Key Decisions Affecting the Success of Commercialising Technology Innovations: Insights and Food for Thought from the Software Industry

Thorsten Kliewe[1] Philipp Marquardt[2]
1 Münster University of Applied Sciences, Muenster, Germany
2 Deloitte Consulting GmbH, Dusseldorf, Germany
(E-mail: kliewe@fh-muenster.de, pmarquardt@deloitte.de)

Abstract The paper assesses key decisions having a significant impact on the success of commercialising technology. Taking the software industry as an example, the paper analyses the characteristics of software and its market, and presents four different key decisions. The paper implies that decisions especially concerning the product/service character, value drivers, the launch date as well as the selling strategy significantly affect the commercialisation success. Looking closer into these decisions, new models have been developed. Overall, the paper draws the conclusion that commercialising software is a complex challenge which requires a strategic and market-oriented approach.
Key words key decisions, technology commercialisation, software industry

1 Introduction

Caused by major economic changes such as globalisation [1] and faster technological development [2], many companies today face the challenge of faster developing and commercialising new products, services and processes then ever before [3]. However, innovation is not only crucial to survive in existing, increasingly competitive markets [4]. The fact that innovation is seen as the most important driver of future growth is also based on its ability to develop new markets and gain sustainable competitive advantages within them [5, 6].

Recognising the centrality of innovation in today's knowledge economy, companies look to spend more and more money on Research & Development (R&D) [7, 8] and break new ground by opening their research and innovation process [9]. Integrating customers, suppliers and/or universities improves the companies' innovation capabilities [10] and helps to get new impetus, to discover new market opportunities and to develop new ideas or technologies. While this open innovation policy and an increase of R&D activities create value for the company and its different stakeholders (including its customers), this value has also to be captured. Creating value is, without doubt, a basic prerequisite for innovation. However, the significance of a target-oriented commercialisation process should not be underestimated in terms of its contribution to the innovation's market success. In this context, knowing the key decisions influencing the likelihood of a commercialisation success and knowing what needs to be taken into account within these decisions is still a not well researched field.

Therefore, this paper aims to contribute knowledge about the complex decision-making challenge facing projects in order to increase the likely success of commercialisation. Taking the business-to-business software industry as an example, the paper outlines key questions software professional should consider when commercialising new software products. However, the aim is not only to provide information to software professionals. Rather, the paper is intended to provide significant insights and food for thought for any professional involved in the commercialisation of products and services.

First, a brief literature review is provided about the characteristics of software and its market, leading to key questions within the software commercialisation process. Following that, these questions are detailed separately with different tools/charts presented. The paper concludes with directions for further research.

2 Characteristics of Software and Its Market

Since the characteristics of a product or service as well as the characteristics of the potential market have a significant impact on the commercialisation process, this chapter analyses both the software and its market in order to establish a basis for giving implications for the commercialisation process.

Thereby it has to be noted that the present paper only considers application software in business-to-business markets.

2.1 Definition and characteristics of software

Investigating the software commercialisation process requires a market-oriented definition of software. Therefore, this paper addresses the need for differentiation and distinguishes between a technical definition of software (software in a narrow sense) and a more market-oriented one (software in a broader sense). In a *narrow sense*, the term software refers to all computer programs contributing to solve information processing tasks [11]. However, this technical definition does not reflect the service offer as perceived by the customers in the market [12]. By contrast, software in a *broader sense* includes additional tangible and intangible benefits such as product documentations, consulting and maintenance services, and implementation assistance [13]. Due to the fact that these value-adding services influence the customer's buying process and consequently the software's commercialisation, the broader definition will be applied in the following.

With respect to its commercialisation, software shows some characteristics which differentiate it from many classical products. These characteristics include the software's (1) product and service character, (2) its innovation character, (3) its complexity, (4) its system and integration character, and (5) its cost structure.

- First, software vendors can provide software in three different ways: As individual software, standard software, or component software. The differentiation criterion of these three market offers is the re-utilisation ability [14]. Individual software is programmed custom-made and therefore is not at all transferable or is barely transferable to other system environments (the software is aligned to the organisational structure) [14, 15]. Standard software, on the other hand, is designed for a wide circle of customers and can be integrated in many different hardware, software and organizational structures (if necessary, the organisational structure needs to be aligned to the software) [16]. Lastly, component software connects both extreme cases (individual and standard software) and provides custom-made software by combining reusable, standardised components [17]. As a result, individual software has service character (custom-made) while standard software has product character (exists already at the time of purchase), and component software is in between (custom-made by combining existing components). However, since value-adding services complete the market offer (see definition above), each case has to be considered isolated. For example, standard software can have significant service character, if consulting and implementation services represent a big share of the market offer/purchasing price.
- Second, software has an innovative character within the purchasing organisation, since organisations mostly buy not-yet known/used software with the purpose of automating, rationalising or better managing processes of the operational structure (innovation within the buyer organisation) [15].
- Third, software is often perceived as complex due to the huge amount of functions as well as the amount and diversity of interconnections between the functions [18]. Thereby it has to be noted that the complexity is perceived subjective and depends on the customer's product knowledge (which is often times small due to the software's innovative character).
- Fourth, software has system and integration character since it interacts with other software and its environment (e.g. the software users or the company's operational structure) [19, 20]. Due to this interaction, network effects apply in many software products meaning that the software's compatibility to its environment has a significant impact on the market diffusion (e.g. easy information exchange through open interface standards).
- Fifth, development costs (e.g. programming the software, writing the documentation) of software are very high whereas (re-)production costs (e.g. burning a DVD, printing the documentation) are very low [14]. While the development costs of individual software are fully paid by the buyer, standard software vendors need to sell lots of units to cover the investment costs. On the other hand, offering standard software provides the opportunity to generate high return on sales after reaching the break-even-point [21].

2.2 Characteristics of the software market

Analogues to the previous chapter which looked at the characteristics of software, this chapter illustrates key characteristics of the software market. The analysis of the company's environment will act as the starting point to derive recommendations how to gain competitive advantages and how to avoid mistakes in the market cultivation. The software market's key characteristics include (1) market

entry barriers, (2) market structure, (3) customer loyalty, (4) lack of skilled staff as well as staff turnover, (5) software piracy, (6) buying centre structures.

- First, the market for software is characterised by high market entry barriers in the form of software engineering capabilities and fast software development (time to market issue). Furthermore, the software market displays specific market entry barriers depending on the type of software (individual, standard or component software). Financial market entry barriers for companies willing to provide individual software are quite low since the development costs will be paid by the client (low financial risk). In contrast, companies planning to entry the market of standard software will face high financial market entry barriers, since they have to take a high investment and a high risk for programming software whose market success is unclear [14, 22].
- Second, the market of software shows different market structures depending on the type of software. On one hand, the standard software market is characterised by a strong vendor concentration (due to network effects) and globalisation (due to very low marginal cost). The market for individual software, on the other hand, is strongly fragmented and more regional (however with a tendency to globalization) [23], because of nearly constant marginal costs and the fact that trust is of particular importance between vendor and client.
- Third, customers of business-to-business software are highly loyal due to high switching costs which make the switch uneconomical [24]. These switching costs are primary caused by the costly purchase of new software and its complex integration in the organisational environment.
- Fourth, the software market lacks qualified staff [25] and is characterised by high staff turnover since staff is crucial to develop high-quality and innovative products, and take them to market as soon as possible (compare chapter 2.1).
- Fifth, software piracy is daily business in the software market. The problem's cause is that after cracking the software's protection for the first time, the software can be copied limitless without any quality loss.
- Sixth, the software buying process in business-to-business markets is formalised, lengthy, collective and passes different phases (buying centre structures) [15]

The following table 1 summarises the key characteristics gained from the analysis and derives key questions to be taken into account within the software's commercialisation process.

Table 1 Key Questions Derived

#	Background	Key questions derived
1	Different ways/software types exist to solve customer problems (individual, standard or component software). Furthermore, the right value-adding services have to be chosen to create an attractive market offering.	(1) What is the right product/service strategy?
2	The commercialisation process has a goal conflict. On one hand, the software should be e.g. of high-quality, easy to use (less complexity) and highly compatible. On the other hand, the software needs to be taken to market as fast as possible.	(2) What are the right value drivers to generate and capture the maximum value? (3) What is the right launch date?
3	Software is innovative and complex, and therefore a risky product. Furthermore, the purchasing decision is made by a buying centre.	(4)What is the right selling strategy to convince the buying centre?

3 Key Decisions Affecting the Commercialisation Success

Linked with the analysis of software and its market presented in the previous chapter, this chapter discusses the four key questions derived. While the following remarks refer to the software industry, readers should not only understand the decision's significance in the software market. Rather, they should understand the decision's background and try to ask themselves whether or not this decision is also important in their own business area.

3.1 Choosing the right product / service strategy

In every business, choosing the right product/service strategy is significant for creating attractive market offerings. This is especially true in the software market due to high investment risks on the customer-side and the linked high market risks on the vendor side. Doing the decision, two aspects have

to be taken into account, namely (1) choosing the right software type and delivery model and (2) deciding which value-adding services have to be provided by the vendor itself and which can be provided by others.

The software type and delivery model

Since different software types exist, it has to be determined which best fits with the customer's problem to solve. Even if non-rational factors have an effect on the purchasing process [15], the process is primary effected by the degree to which the product solves the problem (benefits) and the price the customer has to pay for it (costs).

- *Standard software* has the best cost-benefit ratio when the customer's problems are alike, since it can be sold to a wide range of customers (advantage to individual software) and does not require any selection, modification and assembling of components (advantage over component software). However, the unit costs of standard software increase over-proportionally to the software's degree of differentiation. This means that the more versions of standard software (such as Home, Premium and Luxus) that are offered, the more the cost-benefit ratio declines. The reason for this lies in the fact that the software has to be aligned to more diversified information processing problems and other organizational processes (e.g. production, marketing and sales) are required in the vendor company.
- *Component software* has the best cost-benefit ratio when the customers' problems are diversified to a medium degree. The advantage over standard software is that component software vendors can solve problems of smaller customer groups more cheaply since they just have to select, modify and assemble components and do not have to launch a new version as standard software. The advantage over individual software is that the customer group's problems overlap to some degree so that the economies of scale apply (re-using components).
- *Individual software* has the best cost benefit ratio when the customer's problem are highly diversified, meaning that the costs for selecting, modifying and assembling components are more expensive than programming the software completely from the bottom up.

Figure 1 summarises these remarks graphically and provides a tool which vendors can use to find the optimal software type depending on the diversity of the customers' problems. Vendors can simply state the diversity on the x-axis and imagine a vertical line from this point. The curve/line which is crossed first by this vertical line shows the optimal software type for this particular case (it has to be noted that the curves are not universal; they need to be aligned to the specific case).

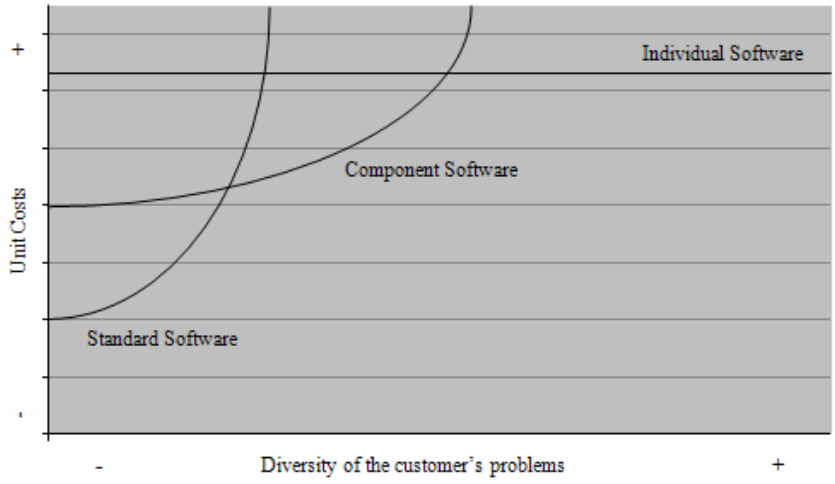

Figure 1 A Model for Finding the Right Software Type

With respect to the way how the software should be delivered, the vendor has to choose between a standalone software offering (installed on the buyer's IT infrastructure) and the Software-as-a-Service (SaaS) delivery model (installed on the vendor's IT infrastructure and usually accessed by the buyer over the internet). Factors connected with this decision are e.g.:

- The vendor's IT infrastructure
- The customers' infrastructure
- The software piracy issue (e.g. SaaS products are easier to protect)
- Cost models (e.g. according to time, bandwidth, storage or data transfer capacity for SaaS products)

- The operating life expectancy
- Implementation and maintenance costs

The need for value-adding services

Commercialising software requires looking beyond the program itself (compare the definition in chapter 2.1). Due to the fact that software is a complex and risky product, customers demand value-adding services such as product documentations and consulting, implementation and maintenance services. Offering these kind of services (before, during and after the purchase) enables vendors to differentiate themselves from competitors, to reduce the customer' uncertainty and to develop customer relationships. Therefore, vendors need first to identify demanded value-adding services before identifying and evaluating the benefits of providing the services itself, giving them to partners or just leaving it to the market. For example, maintenance services do not only generate revenue for the providing company, but also increase switching costs, contribute to developing the relationships with the maintenance customer, increase the company's reputation due to long-term relationships, and ability to identify (future) problems and needs which in turn contributes to new product development [15].

In accordance with the two outlined factors, vendor need to design their optimal market offering. Since the product and service character of this offering is significant for its marketing, their share has to be determined. Figure 2 presents a tool allowing vendors finding this share by stating the amount of value added services (price share) and the degree of integrating an external factor. Thereby, the internal factor refers to the degree to which customer can influence the market offering.

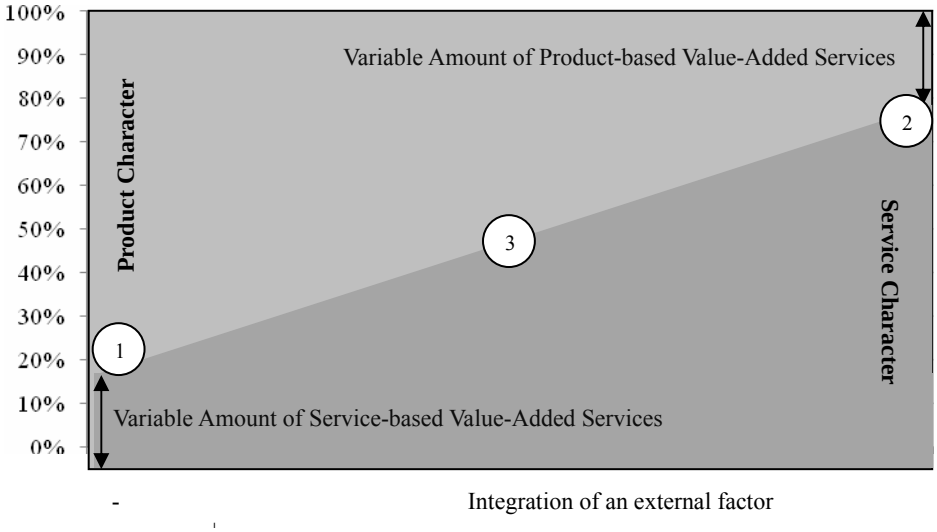

Figure 2 A Tool to Determine a Software's Product and Service Character

Figure 2 shows product and service-based value-added services on the top/bottom. For example, documentation is product-based while consulting services are service-based. Since these value-adding services influence the product and service character, the degree of integration of an external factor can only influence the products final characteristic to a certain degree (in this case the minimum service and product character is 20% with the maximum being 80%).

Three examples may contribute to understand the chart better:

- Customers of standard software have no influence on the products features (the software already exists). As a result, the integration of the external factor is zero. Assuming that the amount of service-based value-added-services is 20% of the whole price, this standard software product would have 80% product character (20% due to product-based value-added services and 60% due to the missing integration of an external factor). Compare point 1 in figure 2.
- Customers of individual software can often times define (nearly) everything within the developments process since the software is build from the bottom up. Assuming the amount of product-based value-adding services is 20% and everything is programmed custom-made with respect to the customer's problem (high integration of an external factor), the individual software product has a

service character of 80% (20% due to service-based value-added services and 60% due to the maximum integration of an external factor). Compare point 2 in figure 2.

- Customers of component software can integrate their requirements on the software to a certain degree (through selecting and modifying single components). Assuming the amount of product and service-based value-added services is 20% each, and customers can influence the development to a degree of 50%, the resulting product has an equal product and service character of 50%. Compare point 3 in figure 2.

Depending on the product and service character found by applying the model shown in 2, vendors need to developed their marketing and commercialization strategy (e.g. the higher the service character, the higher the product risk which in turn requires a strong communication of the vendor's capabilities).

3.2 Choosing the right value drivers

Since developing and commercialising software takes a lot of time, requires a large investment, needs a strategic approach (e.g. to use the network effects) and is risky, software projects should be well organised in order to create and capture the maximum value. As with many other products, software has a goal conflict between (a) developing a high-quality, easy to use and highly compatible product, and (b) taking the product to market as fast as possible. Furthermore, high switching costs in business-to-business markets necessitate that the product creates significant higher and sustainable value compared to existing software solutions. Briefly, vendors need to identify, develop and monitor the software commercialisation's value drivers.

As an example, table 2 names some value drivers, explains their legitimation and outlines the status and estimated value.

Table 2 Value Driver List

Value Driver	Legitimation	Status / estimated value
Hiring the best developers	Ensures fast development and a product of high quality.	Value generated (02/2008) US$ 20k
Internal prototype	Enables industry to imagine/understand the value of the planned software which is turn helps to get a first impression of the market acceptance. Furthermore, it provides new (external) developers with a clear understanding of the designated target.	Value generated (06/2008) US$ 40k
IP protection	Provides the vendor, investors and future interested parties (e.g. licensees) with security around uniqueness	Value generated (07/2008) US$ 20k
Flexible architecture	Ensures that the software can interact with a variety of other systems (network effects)	Not generated yet (target: 05/2009) US$ 40k
Lead user integration	Generates revenue, helps to find bugs, creates market attention	Not generated yet (target: 05/2009) US$ 40k

The following figure 3 continues the example shown in table 2 and presents a value driver map showing the value created (not the value invested) on the y-axis and the value drivers on the x-axis. The map outlines how much value is created through each value driver, how much value has already been created (when the value is created successive) and how much value is expected to be created at the end.

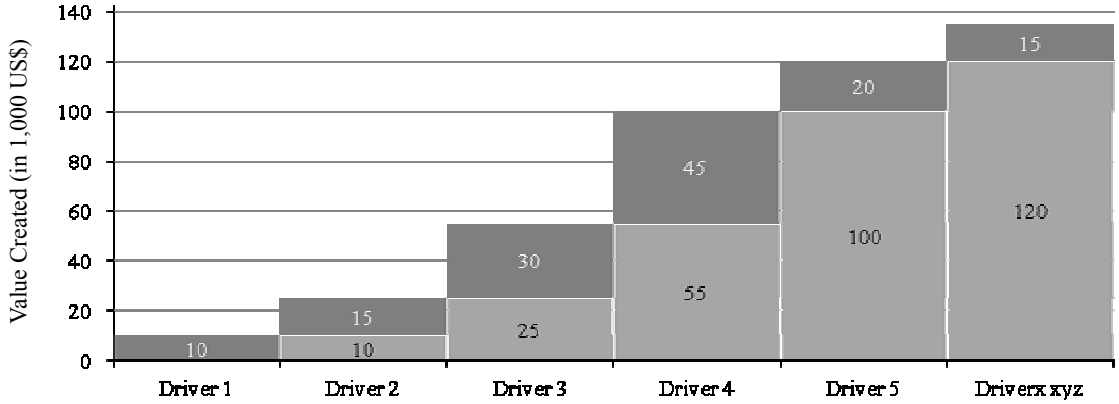

Figure 3 Value Driver Map

3.3 Choosing the right launch date

Due to the software's strategic importance in many industry sectors, business-to-business customers expect software products of high quality leading to the fact that quality is seen as a key success factor of most software products and companies. However, time also evolves more and more into a competitive factor since shorter product life cycles (PLCs) require the immediate coverage of the investment costs (standard and component software). Thereby, it has also to be noted that software has to face a significant price collapse during the PLC.

All this results in the fact that some vendors of standard and component software accept questionable compromises between the software's quality and launch date since earlier market entries increase the probability to obtain a promising market position, to set standards and to use network effects. While non software is error-free when it enters the market, vendors should consider the following factors when defining the launch date:

- The probability that an error/a bug occurs (e.g. in how many test environments has the software been tested)
- The estimated amount of damage (e.g. what effects would an error/a bug have on the customer's ability to fulfill their tasks)
- The opportunity and efforts needed to update the software afterwards (e.g. how easy would it be to detect the source of error and to solve the problem)

Within the decision about the best launch date, vendors should also consider actions which can fasten the development and therefore lower the quality risk by keeping the scheduled launch date. Examples are:

- Hiring more/better qualified staff
- Simultaneous engineering/testing
- 24/7 working days (e.g. shift work using staff in other time zones)

3.4 Choosing the right selling strategy

The commercialisation of complex products whose value is not obvious to all persons involved in the decision-making process raises the question how the customer's uncertainty can be reduced or even fully eliminated. Beside objective criteria such as product features or contractual agreements especially (1) consulting services and (2) customer qualification services [26] can be named as effective instruments. While the boundaries between consulting and customer qualification services are hard to define, their primary goals can be distinguished. Trust-based consulting services have a recommending character, whereas customer qualification services aim to educate customers.

With respect to the decision whether a software company should focus its attention primary on consulting or customer qualification services, the costs per customer has to be taken into account compare figure 4).

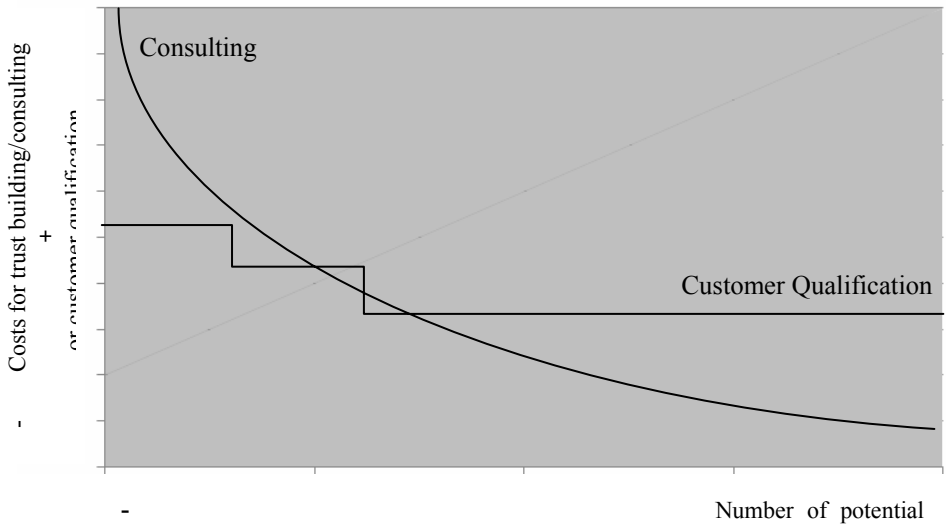

Figure 4 Consulting and Customer Qualification Services

Using *consulting services* to convince the buying centre requires trust. However, building trust is expensive due to high costs for building customer relations and brand development. Furthermore, trust develops over time and results can therefore not be gained quickly. But, building confidence by satisfying customers and showing physical evidence of success (e.g. a magnificent building or being present in important journals) pays off when targeting bigger groups. This is because the high costs are allocated to a wide range of (potential) customers resulting in lower costs per person. In other words: the bigger the target group, the less the costs per person for trust building. However, it has to be noted that the costs for consulting services per person are constant, since no economies of scale can be realised (every customer has to be consulted). As a result, the reduction in cost per person is getting smaller and smaller the bigger the targeted client group (compare figure 4).

Customer qualification costs are constant since each customer has to be qualified by itself. However, with a rising number of potential clients it can be worth to reduce the qualification cost by providing customer qualification products (e.g. brochures or product documentations) or services (e.g. workshops bringing together different clients) to more than just one customer. This usage of the economies of scale is shown in figure 3 with the "steps".

4 Conclusion

Today's markets are characterised by intensification and rapid changes of the competitive landscape leading to a strong need for successfully commercialising innovations. This paper looked at the business-to-business software market with the purpose of identifying and explaining key decisions affecting the commercialisation success of software products. Thereby, decisions with respect to the product and service character, value drivers, the launch date as well as the selling strategy have been found to be crucial. Taking a close look to these decisions different new models have been developed and found to be valuable in the decision-making process. Overall the paper has shown that commercialising software is a complex task requiring a strategic and market-oriented approach. While the paper provided significant insights into the software business, it was also intended to provide food for though for other industry sectors.

In accordance with this, different recommendations for future research can be given. First, research needs to empirically validate the results presented. Second, research is required to investigate to what degree the results can be generalised meaning that further research needs to look at other industry sectors in order to determine their specific key decisions within the commercialisation process. Third, research needs to look at the decisions itself and to analyse the need for adaptation or generalisation. For example, other markets should be examined with respect to the factors influencing the decision if mass production (standard software), mass customisation (component software) or custom-made (individual software) is the most adequate market offering. Beyond this, research investigating value drivers and

their legitimation across different industries in order to provide a practical usable toolbox for commercialisation professionals would be of high value.

References

[1] Gummesson, E. Relationship Marketing in the New Economy. Journal of Relationship Marketing, 2002, 1(1): 37-58

[2] Santoro, M. & Chakrabarti, A. K. Firm Size and Technology Centrality in Industry-University Interactions. Research Policy, 2002, 31: 1163-1180

[3] Szerb, L. The Changing Role of Entrepreneur and Entrepreneurship in Network Organisations. In: Lengyel, K. (ed.) Knowledge Transfer, Small and Medium-Sized Enterprises, and Regional Development in Hungary, 2003: 81-95

[4] Siguaw, J. A., Bakes, T. L. & Simpson, P. M. Preliminary Evidence on the Composition of Relational Exchange and its Outcomes: The Distributor Perspective. Journal of Business Research, 2003, 56: 311-322

[5] IBM. Expanding the Innovation Horizon: Global CEO Study. 2006

[6] Tucker, R. B. Driving Growth Through Innovation. San Francisco, Berrett-Koehler, 2002

[7] Tübke , A. & van Bavel, R. The 2006 EU Survey on R&D Investment Business Trends. JRC Scientific and Technical Reports, 2007

[8] European Union. Monitoring industrial research: the 2007 EU industrial R&D investment scoreboard. 2008

[9] Chesbrough, H. Open Business Models: How to thrive in the new innovation landscape. Boston: Harvard Business School Press, 2006

[10] Howard, J. The emerging business of knowledge transfer: Creating value from intellectual products and services. Technical report, Australian Government: Department of Education, Science and Training, 2005

[11] Seibt, D. Organisation von Softwaresystemen. Wiesbaden: Gabler, 1972

[12] Müller, R. Erfolgsfaktoren kleiner und mittlerer Softwareproduktunternehmen im Lebenszyklus. Frankfurt am Main: Verlag Peter Lang GmbH, 1999

[13] Müller, M. Strategien für Software-Unternehmen. Zeitschrift für Planung, 1990, 4

[14] Gerhardt, T. Strategie und Struktur in der deutschen Softwareindustrie. München, 1991

[15] Baaken, T. & Launen, M. Software-Marketing. München: Vahlen, 1993

[16] Wolle, B. Grundlagen des Software-Marketing. Wiesbaden: Vieweg+Teubner, 2005

[17] Schuster, E. http://wi2.wiwi.uni-augsburg.de/downloads/gifiles/WKBA2/s051.pdf, Accessed on 27 September 2006

[18] Baecker, D. Einfache Komplexität. In: Ahlemeyer & Königswieser (ed) Komplexität managen: Strategien, Konzepte und Fallbeispiele, 1997, 21-50

[19] Kremar, H. Informationsmanagement. Berlin: 2000

[20] Teubner, A. Organisations- und Informationssystemgestaltung. Wiesbaden: Gabler, 1999

[21] Kittklaus, H.-B., Rau, C. & Schulz, J. Software-Produkt-Management. Berlin: Springer, 2004

[22] Anderson, C. A world gone soft. A survey of the software industry. Economist, 1996, 3-22

[23] Hoch, D. J., Roeding, C. R. & Purkert, G. Erfolgreiche Software-Unternehmen. München: Carl Hanser Verlag, 2000

[24] Shy, O. The economics of network industries. Cambridge University Press, 2001

[25] Bundesministerium für Bildung und Forschung. http://www.bmbf.de/pub/it-forschung_2006.pdf, Accessed on 28 September 2006

[26] Baaken, T. Qualifizierung des Kunden als integrative Aufgabe im Technologie-Marketing. In: Töpfer, A. & Sommerlatte, T. Technologie - Marketing: Die Integration von Technologie und Marketing als strategischer Erfolgsfaktor, Landsberg/Lech, 1991, 203-219