University of Augsburg
Prof. Dr. Hans Ulrich Buhl
Research Center
Finance & Information Management
Department of Information Systems
Engineering & Financial Management

Universität Augsburg
University

Discussion Paper WI-93

On the Impact of Standardization on the provision of ERP-Systems as Mission Critical Business Infrastructure

by

Bernd Reitwiesner, Stefan Volkert

June 2001

in: Dittrich, K., Egyedi, T. M., ed., Standards, Compatibility and Infrastructure Development. Proceedings of the 6th EURAS Workshop, Delft University of Technology, Delft, (Netherlands), June 2001, Delft University of Technology, Delft, 2001, p.183-202

Universität Augsburg, D-86135 Augsburg
Visitors: Universitätsstr. 12, 86159 Augsburg
Phone: +49 821 598-4801 (Fax: -4899)
www.fim-online.eu

On the Impact of Standardization on the provision of ERP-Systems as Mission Critical Business Infrastructure

Bernd Reitwiesner, Stefan Volkert

Dipl. Wirtsch. Inf. Bernd Reitwiesner
Dipl. Wirtsch. Inf. Stefan Volkert
University of Augsburg, Business School, Department of Information Systems
Universitätsstr. 16, 86135 Augsburg
{Bernd.Reitwiesner|Stefan.Volkert}@WiSo.Uni-Augsburg.DE

On the Impact of Standardization on the provision of ERP-Systems as Mission Critical Business Infrastructure

Abstract – Although, from a technical point of view, business frameworks appear to be a promising means to provide the industry with commercial, mission critical business management software in a cost effective manner and within a competitive time to market, up to now, only very few framework vendors like IBM or Navision have entered the market for ERP-software. In this paper we discuss factors for framework based software to be successful.

When analyzing the economics of software development with or without the usage of frameworks, there are two particularities which have to be considered: one of them is the network effect, which gives an advantage to software products already established in the market. The other is the effect of compatibility decisions, which allows the compatible software product to profit from the network effect of the product being compatible with. Therefore standards play a crucial role in the market game, as they can be considered as a means to provide compatibility and interoperability.

We carry out our analysis by means of a microeconomic model, which incorporates these two effects and allows to derive recommendations on the strategic positioning of a software vendor in the competition for standardization. In many cases competitors having a high market share are likely to obstruct standardization processes. But we can show, that for some vendors of standard software based ERP-packages the support of the standardization process can be the more preferable strategic option.

We finish the paper with an analysis of welfare implications of standards for the development of ERP-Software, pointing out that socially optimal results are not guaranteed by market forces themselves. Therefore, governmental bodies are asked to foster standardization processes in this field, to support the provision of a high quality software infrastructure for industry.

1 Introduction

Development of information systems by frameworks has been discussed for some years as one solution to the rapidly growing demand for flexible and extendable business applications, which have to be available in reasonable time, with reasonable costs at an adequate quality level.[1] The software industry is, compared to other industries dealing with physical goods, rather at its beginning. Therefore the development of other industries can serve as a role model for the software engineering branch. The vision of using software frameworks is to adopt construction principles successfully established, e.g. in the automobile industry: you take a platform and put the required standardized modules together according to the customers' needs. So that finally a fully individualized car - although consisting of standard components - leaves the assembly room. In software terms, the platform is the framework and the modules are the components which fit into the framework. The aim is to be able to develop business software which is as specific as individual software, but a cost comparable to that of a standard software-based solution, thereby combining the advantages of both methods. Although framework-based industrial style development of individual business applications is a great vision, there are not really many application frameworks available or could be regarded as established in the market for enterprise resource planning (ERP) software. Apart from IBM with its San Francisco Framework only a few smaller software companies adhering to framework technology. But recently two developments can be observed: firstly, further important software companies are announcing their new strategies, as e.g. Microsoft's .NET and Oracle's .NOW initiative, in which frameworks play a major role. Secondly, producers of standard software are adapting to changing markets, too by publishing their interfaces and thereby enabling third parties to offer software add-ons for their systems. In the future we can expect further interesting changes in the market for ERP-Systems. By analyzing the software market for ERP-Systems one can see that the - so far missing - success of the framework technology is not really a matter of technology issues[2] but merely a question of economic and political matters like market share, market entry barriers, standardization and so on. Therefore in this paper we want to take the bird's eye view onto the competition between standard software solutions and framework based solutions for enterprise resource planning systems. By analyzing the market positions of these two technologies by means of a microeconomic model, we will be able to point out the important role of compatibility and standardization in this competition. Furthermore we will be able to address social welfare issues and come to a conclusion whether governmental intervention might be a suitable means to develop the market.

Before we can go into a detailed analysis of the market situation, we have to describe the competition between these two technological approaches to the provision of software solutions for business applications and work out the specifics of this competition. Furthermore, since there is are no unique notions of frameworks and standardsoftware, we first have to clarify the types of software we want to have a look at.

STANDARD SOFTWARE

---

1 See Ferstl / Sinz / Hammel et. al. (1997), p.24.

2 Ivanov (1996), Philippow/ Ivanov (1997) and Philippow / Ivanov / Preißel (1998) give a good survey of the problems arising in this area.

Although there are some definitions for standard software we focus in our paper on the type of business software outlined by its task: to support several domains (e.g. logistics, sales, accounting, human resources management and production data management)[3]. The major feature of this type of standardsoftware is the ability of customizing the software to the specific needs within an organisation, without changing the sourcecode. This is done by adjusting an extensive set of predefined parameters.

FRAMEWORKS AND FRAMEWORK BASED SOFTWARE

Component-based software or Componentware means a concept of developing application systems in which the software is neither produced solely for one system nor it is relying on just one piece of standardsoftware. A component is a piece of software engineered for reuse. In order to deploy components an infrastructure is needed which coordinates the single components thereby fitting them together. This infrastructure is called "framework". Therefore a framework is also a software. In a broader sense the notion framework stands for a set of methods aiming to enhance the productivity of software development by means of reuse. The scheme of a framework is held as generic as possible to enable fast and flexible adaptations referring to each customers' needs. The framework user completes his framework by adding the required application components. In general, several types of frameworks can be distinguished[4]. In our paper we concentrate on domain-specific frameworks, which support the development of applications in a certain problem domain.

THE SPECIFICS OF THE COMPETITION BETWEEN FRAMEWORKS AND STANDARD SOFTWARE

Apart from the technical and development problems mentioned at the beginning of this introduction, there are some particularities which have a great impact onto the relative market positions of the competing technologies. They are related to the network effect, being a positive externality, which causes the utility of a consumer of a certain product to increase with the number of other users of the same product. Network effects play an important role in software markets and may result from different causes. For standard software, a potential new user of a standard resource planning system will trust the processes implemented in the software the more the more other companies are already using the same product of a certain vendor and thus having made it the backbone of their business.[5] In addition to such more psychological effects, network effects may result from the need of interoperability, e.g. if two companies want to cooperate in a business network to form a virtual organization and thus being dependent on the ability to support inter-organizational business processes with the help of information systems being able to interoperate. For framework based solutions the number of components available fitting into a specific framework will increase with the number of users of the same framework.

If two different types of standard software, two different types of frameworks or a framework and a standard software are able to interoperate or if one of them is able to call the functionality of the other, we call them compatible.[6] One product can profit from the network effects of

---

[3] Mertens, Bodendorf, et al, (2000)

[4] For different classification schemes see e.g. Schmitzer, B. 2000 and Ivanov, P. 1996

[5] We get evidence for that statement, if we have a look at SAP, which advertises it's software by proclaiming that "more than half of the world's 500 top companies" are using its software. (See http://www.sap-ag.de/company/profile_long.htm; downloaded 5/20/2000)

[6] An exact definition of the notion of compatibility we want to use for the purposes of this paper will be given in chapter 2. An overview of different types of compatibility can be found e.g. in Shy (1996) pp. 253 and in Wiese (1997).

another product, if it is partially or fully compatible. Standardization, if a standard is accepted by the majority of market players, allows the different products available in a domain to be equally compatible to each other. But we can look at it also the other way round: the need for compatibility may lead to the emergence of a standard either as a proprietary de-facto-standard or as a true standard by undergoing a formal standardization process. Thus compatibility or standardization decisions have a great impact onto the competition in software markets.

We want to analyze the competition between framework technology and standard software on the basis of a market model which allows us to incorporate the impact of network effects and compatibility decisions onto the market positions of these two competing technologies. We introduce the model and its basic assumptions in chapter 2, which allows us to gain first results about the market shares and profits of these two technologies when we are not assuming a standard to be established. In chapter 3, we want to analyze how the introduction of a standard influences the competition. Since we can assume standard software to have a great network advantage compared to the framework technology, we enter into the question whether standard software industry could block the market entry of a framework technology vendor and how a standardization will affect the chances for a market entry of the new technology (chapter 4). Welfare considerations in chapter 5 close our analysis, before we summarize the main policy implications for software industry and standardization organizations.

2 The competition between the technologies

By means of customization through parametrization standard software vendors promise to offer solutions fitting the individual business needs of the user. But taking no account of its customization facilities, standard software will not allow for solutions fitting exactly to the customers' requirements and being as individual as individually developed software.[7] Customization often means the adoption of the users' business processes to the reference processes supported by the standard software, but not vice versa. For many emerging application domains, like supply chain management or customer relationship management, standard software is not able to provide solutions at best practice levels on time. [8]

On the contrary, framework based solutions allow for business applications being as individual as individually developed software[9] by means of adoption and extension of available or insertion of additional components. If one takes into account that, for example a typical application developed using IBM's San Francisco Framework consists to a degree of 40% of the framework and its related components, which are completed by an individual user interface, country and industry specifics, business rules and individual components,[10] it is obvious that a framework based application can be considered to be exactly matching the users needs. Even better, standard components can be used for standard application domains like financial accounting and individual or individualized components can be composed into the framework for the most mission critical parts of the information system, where a differentiation from

---

[7] See Vaske (2000), p. 30.

[8] There might be economic reasons as well for the standard software vendor for not offering software at best business practice levels. For quality considerations based on an economic perspective when designing standard software solutions see Reitwiesner / Will (1997).

[9] See Schmitzer (2000), S. 10.

[10] See Scheer (1998), p. 116.

competitors is desired. E.g. when looking at a bank, this could be the components dealing with risk management.

Although framework-based solutions can be regarded as optimally fitting the users needs, as opposed to standard software which is mainly designed to support the reference processes and variations coverable through customizing, standard software based solutions are not necessarily cheaper to develop than framework based solutions. This is due to the fact that in enterprise wide implementation projects of standard software the mere customization is only one of the necessary project activities. Independently from the chosen technology, in every ERP-implementation project the phases business modeling, requirements engineering, analysis and test are required. Furthermore, the costs for the redesign of the established business processes to be compatible with the reference processes of the standard solution must not be left out of consideration. Therefore, we consider the implementation of a framework based solution not to be more expensive than a standard software based solution to be a very reasonable rating of the respective cost situations.

To be able to analyze the competition between the established technology of standard software and the (still) emerging framework technology, we have to abstract from the technical details and problems linked to framework technology and reduce our analysis to the factors determining the market position of these two technologies in the market for enterprise resource planning systems (ERP systems):

- the degree to which a standard solution meets the business requirements of a user;
- the number of users already having installed a software of a certain type and the impact of the size of this installed base on the utility of the users, usually denoted as *network effect*;
- compatibility and / or standardization decisions.

We capture these factors in a market model which is based on Hotelling's[11] approach to model product characteristics as locations in a linear product space.

2.1 The Model and related assumptions

We assume the competition to occur between two suppliers *S* and *F*, both offering a complete enterprise resource planning system. *S* provides its solution as a standard software, *F* is providing solutions based on a framework and the related necessary components. We describe an oligopolistic competition, the number of participants can be reduced to 2 without loss of generality. The competition between *S* and *F* is characterized in detail by the following assumptions:[12]

(A1) Software users are considered to have requirements which can be characterized as dots on a horizontal line with the length 1. We assume the users to be uniformly distributed on this interval, so that at each point represents a single user. The position of an individual user is denoted by $h \in (0,1]$.

(A2) The total demand for software is fixed, it is completely inelastic. The suppliers compete for the size of their share of the total demand. Every consumer is supposed to buy one (marginal) unit of software. The total demand of all users is normalized to 1.

---

[11] Hotelling (1929). A survey of this category of microeconomic modeling gives Shy (1995) as well as Pfähler / Wiese (1998).

[12] The assumptions and the structure of the model follow a model presented in Pfähler / Wiese (1998), pp. 295. We modify this model by introducing asymmetric product characteristics and by taking marginal and fixed production costs into the considerations.

(A3) The software solutions are (horizontally) differentiated in the following way: the framework solution provided by *F* is supposed to meet exactly the requirements of any user along the horizontal line. The solution provided by *S* using the standard software approach is supposed to meet the requirements of users whose requirements are characterized by position a_S in the linear product space, i.e. the standard software is located at one point (denoted by a_S) in product space. All users with requirements different from a_S have to face "distance costs" (see figure 1), which are a linear function of the difference between the (position of the) users requirements and the (position of the) properties of the software solution.

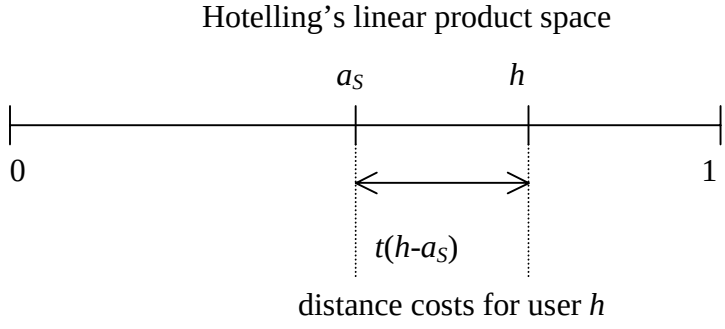

Figure 1: Distance costs in Hotelling's linear product space

The distance costs $d_S(h)$ and $d_F(h)$ can be computed as:

$$d_S(h) = t \cdot | h - a_S | \qquad \text{for the users of } S \tag{1}$$

$$d_F(h) = 0 \quad \forall h \in (0,1] \qquad \text{for the users of } F \tag{2}$$

The distance costs quantify the monetary equivalent loss of a user with requirements *h* when having to use a software solution with properties a_S. *t* represents the cost unit rate per distance: the higher *t*, the more important it is for the user to have a software which exactly meets his needs and the higher is the monetary equivalent loss per unit of distance in product space.

(A4) *S* is supposed to choose the position $a_S = 0{,}5$ in the middle of the product space, since he maximizes the size of the symmetric interval of users on the left and the right hand side possibly choosing his solution.

(A5) Both suppliers are assumed to produce with identical marginal costs and the same costs per unit of software *c* respectively. Although we assume both suppliers already to be established in the market, there are identical fixed costs *FC* for both suppliers every period to adjust the software to current developments. These fixed costs include the costs for a possible changing of the degree of compatibility s_S and s_F respectively. The arguments at the beginning of this chapter show that the assumption of identical costs for both types of technologies is not at all unrealistic. Furthermore, it allows to emphasize and to focus our analysis on the strategic role of the factors pointed out at the beginning of this chapter.

(A6) We assume duopolistic competition, i.e. the market shares of both competitors are greater than zero.

(A7) Both suppliers are supposed to maximize their profits, which can be computed as

$$\pi_l = (p_l - c) \cdot x_l - FC \quad l \in \{S, F\}. \tag{3}$$

They maximize their profits only with regard to the current period. Each supplier is fully informed about his own cost structure, the cost structure of the competitor and about the preferences and the willingness to pay of the users.

(A8) The users choose the solution with the greater consumer surplus CS_l (with $l \in \{S, F\}$). The consumer surplus consists of the basic willingness to pay Z minus the effective price of the software solution p_l, minus the distance costs $d_l(h)$ plus the utility derived from the network effect. Thus, the consumer surplus of a user with address h when purchasing a solution from F or S can be formulated as:

$$CS_F = Z - p_F - t \cdot 0 + en_F \tag{4}$$

$$CS_S = Z - p_S - t \cdot | h - a_S | + en_S \tag{5}$$

$e > 0$ is a measure for the general valuation of the network effects of the users and is assumed to be equal for standard software and framework based solutions. n_l denotes the network size:

$$n_S = x_S^i + x_S^{ex} + s_S (x_F^i + x_F^{ex}) \tag{6}$$

$$n_F = x_F^i + x_F^{ex} + s_F (x_S^i + x_S^{ex}) \tag{7}$$

The network size n_l of a technology is determined by three factors:

- the size of the installed base x_l^i (i.e. the number of installations resulting from the last periods);
- the expected size of the market share in the current period x_l^{ex};
- the participation in the installed base and the expected market share of the competing technology $s_l (x_m^i + x_m^{ex})$ $(with\ m, l \in \{S, F\}\ and\ l \neq m)$.

The *degree of compatibility* $s_l \in [0,1]$ determines to which degree technology l participates in the number of installations of technology m sold in the previous and the current period. If e.g. F decides to be fully compatible to the technology of S by choosing $s_F = 1$, then his users not only profit from the network effect of technology F, but fully profit from the network effect of technology S as well.

(A9) The users are supposed to have complete information about the prices and the degrees of compatibility of both technologies. In particular with regard to the expected market share x_l^{ex} $(l \in \{F, S]\})$, we assume that the users have rationale expectations, i.e. the users' expectations about the market share of technology l exactly match the result of the competition. Formally this can be described as:

$$x_l^{ex} = x_l$$

(A10) The competition between F and S takes place in two stages: in the first stage (*compatibility competition*) the competitors decide on the degree of compatibility s_F and s_S they will choose when designing their products. In the second stage (*price competition*) the de-

grees of compatibility and by that the network advantages/disadvantages of the technologies are fixed and the competitors try to price their products in a way that maximizes their profits. The decisions on both stages of the competition have to be made simultaneously, i.e. competitor *F* doesn't know how *S* will decide and vice versa. The outcome of the decisions of the first stage determines the strategic position of the competitors in the price competition. The technologies can be incompatible $(s_I = 0)$, partially compatible $(s_I \in]0,1[)$ or fully compatible $(s_I = 1)$.

Beware that compatibility is not a symmetric relation. Let us, for instance have a look at the competition between the Compact Disc (CD) and the new Digital Versatile Disc (DVD), there we have asymmetric compatibility: one can't play DVD's on a CD-player, whereas it is possible to use CD's on a DVD-player. Looking at software, we have asymmetric compatibility if software vendor *A* knows the interface specifications of the software product of software vendor *B* and can make function calls to the software of *B* whereas *A* doesn't publish its interface specifications thereby not giving *B* any possibility to be able to produce compatible to *A*. For the purpose of this paper we ***define the degree of compatibility*** as follows:

- We treat the compatibility between two technologies, i.e. the compatibility between the standard software solution as a whole and the framework (including all related necessary components) as a whole. We do not look at the compatibility between the framework and its single components[13].
- Partial compatibility is assumed to have the following meaning: a degree of compatibility e.g. 0.6 of software A with software B means that 60 % of the functionality of software B can be used / called up by software A.

Using these assumptions, we can deduct the demand x_F for solutions based on *F* as well as the demand x_S for solutions based on *S* in the current period. Each user chooses the solution providing the greater consumer surplus (see (A8)). Thus, a user chooses the framework if the following inequality holds:

$$CS_F \geq CS_S \Leftrightarrow | h - a_S | \geq \frac{1}{t}[(p_F - p_S) + (en_S - en_F)] \tag{8}$$

We see that users prefer the framework based solution, if their *distance* from the properties of the standard software based solution is greater than a possible ***price advantage*** $(p_F - p_S)$ or a possible ***network advantage*** $(en_S - en_F)$ of the standard software based solution. For matters of clarity we introduce the following simplifying assumption:

(A11) The indifferent consumer prefers the framework based solution.

If we resolve the absolute value $| h - a_S |$ with the help of a fall differentiation $(h \geq a_S \; vs. h < a_S)$ and take into account (A4) we get the demand for solutions from *F* and *S*:

$$x_F = 1 - \frac{2}{t}[(p_F - p_S) + e(n_S - n_F)] \tag{9}$$

$$x_S = 0 + \frac{2}{t}[(p_F - p_S) + e(n_S - n_F)] \tag{10}$$

---

[13] We assume full compatibility between the framework and its single components as well as between the components itself.

Taking into consideration the assumption (A9) of rationale expectations of the consumers about the technology decisions in this period, we can rewrite the demand functions (9) and (10) to:

$$x_F = \underbrace{1}_{a} - \underbrace{2\lambda[(p_F - p_S)}_{b} + \underbrace{e(\Delta n^i + s_S - 1)]}_{c} \tag{11}$$

$$x_S = \underbrace{0}_{a} + \underbrace{2\lambda[(p_F - p_S)}_{b} + \underbrace{e(\Delta n^i + s_S - 1)]}_{c} \tag{12}$$

with:

$$\lambda = \frac{1}{t - 2e(2 - s_S - s_F)} \qquad \text{intensity of competition} \tag{13}$$

$$\Delta n^i = (x_S^i - x_F^i) + (s_S x_F^i - s_F x_S^i) \qquad \text{network advantage for } S \tag{14}$$

The network advantage consists of an advantage resulting from the installed base $(x_S^i - x_F^i)$ and a compatibility advantage $(s_S x_F^i - s_F x_S^i)$. With regard to the intensity of competition we make the following assumption:

(A12) We assume the intensity of competition λ to be positive.[14]

As the demand functions are fundamental for our further investigations, let us have a look at the single terms and their economic interpretation. Furthermore, we will be able to draw some first conclusions on how the parameters will work. Interpretation of (11) and (12) will be easier if we split them up into three terms:

(a) The natural market share:

As it was assumed in (A3), software produced with the help of frameworks can be considered as tailored directly to the customer's requirements, i.e. customers are not confronted with distance costs. Therefore, a customer being asked to decide between framework based software and standard software will, when neglecting any other factors we come to later on, always choose framework based software. In our formulae this aspect is represented by the resulting natural market share of 1 for frameworks and zero for standard software.

(b) The price advantage:

For the explanation let's assume standard software is offered at a smaller price than framework based software. The greater the price difference the more customers will decide for the standard software based solution and thereby increase the basic market share of the standard software based solution.

(c) The network advantage:

This term incorporates the network advantage which is very important in software markets as discussed in chapter 1. For the term to be comprehensible, let us assume that standard software is already established on the software market and can profit from its large

[14] By this assumption we exclude the need to take into account very specific properties of demand functions which are usually assumed for Giffen-Goods. See Pfähler / Wiese (1998), p. 311.

number of installations in former periods whereas framework based software has just entered the market. The greater the difference between the number of installations of standard software and framework based software, the greater the market share of standard software will be. Beware that *F* can reduce his network disadvantage by choosing a high degree of compatibility s_F, since the degrees of compatibility s_S and s_F have an impact onto the term Δn^i via the term $(s_S x_F^i - s_F x_S^i)$. While Δn^i represents the network advantage resulting from former periods, the remaining term (s_S-1) represents the impact of the sales differences in the current period onto the network advantage. Therefore, the network advantage will work for *S*, if the advantage from former periods is larger than the total demand of the current period[15] less the degree of compatibility of *S*: $\Delta n^i > 1 - s_S$.

Of course, (b) and (c) may also work negatively for the standard software vendor, in case standard software is more expensive than framework based software or the installed base of frameworks (x_F^i) is greater than that of standard software (x_S^i). The effects of (b) and (c) will be factored by the intensity of competition λ. This is a measure, to what extent price and network advantages influence the market share.

Now, having explained the general structure of the market share formulae, we are able to present some interesting first results:

(S1) The greater the sum of the two degrees of compatibility the smaller the intensity of competition λ.

(S2) *A unilateral increase of the degree of compatibility does not have a unique impact on demand.*

This phenomenon results because of the effect of a variation of the degree of compatibility onto the intensity of competition as well as onto the network advantage and can be shown by discussing the first order differentiation $\partial x_S / \partial s_S$ and $\partial x_F / \partial s_F$ respectively. With respect to a network advantage, increasing uniliteral compatibility of the competitor causes a decreasing network advantage. Thus, increasing compatibility favors the competitor with the network disadvantage. The intensity of competition will decrease along with an increasing degree of compatibility s_S or s_F. A smaller intensity of competition will result in the network advantage and price advantage (b) and (c) to be less important for the market share. This favors the competitor with the price / network disadvantage. A unilateral increase of the degree of compatibility thus has a positive effect for the competitor with the price / network disadvantage. For the competitor with the price / network advantage, the effect may be ambiguous. He is disfavored by the decreasing network advantage while the network advantage itself becomes less important because of the decreasing intensity of competition. Which of the two contrary effects will be stronger depends on the explicit values.[16]

(S3) *Fully bilateral compatibility minimizes the intensity of competition and reduces any network advantages or disadvantages to zero.*

The main question raised in the introduction was the question for the market position of frameworks in the competition with standard software. Our analysis of the demand for the respective technologies yields a clear natural preference for the framework technology and shows that the standard software can get market shares greater than zero only if it can offer

---

[15] Which is assumed to be 1, see (A2).

[16] This result corresponds to a result of the model we used as a basis for our analysis, see Pfähler / Wiese (1998), p. 311.

either price or network advantages. Our brief discussion of the market situation in chapter 1 shows, that we can assume the standard software technology to have a significant network advantage. Whether *S* can use the network advantage to prevent *F* from entering the market will be discussed in chapter 3. *F* can reduce the network disadvantage by choosing to produce at least partially compatible to *S* (i.e. $s_F > 0$). Since a market share greater than zero is not enough for a technology to be successful, we analyze the respective profit of *F* and *S* in the resulting market equilibrium in the next chapter 2.2.

2.2 Results of the price competition with given compatibility

Let us assume for the moment, that both vendors accept their degree of compatibility as given. The strategic variable left is the price of their product. In chapter 3 we will extend our model to a two step-approach for the case of standardization and thereby allow the degree of compatibility to be a strategic variable as well. As we have assumed duopolistic competition (see (A6)), it is necessary to take into account the reaction of one's competitor when determining one's own price. The profit maximizing Nash-equilibrium price for *F* and *S* can be found at

$$p_F^B = \frac{1}{3}[3c + \frac{1}{\lambda} - e(\Delta n^i + s_S - 1)] \tag{15}$$

$$p_S^B = \frac{1}{3}[3c + \frac{1}{2\lambda} + e(\Delta n^i + s_S - 1)] \quad . \tag{16}$$

The corresponding market shares are

$$x_F^B = \frac{2}{3} - \frac{2}{3}\lambda e(\Delta n^i + s_S - 1) \tag{17}$$

for the framework vendor, and

$$x_S^B = \frac{1}{3} + \frac{2}{3}\lambda e(\Delta n^i + s_S - 1) \tag{18}$$

for the supplier of standard software. At the bottom line, the following profits can be earned:

$$\pi_F^B = \frac{4}{18\lambda} - \frac{8}{18}e(\Delta n^i + s_S - 1) + \frac{4}{18}\lambda e^2(\Delta n^i + s_S - 1)^2 - FC \tag{19}$$

$$\pi_S^B = \frac{1}{18\lambda} + \frac{4}{18}e(\Delta n^i + s_S - 1) + \frac{4}{18}\lambda e^2(\Delta n^i + s_S - 1)^2 - FC \tag{20}$$

In chapter 2.1 we have already had a closer look at the demand functions x_F and x_S. In the market equilibrium, the natural demand for framework based technology takes the value 2/3 and the natural demand for the standard software based solution takes the value 1/3. As far as a variation of the degrees of compatibility is concerned the statements (S1), (S2) and (S3) do still hold.

When analyzing the individual profit functions, there are two relevant questions:

(a) Is the profit greater than zero and therefore the extremum we calculated for the profit in the equilibrium really a maximum ?

(b) Which competitor makes more profit?

Although question (a) is difficult to answer from analyzing π_F^B (19) and π_S^B (20), we can prove both profits to be greater than zero by analyzing (15) in combination with (17) and (16) in combination with (18) respectively: we can show that prices turn out to be greater than the marginal costs *c*, as long as the market shares are greater than zero. The market shares are

always greater than zero because of assumption (A6). Thus both competitors can gain a contribution margin greater than zero and therefore gain profits greater than zero as long as the fixed costs *FC* are sufficiently small.

The answer to question (b) depends very much on the set of parameters. The condition for π_F^B to be greater than π_S^B can be written as:

$$\pi_F^B > \pi_S^B \Leftrightarrow \frac{1}{6\lambda} - \frac{2}{3}e(\Delta n^i + s_S - 1) > 0 \qquad (21)$$

Again, we can state that the profit of the framework vendor will surmount the profit of the standard software vendor in case of a small intensity of competition[17] and small network advantages of the standard software over frameworks (which e.g. can be reached by a large s_F). Hence, we can point out that for typical scenarios, where the number of software installations of *S* succeeds those of *F* to a great deal, it is necessary for a success of framework based software to be sufficiently compatible.

Thus, at the end of this chapter we are able to answer one of the questions raised in our introduction: basically, if frameworks are able to get a market share larger than zero at all, then the development costs for the Framework can be covered. Furthermore, extra profits can be drawn under the conditions shown in this section. We were able to point out that a large degree of compatibility of *F* favors this development. But, as we discuss at the beginning of the next chapter, it may be difficult for *F* to be able to reach a high degree of compatibility with software from *S*. In the following chapter we discuss how standardization can promote the market position of frameworks.

3 Introducing a standard into the competition

In the last chapter, we assumed each of the competitors decides on his own whether his software should be compatible with other products and to what extend. We have seen already, that due to the network effects, for some settings a high degree of compatibility may be advantageous. But we have to bear in mind that, in case of software, a profit maximizing degree of compatibility usually cannot be reached without agreement of the vendor of the product one wants to be compatible to. This is true mainly for two reasons. The first reason is technologically oriented: to be compatible, in our setting, means to be able to use functionality provided by some other piece of software. Therefore, one has to know the detailed specification of the interfaces and is dependent on their publication. It is needless to say that compatibility can only be reached if the software provides interfaces at all. The second reason is of legal nature: in most countries software is accepted as an intellectual property, that means the owner solely can decide on the terms of its use.

As interoperability, usually reached by compatibility, in the field of information technology is so important, standards play a crucial role. From our software-perspective we consider a standard to be a common set of functionalities which can be incorporated within a piece of software, two or more parties have agreed on. Interoperability between software components which adopt the standard is guaranteed. The standard is adopted by a software vendor to the extent his own functionality is compatible with the common set. Software standards can be set either by a governmental non-profit-organization, or, more often by a manufacturers' consortium. We should not neglect that because of their importance in the software market, some

[17] Remember that the intensity of competition is small if the degree(s) of compatibility are large (see (S1)).

single software vendors are able to set *de-facto*-standards. Nevertheless, for the discussion in this paper it is not necessary how the standard has been or will be set.

In the following we discuss the effects of a common standard within our economic model. A common standard leads to the same degree of compatibility for both technologies. When replacing s_F and s_S by s within formulae (15) to (21) the findings of the last chapter are applicable in this context as well, so that we do not need to repeat the analysis and can pay attention to the question of an optimal degree of standardization.

THE OPTIMAL DEGREE OF STANDARDIZATION

We assume both competitors to be willing to agree to a standard. Nevertheless, they might have divergent preferences as far as the degree of standardization is concerned. Thus we analyze what profit-maximizing degree of standardization each of the vendors would strive to individually in such a standardization game. Primarily, we want to know, whether in the case of a substantial network advantage of the standard software based solution, *S* will vote for low and F will vote for a high degree of standardization *s*.

We enter into that question by analyzing, for which size of installed base of *S* profits of *S* and *F* increase or decrease with an increasing degree of standardization. I.e. we look for the range of the network advantage of S, where $\partial\pi_s^B / \partial s$ and $\partial\pi_F^B / \partial s$ is greater or smaller than zero.

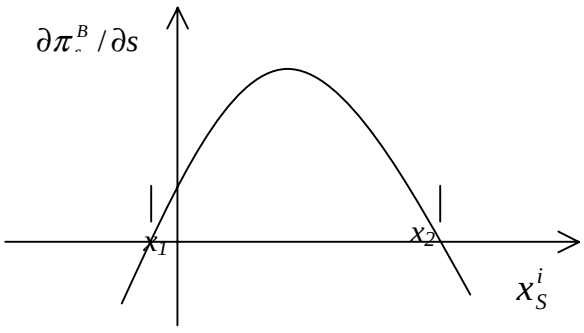

Figure 2

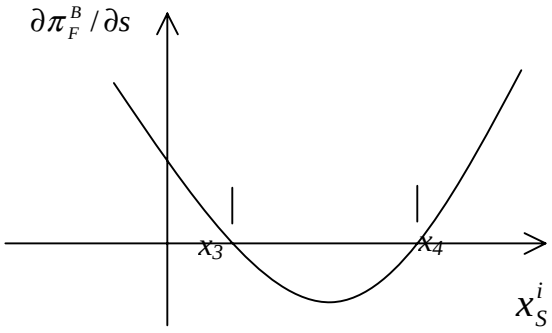

Figure 3

(a) Profit analysis of S with variable s, depending on x_S^i

Figure (2) shows the dependency between $\partial\pi_s^B / \partial s$ and x_S^i. The graph is a parabole with two zeroes. The right zero always is a positive value of x_S^i. The left zero may be a positive or even negative value of x_S^i. The figure shows that we have at least one area (for $x_S^i > x_2$.) in which an increasing degree of standardization reduces the standard software vendor's profit. On the contrary, we see that for a sufficiently small installed base of the standard software vendors product (for $x_1 < x_S^i < x_2$.) an increasing degree of standardization will be favored by the standard software vendor as well. Thus, the established standard software vendor should not counteract ambitions to increase the degree of standardization in any case but only if his installed base is sufficiently large.

(b) Profit analysis of F with variable s, depending on x_S^i

Figure (3) shows a typical dependency between $\partial \pi_F^B / \partial s$ and x_S^i. The graph shows a parabole with two zeroes. The interesting section is $x_S^i > x_4$: we have an area in which an increasing degree of standardization increases the profit of *F*. That is because he is able to make use of the network effect caused by *S*'s installed base. [18]

(c) Profit analysis for s = 1

s =1 means both competitors fully support the common standard. Interoperability is guaranteed to its maximum extent. We analyze the profits by comparing the corresponding profit-functions and get the condition

$$\pi_S^B \geq \pi_F^B \Leftrightarrow \frac{t}{6} \leq 0 . \qquad (22)$$

From (A3) we know that *t* always is positive. Therefore condition (22) cannot be fulfilled. This shows that the profit of *F* will always be larger than the profit of *S* if it comes to total standardization. As a result, one can imagine that *S* will try to avoid that a mandatory standard will be enforced.

We finish this chapter by pointing out that the market itself does not guarantee that both vendors work along to support a standard. Most probably the contrary is true: companies with a network advantage will strive for small values of *s* whereas companies with a network disadvantage are likely willing to take part on the exploitation of their competitor's network effect by increasing *s*. We come back to this phenomenon in the last chapter, where we discuss policy implications for the standard software vendor and the vendor of the framework-based ERP-software.

4 Can *S* prevent the market entry of the framework supplier *F* ?

In the introduction we raised the question, whether it is worth to continue the investments into the development of framework technology from an economic point of view. So far, we have analyzed possible market shares for the framework technology and tried to find out whether in the case of a market share greater than zero the framework supplier will be able to regain his development costs for the framework and to show the conditions for which he can gain extra profits. Now we will enter into the question whether there is a chance that the standard software vendor can prevent the mere market entry of the framework technology vendor. In reality, under certain circumstances the market entry can be considered as a failure even if the framework vendor can a gain a positive market share which is too small to be considered as relevant. But in this formulation the condition for a market entry failure is too fuzzy to be covered by a formal model. Generally, the market entry of F can be considered as a failure, if the market share x_F^B of F in the Nash-equilibrium (see equation (17)) equals zero. In the model we also analyze the scenario for a market share smaller than zero ($x_F^B < 0$) to be able to get a better insight into the impact of the different factors onto the market process. To analyze that formally, we have to adjust some assumptions to this new situation of one competitor not being already established in the market:

(A13) Since we discuss a possible market entry of *F*, we can be sure that the size of the installed basis equals zero: $x_F^i = 0$.

[18] For values of s sufficiently close to 1, the profit cannot be increased when s is increased.

(A14) S is assumed already to be established. Thus S, when deciding about his policy, has to take into account only the regular periodical adoption costs FC, which occur every period. Whereas F will have to take into account the full amount of development costs for the framework and the related components. We denote them by FC^{entry} and hence assume them to be greater than FC: $FC^{entry} > FC$.

Additionally, (A6) is not assumed to hold for the purposes of this chapter.

Without any specific assumptions with regard to the degrees of compatibility, we can basically state that the prevention of the market entry of F is possible, if the installed base of S's standard software x_S^i will be sufficiently large. This result is not at all trivial, especially if one recalls the fact that the "natural demand" for standard software solutions is zero (see equation (12)) since there are no distance costs for users of the framework technology.

We formally show that for the case of standardization. If we assume standardization, the market share of the framework vendor will be forced below zero if the following inequality holds:

$$x_F^B(s) \le 0 \Leftrightarrow \frac{1}{\lambda} \le e(1-s)(x_S^i - 1) \tag{23}$$

$$x_F^B(s) \le 0 \Leftrightarrow x_S^i \ge \frac{\frac{t}{e} - 3 + 3s}{1-s} \tag{24}$$

We can compute the limit degree of compatibility, up to which market entry prevention is possible:

$$x_F^B(s) < 0 \Leftrightarrow s < 1 - \frac{t}{e(3 + x_S^i)} \tag{25}$$

We see that the greater the installed base x_S^i of the standard software vendor, the greater is the possible degree of standardization s up to which market entry prevention is possible. Nevertheless, if standardization allows to enforce full compatibility ($s = 1$), inequality (25) will not hold for any x_S^i. Setting a standard at a high degree of standardization can thus help to keep the market open for emerging technologies. The higher the degree of standardization which can be reached in the standardization process, the more difficult the will be c.p. the market entry prevention.

5 Summary and Policy Implications

ERP-Systems provide industry with the information systems infrastructure necessary for their daily business. Vendors of traditional standardized software, which can be adopted to the customers needs by parametrization, compete with the vendors of framework-based software in this important market for ERP-systems. Both types of vendors face different challenges, which result from the different product characteristics of the software solutions offered and their different market positions. Therefore we analyzed the competition between standard software based and Framework-based software solutions for the provision of ERP-systems by means of a game theoretic model.

As opposed to standard software based systems framework-based systems can fully meet the requirements of the user and therefore are advantageous compared to standard software based solutions as far as the mere product characteristics are concerned. Despite these preferable product characteristics, there are not really too many business frameworks offered which sup-

port the implementation of ERP-systems based on standard components. On the contrary, vendors of traditional standard software still dominate that market. This is due to the high relevance of network effects for the selection of standardized business software. Therefore, a large installed based of an established ERP-vendor can be a real market entry barrier for Framework-based software. The investment into the development of a business framework and its related components from scratch yields a high risk due to this market entry barrier. Thus it is a real question, whether a single software vendor should take that risk to develop this new kind of software, which promises to provide industry with a better software infrastructure for business.

We could show, that if the Framework-based solutions can master the market entry barrier, it will be possible to regain the investments necessary to develop the new technology. Moreover, the vendor of the Framework-based solution can gain extra profits under the conditions shown in section 2.2. The risk of the development of this new technology therefore is worth being taken as long as there is a chance to successfully enter the market. The network advantage of the established software vendor is the main obstacle to a market entry. That's where the important role of compatibility and standardization comes to influence the competition. A widely accepted standard for the interaction of software components can reduce the market entry barriers by enabling the Framework-based software to participate from the network effects of the established software products. Furthermore, if it is possible to set a standard which enables full mutual compatibility, the vendor of the Framework-based software will always make greater profits than the vendor of the standard software. Nevertheless market entry prevention is also possible in case of an existing standard, if the degree of standardization which results from the standardization competition is small enough (i.e. smaller than the limit degree of compatibility). Thus the established vendor of standard software should not obstruct the standardization process. On the contrary, he should try to participate in the standardization process in order to reach the optimum degree of standardization with respect to its own market position.

On the other hand, although the standard software vendors can be considered to be the established market players, they have to decide about new strategies to face the challenges of the future competition in the market for ERP-systems. This is due to the fact that nearly all of their main customers - large industrial companies - are already well equipped with this type of software. Therefore the number of potential new customers is getting smaller and smaller, since smaller companies often can't afford this expensive type of software. Therefore they also have to decide how they can survive in the long run. Mainly two reasonable strategic options can be seen: one of them is to improve the possibilities for the adaptation of the standard software to the user specific requirements, the other is to decompose the existing software into components which can be sold individually to those customers which can't afford or wouldn't need the whole system. The effect of these two strategic options onto the relative market positions of the two types of software vendors can also be discussed on base of our model and its results.

The first of the two strategic options tries to compensate the central disadvantage of the standard software which has to be customized by means of parametrization: compared to framework based software this type of software solutions does not allow to exactly meet the customers needs. The customers therefore draw less utility out of the use of a standard software-based solution than they would be able to draw from a Framework-based solution. In the model, this disadvantage can be balanced by giving the consumers a price advantage or by giving them a network advantage. In reality, the product characteristics of the standard software based solution could be changed as well in order to provide the user with a higher degree of fit to his individual requirements. To obtain that, one could try to make the parametrization facilities more generic. But to stick to the parametrization paradigm is one of the central pre-

requisites for an upgrade which maintains the investments into the existing tailoring of the standard software.[19] Therefore the value of this strategic option is limited, since parametrization only allows for the variation of the way a specific function is executed at runtime, but not for the variation or completion of the predefined functionality itself.

The other strategic option, which is already being made use of by some vendors of standard software, is to decompose a large monolithic software system into components, which can be used and sold individually.[20] As a consequence, this software is becoming interesting for smaller companies, which couldn't afford the system as a whole. But if a company wants to make effective use of these single components, it has to be able to combine them with software from other vendors. Therefore, the established software vendor is dependend on the existence of a standard for the interaction of components as well. As a consequence, if he sticks to this standard, he will risk to lose his network advantages resulting from the high installed base.Thus, if the established standard software vendors realizes the strategy of the componentization of his existing system, he has to carefully decide about the degree of compatibility he is going to implement when defining the components.

In chapter 3 we could show with regard to that question, that if there's only a small network advantage resulting from the installed base a cooperation in the standardization process leading to a high degree of standardization seems to be the reasonable thing to do. If the network advantage resulting from the installed base is sufficiently large on the other hand, the limit degree of compatibility should not be exceeded, since market entry prevention is still possible up to this degree of compatibility. If it comes to total compatibility (i.e. $s = 1$), the provider of the Framework-based solution will always make higher profits than the provider of the standard software based solution. The higher the degree of standardization which results from the standardization competition, the more the vendor of the standard software based solution will have to work on the improvement of the product characteristics, i.e. the improvement of the customization facilities to reach a higher degree of fit to the users requirements. If a potential improvement of the customization facilities does not promise to really lead to a great improvement of the product characteristics, the obstruction of the standardization process will be the better thing to do for the established provider of standard software based solution, since the market entry barriers are being preserved.

In reality, the question whether the componentization of an existing monolithic system will be the right choice for the established vendors of standard software based ERP-systems to face the challenges provided by the desirable product characteristics of component-based systems will also depend on further factors not directly covered by the model. The electronic commerce with its increasing number of transactions between different companies being automatically processed on electronic marketplaces also fosters the demand for standardized business processes. So there are factors which could lead to a high degree of standardization which can't be controlled solely by the software vendors. If it can be foreseen that the market entry of the providers of Framework based solutions cannot be prevented, the effective embarkment on the strategy of the componentization of existing systems is strongly recommended for the vendors of 'traditional' standard software. The software users surely would welcome such a development, since it would lead to a better provision of business management software, which is the core infrastructure for successful business in the information age.

---

[19] See Brehm/Heinzl/Markus (2001), p. 7.

[20] E.g. SAP is planning for the future, also to sell individual components of mySAP.com, instead of only selling the system as a whole (see N.N. (2001)).

6 Limitations and Outlook

The model analysis applied here was designed quite simple to ensure mathematical tractability and easy interpretation. This obviously implies a number of limitations. In the following three of the most obvious limitations will briefly be discussed as well as ways to overcome them.

- In (A5) we assumed the marginal costs as well as the fixed costs for *S* and *F* to be identical. We presented arguments for this assumption to be reasonable, nevertheless different costs can be included in our model.
- We have used a very simple notion of compatibility. This is sufficient for our aim to discuss the relationship between the standard software and the framework as a whole. Economic literature provides more complex models which would allow to include a triple relationship between providers of pure frameworks, components and standard software.[21]
- Our model does not incorporate time. Observing software markets, one has to agree that time-to-market plays an important role. An extension of our model could incorporate the preferences of users for time-to-market as an element of the effective price in the same way we included the distance costs.

7 References

Bohrer, K.A. 1998: Architecture of the San Francisco Frameworks. In: IBM Systems Journal 37 (1998).

Brehm, L./Heinzl, A./Markus, M.L. (2001): Tailoring ERP Systems: A Spectrum of Choices and their Implications. In: Proceedings of the Hawai'i International Conference on System Sciences, January 3-6, Maui, Hawaii.

Ferstl, O./Sinz, E.J./Hammel, C./Schlitt, M./Wolf, S. 1997: Bausteine für komponentenbasierte Anwendungssysteme. In: Höhere Mathematik und Datenverarbeitung (HMD) 197/1997, pp. 24 - 46.

Gamma, E./Helm, R./Johnson, R./Vlissides, J./Booch, G. 1995: Design Patterns – Elements of Reusable Object-oriented Software, Addison-Wesley.

Hotelling, H. 1929: Stability in Competition. In: Economic Journal 39, pp. 41-57.

Ivanov, E. 1996: Entwicklung und Anwendung von Frameworks – Probleme und Lösungsansätze, 1996, p. 3-8. (http://www.theoinf.tu-ilmenau.de/~ivanov/scripts/studie.ps, downloaded 12/2/1999)

Mertens, P./Bodendorf, F./König, W./Picot, A./Schumann, M.: Grundzüge der Wirtschaftsinformatik, 6. Auflage, 2000.

Philippow, I. / Ivanov, E. 1997: Methodische Entwicklung von Frameworks. In: 42. Internationales Wissenschaftliches Kolloquium der TU Ilmenau 22.-25.9. 1997, Band 1, pp. 15-23.

Philippow, I. / Ivanov, E. / Preißel, R. 1998: A Method for the Development and Application of Frameworks. The Third Conference on Integrated Design & Process Technology (IDPD, incorporated ESPA, IEEE), Berlin, July 6-9 1998, IDPT-Vol.4, pp. 938-946.

[21] See Wiese (1997) and Shy (1995) pp. 253

Pfähler, W. / Wiese, H. 1998: Unternehmensstrategien im Wettbewerb, Eine spieltheoretische Analyse, Berlin, Springer Verlag.

N.N. 2001: SAP verkauft MySap auch in Modulen. In: Computerwoche, 27. Jahrgang (2001) Nr. 5, S. 1.

Reitwiesner, B. / Will, A. 1997: Best practice oder common practice: Eine Frage der Wirtschaftlichkeit. In: Wirtschaftsinformatik 39 (1997) 6, pp. 640 - 641.

Scheer, A.W. 1998: ARIS - Vom Geschäftsprozeß zum Anwendungssystem. Berlin, Springer Verlag.

Shy, O. 1995: Industrial Organization, Theory and Applications, Cambridge (Mass.), MIT Press.

Stiel, H. 2000: Sticht Desktop-Management den Framework-Ansatz aus? – Klein anfangen, statt groß scheitern. In: Computerwoche 19/2000, pp. 14-16.

Schmitzer, B. 2000: Klassifikationsaspekte betriebswirtschaftlich orientierter Frameworks. FORWIN-Bericht-Nr.: FWN-2000-06, Bayerischer Forschungsverbund Wirtschaftsinformatik, http://www.forwin.de/forwin_berichte.html (downloaded 2000/03/12).

Vaske, H. 2000: Individualsoftware trotzt den “Modewellen” in der IT. In: Computerwoche 8/2000, pp.30 - 31.

Wiese, H. 1997: Compatibility, Business Strategy and Market Structure - a Selective Survey. In: EURAS Yearbook of Standardization, Volume 1, München, Accedo Verlagsgesellschaft, pp. 283 - 308.

Woeckener, B. 1999: Network Effects, Compatibility Decisions, and Monopolization. In: Zeitschrift für Wirtschafts- u. Sozialwissenschaften (ZWS) 119 (1999) 1, pp. 23 - 44.