SOFTWARE PACKAGES

What you always wanted to know when buying software but were afraid to ask

JAN DAMSGAARD[1] and JAN KARLSBJERG[2]

[1] *Dept. of Information Systems, Weatherhead School of Management, Case Western Reserve University, USA*

[2] *Dept. of Computer Science, Aalborg University, Denmark*

Abstract: This paper presents seven alerts that inform organizations and individuals buying software packages. The research is based on our own studies as well as a review of theoretical and empirical studies of modern information systems and the networks associated with these systems. The paper departs from a monolithic view of buying software as an atomic event that is based solely on the software's independent features and its immediate price. Instead we promote a pluralistic multi-organizational view of buying software as a continuous process of trying to match available packages with a base of already installed information systems while anticipating future needs. We have formulated seven alerts that both researchers and practitioners should consider when studying these building blocks of e-business.

1. INTRODUCTION

Software systems pervade our lives, both as professionals working in some organization, and as private persons. Such systems are not uniquely designed for each use situation, but instead they are built around some standard, that allows (and by exclusion disallows) communication with other software systems. Organizations and individuals alike therefore have to make decisions about which Internet browser and email program to purchase. And organizations are making substantial investments in terms of both money and time resources when it comes to the choice of organization-

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35692-1_36

K. V. Andersen et al. (eds.), *Seeking Success in E-Business*

© IFIP International Federation for Information Processing 2003

wide systems like financial software or enterprise resource planning (ERP) software, customer relationship management (CRM) software, network operating systems, etc. For these and many other software choices the notion of networking is important, as the acquired software will meet its full potential for the organization only when it is successfully integrated with other systems inside the organization, and in some cases even with systems of other organizations.

Very few organizations develop systems from scratch. Instead companies are buying standard software and adding it to a cumulative pool of standard systems that already exists in the organization. The result is that today an information system is not built as a "stand alone" micro cosmos; instead the system must obey many rules set by the installed base of existing IT systems in the organization with which it must interact in order to perform its intended function (Hanseth and Braa 1999). This paper argues for the urgent need to put the study of IT standards on the research agenda.

We wish to emphasize that the management of information technology standards is crucial in order to achieve the coherence of organizational information systems that is necessary to graduate from doing electronic commerce into doing electronic business. With some notable exceptions (e.g. Bensaou 1999; George 2000; Markus 2001), unfortunately IS scholars have so far had little advice to companies and CIOs on how to maneuver in the seemingly erratic world of standard based IS systems. This lack of advice is particularly alarming when it comes to developing e-business systems, as it is generally accepted that the e-business user interface is just the tip of the iceberg and that all successful e-business are built on top of a coherent and cohesive IT infrastructure made up of standard software systems. And there is ample evidence of e-businesses collapses due to lack of integration between different software systems.

2. SOFTWARE PACKAGES

In the early years of commercial use of computers (starting in the mid 1950s), all software systems were developed in-house out of simple necessity, since no software industry existed at the time (George 2000). As the software industry emerged, eventually most companies outsourced most of their software development to specialized software companies. Most software products were, however, still developed independently for each organization, i.e. there was little market standardization in the product, and little standardization in the production process. The next step was that each software producer developed its own set of standards in order to capture economies of scale in developing the software once for multiple customers

(Attewell 1992). This standardization also benefited the software buyers by lowering transaction costs as it was now possible to choose among a set of applications rather than merely choose among a set of software producing companies. Standardization also gives both producers and buyers of software a way of blackboxing knowledge by embedding it into the standardized components of the systems, thereby freeing resources for handling the exceptions (Scarbrough 1995).

For the purposes of this paper we define a standard software package as follows:

> A "standard software package" is a collection of software components that combined perform a set of generalized tasks that are applicable to a wide range of users (be they individuals or organizations). The software packaged is "standard" because the core components are identical across all of its implementations. Customization of the software may be performed to fit the software with use requirements unique to the individual implementations. This customization is implemented by adjusting program parameters, installing add-on modules, interfacing the software with other software systems, etc.

Three well-known categories of standard software packages are office application suites (including software like word processing, spreadsheets, and presentation software), database management software (DBMS), and enterprise resource planning (ERP) software. Some standard software packages require little customization on the part of the users before they are usable by an individual or an organization, while other standard software packages are mere tools on top of which the specific functionalities required by the user must be implemented in the customization process. In some cases, some customizations may be common among several customers, and the producer can then offer "standard customizations" on top of which only the site-specific customizations need be made. The German ERP producer SAP AG provides an example of this setup, since its system SAP R/3 exists in several standard customizations targeted at different industry sectors, e.g. chemical producing companies, retail store chains, and the pulp & paper industry.

Organizations are increasingly relying on standard software packages. George (2000) reports a threefold increase in the time corporate IS groups spend working on packaged applications from 1997 to 1998. The move from one-off implementations towards standard systems is accelerated by the increasing maturity of standard software packages due to economies of scale, and further by what looks like a flocking effect exhibited by the buyers of standard systems - creating winner-takes-all situations in many markets for standard software systems: large-scale consolidation on the producer side (Liebowitz and Margolis 1999; Shapiro and Varian 1999).

At first glance, these developments in the market for software systems may appear to be to great advantage for user organizations buying software since both the quality of products offered and the stability of the market increases. But as we shall illustrate in the seven alerts, standards have interesting properties that enable some or all participants in the market to use standards as a tool to attempt to manipulate the market.

It can be argued that in some large segments the market has narrowed down the range of options available to new buyers. For example bundling and co-branding agreements between hardware and software vendors often makes it impossible to buy a particular PC model without also paying for a license for Microsoft's Windows operating system. Another example concerns availability of particular versions of software products. Due to the high cost of supporting multiple versions of a software product, most software producers choose to let older versions expire, making it impossible for new users to buy a license to a particular version of the software which they may prefer over newer versions of the software.

We argue that strong effects of the market forces such as the two examples described above do not relieve the buying organization or individual from the responsibility of choosing the standards that suit their particular situations. Rather, these effects of the market forces make it all the more important for individual users and organizations alike to be aware of standardization issues and heed our standards alerts.

3. SEVEN ALERTS

3.1 Alert 1: When you buy a software package you join its network

The development from in-house development of software towards standard software packages puts the focus on standards and networks, since the users and producers of any given standard form a network (Shapiro and Varian 1999). Thus a decision to buy a particular product includes the implication of joining the network associated with the standard implemented by this product. The network is virtual in the sense that the members may not know each other, but they share a common interest in protecting their investments and therefore also in the evolution of the software. This is true whether or not the standard in question is a communication standard, because for all standards the adopters share common interests and

communicate indirectly if not directly, for example through training and education of personnel, etc. (Shapiro and Varian, 1999).

The value of a network is determined by a combination of two factors: the number of members of the network and the interactions that take place in the network. First, Metcalfe's law (Shapiro and Varian, 1999) says that for a network with N participants, the potential value of the network is proportional to the number of possible interconnections in the network, which is N*(N-1) or approximately N squared. Thus the potential value of a network A with ten times as many members as a competing network B is not just ten times the value of network B, but a hundred times the value. Secondly, the network effects of a network are determined by the quantity and quality of actual interconnections and interaction in the network. If the network is sparsely connected because only few of the possible connections are realized, the network has little value; alternatively if the network enjoys rich communication and many interconnections, the members of the network reap large benefits of participating in the network.

Software companies own the network that is formed by their proprietary software and have the power to determine the evolution of the network. This power can be challenged, however, if a sufficiently large group of important customers agree to form a user group to influence the software producer's decisions regarding the software. Not all networks are subject to such power struggles based on the financial strength of the network's participants. By definition open standards are not owned by a single entity but rather by the community interested in the standard, i.e. the network of interested parties (Kindleberger 1983). Examples of open standards in information systems are most of the Internet standards such as the email protocols POP3 and SMTP (post office protocol 3, simple mail transfer protocol) which are implemented in every email client program used by Internet users worldwide regardless of which software producer developed the particular email client program.

The open source movement that lies behind applications such as sendmail and even operating systems such as Linux is a special case of open standards. In open source systems, the software producers not only comply with the specifications of an open standard, they also make the inner workings of their implementation freely available to other developers and encourage improvements to their software products. Thus a community of users and developers is designed specifically to prevent ownership and to promote shared software (Raymond 1997; O'Reilly 1998; Ljungberg 2000). The open source communities or networks surrounding various information systems appear unattractive to some user organizations because these networks lack central guidance which may lead to uncertainty in the future of the standard. Other organizations view these same properties of the open

source networks as strengths since they protect the standard from the whims and opportunistic actions of individual stake holders, thus enhancing the continuity of the standard. We shall not conclude on this debate over open source here, but merely emphasize that organizations and individuals who choose to adopt a particular software package beware the intimate connection between any software package and its network.

3.2 Alert 2: Take a long-term perspective: Look ahead but reason back

Earlier when software was designed for a specific closed environment with well defined systems requirements, the focus was on immediate costs versus benefits when investment decisions were to be made. Often new software systems had both very short range and reach (Keen 1991; Weill and Broadbent 2000), i.e. they were standalone systems built to solve one particular task within a single organization, often even within a narrow setting in the organization. Thus, the buyer organization only had to consider short-range and short-reach implications of the software acquisition. As the standalone systems grew in numbers and size, a need emerged to move data between applications. The solution was standardized software platforms, e.g. operating systems, file systems, databases management systems.

Many standards choices made in the early stages of computer use in an organization or in an industry as a whole prove to have surprisingly durable consequences as both software standards and data have shown very long lifespans, as any organization with legacy systems can attest to. Now, in the age of open standards and interconnected systems it is vital for organizations to consider each standards choice as a long-range and long-reach decision with long-term implications, because the network is not limited to the users of that one application or standard. Rather the network includes a family of related and compatible software products, as well as a host of suppliers, institutions and other users surrounding this family of products. And it is important to choose the network that will provide the best long term benefits for the organization as the technologies of the network evolve. Further, the growing importance of system interconnections within the organization will mean that the standards choice has consequences for other parts of the organization whose software standards, implementations and interests may not originally have been considered in the decision process regarding the new software acquisition.

Two strategies for this choice are widely known as "best of suite" and "best of breed". In the "best of suite" strategy, the organization chooses the one best product or supplier that has the best all-round match with the organization's requirements. The reasoning behind this decision is to

optimize the internal consistency between the various components of the overall system. The risk of this strategy is of course that of putting all of one's eggs in one basket, and that this trust may later prove to have been misplaced when the standard or vendor starts to move in a different direction compared with the organization's own requirements. In the "best of breed" strategy, the organization goes cherry-picking to find elements from several standards or suppliers that fit the organization's every requirement. The motivation behind this decision is to achieve optimal solutions. The risk - or rather the cost - of this strategy is that the system can be complicated to manage.

The evolution of many software applications is so that first an innovation is custom built by innovators and early adopters. Later several competing systems become available on the market (a breed of systems) and finally the systems become a commodity and bundled into a larger suite of standard software, like Microsoft Office Suite that contains a word processor, spreadsheet, presentation software, web site creation software, and database software. Awareness of this evolution trajectory for software applications helps the buying individual or organization focus on the potential long-term consequences of every software application acquisition.

We stress these long-term perspectives of software standards even as the pace of evolution in the computer industry reduces the effective lifespan of most products to very few years. When a personal computer is four years old, it is essentially unusable for any current software application. The machine will be at least an order of magnitude slower than contemporary computers, and very likely the replacement and upgrade parts for it have long since been unlisted by the supplier. Software products also typically have a high turnover with updates being released every year for most products. But the long lifespan of standards and data necessitates awareness that the standards choice involves participation in a network that may last one or two decades or longer.

An organization's initial choice of a standard software package for a particular task that has not previously been handled by a software package is a unique situation. There are fewer concerns about compatibility, both regarding existing technical systems, organizational processes, and knowledge management - old habits do not have to be unlearned. This situation is analogous to the choice of driving in the left side or the right side of the road. To the novice driver the decision is arbitrary; it does not matter in what side she drives. But to the experienced driver changing side represents a big challenge of "unlearning" as those of us who have tried it can attest to. The costs of changing to a different standard can be astounding to organizations with existing investments in technology and training.

3.3 Alert 3: When choosing a software package, there is safety in numbers

Earlier when ordering a software product an organization in effect committed itself to a particular (often local) producer and this producer's ability to produce added functionality as requirements grew. Now, as the commitments move from a local producer towards global producers and standards, user organizations must be wary of the development and evolutions of these standards and producers, a task that is likely much more demanding than keeping track of a single local producer.

As IT standards evolve and compete against each other, they may converge or diverge on some features, i.e. reaching or breaking compatibility with each other. As we shall describe in further detail in Alert 4, this development is caused by some party's perceived advantage in changing the degree of compatibility or interoperability between competing software standards. User organizations must be aware of the developments and seek to avoid the pitfalls of standards evolutions: blind alleys and one-way streets, which we describe next.

The blind alley scenario refers to the situation where an organization has chosen a standard that is now about to lose to competing standards. David (1986) uses the term "angry orphans" to describe the losing standards. He points out that such standards or technologies often show a sudden rapid development as they are losing the battle. For example the greatest speed of innovation in sail ships happened as the steam engine was taking over as the leading technology on the sea. Thus, though an organization should seek to avoid the blind alley scenario, the death throws of a losing standard may actually prove to be invigorating for the organization's IT systems. And if the losing standard manages to capture a niche market network, it may sustain itself for several years - or even perpetually, giving user organizations the choice of staying with the incumbent producer or the time to look for migration paths towards the wide boulevard, where there is safety in numbers. At other times the "angry orphans" may seek to counter the establishment of a new reign by holding on to resources that are critical for establishing a new. A good example is frequency allocation for cellular phones in the US where coalitions of companies providing first and second generation cellular phone services refuse to give up their control over a certain part of the radio spectrum, which in effect blocks a fast establishment of a third generation network.

A popular route taken by users who want to protect themselves against the blind alley scenario is the flocking strategy where users choose a standard based on the historic and current size and success of its associated network. As long as the network remains large, there will be a commercial

incentive for both the original owner of the network as well as competing software producers to cater to the users of the standard. The risk for the users is that the network owner may end up being powerful enough to force users into a one-way street scenario.

The one-way street scenario describes the common situation where one is left with little choice when it comes to buying upgrades or expansions to existing systems. Often the purchase of a particular product in effect obliges the user to place future product purchases in the same product family, because the product or standard has low compatibility with other products or standards. This happens for example when the producer is ahead of the competition or if the producer has intentionally differentiated its implementation of the standard from the competition for competitive reasons rather than technological necessity. In this situation, the user organization may find itself so committed to the standard that the switching costs involved with moving to another software standard are prohibitive, and the organization is in effect locked-in to the standard software for better or for worse.

3.4 Alert 4: Focus on compatibility and avoid false gold

As we mentioned in Alert 2, the actual life expectancy of the data stored in software systems is often much longer than was expected at the time of the initial implementation. For user organizations this makes backward compatibility increasingly important, and the user organization must abstract from individual products and their features and instead focus on compatibility, product types and functionality.

In some cases several software products adhere to one common standard, enabling user organizations to choose among competing products based on other product features such as price, performance, usability, etc. Most often, however, compatibility is not a clear black and white issue because producers differentiate their products by adding proprietary features and unwarranted extensions to a standard (Besen and Farrell 1994; Shapiro and Varian 1998). Microsoft calls its execution of this competitive business technique "embrace, extend, and extinguish" (Kingmand 2001).

A user organization will often find these proprietary features attractive, but it is important to be aware that proprietary features are in reality false gold: Every time a proprietary feature is included in the organization's practice of use, the switching costs are raised, meaning that it will be harder to pull away from that producer if and when the organization should ever want to do so, for example when contracts are up for renewal or renegotiation. Instead, user organizations should keep their options open by staying close to compatible standards and look for gateway standards as a

way to break an existing lock-in (David and Bunn, 1988). The break-down of standards happens in many cases where there is no central governance of the standard by a central player or authority, and even if such governance does exist standards often break down anyway as competitors test and extend the limits of the standard.

3.5 Alert 5: Choose a software package with an accessible base of knowledge

Earlier user organizations needed proprietary knowledge to operate the systems built especially for them. Today a standard system guarantees access to standardized knowledge both in its use and exploitation. Ideally, the network of organizations using a technology is matched by a network of individuals with knowledge and skills specific to this standard. If the two networks are not aligned, both user organizations and producer organizations will experience problems. One example of misaligned networks is that of ERP systems, where the number of organizations using ERP systems has not been matched by the number of people with knowledge about the setup and configuration of ERP systems, resulting in disproportionately high costs for the people component of ERP implementations.

Software producers employ various strategies for ensuring a network of knowledgeable users. One such strategy is to produce free or low cost versions of a product so people interested in the technology will be more likely to sample the products. One variation of this strategy is to make "academic versions" of the software available as free downloads or to bundle the products with textbooks used in educational institutions. Another variation is the bundling of products into product suites, resulting in a very wide distribution of even highly specialized software components (Shapiro and Varian 1999). By bundling feature-limited versions of the specialized software products, the vendor can even raise awareness about the products without serious loss of revenue from the sale of the full versions of the specialized software products. An example of feature limitation occurs in Microsoft's workstation (non-server) versions of Windows 2000 where the server modules (such as the web and file transfer protocol (FTP) modules) have been programmatically limited to ten simultaneous connections, effectively ensuring that organizations employing such services on some of their computers will invest in the much more expensive Windows 2000 server licenses for these computers.

A different strategy for building a network of knowledgeable users is formalization and institutionalization of the skills involved such as the establishing of authorized, branded training (courses and exams) for a technology, for example the multitude of titles in Microsoft's regime, such as

Microsoft Certified Systems Engineer. The process of institutionalizing skills is more complex for open standards such as open source products (sendmail, emacs, Linux, etc.) where there may be no single trusted certifying institution corresponding to the owner of the standard. Instead other forms of legitimization are used such as a person's ranking in recommender-systems such as discussion web sites for a particular product or community. Such online communities also make it very easy to see the contributions a particular member has made over the lifetime of the project, enabling potential customers or employers to get a quite detailed mental picture of the person's knowledge about a particular product, as well as her ability to explain technical issues to other users.

The co-development of the two networks (that of the technology and that of its users) has a high inertia to the point of being irreversible. The existing network of knowledgeable users of an incumbent standard forms an entry barrier that is hard to break through by a new, competing standard that has only a small network. If the new standard is owned and the owner is willing and able to invest in the standard, one way for the new standard to achieve a critical mass of users is for the standard owner to bear some or all of the switching costs for the users or user organizations willing to switch (Shapiro and Varian 1998; Shapiro and Varian 1999). Due to Metcalfe's law (see Alert 1) the owner of the network standard will be fighting an uphill battle until the networks are of comparable size. An alternative approach is to introduce gateway features into the new standard easing the transition from a competitor's product (David and Bunn 1988). When Microsoft Word was winning over the majority of the word processor market from WordPerfect they circumvented the knowledge barriers by providing WordPerfect users an easy passage. Microsoft Word featured two gateway technology features: An alternative user interface where Microsoft Word could be made to emulate the keyboard shortcuts of WordPerfect, and "Help for WordPerfect users" where the use of Microsoft Word was explained in terms that WordPerfect users could relate to. The latter feature still exists in the newest installment of Microsoft Word (Microsoft Office XP) that was released in 2001.

3.6 Alert 6: Choose a software package with the right type of standardization

Information technology standards are socio-technical constructs that comprise more than just technical specifications. Standardization can be achieved at many levels. Here we give an overview of the more common types of standardization, because it is important to choose the type that is right for the particular organization.

In standardization of output, the system's only compatibility restraint is that it must produce an output that can be used by the recipient users or systems. One example is that of web page production, where different users may use very different production techniques as long as their products fulfill agreed-upon requirements, such as a persistent look and feel when rendered on the five most popular web browser versions in use by the site's intended audience. This standardization strategy has the strength of allowing its users to optimize and personalize their data production methods without depending on consensus decisions. The strategy also has serious drawbacks if at any point in time the users need to share input or intermediate data. We would not recommend that this strategy be employed as the exclusive standardization strategy in an organization.

Standardization of user interface is a popular strategy employed to limit the training requirements as implementation decisions are guided by the similarities of the user interfaces. After some experimental implementations of information systems in a particular field, a dominant design emerges (Webster 1991; Teece 1997). Referring to web design Nielsen (2000) reasons that users spend most of their time on other sites, and therefore they prefer any new web site to be designed like the other sites they already know. As a consequence the dominant designs sometimes become static and end up as anachronisms when the surrounding systems change. For example the diskette icon featured in most MS Windows applications invokes the "save" function, even though few files are ever saved to diskettes these days, and some personal computers do not even contain diskette drives.

An organization might choose standardization of data files for one of two reasons, either seeking backward compatibility with existing data stored in legacy systems, or in an attempt to ensure access to the data using other information systems in the future. Employing this strategy, an organization can choose between a number of compatible software packages, thus bringing the simple advantage of choice in a market that may otherwise be dominated by a single producer. The disadvantage is that the user organization must abstain from using any proprietary features of the programs chosen (the false gold of Alert 4) in order to maintain data standardization. Examples of data standards with wide vendor support are the all-purpose information formatting language XML, the database language SQL, and Microsoft's rich text format (RTF).

More advanced modes of standardization of data interfaces include interconnectivity and interoperability (Bailey, McKnight et al. 1995). Interoperable information systems are able to communicate during the execution of a particular task. An everyday example is when the functionality of a electronic spread sheet program is employed by a word processing program to perform a calculation in a text document. More

advanced implementations allow interoperability between systems running on different computers, even in different locations or different organizations. Recently implemented protocols such as Microsoft's .NET initiative and XML-RPC (extensible markup language - remote procedure call) allow relative simple runtime integration of information systems using standardized communication mechanisms over the common Internet. This will have far-reaching implications for the implementation of standard software packages and inter-organizational information systems in the coming years.

Organizations may choose standardization of skills by employing only people with a particular training or education, or if necessary to carry the cost of training new employees to some formalized level of training (see Alert 5). This strategy helps ensure uniform performance and results of the work carried out by the employees, and fewer control mechanisms have to be in place to assure the quality of the output produced. Organizations can choose two types of skills on which to standardize: specific or generic skills. Specific or proprietary skills encompass a user's qualifications with a particular product or product suite, where the skills are certified by the product producer or a trusted third party. Every major vendor in the technology market has such certification programs, and many are even updated on a continuous basis, forcing certificate holders to take new exams in order to preserve their certificate status. Generic skills are skills such as programming, business knowledge, etc. that can be acquired at universities or similar educational institutes.

3.7 Alert 7: Focus more on use and less on waiting

In 1965 Intel co-founder Gordon E. Moore famously predicted that the number of transistors that can be placed on a single integrated circuit would continue to double every 18 months for the next ten years (Moore 1965). The press dubbed this prediction Moore's Law, and though the initial predictions only dared include the time until 1975, the law has held true to this date as measured by the number of transistors incorporated in the top processor models from Intel over the years (Intel 2000). The popularized version of Moore's law states that for a constant price, computer performance doubles every 18 months. Software production actually has negative scaling: When a software product grows it inherently becomes more complicated and demands more development resources (Brooks 1995). But successful standard software packages mature with time, and innovation and market forces combine to establish a small set of dominant software packages from which new buyers can choose.

This may lead to a wait-and-see position for buyers until they can determine which packages will prevail (Au and Kauffman 2001). Our advice to buyers is not to fall into this trap for the following two reasons. First of all the winner will only emerge when companies actually buy software, so by being part of the selection process there is a greater chance that software that fits the needs of the company will succeed. Secondly change is inevitable and waiting only leads to greater discrepancy between a company's existing software base and the available software packages. In a worst case scenario from a buyer's point of view a late purchase of some package forces the replacement of the existing base of software.

Another trap buyers can fall into is waiting for the emergence of a software package that is a perfect fit with the buyer's requirements.

4. CONCLUSIONS

In this paper we have described and highlighted seven connotations that are related to buying a standard software package but less obvious than the two factors of price and immediate features. The seven alerts are depicted in table 1.

1. When you buy a software package you join its network
2. Take a long-term perspective: Look ahead but reason back
3. When choosing a software package, there is safety in numbers
4. Focus on compatibility and avoid false gold
5. Choose a software package with an accessible base of knowledge
6. Choose a software package with the right type of standardization
7. Focus more on use and less on waiting

Table 1. The seven alerts

We have advocated a broad, networked view of the purchasing process and argued that the purchase of a software package is not an atomic, stand-alone event similar to buying a piece of well established and packaged technology. Instead software packages are networked and built around standards that allow (and sometimes disallow (Wilson 2001)) more or less seamless connection to other software systems. We promote a pluralistic multi-organizational view of buying software as a continuous process of trying to match available software packages with a base of already installed software systems while anticipating future needs. We have formulated seven alerts that both researchers and practitioners should consider when studying these building blocks of e-business.

The first alert emphasizes that all software packages form networks of which their users become part. Secondly standards decisions should not only

be based on immediate benefits but should take a long-term perspective, weighing in the past and the future. The third alert argues that the size and value of a network formed by a software package means that there is safety in numbers when buying software packages. The fourth alert states that even though additional features are attractive they may be deceiving and in effect lock-in the buyer to the software package, and therefore it is better to focus on compatibility. Alert number five stresses the importance of not only considering the artifact but also the entire support infrastructure and complementary assets surrounding the package. Software packages can be standardized along a number of dimensions, and alert six stresses the importance of choosing the right type of standardization. Finally alert seven urges the buyer to take an active part in the selection process and use of a software package.

We hope that these seven alerts will be used in practice by IT managers when evaluating different software alternatives. We have designed the alerts so that they each provide a particular yet practical concern when purchasing software.

One fruitful area of further research is the need to study in detail the evolution of information infrastructures built up by software packages, and how interoperability and standards decisions by software houses lead to success or failure for their software. Especially the time criticality and scalability of an e-business infrastructure is particularly interesting. We are in the midst of performing longitudinal field studies that follows the evolution of large-scale IT infrastructures that are comprised of a multitude of standard software packages.

ACKNOWLEDGEMENTS

This research was carried out as part of the PITNIT project. PITNIT is supported by the Danish Research Agency, grant number 9900102. URL: http://www.cs.auc.dk/research/IS/PITNIT/

REFERENCES

Attewell, P. (1992). "Technology diffusion and organizational learning: The case of business computing." Organization Science 3(1): 1-19.

Au, Y. A. and R. J. Kauffman (2001). Should We Wait? Network Externalities and Electronic Billing Adoption. Hawaii International Conference on System Sciences, Hawaii.

Bailey, J., L. McKnight, et al. (1995). "The economics of advanced services in an open communications infrastructure: transaction costs, production costs, and network externalities." Information Infrastructure and Policy 4: 255-277.

Bensaou, M. (1999). Electronically-Mediated Partnerships: The Use of CAD Technologies in Supplier Relations. Proceedings of ICIS 1999.

Besen, S. M. and J. Farrell (1994). "Choosing how to compete: strategies and tactics in standardization." Journal of economic perspectives 8(2 - spring 1994): 117-131.

Brooks, F. P. (1995). The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition (2nd Edition), Addison Wesley Longman, Inc.

David, P. A. (1986). Narrow windows, blind giants and angry orphans: The dynamics of systems rivalries and dilemmas of technology policy. Technological Innovation Project - working paper no. 10.

David, P. A. and J. A. Bunn (1988). "The economics of gateway technologies and network evolution: Lessons from electricity supply history." Information Economics and Policy 3: 165-202.

George, J. F. (2000). The Origins of Software: Acquiring Systems at the End of the Century. Framing the Domains of IT Management: Projecting the Future Through the Past. R. Zmud. Cincinnati, Ohio, Pinnaflex Educational Resources, Inc.: 263-284.

Hanseth, O. and K. Braa (1999). Hunting for the treasure at the end of the rainbow: standardizing corporate IT infrastructure. IFIP TC8 WG 8.2 - New Information Technologies in Organizational Processes: Field Studies and Theoretical Reflections on the Future of Work, St. Louis, MO, Kluwer Academic Publishers.

Intel (2000). "Moore's Law", http://www.intel.com/research/silicon/mooreslaw.htm

Keen, P. G. W. (1991). Shaping the Future: Business Design Through Information Technology, Harvard Business School Press.

Kindleberger, C. P. (1983). "Standards as public, collective and private goods." KYKLOS - International review for social sciences 36 (1983): 377-396.

Kingmand, H. (2001). "The Microsoft Standard is Anything But" ZDNet News, http://www.zdnet.com/zdnn/stories/comment/0,5859,2784051,00.html

Liebowitz, S. J. and S. E. Margolis (1999). Winners, Losers & Microsoft, Competition and antitrust in High Technology. Oakland, California, The independent institute.

Ljungberg, J. (2000). "Open Source Movements as a Model for Organizing." European Journal of Information Systems 9(4): 208-216.

Markus, M. L. (2001). Process Integration In The Chemical Industry, (Working paper).

Moore, G. E. (1965). "Cramming more components into integrated circuits." Electronics 38(8).

Nielsen, J. (2000). "End of Web Design", http://www.useit.com/alertbox/20000723.html

O'Reilly, T. (1998). The open-source revolution. Release 1.0, Ester Dyson's Monthly Report: 3-26.

Raymond, E. S. (1997). "The Cathedral and the Bazaar", http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/

Scarbrough, H. (1995). "Blackboxes, Hostages and Prisoners." Organization Studies: 991-1019.

Shapiro, C. and H. R. Varian (1998). "The art of standards wars." California Management Review 41(2): 8-32.

Shapiro, C. and H. R. Varian (1999). Information Rules: a strategic guide to the network economy. Boston, Massachusetts, Harvard Business School Press.

Teece, D. J. (1997). Capturing Value from Technological Innovation: Integration, Strategic Partnering, and Licensing Decisions. Managing Strategic Innovation and Change. A Collection of Readings. New York, Oxford Press: 287-306.

Webster, J. (1991). Advanced manufacturing technologies: work organisation and social relations crystallised. A Sociology of monsters: essays on power, technology and domination. J. Law. London and New York, Routledge: 192-221.

Weill, P. and M. Broadbent (2000). Managing the IT Infrastructure: A Strategic Choice. Framing the Domains of IT Management: Projecting the Future Through the Past. R. Zmud. Cincinnati, Ohio, Pinnaflex Educational Resources, Inc.: 329-353.

Wilson, D. (2001). "Talkin' to Me? Not if AOL Has Its Way" Los Angeles Times, http://www.latimes.com/technology/la-000093189nov22.column?coll=la%2 Dheadlines%2Dtechnology