rarely possible given the shortage of staff and their lack of immediately applicable customer skills – it is possible to bring in customers to train developers in the customer's major applications, current and desired.

6. *Encouraging interaction with customers.* Bringing customers through and letting them watch the work ensures that each technical employee knows how to talk to customers about products without letting the cat out of the bag. Several of the other tactics ensure this colloquial ability. Then each employee becomes a consultant and a source of product innovation. Marketing can then concentrate on sales.

7. *Making rewards contingent on product success at each level.* Reward employees for successful product innovations. Reward managers and supervisors for producing successful products. Reward executives for building successful product lines. Study successes and failures and maintain a database of product-development experience and evaluation. Keep this data over a long term – a short-term success may only divert from long-

term planning. A critical ingredient is learning how to measure success and apportion reward in team environments. If A re-uses B's software to build a product, under C's supervision, who gets the reward?

(C) Shift to product management

8. *Building management systems that reward good product management.* The answer to the question above comes from a management system oriented not towards projects but products. Each developer, supervisor and manager shares in the rewards of profit and in the costs of development. A royalty system is not infeasible and IPSEs (Integrated Project [here, read 'Product'] Support Environment) can be built or purchased to support this philosophy. Whereas a project dissolves at product release (often well in advance) and project management is considered over, product management lasts from inception through re-use, especially if rewards come from sales. Defining subproducts which improve over time and enhance the main product should be rewarded; promoting product lines rather than one-shot efforts for custom software is likewise rewardable. Employees need access, however, to precise feedback on performance, acceptability, and changing customer needs.

9. *Building or acquiring product-management systems.* The successful firms employ methodologies that are product-based, object (product)-oriented rather than task oriented, and aimed at increasing developer involvement in the product as a whole rather than merely in time-money relationships. These systems will naturally stress functionality, maintainability, and re-use over cost, elegance, and time. Product costs may be higher (Sommerville and Morrison),[16] but management of people in this environment is easier when people are responsible for objects (software) within a coherent whole (products). Supervision techniques will have to shift, too, toward product evaluation. Supervisors will have to make judgements of product viability over its expected lifetime rather than completion of software relative to specs.

10. *Encouraging development methods that allow re-usability and rapid prototyping while accumulating learning in a form developers can use.* These methods stress easy access to a database of developer experience and software that evaluates its own usability for specific goals. This seems to be the design philosophy behind the Ada Programming Support Environment (APSE). Without such a facility, developers re-invent wheels, re-experience traumas and have to spend a lot of time writing documentation. By the way, prototyping without memory is worse than continuous re-development, since it is impossible to insure the lack of time-consuming and product-debilitating loops or learning from mistakes. The experience of the 'prototypee' (who may be a seconded customer) can be more important than specific design decisions of developers. It's not clear that such methods exist, however; home-grown may be the wave of the near future.

The same ten tactics are appropriate for software developers, although the cost is high and systematic management techniques are as yet undeveloped. Clearly a great deal of work has to be done before automated techniques catch up with the needs. Until then success cannot be assured and informality rules the day. Our most successful firms have almost no formal project management.

6. LESSONS FOR DATA PROCESSING

These ten tactics can be applied to data-processing departments within larger firms – some are in fact easier since the need for security may not be as large within the firm's boundaries. In most firms, user-developed applications are increasing rapidly in number, size and sophistication, often with active DP department encouragement. Users, rather than developers, know how to use and have extensive experience with 4GLs. It may be that those still in DP are those left behind in the 'user revolution'. Clearly, however, the large, real-time, distributed, security-conscious systems of the 1990s will require experienced systems personnel in increased numbers and will pose increased levels of management challenge regardless (and perhaps because) of technological developments, particularly as DP departments are called to account.

The translation of the ten tactics to a DP-appropriate wording is relatively straightforward.

1. Hiring workers who share your (the firm's) values. These may be users who are changing careers.
2. Rotating staff through as many positions as are available rapidly.
3. Watching out for growth problems (especially in maintenance, relative ageing of supervisory staff, and resource conflicts between production and development).
4. Rotating users on to development teams.
5. Training development staff in the firm's applications; rotating them through positions in other divisions to learn them.
6. Encouraging the development staff to become consultants within the firm on information systems solutions.
7. Making rewards contingent on the success of products and the establishment of product lines.
8. Building management systems that reward good product management.
9. Building product- rather than project-management systems, in consultation with users and upper management.
10. Encouraging development methods that stress re-usability and prototyping.

7. CONCLUSION

Human resource management strongly influences success in small high-technology firms that produce software products. As a result, ten tactics with a human-resource component or implication appear, based on our research in these firms. The bases of these strategies are management of the culture, market awareness and product management. Technology can assist, but the techniques themselves require attention to the human resource component for successful implementation. Each of the strategies is appropriate in a data-processing environment, too, with appropriate rewards approximating to 'success' available there, too.

REFERENCES

1. J. Brancheau and J. Wetherbe, Key issues in information systems Management. *Management Information Systems Quarterly*, **11** (1), 23–45 (1987).
2. J. D. Couger and R. Zawacki, *Motivating and Managing Computer Personnel*. New York, Wiley (1980).
3. W. DeLeon, Firm size and characteristics of computer use. *Management Information Systems Quarterly*, **5** (4), 65–77 (1981).
4. G. Dickman, R. Leitheiser, J. Wetherbe and M. Nechis, Key information systems issues for the 1980s. *Management Information Systems Quarterly*, **8** (3), 135–159 (1984).
5. Vivien Fleck and Elizabeth Garnsey, *Strategy and Internal Constraints in a High-Technology Firm: The Management of Growth at Acorn Computers*. Research Paper 2/87, Management Studies Group, Engineering Department, University of Cambridge (1987).
6. Anna-Maria Garden, *Job Excitement, Motivation and Satisfaction of Software Professionals*. Working paper, High Tech Management Unit, London Business School (1988).
7. T. Guimares and Y. Gupta, Measuring top management satisfaction with the MIS department. *Omega* **16** (1), 17–24 (1988).
8. C. Hartog and M. Herbert, 1985 opinion survey of MIS managers. *MIS Quarterly* **10** (12), 351–361 (1986).
9. B. Ives and G. Learmouth, The information system as a competitive weapon. *Communications of the ACM*, **27** (12), 1193–1201 (1984).
10. J. Kotter and V. Sathe, Problems of human resource management in rapidly growing companies. *California Management Review*, **22** (2), (1979).
11. P. Licker, The data processing department as a business: British and North American approaches. Proceedings, Conference on Managers and Managing in Britain and Canada, British Association for Canadian Studies, Birmingham, 16 March 1988.
12. H. Lucas, Organizational power and the information services department. *Communications of the ACM*, **27** (1), 58–65 (1984).
13. L. Markus and N. Bjorn-Andersen, Power over users: its exercise by system professionals. *Communications of the ACM*, **30** (6), 498–504 (1987).
14. M. Porter and V. Millar, How information gives you competitive advantage. *Harvard Business Review*, **63** (4), 179–185 (1985).
15. C. Saunders and R. Scammell, Organizational power and the information services department: reexamination. *Communications of the ACM*, **29** (2), 142–147 (1986).
16. I. Sommerville and R. Morrison, *Software Development with ADA*. Wokingham, England, Addison-Wesley (1987).
17. R. D. Teach, F. A. Tarpley, Jr and R. G. Schwartz, *Software Venture Teams*. Proceedings, Babson Entrepreneurship Conference, Babson College, Wellesley, MA (1986).

Announcement

9–11 JULY 1990

BNCOD-8, the Eighth British National Conference on Databases, the University of York, England

Call for Papers

The Eighth British National Conference on Databases (BNCOD-8) is to be held at the University of York from 9 to 11 July 1990.

Themes

The Programme Committee will welcome all papers within the database systems area. They may address any topic relevant to the design, implementation, and application of database technology. Papers are particularly sought in new database application areas such as Engineering Data Management, Office Information Systems, Software Development Environments, and Geographical, Spatial and Image Systems. Other topics of interest include:

- Object-oriented database systems
- End-user interfaces to database systems
- Data and knowledge bases
- Extensible database systems
- Real-time database systems
- Design methodologies and CASE tools

Authors from all countries – from both industry and academia – are invited to submit papers. These should normally be about 5,000 words in length, and should be with the conference organisers before **Monday, 8 January 1990.**

Important dates

Receipt of full papers 8 January 1990
Notification of acceptance 23 February 1990
Camera-ready copy due 30 March 1990

Please send all papers to:
Dr Peter Hitchcock, BNCOD-8, Department of Computer Science, University of York, Heslington, York YO1 5DD, England

For any queries contact **Peter Hitchcock** (tel: 0904-432745; email: ph@uk.ac.york.minster), **Alan Brown** (tel: 0904-432746; email: alanb@uk.ac.york.minster), or **Rebecca Wise** (tel: 0904-432782; email: rebecca@uk.ac.york.minster).