Author(s): Käkölä, Timo

Title: Software business models and contexts for software innovation: key areas software business research

Year: 2003

Version:

Please cite the original version:

Käkölä, T., "Software business models and contexts for software innovation: key areas software business research," System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on , vol., no., pp. 8 pp., 6-9 Jan. 2003
doi: 10.1109/HICSS.2003.1174425

Software Business Models and Contexts for Software Innovation: Key Areas for Software Business Research

Timo Käkölä
Software Business Program
Department of Computer Science and Information Systems
University of Jyväskylä
P.O. Box - 35, FIN-40351, Jyväskylä, Finland
E-mail: timokk@cc.jyu.fi

Abstract

This paper examines business, design, and product development aspects of software business models. Contexts of small and large companies for creating software innovations are also analysed. Finally, software business research is called for and an agenda for software business research is presented to better understand the dynamics of the software industry and help create and manage successful software-intensive ventures.

Keywords: software business, software business models, software innovation, software industry, venturing, corporate venturing.

1. Introduction

Software industry is one of the fastest growing industries in the world. The importance of software industry is magnified because (Hoch, et al. 1999):

- Software is increasingly becoming a key enabler of other industries.
- Software is more and more commonly embedded in the products and services of other industries.
- Other industries are becoming increasingly knowledge driven and thus more similar in their management problems to the software industry.
- The operations and competitiveness of e-businesses rely on effective use of software.

Software products and services market can be divided into five major industry segments:

- Professional software services (planning, building, integrating, and maintaining customized software systems for individual customers).
- Enterprise solutions (relying on both products and professional services that adapt and integrate the products for customer needs).
- Packaged mass-market software (designing and selling of software products for the public).
- Internet-based applications rented by Application Service Providers.
- Embedded software including services.

Software industry is difficult to analyze because it penetrates other industries and evolves rapidly and in unexpected ways. For example, a powerful shift from customized software systems toward packaged off-the-shelf enterprise solutions has taken place. This shift has broad ramifications for the industry. It decreases the need for systems development and software engineering and increases the need for people and services companies with excellent systems integration, change management, and social competencies. These competencies are needed to introduce complex software packages in organizations so that resistance and inertia can be overcome and full business benefits can be reached. Another powerful trend is ”servicization” of the industry, that is, vendors, instead of selling software products, rent or lease software-based services to customers through fixed and mobile Internet. This services-based business model is appealing because companies and consumers are interested in the benefits provided by the services, not in software itself. For example, location-based services (e.g., finding the closest movie theater showing a specified movie) are offered to customers via mobile terminals by using complex hardware and software systems but these systems are almost transparent to the customer. Because of these and other trends, software industry will be very different from now by the end of this decade.

Software businesses tend to be challenging to manage. Each industry segment requires a very different business model. Moreover, businesses operating in small local

Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)
0-7695-1874-5/03 $17.00 © 2002 IEEE

markets such as Finland usually have to operate in several segments in order to grow, thus requiring them to run multiple business models in parallel. For example, some Finnish enterprise solution providers sell standardized products, rent application services through Internet, and sell training and consulting services so customers can integrate and adopt these products and services. Such a diversified strategy may succeed as long as a company stays in local markets, but it is a complicated and expensive model to implement, thus making the company vulnerable to attacks by larger, more focused competitors. The strategy is very unlikely to work if the company wants to target international markets to reap significant growth opportunities. Indeed, software businesses in small local markets often spread their resources too thinly. For example, they may create customized, unrelated systems simply because somebody is willing to sponsor such development projects. Success in the international markets typically requires a focused, product family-based business model: developing and marketing a holistic product family that offers a complete solution to a well-specified set of problems in a clearly specified market segment (c.f., Jacobsen et al., 2001). The gap between the professional services model and the product family-model is so wide that it is easy to fall in the middle and fail.

This paper will examine business, design, and product development aspects of software business models. Contexts of small and large companies for creating software innovations are also analysed. Finally, software business research is called for and an agenda for software business research is presented to better understand the dynamics of software industry and help create and manage successful software-intensive ventures.

2. Business strategies for software businesses

Successful software businesses, like businesses in other industries, excel and focus on one of three business strategies (Rifkin 2001, Treacy and Wiersema 1995):

- Operational excellence
- Customer intimacy
- Product (or service) innovativeness

Operationally excellent organizations provide their customers with the best total cost. They have superior quality and programmer productivity, low costs, and a stable and highly focused product or service, which is delivered excellently and at a competitive price. Most research in software engineering and software process improvement (e.g., the capability maturity model CMM) focuses on designing operationally excellent organizations (Rifkin 2001, Seppänen, et al. 2001).

Customer-intimate organizations provide their customers with the best complete solutions. They seek long-term relationships to understand the businesses and underlying value networks of their customers well and provide customers with customized services and/or products and maximum benefits. Professional software service providers typically use this strategy. They benefit from relatively low market risk because even during economic downturns they are likely to be able to reap enough business from their partners to survive. On the other hand, it is difficult for them to create superior spearhead technologies and reach high operational efficiency, thus making them vulnerable to competitors with technological leadership and superior pricing. Their growth opportunities are also limited due to customer-centric and human resource intensive nature of this business.

Product-innovative organizations provide customers with the best products and target mass-markets. They compete by trying to launch new products faster than their competitors, innovate features markets are most willing to pay for, cannibalize their products before their competitors can do it, and move rapidly to new products and uncontested markets (Cusumano and Selby 1995). If they are successful with these competitive strategies, they can generate maximum revenue streams and optimal competitive positions. They operate within a context of rapid technological advances, short product life cycles, organizational transitions, and turbulent markets. Their environments are increasingly competitive and global. Product innovation strategy dominates all software industry segments except for the professional software services segment. Yet, there is limited research covering this strategy (Seppänen, et al. 2001).

Software ventures are always located within one or more business webs (Hoch, et al. 1999). Each web is built around a common platform such as an operating system or an enterprise resource planning product. Shapers of the web are companies that build the web by creating the platform that has a truly substantiated value proposition for the customers. They typically leverage the product-innovation strategy. They use various types of partners such as R&D, complementary product or service, marketing, and implementation and maintenance partners to

- close the gaps in product portfolio, R&D expertise, and distribution channels,
- focus on their key competencies,
- accelerate time-to-market, and
- increase market penetration.

The webs compete with each other. The more partners there are in a web and the higher their product or service quality is, the more successful the web is likely to be. Therefore, shapers need to provide potential partners with excellent incentives to join the web by giving them a big enough share of the business.

Software businesses need to crystallize unique strategies by properly balancing aspects of all three

strategic alternatives and leveraging partnering optimally. Excelling in one alternative is necessary but not sufficient for success (Treacy and Wiersema 1995). Moreover, a central challenge for software companies is to implement and dynamically adjust such strategies that maintain a proper balance between short-term efficiency and long-term effectiveness (Brown and Eisenhardt 1995).

Efficiency ("doing things right") requires discipline and exploitation of existing business and technology strategies, paradigms (i.e., ways of thinking and values and norms for judging what is right), technologies (e.g., programming languages and tools, software components and plug-ins, software engineering techniques), ways of organizing and working (e.g., system and software product creation processes), ideas, and business and application domain knowledge. As a result, incremental returns on these existing competencies are achieved (e.g., new releases with slightly enhanced feature sets are issued in accordance with pre-established schedules) and low risk of failure is facilitated on the short term. Operationally excellent software businesses focus on efficiency. They are like symphony orchestras that have clear roles for each musician, play according to the same notes in an exactly planned way, and are clearly managed. Operational excellence reduces the need for "fire-fighting," thus freeing resources for innovation. But an overly emphasis on efficiency is likely to lead to increased bureaucracy that may stifle the organization.

Effectiveness ("doing right things") requires both discipline for consistent execution and creativity for adaptive innovation (Boynton and Victor 1991, Brown and Eisenhardt 1998, Daft and Weick 1984, Huber 1991, Nonaka 1994, Senge 1990). Creativity involves exploration, experimentation, and improvisation with new ideas, paradigms, technologies, processes, strategies, and knowledge to develop alternatives that radically improve on old ones (e.g., a new product family that makes existing products obsolete through its superior features based on a novel underlying technology). Explorative organizations might best be characterized by the metaphor of the rock'n roll band. Exploration can yield high returns on competencies on a long term but carries a significant risk of failure. The organizational culture must tolerate failures (as long as they trigger learning and are not repeated), foster requisite variety (Ashby, 1956) in the backgrounds, competencies, and personal characteristics of personnel, facilitate frequent informal interactions that build and rely on high level of trust, and focus on creating something new. However, exploration without discipline will lead to chaos. Therefore, the best-suited metaphor for an effective software business might be a jazz group that actively seeks and adopts new ideas, is able and highly motivated to improvise but knows that there are rules that need to be followed, and has a goal-oriented, sensitive management.

In turbulent environments markets are typically rapidly changing and ambiguous, that is, software companies cannot search for and find all the information that would help reduce uncertainty and risks in business development because they do not even know which questions should be asked and answered. In such contexts, software products and services can seldom, if ever, be specified completely at the outset of the development processes. Often projects even need to be started without clear pictures of their goals and expected deliverables (Knauber, et al. 2000) because flexibility and speed are the primary organizational drivers. Competitiveness in dynamic markets thus depends heavily on the abilities of software companies to devise business models that facilitate dynamic balancing between exploitation and exploration and help converge planning and execution of actions in time to the maximum extent. Such models must enable and leverage risk taking, moving fast, and developing effective business and application domain competencies, software engineering and process competencies, organizational and information systems designs, and product development, marketing, and distribution strategies (Cusumano and Selby 1995, Cusumano and Yoffie 1998, Dybå 2000, Knauber, et al. 2000).

3. Independent and corporate ventures as creators of software innovations

A multitude of contexts of innovation, organizational designs, and business models are available for software ventures. In the following, we will discuss two contexts of software innovation and their strengths and weaknesses:

- Traditional start-ups that are financed by angel investors and venture capitalists.
- Corporate software ventures that operate as relatively independent venture projects of much larger companies.

The role of mergers and acquisitions in corporate renewal will also be addressed.

3.1. Benefits and drawbacks of software ventures

Start-ups can be more agile than industrial giants for many reasons. In the early stages of their life cycles they are typically small, have little bureaucracy, and operate in one geographical location. They can utilize simple and modern information systems because they are not burdened by outdated legacy systems. They can also focus on a few key technologies and products instead of spreading their resources simultaneously to many businesses and to both developing new products and maintaining old ones. In order to succeed, start-ups typically have to focus on niche markets and have clear missions and strategic objectives. For example,

professional service start-ups offering individually tailored systems and services specialize in creating long term one-to-one relationships with their customers. They should not try to compete with companies that offer highly productized systems and services for mass markets requiring inexpensive, highly standardized solutions and for customers willing to adjust their ways of working to adopt these solutions.

Socialization within small companies is relatively easy and people tend to know and get along with each other well, which facilitates the formation of trust. Many people are generalists rather than specialists because very strict and bureaucratic division of labor is seldom required or even possible in a small organization. For example, each software product developer may be involved in all phases from concept creation to implementation and testing. This improves *the redundancy of knowledge*, that is, people know their own and fellow workers' roles well and can develop a shared understanding of their roles with respect to the strategic mission and objectives of the company, which improves their motivation and the transparency of processes. Communication, coordination, and collaboration within small start-ups are thus relatively easy and people can improvise solutions to nonroutine situations quickly. Incentives are also high, thus increasing motivation. Building a start-up facilitates career development (at least in the U.S. and many other countries with similar cultures) even if the start-up failed. And if the company succeeds, entrepreneurs can generate enormous wealth. As a result, start-ups can innovate new product, service, and business process concepts rapidly, effectively, flexibly, and without extensive managerial control.

On the other hand, small start-ups face many challenges in their operations. They are likely to be highly dependent on a few key people. They have to develop products fast and get them to markets quickly. They also have to create brand recognition in all target markets in order to grow quickly, reap significant market share and become a major player in their industry segment. For example, significant potential customers with high growth strategies, international operations, and turbulent markets expect software products and services to be provided in a scalable way. Angel investors and venture capital (VC) companies also expect ventures to grow and internationalize quickly because this increases the valuations of the ventures and makes it possible for the angels and VCs to realize high returns on their investments as quickly as possible.

Start-ups thus operate in fairly chaotic conditions and their product creation and delivery processes are informal and unlikely to be well documented, which makes meeting these growth objectives challenging. Without a holistic, well-defined process even a small project is likely to face problems such as (Russ and McGregor 2000):

- doing an outstanding job with those development phases its members care most about and doing little or nothing about the other phases,
- quest for completeness in a rapidly changing domain, that is, by the time a team completes a phase of the development process, some of its work is out of date, and
- not knowing what to do next, thus wasting time trying to complete activities for which there is insufficient preparation.

If ventures are not able to deal with these challenges effectively, they will face difficulties in convincing potential customers about their credibility and quality. Customers will hesitate in their purchasing decisions if they cannot be relatively sure that their software suppliers can provide them with stable service levels and continuing product development. With insufficient credibility start-ups will also have difficulties attracting and recruiting enough highly skilled workers and managers.

3.2. Benefits and drawbacks of corporate software ventures

Large high-tech corporations that are dominant players in their markets typically face a major risk: an unforeseen technology

- creates completely new markets where the corporation does not operate and customers move to these markets (Porter 1980) or
- makes the current key technology obsolete thereby radically reducing or completely eliminating demand in most important markets of the corporation (Christensen 1999).

These unforeseen technologies are often developed by innovative start-ups. Corporations have two major ways of alleviating this risk: they can create or acquire ventures. The first option, corporate venturing, aims at "combining the established reputation, massive talent, and powerful resources of a major established company with the innovativeness, flexibility, aggressiveness, and frugality of a start-up to move the big company smartly ahead in sales and profits" (Vesper 1999, p. 1-26).

Corporate ventures create, experiment with, and combine new technologies to come up with new business and product concepts for mass markets that may not be mature enough to adopt such products for years or ever. Corporate ventures can be highly successful from the viewpoint of the corporation even if their concepts fail. For example, ventures can be used to create and experiment with new partnerships with other companies. If partnering with a certain company works well and yields new strategic competencies that can be leveraged in

creating new products or entering new markets, it can be exercised to create long-term collaboration even if the first concept(s) of the venture would not succeed.

According to Vesper (Vesper 1999, p. 1-29), several aspects may distinguish a venture from a conventional research or engineering project:

- "A venture connotes a more complete business, including such elements as not only technical development but also profit responsibility, overall business planning, follow-through to market, production, selling, and servicing. In contrast, a project typically is more limited to particular functional specialties and lacks profit-and-loss responsibility.
- A venture usually involves a new direction for the firm or a more radical change in the way it does business, whereas a project connotes a more limited innovation, usually in line with the accustomed strategy and direction of the company.
- A venture needs greater autonomy than a project because it fits less well with the company's customary procedures. This autonomy may come about by "hiding" the venture, by separating it geographically, or by housing it in a special organizational unit capable of shielding it from the normal company activities."

Ventures can overcome most hurdles faced by traditional independent start-ups more easily than start-ups provided that their parents have solid resource bases such as highly appreciated and well-recognized brand image, solid financial situation, innovative technology strategy, and well-functioning technology development and marketing. For example, corporate ventures can create solid businesses relatively quickly by innovatively combining existing core knowledge and technologies within the corporation. After all, brand image as well as marketing and distribution channels and expertise exist already and market creation or penetration need not be invested in as heavily as in traditional start-ups.

On the other hand, corporate ventures also face many challenges with respect to the available resources and processes. Indeed, stories of successful corporate ventures are scarcely available in the scientific literature (Vesper 1999). Ventures have to sell their ideas aggressively and compete ferociously for resources within the corporation because larger business units and other already more established ventures are likely to have more leverage and negotiation power for internal resources. In the worst cases such challenges can lead to situations where ventures stand alone without adequate support. As a result, the business objectives may be unrealistic, the business focus can be misdirected, or software technology transfer from partners and the corporation may not be executed successfully. For example, ventures can save time in the beginning of their operations by hastily adopting tools and technologies only to spend excessive time later when their systems and other deliverables are not scalable enough or do not integrate well enough with existing technologies of customers.

Corporate ventures face other resource-related challenges as well (Vesper 1999). As boundary spanners between ventures and their environment, venture leaders (i.e., venture project and product managers) typically face a huge workload. They have to interact with numerous, often globally distributed stakeholders including external customers, partners, and subcontractors as well as related projects and financiers within the corporation. Yet, incentives are not excellent. Although their own financial assets are not at risk in case of a failure, failed ventures may hamper their career development. Success can help in career development but will not generate exceptional financial wealth for any members of the venture team. Especially during economic downturns corporations are also reluctant to invest heavily in ventures and expect results for a long time. As a result, pressure to deliver fast and at low cost is immense.

3.3. The role of mergers and acquisitions in corporate renewal

Due to the above-mentioned challenges, high-tech corporations seldom, if ever, rely solely on corporate venturing for corporate renewal. This is especially true in the case of publicly listed companies that have high market capitalization. They tend to use their stock aggressively to acquire other companies, including innovative start-ups, in order to gain

- experienced staff to their management and development teams,
- spearhead technologies to complement existing product lines so that complete solutions can be sold to strategic markets, and/or
- new distribution channels and significant market positions.

Acquisitions can help corporations move quickly to new competitive positions if *both* great people *and* technologies or market positions are obtained and successfully integrated in the strategies and operations of the corporations (Cusumano and Yoffie 1998, p. 302). However, meeting these requirements is far from trivial. For example, the mergers of companies often involve significant anxiety, causing many key people to leave. Whenever acquisitions fail in meeting any of these requirements, they can significantly slow down the process of corporate renewal.

4. Design and development strategies of software ventures

This paper has outlined core business strategies for software businesses and discussed the roles of acquisitions and independent and corporate venturing in software innovation. However, the importance of design and development strategies should not be downplayed. These strategies and their implementation will determine whether the engineering organizations of software businesses are fast, light, and agile enough to enable strategic moves (e.g., rapid movement to new markets where competition is weak) that are critical for survival and the continued growth of business.

There are at least five design strategies that can be used to achieve strategic leverage over competitors (the first four of those have been presented by (Cusumano and Yoffie 1998)):

- Design a product for multiple markets concurrently. These markets exist typically in different business webs built around different operating system, enterprise software, or application platforms. This strategy can help implement multiple competitive strategies. For example, a corporate venture may try to design an innovative platform and build a business web around it leveraging the muscles of its parent. An independent start-up can design products to existing webs where the competition is weak. However, there are many challenges with this strategy. For example, it is difficult to write software that works equally well across platforms. Programming productivity and product performance are typically lower compared to platform specific products. Competitors are probably able to operate in multiple platforms as well.
- Design and redesign products to have more modular architectures. This strategy encourages organizational designs based on small teams working around shared components and facilitates co-design between partners across organizational and geographical boundaries. However, modularity cannot be achieved without some bureaucratic rules and organizational control. As a result, organizations need to balance between the degree of product and process innovativeness and the degree of modularity thay want.
- Design common components that multiple product teams can share. Componentization makes unit-testing and testing automation easier and a lot of work can be eliminated altogether through effective reuse of components. Indeed, the essence of platform based design is in modular architecture and shared components. However, this strategy is far from easy to implement. For example, product developers are likely to have the best expertise concerning the components especially in ventures that have only one or a few products in their product line. Developers may be unwilling to relinquish control of the components to a special organizational unit that would be primarily responsible for the development and maintenance of the components as corporate assets. Yet, large product-lines are challenging to develop without such specialized organizations.
- Design new products and features for parallel development. The objectives of this strategy are to reduce cycle time through overlapping work, enhance awareness of the needs of the markets and quick market movements because of reduced product release interval. Typically, the next and next-plus-one versions of the product as well as bug fixes can be done in parallel. The challenges include the difficulty of controlling release schedules and the burden involved with managing multiple versions of the same product.
- Metamodelling as an enabler of the four other design strategies. Metamodelling is important because it lets designers create domain-specific languages that are well-understood by the intended users of the software products in those domains. These languages can then be used to automatically generate software products. As a result, communication between the users and designers is greatly facilitated and complex mappings between the language of users and the languages of software design and implementation are (at least partly) eliminated or automated, thus significantly accelerating time-to-market and reducing resource requirements.

Product development strategies and processes available to software ventures also warrant further investigation. For example, fast, agile, and light but, at the same time, robust and scaleable practices need to be generated for software ventures, enabling them to establish efficient and effective product creation and delivery processes during the creation stage of their life cycles and fostering rapid but profitable growth.

5. Understanding and alleviating the challenges faced by software ventures

Individual design strategies and development processes have been investigated in great depth. There is also some material on business models available for software and e-business start-ups (e.g., Afuah and Tucci 2000, Rajala, et al. 2001). But there is still little research available that would study software businesses holistically, thus integrating research on their business, competitive, marketing, design, and development strategies. For example, I have found little previous research that would identify, prioritize, and address the risks faced by

corporate software ventures and propose socio-technical design solutions to alleviate these risks.

An international software business research community focusing on these and many other important issues is gradually forming. The community needs to deal with challenging, interdisciplinary projects and problems. *Software business* is a much broader concept than, for example, *software engineering* or *software production*. *Software business* refers to a holistic, process-driven, cross-functional, and multi-disciplinary view of software companies. According to this view, all functional areas of a software business such as product planning and management, systems and software engineering, customer support, and information and knowledge, human resource, marketing, legal, and financial management need to be addressed in a balanced way (c.f., Reo 2000). As a result, software business research can draw upon and support the strategic development of software businesses. S*oftware production* is an important functional discipline but can only realize its potential in full when it is seen in the holistic context of *software business*.

Software business research topics include but are not limited to leadership, managerial, organizational, contractual, and product creation and delivery practices as well as competitive strategies and knowledge management systems of software companies. Such a broad, multi-disciplinary view is necessary to understand these companies holistically. Without such an understanding it is difficult to suggest relatively detailed but generic solutions to specific strategic, process, product or service, and knowledge management challenges of software companies that are complex and rapidly evolving systems.

It is beyond the scope of this paper to try to present an exhaustive list of detailed software business research topics. However, a few topics are discussed to stimulate research. The *creation* stage of software businesses is an especially interesting domain. The stage is finished when the venture is ready to perform an exit successfully or has to exit unsuccessfully. Exit can be completed successfully through merger/acquisition, initial public offering (IPO), or (within a corporate context) technology transfer from the corporate venture to a business unit that can incorporate the deliverables (e.g., product (line) and business concept, key people) of the venture in its business. Unsuccessful exit typically means that the venture fails in its business and has to stop its operations prematurely. However, it is not implied that stages after a successful exit would not be important in software business. It is simply argued that strategies, policies, and systems established and collective experiences and lessons learnt during the creation stage are likely to have the most significant and long-lasting influence on the future developments of a software venture. The role of start-up ventures as major sources of new knowledge and innovations is also clear. This stage thus deserves special attention.

More research is needed on software ventures that focus on product innovation strategy, create and leverage new technologies in the area of digital convergence, target international markets, can serve as shapers of their webs, and have high growth objectives. They are likely to have much broader and deeper impact on economic, technological, and competence development of their industry segments than ventures focused on serving their current markets with incremental technology and product development.

Longitudinal analysis of software ventures should also be emphasized. It is especially useful to study the historical evolution of business models, product strategies, organizational and product creation and delivery process designs, and knowledge management systems of such ventures that have already moved well beyond the creation stage. In such studies it is possible to investigate which strategies and operations worked and which ones did not work, thus allowing the sharing of experiences from more advanced ventures to those struggling in earlier phases of their life cycles.

In the context of established ventures, it is especially interesting to study situations where a software business is making or has made a major change in its strategy. For example, many companies with an extensive background in professional software services are trying to move to product business or become players in both professional service and product business. Such transformations are important but very cumbersome to implement successfully, thus calling for extensive research in the area (Käppi 2002). Agile product development methods are a central research issue in the context of start-ups because agility is vital for start-ups to succeed in dynamic markets (Kalermo and Rissanen, 2002). Multi-project management is an interesting research issue in the context of corporate ventures. Corporate ventures can build highly innovative new products by linking previously separate concepts and technologies together if they can draw on the knowledge, components, and platforms of on-going and past projects of the parent effectively (Cusumano and Nobeoka 1998).

6. References

[1] Ashby, W.R., *An Introduction to Cybernetics*, *Chapman & Hall, London,UK (1956)*.

Afuah, A., and Tucci, C.L., *Internet Business Models and Strategies: Text and Cases*, McGraw-Hill Higher Education, New York (2000).

Boynton, A.C., and Victor, B., "Beyond Flexibility: Building and Managing the Dynamically Stable Organization"*, California Management Review*, Fall (1991), 53-66.

Brown, S.L., and Eisenhardt, K.M., "Product Development: Past Research, Present Findings, and Future Directions", *Academy of Management Review* 20, 2 (1995), 343-378.

Brown, S.L. and Eisenhardt, K.M., *Competing on the edge: Strategy as Structured Chaos*, Harvard Business School Press, Boston, MA (1998).

Christensen, C.M., "The Evolution of Innovation", *The Technology Management Handbook*, R.C. Dorf (Ed.), CRC Press LLC, Boca Raton, FL (1999), 3.2-3.11.

Cusumano, M.A., and Nobeoka, K., Thinking Beyond Lean: How Multi-Project Management Is Transforming Product Development at Toyota and Other Companies, The Free Press (1998).

Cusumano, M.A., and Selby, R.W., *Microsoft Secrets: How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People*, The Free Press (1995).

Cusumano, M.A. and Yoffie, D.B., *Competing on Internet Time: Lessons from Netscape and Its Battle with Microsoft*, The Free Press (1998).

Daft, R.L. and Weick, K.E., "Toward a Model of Organizations as Interpretation Systems", *Academy of Management Review* 9, 2 (1984), 284-295.

Dybå, T., "Improvisation in Small Software Organizations", *IEEE Software* 17, 5 (2000), 82-87.

Hoch, D.J., Roeding, C.R., Purkert, G., Lindner, S.K., and Mueller, R., *Secrets of Software Success: Management Insights from 100 Software Firms Around the World*, Harvard Business School Press, Boston, MA (1999).

Huber, G.B., "Organizational Learning: The Contributing Processes and the Literatures", *Organization Science* 2, 1 (1991), 88-115.

Jacobsen, K., Paulin, W.L., Vurpillat, V.V., Nukari, J., Peltola E., and Saukkonen, J., *Launching Your Software Business in America: A Handbook for Finnish Entrepreneurs*, Tekes: National Technology Agency, Helsinki, Finland (2001).

Kaplan, R.S. and Norton, D.P., *Translating Strategy Into Action – The Balanced Scorecard*, Harvard Business School Press (1996).

Knauber, P., Muthig, D., Schmid, K., and Widen, T., "Applying Product Line Concepts in Small and Medium-Sized Organizations", *IEEE Software* 17, 5 (2000), 88-95.

Käppi, R., *Organizational transformation from tailored software production to international Modified-Off-The-Shelf software business model*, University of Jyväskylä, M.Sc. thesis, http://www.cs.jyu.fi/sb (2002).

Nonaka, I., "A Dynamic Theory of Organizational Knowledge Creation", *Organization Science* 5, 1 (1994), 14-37.

Porter, M.E., *Competitive Strategy: Techniques for Analyzing Industries and Competitors*, Free Press (1980).

Rajala, R., Rossi, M., Tuunainen, V.K., and Korri, S. *Software Business Models: A Framework for Analyzing Software Industry*, The National Technology Agency of Finland (2001).

Reo, D.A., "The Balanced IT Scorecard: Quality of Strategy vs. Strategy Execution", European Software Institute, http://www.esi.es (2000).

Rifkin, S., "What Makes Measuring Software So Hard?", *IEEE Software* 18, 3 (2001), 41-45.

Russ, M.L., and McGregor, J.D., "A Software Development Process for Small Projects", *IEEE Software* 17, 5 (2000), 96-101.

Salo, A., and Käkölä, T., "Groupware Support for Requirements Management in New Product Development", *Forthcoming in Journal of Organizational Computing and Electronic Commerce* (2002).

Senge, P., *The Fifth Discipline: The Art and Practice of the Learning Organization*, Doubleday, New York (1990).

Seppänen, V., Käkölä, T., Pitkänen, O., Sulonen, R., and Sääksjärvi, M., *Ohjelmistoalan tutkimustoiminta Yhdysvalloissa*, The National Technology Agency of Finland, 109 (2001).

Treacy, M., and Wiersema, F., *The Discipline of Market Leaders: Choose Your Customers, Narrow Your Focus, Dominate Your Market*, Addison-Wesley, Reading, MA (1995).

Vesper, K.H., "Internal Ventures", *The Technology Management Handbook*, R.C. Dorf (Ed.), CRC Press LLC, Boca Raton, FL (1999), 1.26-1.32.