SELLING PACKAGED SOFTWARE: AN ETHICAL ANALYSIS

Adam, Alison, IS, Organisation and Society Research Centre, University of Salford, Ashworth Building, SALFORD, M5 4WT, UK, a.adam@salford.ac.uk

Light, Ben, IS, Organisation and Society Research Centre, University of Salford, Ashworth Building, SALFORD, M5 4WT, UK, b.light@salford.ac.uk

Abstract

Within the IS literature there is little discussion on selling software products in general and especially from the ethical point of view. Similarly, within computer ethics, although there is much interest in professionalism and professional codes, in terms of accountability and responsibility, the spotlight tends to play on safety-critical or life-critical systems, rather than on software oriented towards the more mundane aspects of work organisation and society. With this research gap in mind, we offer a preliminary ethical investigation of packaged software selling. Through an analysis of the features of competition in the market, the global nature of the packaged software market and the nature of product development we conclude that professionalism, as usually conceived in computer ethics, does not apply particularly well to software vendors. Thus, we call for a broader definition of professionalism to include software vendors, not just software developers. Moreover, we acknowledge that with intermediaries, such as implementation consultants, involved in software selling, and the packaged software industry more generally, there are even more "hands" involved. Therefore, we contend that this is an area worthy of further study, which is likely to yield more on the question of accountability.

Keywords: Packaged Software, Software Vendors, Ethics, Accountability.

1 INTRODUCTION

Commentators have noted that IS research papers rarely remark explicitly on ethical issues (Adam and Bell, 2003). Similarly there is minimal intersection between computer ethics writing and mainstream IS. We argue that many topics, of much potential interest to computer ethics and IS, slip down the cracks and therefore remain under or unexplored. One such topic involves the ethical issues surrounding the activities of software vendors, particularly in relation to selling software packages. From the IS literature there is little discussion on selling software products in general and especially from the ethical point of view. This is unexpected as the failure of software projects is still evident, with some of this arguably linked with vendor capabilities and highly embroidered sales pitches rather than problems with the 'technology' per se (Lynch 1984, Bicknell 1998, Martin 1998, Bingi et al. 1999, Sumner 2000). Similarly, within computer ethics, although there is much interest in professionalism and professional codes (Gotterbarn 1997), and some concern with accountability and responsibility, the spotlight tends to play on safety-critical or life-critical systems (Nissenbaum 1995), rather than on software oriented towards the more mundane aspects of work organisation and society. With this research gap in mind, we offer a preliminary ethical investigation of package software selling. In this paper, we are limited in the ground we can cover and therefore we focus upon the work organisation and larger packages (such as that for workflow or enterprise support for example) although our work clearly has resonance for domestic contexts and other forms packaged software. Indeed, it is our intention to extend this study to incorporate these.

The rise of the packaged software industry over the last two decades has been considerable. In 1984, a report suggested that sales would reach US$ 10 billion in that year (Business Week 1984) and by 2001, it was estimated that world packaged software markets totalled US$ 196 billion (OECD 2002). Even some in-house IS departments have become packaged software companies, themselves, e.g. 'Software Corp' (Dube 1998). The software service industry has also grown extensively in recent years. Forrester Research (1998) reports that this segment has increased by 16 per cent per annum for more than a decade and accounted for US$ 180 billion in 1997. Significantly, in contrast to custom development, we can see that packaged software is primarily thought of in market terms and this has various ethical implications (although we acknowledge custom development can be commercial too). Our analysis starts with the topic of professionalism. We conclude that professionalism, as usually conceived in computer ethics, does not apply particularly well to software vendors, therefore we call for a broader definition of professionalism to include the activities of software vendors, who generally develop *and* sell software. The problem with discussions of professionalism in IS, from the computer ethics domain, is that they focus on software development, without clearly recognizing that the selling activity adds an extra ethical dimension requiring explicit exploration.

Based upon this discussion, we consider a number of aspects of software vendors' activities in terms of the ethical issues involved. This includes a review of the features of competition in the market, the global nature of the packaged software market and the nature of product development. Hence we are offering a re-interpretation of these in ethical terms, a re-interpretation which seems surprisingly absent from the IS literature on software vendors. We focus on the topic of accountability as identified by Nissenbaum (1995), and, in particular, seek to extend it beyond the usual computer ethics focus on safety-critical systems towards accountability for selling packaged software.

2 PROFESSIONALISM AND THE PROBLEM OF ACCOUNTABILITY

Professionalism is one of the major topics in contemporary computer ethics. There are several strands involved, including the development of professional bodies and codes of conduct which can be used to regulate members of the profession, more specifically members of professional computing bodies. However, the development of professional ethical codes is as much related to the IS community's attempts at becoming a distinct, recognisable profession, as it is to do with regulating the activities of its members (Gotterbarn 1997). Yet the IS profession hardly fits any of the standard indicators for

professional groups such as standard education/training and compulsory membership of a professional body (Johnson 2001). IS professionals come from a wide variety of educational backgrounds. Indeed, Bill Gates is famous for having dropped out of Harvard University. It is not necessary to be a member of such bodies as the ACM or BCS to write and sell software. As Tavani (2004, p. 93) notes, professional codes of ethics have no "teeth." This means that it is very hard to see how IS professionals could be subject to effective regulation and this is a perennial problem in the world of computer ethics (Gotterbarn 1997, Johnson 2001). Ladd (1995) argues that there is often confusion as to macroethical issues - those that confront the whole profession and microethical issues - those that may confront an individual.

However, we argue that there are several further issues at stake, which have seldom been identified, let alone addressed. Firstly the development of structured techniques in producing software and IS can be seen as an attempt at regulating the work of, perhaps potentially unruly, software producers. Ethical codes are part of the moral regulation of software/IS professionals. Regulating work and regulating moral concerns are more connected than is often recognised. Using structured techniques to produce good IS and good programs is also part of being a good IS professional in the moral sense. Structured development techniques and moral codes are both designed to impact the developers of systems, although with the caveats above, it is not clear that they must necessarily be related. Indeed Gotterbarn (2000) separates out codes of ethics as aspirational, codes of conduct as addressing attitude and behaviour while codes of practice address operational activities. Yet one of the important shortcomings, to date, in professional computer ethics is that it has not focused on software vendors as a distinct and possibly special professional group. This means that the good practice advocated by systems development methodologies is relevant to developing systems but is not particularly relevant to selling. There is also the question that codes of ethics do not generally address the question of selling software and the ethical concerns involved. However, we should note that the joint IEEE Computer Society and ACM Codes of Ethics for Software Engineers do, in part, address the question of the software product in Principle 3 which relates to software products: "Software engineers shall ensure that their products and related modifications meet the highest professional standards possible." Tavani (2004, p. 325) and where fifteen sub-principles relate to cost, methods, qualifications etc. Nevertheless, it is hard to see how such a code could possibly be enforced amongst software vendors.

Tavani (2004, pp.106-7) following (Nissenbaum 1995) has discussed moral responsibility and its relationship to legal liability and accountability. Although not specifically aimed at selling software this discussion bears some considerable relevance to the activities of software vendors. Liability is a legal concept where "strict liability" involves liability to compensate even though there is no faulty action, that is, no blame. Nissenbaum's (1995, p. 527) view of accountability is broader than responsibility or even liability – blame may or may not be involved but someone or some group or organisation is answerable: "there will be someone, or several people, to answer not only for malfunctions in life-critical systems that cause or risk grave injuries and cause infrastructure and large monetary losses, but even for the malfunctions that cause individual losses of time, convenience, and contentment."

Despite the way that we are increasingly reliant on computer systems, Nissenbaum believes that the concept of accountability has been increasingly undermined. Much of the barrier towards attributing accountability, she argues, lies in the "many hands" problem. This refers to the way that IS are developed in large, complex organisational settings where many hands contribute to the final system. Using the concept of accountability moves the spotlight away from the level of individuals, towards the group, in assigning moral responsibility. Furthermore, this recognises that accountability is not the same as legal liability which involves quite a narrow definition involving liability to compensate.

Thinking further about the concept of accountability we see how the pendulum seems to have swung in favour of software producers. Because ownership of software implies a set of rights, such as copyright, Nissenbaum (1995) believes it should also imply a set of responsibilities - including being liable. Thus, there is a paradox in that liability laws may protect the public against potential harm, but, at the same time, software vendors usually deny accountability for software errors. Software is often

sold "as is." Nissenbaum (1995, p. 534) argues, "the trend in the software industry is to demand maximal property protection while denying, to the extent possible, accountability."

Hence, based on the discussion above, we see the main ethical issues surrounding software vendors to be as follows. First, discussions of professionalism and the development of ethical codes has not been directed at the activities of software vendors and so do not cover such activities particularly well. Second, the question of accountability versus rights of software vendors needs to be addressed more robustly. These are the main ethical questions which we address in the discussions below.

3 COMPETING IN THE PACKAGED SOFTWARE INDUSTRY

For vendors, many of the benefits of selling packaged software derive largely from economies of scale. In order to attain these economies, products have to be standardised so they have broad appeal in the market (Robson 1997, Fan et al. 2000). Thus, the vendor need only develop the software once and can distribute the cost of development and ongoing maintenance over a large base (Gremillion 1982, Dube 1998, Butler 1999). Consequently, this means that the vendor has to determine how they will compete in this environment and within which market/s they wish to operate.

3.1 Market Segmentation

From a work organisation perspective, packaged software vendors may, for instance, target a mass market with an application type (e.g. Microsoft Windows), a particular customer organisation size (e.g. SAP versus Sage for accounting products), or sector (e.g. IBM's Corebank in the banking industry). Vendors may also choose to manage product delivery schedules and consumption patterns, for example by segmenting the market via the use of editions. Editions serve to differentiate among similar low and high-end products, particularly via differences in functionality and price. What this essentially means is that a vendor may produce multiple editions of their products for various markets e.g. domestic, business, small business, or sector specific, such as education. This segmentation activity may of course also be combined with targeting a particular market. That is, various editions may be made available with a market sector. This is done mainly to allow decisions to be made regarding tradeoffs of cost and functionality (Raghunathan 2000).

In respect of ethics, a way of reading these activities is that the vendor uses them as control mechanisms. As the vendor effectively determines who their market will be, how much they will pay and what they will receive, and when (if ever) they will receive it, the consumer organisation's palette of choice is limited. The pressure for profit further militates against the software industry becoming more accountable in the sense described by Nissenbaum (1995).

3.2 Time to Market Considerations

A number of studies have also indicated that time to market is of competitive importance (Carmel and Becker 1995, Carmel and Sawyer, 1998). It has even been argued, that, in this segment of the software industry, time-to-completion is the most important issue and that future competition will be based upon accelerated release cycles (Carmel 1995b). However, it has been further proposed by the same author that many packaged software companies have not fully recognised cycle time as a competitive concept and rather, that other pressures dominate such as features, performance, customer service, adherence to standards and price (Carmel 1995a). In order to improve cycle times, it is argued that traditional software development life cycle approaches do not work, and that new ones need to evolve (Carmel and Becker 1995, Dube 1998). However, purchase is often motivated by expectations of the direction of vendor products as much as by specific internal needs (Butler 1999, Sawyer 2001). That is, a consumer organisation will be interested in the development trajectory of a vendor.

Furthermore, the vendor's perceived strength and stability is also a significant consideration (Chau 1994). In one study was suggested that universities were not likely to make an investment in an ERP package unless they were convinced that the vendor was totally reliable (Oliver and Romm 2000). Thus, the quality of products and the direction in which they evolve is important as trust from a customer is reportedly often gained through vendor responsiveness and dependability (Gefen 2002). Therefore, for vendors to sell products it is not good enough just to have them to market first, they need to release products that are perceived to be of good quality too.

3.3 Marketing Software

A further part of the story appears to reside with the business of what the software vendor does, or does not, tell the purchaser. Benchmarking information is not usually published and independent reviewers may be prevented from publishing benchmark data (Vogt 2001). Purchasers are unlikely to get much by way of a trial of the product (or indeed this might not be possible). As Vogt (2001) points out, it is as if software vendors can somehow escape the demands of a free-market economy where it might be expected that informed consumers make choices and producers/suppliers succeed and fail based upon the quality of their products. Software vendors have escaped much of the benchmarking accountability which would be expected in marketing other types of product. Small wonder, then, that the commercial packaged software market cuts right against the grain of the open source software movement where sharing rather than hiding information about products becomes part of its avowedly ethical mission.

Consequently, whilst vendors may control landscape of selection to a fair degree, they clearly have to respond to market demands to some extent. Yet, vendors may attempt to circumvent this by engaging in processes of convincing consumer organisations that their products are the 'best' and that they should buy into their vision for the package especially where comparable alternatives exist. Friedman and Cornford (1989) call this "salesmanship", the idea that users may be taught what is good for them and to do things in a new way. Thus, it is possible to argue that, due to the competitive nature of the packaged software market, purchase may be directly/indirectly or implicitly/explicitly influenced by vendors who bring their own agendas to "the table". It is here where Nissenbaum's (1995) concept of accountability can be seen as at its most important. Not only do we expect software to be free from error, and to hold those responsible for production of the software if errors are encountered. But, in addition to errors, the IEEE-CS/ACM software engineering code exhorts its members not to make false claims as to functionality of a product and to be realistic about costs and timescales. However there is no way, at present, of ensuring that software vendors behave in an ethical way. Indeed, by valorising doubtful practices under the rubric of "salesmanship" the art of selling software may, forever, be tarred with the brush of "used car salesmanship" where it is seen as clever to convince a user organisation that a product is best.

4 GLOBALISATION IN THE PACKAGED SOFTWARE INDUSTRY

The packaged software industry has rapidly become global in nature. In 1991, the US held 78 per cent of the market, with Europe in second place with 16 per cent (Brouthers and van't Kruis 1997). By 1997, the US still dominated the industry although Japan and Europe had both increased their shares of the market significantly and overall worldwide growth was evident (OECD 2000).

4.1 Packaged Software as a 'Global' Product?

An important issue in this respect is the question of whether the origin of a packaged software product relates to the inscribed assumptions about the conduct of work that it may display as a result. That is, if people in the US develop software – will it be in agreement with the "ways of life" that exist throughout Europe or Asia? So does it make a difference, in a sense which we would regard as

ethical, where a software package was built? Are there elements inscribed (Akrich 1992) in the design of software packages which reflect the country, or organisation where the software was built and which then can be seen as forcing a way of working upon those that purchase the software from other cultures?

It has been suggested that the origin of the development of a package might be at odds with the country of implementation even though the drive for economies of scale may require a globally acceptable product (Krumbholz et al. 2000). Moreover, Carmel (1997) in attempting to explain the hegemony of the US in the packaged software industry, argues that Japan does not have its creative culture, although he recognises that the "American Culture of Software" is an intersection of characteristics including US culture, immigrant culture and that of computer programmers and software professionals. The latter point relating to the culture of the developers refers to "Mealy's Law" whereby: "The eventual structure of the system reflects the structure of the organization that builds the system" (Yourdon 1986, pp. 81-82).

Yet, Carmel points out that US culture is in fact US "cultures" so we cannot even assume that there is an "American" culture which dominates. So there are two related concerns here. First of all there is the question of imposing values from one culture on another culture. The second related point is that the structure of an organisation drawn from the first culture may then be imposed on an organisation in the second culture. Pollock et al's. (2003) description of the implementation of ERP package module, Campus, in various UK universities is very relevant. The universities themselves did not have compatible enough procedures and standard categories to make this readily achievable in the first place. The supplier attempted to have a global common core. The authors argue that the biography of a piece of software is important. We need to know where it has been in order to understand why it possesses its current functionality. This suggests that all sorts of unwanted functionality may exist by the time a particular organisation gets the package – this unwanted functionality represents a form of imposing a design from an earlier organisation, within a particular culture, onto a new one. For instance, the Campus module was built to accommodate the procedure, common in American Universities of having applicants submit "application fees". There is no such system in the UK so one of the British University users was left with the problem of deciding what to do with 30,000 unwanted accountancy records which were generated each year (Pollock et al. 2003, p. 328)

Moreover, in respect of "Mealy's Law", the problem is that given the rising complexity of packaged software offerings and the characteristics of organisations in the intended markets, developing organisations may lack the structures in place for developers to mirror. One SAP development team found it necessary to involve customers in the development of its treasury module in an attempt to build a product with high market acceptability (Scott and Kaindl 2000). These arguments also assume that developers are free to build products in a manner that they wish when in fact they, and the organisations they belong to, are to some extent, constrained by market demands if they wish to continue in business.

4.2 Packaged Software as a 'Cultured' Product

Of course, the country of origin of the software, the cultural composition of the development team and their environment may impact upon the eventual product but this is not as simple as first suggested. What this means is that the globalisation of the packaged software industry has significant implications where the producer aims for standardisation to achieve economies of scale across diverse geographical and cultural environments. However, it does not necessarily follow that a product built in the US will turn a corner shop in the UK into Wal-Mart.

Thus, in terms of culture, it is perhaps more useful to consider the global argument in relation to the problems of implementing a generic product that has been targeted at a broad market. Undoubtedly, there will be features of the product that will be ingrained with cultures which reflect the origins of its production and the ethos of the organisation where it was developed. The issue in relation to ethics appears to be understanding the implications of this for the context within which the product will be

used. That is, asking the question of whether the package will fit with the cultures of the organisation within which is to be implemented. To paraphrase the popular mantra, one could argue that software vendors are thinking global but not acting local. Of course the matter is more much more complicated. For instance, assumptions are made that those involved in selection are willing and able to find out about this, that cultural changes are not part of the reason for implementation, and that those using the eventual system do not subvert the system to avoid any formal culturally rooted directives in the system.

When we consider that vendors attain benefits from achieving economies of scale, there is little incentive for products to become more local. They may sell something that will not fit otherwise, they may not achieve the sale. Then the organisation may be left with all sorts of undesirable functionality as in the case above where the UK university ended up with thousands of unwanted accountancy records. Worse than this, the package may not have the functionality that the customer requires. If we think of this problem in ethical terms, we can see that the concept of accountability is important. However, we argue that the idea of accountability for error, as in errors or faults in safety critical systems causing harm is not broad enough for software package selling as the discussion above demonstrates. Arguably, the UK university that was faced with thousands of unwanted accountancy records was not strictly being harmed. However it seemed impossible to hold the software vendor accountable for imposing the requirements of another organisation operating within a different culture. Other more serious examples of the potential harm caused by packages though can be seen at 'Threads' where for example, the overall project was reported to have increased in cost five-fold from original estimates (Holland and Light 1999) and at FoxMeyer, where the acquisition, implementation and usage of SAP, was argued to have resulted in bankruptcy proceedings (Bicknell 1998).

5 PACKAGED SOFTWARE PRODUCT DEVELOPMENT

As the market requires customers to buy the products on offer it is widely recommended that there are benefits to be had from including customers in product development activities. For instance, it is made clear by Raghunathan (2000) that market information, in the form of customer input, is essential. This point is also brought out by Carmel and Becker (1995) in their explanation of the requirements of a process model for packaged software product development. This also accords with the findings of Holmström (2001) in her study of the development of the 'Clusterball' online game. However, the extent and nature of the inclusion of consumers in the development process, and for what purpose, is far from simple. This is of particular ethical importance because once those organisations buy a package they become locked into a vendor's product development trajectory.

5.1 Trajectories of Product Development

There are questions surrounding the ways in which the product will develop in the future and the extent to which any purchaser will be able to influence that activity. The ethical issue here involves the question of how far involvement in a product's development in the past guarantees a say in the future or whether it achieves lock-in to a product with little say on its future development. But this is a complex situation. If the software vendor ties the product into one customer it may never be generic enough to sell elsewhere. The vendor's business may collapse leaving the original customer with an unsupported software product.

An early example of this was encountered by one of us in relation to producing software packages for IBM time-sharing systems in the early 1980s. The software house, in question, developed its packages in response to the requirements of one division of a large UK chemical company. The packages were then sold to other parts of the same company at heavily discounted rates. Although the software products were intended to be generic database, query and graphics systems, with the appeal of running on widely available IBM mainframes, and with a level of interactivity far ahead of most competitor systems, they failed to achieve significant market penetration outside the original user organisation.

Part of the reason may relate to the way that the PC market was poised to take off at that time. However, much of the problem for the software vendor was that, regardless of the generic capabilities of the packages, these software packages were perceived as hooked into the chemical company where they had originally been developed.

In another example, one study recounts the product enhancement process of the SAP Treasury module which involved a number of European and US organisations (Scott and Kaindl 2000). The study illustrates just how exclusive the packaged software development process can become. Most of the participating organisations were large global firms from a variety of industries which where pruned by the SAP team. They did this by selecting ones with what they felt to be 'state of the art' knowledge in the area. Moreover, to "ensure mutual goal alignment", they chose organisations that were willing to change their processes. The authors suggest that, as a result, the chosen individuals felt like members of an "elite group", were excited to influence the design and to be among the first customers to have the module. The process demonstrates how the packaged software company used customers to improve its product, but in a way that mostly accommodated them rather than their customers. They wanted the knowledge of the market, but not the arguments and complexity of heterogeneous ways of work. Essentially, the organisations in the study submitted to a form of salesmanship.

One example of conflict within the process, however, is illustrated whereby there were disagreements regarding the linkage of SAP with other applications in place in the participating organisations. The SAP team was unwilling to write interfaces to support these applications. Instead, the most common applications were catered for and users were told to place a request with their software vendors to work with SAP to become partners. This shows how it very difficult for individual user companies to influence the design of packages. Indeed, it has been suggested that even where purchasers do get involved, vendors may not view all requirements as relevant (Pozzebon 2001), especially as they want to maintain a 'generic' product so that they can sell it to as many customers as possible (Bansler and Havn 1994). Even where demands are made in a relatively strong fashion, the power of the vendor outweighed that of the customers even though they were large companies. This demonstrates the way in which a software vendor may make themselves accountable to customers, to some degree, where common applications are catered for or where "big" customers are accommodated, but where full accountability is avoided where smaller customers may have little influence on the trajectory of a package's design.

It is generally considered to be more expensive to market products to new customers rather than existing ones (Gronroos 1994, Buttle 1995, Payne et al. 1999). Consequently, vendors have to balance their desires for increased profitability via economies of scale with customer demands. Andersson and Nilsson (1996) suggest that price competition is less when selling services such as support and maintenance to existing customers, and that contracts for these can generate considerable repeat income. Essentially what this means is that packaged software vendors will have little scope for operating a cost based strategy in relation to the initial sale of licences but they will be more successful in generating profit once the customer is locked-in through maintenance and upgrade services as switching costs increase dramatically.

Although this is a typical reflection of any competitive environment, it does raise issues in respect of ethical product development and selling. Specifically, issues related to the responsiveness of the vendor and the ability of the adopting organisations to influence the development trajectory of a given product may require consideration. Accountability is raised as an issue once again. The customer organisation must trust the software vendor to be accountable for future developments which will not lock them into a product which has begun to take a different direction from that which the original customers envisaged.

5.2 Perceptions of the Robustness of Packaged Software

A further significant attraction of packaged software for many organisations is related to the perception of the reduced risks of implementing what is seen as a 'tried and tested solution' in contrast

to custom development (Golland 1978, Chau 1995). Packages are proffered as designed and tested by the vendor, and in most cases, as having been installed by other organisations allowing for reference site visits by potential purchasers in order to evaluate the product (Heikkila et al. 1991). For example, most packaged software vendor websites contain the lists of high profile company cases that aim to illustrate the potential benefits resulting from the implementation of their product. Thus, it is argued that the conditions for estimating the quality and usefulness of the system, and the implications for work content and organisation, are much better than in traditional development projects (Bansler and Havn 1994).

Yet, again there are problems with these assertions. There is the suggestion that packaged software is 'better built' than custom developed software yet studies suggest that there is a lack of rigour in the product development processes of the packaged software product industry (Carmel 1993, Carmel 1997). To compound this problem, it has been argued that much of product development is opaque to most consumer organisations. Consequently, since production and consumption are separated, vendors tend to be evaluated in terms of their products, not their processes (Sawyer 2001, Howcroft and Light 2002). Therefore, even if a package has been developed in a less than robust fashion, it is possible that this will be ignored by organisations when they decide to implement packaged software and further overlooked in the procurement process. In addition, even if a product is seen to work at a reference site, it does not follow that it will do so in exactly the same fashion in another organisation (Light 2003). This is almost the opposite problem from that of customer involvement in software development. In this case the customer is being asked to purchase the product without being able to see what is underneath. It is analogous to buying a car without being able to look under the bonnet (assuming you know what you are looking at). The point here is that if the customer has little inkling of the vendor's processes, and has only seen the product in a very artificial reference setting, it becomes difficult to hold the vendor accountable if things do not work as expected in the new customer's organisation. Once again accountability is deflected.

6 SUMMARY AND CONCLUSIONS

In conclusion, we have argued that current conceptualisations of Computer Ethics and Professionalism require further development for software vendors and packaged software as these areas have largely been left out of computer ethics research to date. By the same token, the burgeoning IS literature on software packages is relatively quiet on the topic of ethics. By bringing computer ethics to software vendor research we hope to have made a beginning in casting the ethical spotlight on the activities of software vendors. This offers the possibility of a broader definition of professionalism than has usually been the case in IS, suggesting that the topic of software vendors opens up a whole new area of possible research for computer ethics which needs to look to the activities of small specialist companies in addition to its traditional concentration on individuals in larger organisations. Hence more research is required on the concept of accountability and how this can be extended to all sorts of organisations, large and small. We have argued that the topic of professionalism needs extending to include the activities of software vendors, not just software developers, as the industry has rapidly moved into a position where package software dominates. However, calling for a widening of the concept is one thing, making it stick to software vendors is another when the IT industry, in general, is largely unregulated. This suggests that we need ways to bring software vendors into the debate over ethics. This process may begin through discussions in the trade press and through the work of professional bodies such as IFIP, BCS (British Computer Society) and ACM. We have sought to bring Nissenbaum's (1995) conception of accountability more squarely into the software package industry as much the problem of professionalism relates to ways in which the industry may be made more accountable. We regard the concept of "salesmanship" and how this relates to ethics, particularly the ethics of accountability, as an important area for further exploration.

Although space does not permit discussion of it here the question of intermediaries further extends ethical concerns. As Nissenbaum (1995) notes, at least some of the problem of accountability rests in

the “many hands” syndrome. With intermediaries involved in software selling there are even more “hands” involved and so we contend that this is an area worthy of further study which is likely to yield more on the question of accountability.

References

Adam, A. and Bell, F. (2003), "Critical Information Systems Ethics", in *3rd Critical Management Conference* University of Lancaster: Lancaster.

Akrich, M. (1992), "The De-Scription of Technical Objects", in Bijker, W. E. and Law, J. (Eds), *Shaping Technology/Building Society: Studies in Sociotechnical Change*, MIT Press, London, pp. 205-224.

Andersson, R. and Nilsson, A. G. (1996), "The Standard Application Package Market - an Industry in Transition?" in Lundeberg, M. and Sundgren, B. (Eds), *Advancing Your Business: People and Information Systems in Concert*, EFI, Stockholm School of Economics, Stockholm, pp. 1-24.

Bansler, J. and Havn, E. C. (1994), "Information Systems Development with Generic Systems", in Baets, W. R. J. (Ed.) *Proceedings of the 2nd European Conference on Information Systems* Nijenrode University Press: Breukelen, pp. 707-715.

Bicknell, D. (1998), "Sap to Fight Drug Firm's $500m. Suit over R/3 Collapse", *Computer Weekly*, 3 September, p. 3.

Bingi, P., Sharma, M. K. and Godla, J. K. (1999), "Critical Issues Affecting an ERP Implementation", *Information Systems Management*, 16(3), pp. 7-14.

Brouthers, K. D. and van't Kruis, Y. M. (1997), "Competing in Software: Strategies for Europe's Niche Businesses", *Long Range Planning*, 30(4), pp. 518-528.

Business Week (1984), "Software: The New Driving Force", *Business Week*, 27 February, p. 74.

Butler, J. (1999), "Risk Management Skills Needed in a Packaged Software Environment", *Information Systems Management*, 16(3), pp. 15-20.

Buttle, F. (1995), *Relationship Marketing: An Overview (Working Paper No. 305)*, Manchester Business School, Manchester.

Carmel, E. (1993), "How Quality Fits into Packaged Development", *IEEE Software*, 10(5), pp. 85-86.

Carmel, E. (1995a), "Cycle Time in Packaged Software Firms", *Journal of Product Innovation Management*, 12(2), pp. 110-123.

Carmel, E. (1995b), "Time-to-Completion Factors in Packaged Software Development", *Information and Software Technology*, 37(9), pp. 515-520.

Carmel, E. (1997), "American Hegemony in Packaged Software Trade and the "Culture of Software"", *The Information Society*, 13(1), pp. 125-142.

Carmel, E. and Becker, S. (1995), "A Process Model for Packaged Software Development", *IEEE Transactions on Engineering Management*, 42(1), pp. 50-61.

Carmel, E. and Sawyer, S. (1998), "Packaged Software Development Teams: What Makes Them Different?" *Information Technology and People*, 11(1), pp. 7-19.

Chau, P. Y. K. (1994), "Selection of Packaged Software in Small Businesses", *European Journal of Information Systems*, 3(4), pp. 292-302.

Chau, P. Y. K. (1995), "Factors Used in the Selection of Packaged Software in Small Businesses: Views of Owners and Managers", *Information and Management*, 29(2), pp. 71-78.

Dube, L. (1998), "Teams in Packaged Software Development: The Software Corp. Experience", *Information Technology and People*, 11(1), pp. 36-61.

Fan, M., Stallaert, J. and Whinston, A. B. (2000), "The Adoption and Design Methodologies of Component-Based Enterprise Systems", *European Journal of Information Systems*, 9(1), pp. 25-35.

Forrester Research (1998), "Sizing Commerce Software", *Forrester Research*, http://www.forrester.com, Accessed: 6 June 1999.

Friedman, A. L. and Cornford, D. S. (1989), *Computer Systems Development: History, Organization and Implementation*, John Wiley and Sons, Chichester.

Gefen, D. (2002), "Nurturing Clients' Trust to Encourage Engagement Success During the Customization of ERP Systems", *Omega - The International Journal of Management Science,* 30(4), pp. 287-299.
Golland, M. L. (1978), "Buying or Making the Software Package That Is Best for You", *Journal of Systems Management,* 29(8), pp. 48-51.
Gotterbarn, D. (1997), "Software Engineering: A New Professionalism", in Myers, C. (Ed.) *The Responsible Software Engineer: Selected Readings in It Professionalism*, Springer-Verlag, London, pp. 21-31.
Gotterbarn, D. (2000), "Computer Professionals and Your Responsibilities", in Langford, D. (Ed.) *Internet Ethics*, St Martin's Press, New York, pp. 200-219.
Gremillion, L. L. (1982), "Improving Productivity with Application Software Packages", *Business Horizons,* 25(2), pp. 51-54.
Gronroos, C. (1994), "Quo Vadis, Marketing? Toward a Relationship Marketing Paradigm", *Journal of Marketing Management,* 10(5), pp. 347-360.
Heikkila, J., Saarinen, T. and Saaksjarvi, M. (1991), "Success of Software Packages in Small Businesses: An Exploratory Study", *European Journal of Information Systems,* 1(3), pp. 159-169.
Holland, C. and Light, B. (1999), "Global Enterprise Resource Planning Implementation", in *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences* IEEE Computer Society Press: Hawaii, pp. CD-ROM.
Holmström, H. (2001), "Virtual Communities as Platforms for Product Development: An Interpretive Case Study of Customer Involvement in Online Game Development", in Sarkar, S., Storey, V. C. and De Gross, J. I. (Eds), *Proceedings of the 22nd International Conference on Information Systems* Association for Information Systems: New Orleans, USA, pp. 299-306.
Howcroft, D. and Light, B. (2002), "A Study of User Involvement in Packaged Software Selection", in Applegate, L., Galliers, R. D. and De Gross, J. I. (Eds), *Proceedings of the 23rd International Conference on Information Systems* Association for Information Systems: Barcelona, Spain, pp. 69-77.
Johnson, D. G. (2001), *Computer Ethics*, 3rd edn., Prentice Hall, Upper Saddle River, NJ.
Krumbholz, M., Galliers, J. and Coulianos, N. (2000), "Implementing Enterprise Resource Planning Packages in Different Corporate and National Cultures", *Journal of Information Technology,* 15(4), pp. 267-279.
Ladd, J. (1995), "The Quest for a Code of Professional Ethics: An Intellectual and Moral Confusion", in Johnson, D. G. and Nissenbaum, H. (Eds), *Computers, Ethics and Social Values*, Prentice Hall, Englewood Cliffs, NJ, pp. 526-538.
Light, B. (2003), "A Study of Organisational Experiences of Crm Packaged Software", *Business Process Management Journal,* 9(5), pp. 603-616.
Lynch, R. K. (1984), "Implementing Packaged Application Software: Hidden Costs and New Challenges", *Systems, Objectives, Solutions,* 4(4), pp. 227-234.
Martin, M. H. (1998), "An ERP Strategy", *Fortune,* 2 February, pp. 95-97.
Nissenbaum, H. (1995), "Computing and Accountability", in Johnson, D. G. and Nissenbaum, H. (Eds), *Computers, Ethics and Social Values*, Prentice Hall, Englewood Cliffs, NJ, pp. 526-538.
OECD (2000), *Oecd Information Technology Outlook,* OECD, Paris.
OECD (2002), *Oecd Information Technology Outlook (Highlights),* OECD, Paris.
Oliver, D. and Romm, C. (2000), "ERP Systems: The Route to Adoption", in Chung, H. M. (Ed.) *Proceedings of the 6th Americas Conference on Information Systems* Association for Information Systems: Long Beach, USA, pp. 1039-1044.
Payne, A., Christopher, M., Clark, M. and Peck, H. (1999), *Relationship Marketing for Competitive Advantage,* 2nd edn., Butterworth Heinemann, Oxford.
Pollock, N., Williams, R. and Procter, R. (2003), "Fitting Standard Software Packages to Non-Standard Organizations: The 'Biography' of an Enterprise-Wide System", *Technology Analysis and Strategic Management,* 15(3), pp. 317-332.

Pozzebon, M. (2001), "Demystifying the Rhetorical Closure of ERP Packages", in Sarkar, S., Storey, V. C. and De Gross, J. I. (Eds), *Proceedings of the 22nd International Conference on Information Systems* Association for Information Systems: New Orleans, USA, pp. 329-337.

Raghunathan, S. (2000), "Software Editions: An Application of Segmentation Theory to the Packaged Software Market", *Journal of Management Information Systems*, 17(1), pp. 87-113.

Robson, W. (1997), *Strategic Management and Information Systems*, 2nd edn., Pitman Publishing, London.

Sawyer, S. (2001), "A Market-Based Perspective on Information Systems Development", *Communications of the Association for Computing Machinery*, 44(11), pp. 97-102.

Scott, J., E. and Kaindl, L. (2000), "Enhancing Functionality in an Enterprise Software Package", *Information and Management*, 37(3), pp. 111-122.

Sumner, M. (2000), "Risk Factors in Enterprise-Wide/ERP Projects", *Journal of Information Technology*, 15(4), pp. 317-327.

Tavani, H. T. (2004), *Ethics and Technology: Ethical Issues in an Age of Information and Communication Technology*, Wiley, Hoboken, NJ.

Vogt, C. (2001), "What Do So Many Software Vendors Have against a Free Market Economy?" *Infoworld.com*, http://archive.infoworld.com/articles/op/xml/01/04/23/010423opethics.xml, Accessed: 20 October 2003.

Yourdon, E. (1986), *Managing the Structured Techniques: Strategies for Software Development in the 1990s*, 3rd edn., Yourdon Press, Englewood Cliffs.