Pricing of Software Product Lines

Denis Sewerjuk

University of Leipzig
Information Systems Institute
Marschnerstr. 31
04109 Leipzig, Germany
sewerjuk@wifa.uni-leipzig.de

Abstract: This paper examines possibilities and consequences of introducing variability with respect to pricing, accounting, and way of payment into software product lines for developers, vendors, and customers. The global goal is the derivation of an extensible conceptual framework for vendors of SPLs which adds relevant economic variability to the SPL. The capability of economic methods for pricing and accounting for SPLs to provide the desired variability model for setting prices and a variability model for licensing is examined and adjusted. Furthermore, an analysis of possible or existing interdependencies between different methods of pricing, accounting and the way of payment is provided. Nevertheless, a comprehensive validation of this framework is required which will only be marginally discussed in this paper. The proposed framework is not a fixed roadmap but rather a white-box framework which a vendor has to tailor according to his strategic and economical goals. In order to achieve this additional guidance is provided.

1 Introduction

In recent years software product lines (SPL) have emerged as the most promising approach for software development with re-use [CE00, Ba06, Kr06, PB05]. Well-known institutes are researching this topic, like SEI Carnegie Mellon University (www.sei.cmu.edu), LERO (Limerick, Dublin) (www.lero.ie) and Fraunhofer IESE (www.iese.fraunhofer.de). Furthermore, SPL's have been successfully applied in practice and a reasonable number of success stories are reported in literature [BC96, Se07]. The number of highly innovative companies which develop and provide tool support for software product lines is growing. Software product lines bridge individual software development with its advantage of customer-oriented development and standard software with its benefit of fast and relatively inexpensive availability of ready-made solutions.

According to various definitions (e.g. [Ko99], [Ka90], [Wi96]) a software product line is a set of products that are closely related and share a common, managed set of features. The relation depends on different economic criteria among them – and most important -

homogeneous customer groups. Thus, a software product line satisfies the specific needs of a particular market segment, similar distribution channels and given price ranges.

SPL products are characterized by features. A feature is a requirement or a property which is relevant to any stakeholder of the SPL's domain. A customer is interested in features and a vendor is selling a customized SPL-variant with these features and serves the special customer needs. Features span up the variability space and the possibility to create system variants that are specifically adjusted to customer needs, theoretically up to the point where a software system is unique with respect to the individual customer. This is an important prerequisite to charge a price that is oriented towards the value a system variant creates for the customer. Furthermore, it’s difficult if not impossible for the customer to compare this system variant with others.

Since software development is a highly competitive market those vendors will survive who offer customer-oriented solutions and who gain recognition in the market, the economical basics become more important. The first condition is that a vendor can sell to the customer a highly customized product with a lot of functional variability (SPL variant). But an equally important condition that has not been addressed so far is the economical foundation for pricing, accounting and the way of payment for these systems created as members of a product line. Often the emphasis is on value and revenue with respect to cost savings of a software product line and not about the opportunities how to set prices with other economical methods [Ba06]. The economical variability inside an SPL, as described in this paper, provides an additional benefit for both vendors and customers. A vendor gains substantial potential in skimming the highest possible revenue per customer with a suitable price policy and a highly customized product thus becoming irreplaceable with regard to competitors through unique selling propositions. Benefits for the customer are gaining economic influence and control ordering his customized product and the possibility to communicate his preferred price, account and the way of payment. The goal is that a vendor sells a very variable product with a high economical variability to address as many customers as possible.

2 Software Economy

Software economy examines, describes and develops models for price determination and accounting of software in the form of products and services. Regarding the selection possibility of special features of software it tries to answer the following questions:

- How can software be licensed?
- How can the price for software be specified?
- How can software licences be sold?
- What are possible strategies for selling software?

For determining a price the literature suggests several price formation procedures [Si92, Di00, Me00, Pe05, Pe98] which can be more effective than just decreasing sales prices in order to achieve more sales. Generally, the adoption is common that the cost structure of software products makes traditional price formation procedures often impossible [Cl06]. The reason for this is that software products are digital goods and thus have a set of peculiarities compared to material goods. Digital goods are immaterial services for the satisfaction of customer needs. They can be developed by information systems, distributed and occur in form of products or services. An example of a digital product as a complex service is the electronic handling of procurement transactions [Cl06, p. 238]. Software products usually have high one-time production costs (fixed costs) and low duplication costs (variable costs). Furthermore, digital goods do not have any capacity restrictions. Software products are optionally often reproducible without loss of quality or content. Thus, the multiple possession of the same product is possible [Cl06, p. 239]. In addition, digital goods can be changed and distributed without considerable costs, so that cost-effective offering of product versions is much easier [Cl06, p. 242]. Further specifics are that a value gain can result from division or frequent use, for example discussion forums, knowledge platforms or utilisation of communications - as well as present economies of scale and economies of scope [Sk05, p. 2].

As mentioned before there is a set of procedures which can be used to determine the price of the software product. After the first price is set a lot of questions arise for the vendor: "Which price should be asked of which customer?" and "Which accounting conditions should be offered to the customer especially for this price?".

The first question concerns price differentiation. It is the most important instrument for the vendor to gain from the diversity of the customers and offer an almost equal product to different customer segments with different prices [Fa03, p. 485].

The second question concentrates on the accounting, the definition of the payment type on the previously determined price. Both aspects, price and accounting of software, are two elements of software economy. On the one hand the question should be clarified, what are the costs for software development [Bo84, p. 4-21]. On the other hand the information about how to sell software products must be clearly outlined [Co96].

Pricing policy

Today many firms face slowing sales and growth. Pressured by competitors and customers, companies are forced to set lower prices and making their position in the market even worse (downward spiral). When competitors are aggressively pushing prices down and customer demand lower prices, the vendor should reduce the volume of his product, to keep his impact on profitability. Otherwise, a competitive price war could start and this would teach the customer that the demand of a lower price is every time rewarded by a lower price [Cr02, p.1]. The end of such a war is hard to predict.

Pricing policy includes all considerations, decisions and actions of a vendor which are directly related to the price of a product and which have the goal of keeping the revenue of the company as high as possible. Generally, a price could be earned directly by the product or indirectly by sponsoring, advertisement etc. [Ze99].

The price strategy consists of three parts:

1. Price formation,
2. Price aggregation
3. Price differentiation

Price formation

Price formation means how the price for a software product is determined. As mentioned above, various price formation procedures are available [Bl03, Di00, Si92]. A common price formation *criterion* is *cost-orientation*, which observes the production and/or development process of the vendor. The cost-oriented price formation is applicable to all software products, because they all have a similar cost structure, i.e. fixed development costs and variable costs for different services. The most important weakness of the cost-oriented price formation is the disregard of the information about the customers and the competition [Bl03, p. 141].

This is served by market-oriented price formation procedures. In detail, these are the demand-oriented and the competition-oriented price procedures. The initial point of *demand-oriented* price formation is the connection between price and demand, expressed by a demand curve [Bl03, p. 143]. This curve shows the quantitative reaction of the customer as a consequence of a certain price (and/or its change) of the supplier. The *competition-oriented* price formation acts in accordance with the prices of the competitors and the quality of the products and deals explicitly with the fact of the competition actions and reactions [Bl03, P. 143].

Besides the presented classical price formation procedures the *value-based* price formation exists. This formation includes all procedures which are suitable to determine the value and the incentive of a product range for the customer. This among simple cost-benefit-calculations and all well-known methods, which help to determine the relative customer benefit, such as various questioning procedures (direct, indirect, benchmark questionings) or the Conjoint-Measurement [Bl03, p. 143], [Sr06].

Price aggregation

The price aggregation procedures determine the price for a product variant. If the customer selects certain features of a software product line, the prices of the features (for example documentation, artefacts, additional achievement and the like) become an aggregation price (sum of the unit prices). The result is a total price of the software system variant.

Price differentiation

The market, on which the vendor acts, usually consists of customers, who differ regarding various characteristics. These characteristics (for example value, income, sex, age and so on) have influence on the price the customers is willing to pay for a product (maximum price). So the goal of price differentiation is to address as many customers as possible and to skim maximal customer's surplus regarding the diversity of the customers [Fa03, p. 485].

The performance-related price differentiation, also termed "quality-price differentiation", is connected to the software product line approach. Software system variants or similar products (differentiation in the scope of works) are offered at different prices [Sk99, p. 287].

Accounting

The accounting answers the following questions:

- "What will be accounted?" – determination of the accounting criteria and
- "How will be accounted?" – definition of the accounting type (for the whole product or a accounting criterion) as well as
- "How does the vendor get the money?" – determining a payment type.

Accounting criteria are the features of a product which have a price.

The accounting types are divided into use-independent and use-based accountings. Use-based means that for the usage of an accounting criterion a price has to be paid. Use-independent means that the price for an accounting criterion is always to be paid, no matter whether an accounting criterion was taken up or not [Ze99, p. 27], for example a base charge.

The use-based accounting types are divided into single transaction and the frequency of use. The single transaction takes the price for the usage of an accounting criterion depending on the parameters quantity and time. The frequency of use criterion has a certain volume (quantity), for example an encyclopaedia from which the customer only wants to use a certain part and for the use of this part he pays a price. Another example is that the accounting criterion depends on the duration of use (time), for example counting the online time of a customer by an internet service provider. With the frequency of use the price will be aggregated and placed in a bill at the end of a given period [Sk06].

The use-independent accounting type is divided into two parts, namely unique and recurrent accountings. A unique accounting type characterises the blanket payment of a price (for example use basic price, license price and so on) of the whole product or an individual accounting criterion. The recurrent accounting types are periodic payments of the customer. Examples for this category are payments for service and support, maintenance etc., payable monthly or annually.

In addition to the accounting type there is the payment type, namely how and on which conditions the money is transferred from the customer to the vendor (cash, debit, cash on delivery etc.). The conditions support an additional price differentiation, for example discounts, bonuses, allowance and so on.

A feature model for accounting could have this structure, shown in Figure 1:

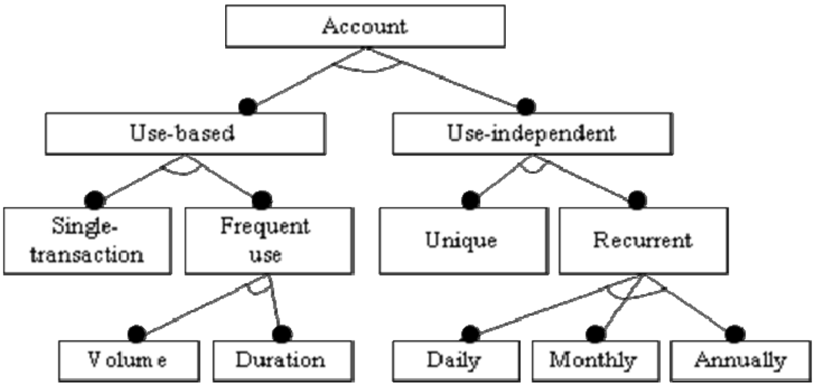

Figure 1: Example of an accounting feature model

2 Pricing of Software Product Lines

The emphasis is on the possibilities for customers and vendors which exist when using software product lines with respect to price strategy and accounting.

All the commonalities and variability of a software product line are described in the *problem space*. This reflects the desired range of applications ("product variants") in the software product line (the "domain") and their interdependencies. So, when producing a product variant, the application developer, the vendors and customers use the problem space definition to describe the desired combination of problem variability to build the product variant [Be06]. The problem space also defines illegal feature combinations, default settings, and default dependencies (some defaults may be computed based on some other features). In the configuration view, the problem space consists of domain-specific concepts and their features with prices and accounting. The specification of a given system requires the selection of features that the desired system should have [CE00, CHE05].

When combining features into a product variant all the mentioned procedures of the price policy are possible: price bundling, price differentiation (shown in Figure 2: aggregation) and price negotiations (discount etc.) (shown in Figure 2: aggregation and differentiation).

An associated *Solution Space* describes the constituent assets of the Product Line (the "platform") and its relation to the problem space, i.e. rules on how elements of the platform are selected when certain values in the problem space are selected as part of a product variant [Sc06]. The solution space has the knowledge about how price and accounting could be combined, and which particularities exits between selected features and their prices.

Figure 2 show the price policy for a product variant of a software product line. As mentioned above, the first step is to form a price for every feature. A feature can have multiple price formation set by the different procedures named above. After a vendor has a price of a feature, a customer select features into his product variant. A vendor has to set feature prices and feature price classification numbers for each feature. Classification numbers could be a ratio, which is needed to know how to aggregate the price using two features in a bundle or the ratio for the accounting, to know which discount should be calculated when the feature is paid immediately cash. It is easy to predict that with growing complexity the effort will increase significantly. That’s why a vendor should set his price definitions and combinations in a wise way, driven by the granularity of the features. Therefore, features should be grouped according to the feature model to reduce the complexity. A customer could use a wizard to customize his product variant when the complexity of a software product line is not very high. When a product line has a high feature complexity, best practice configurations and rules should be used and offered to the customer.

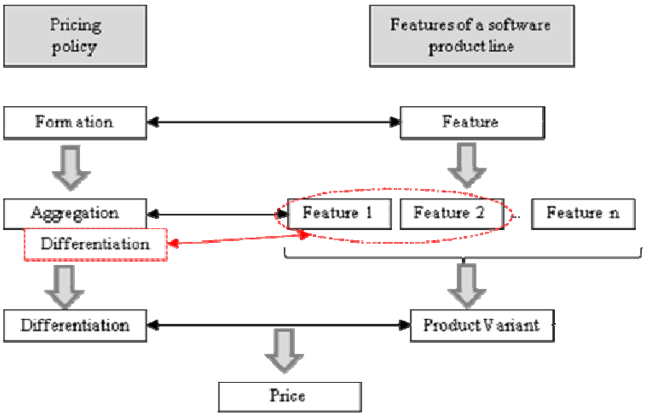

Figure 2: Pricing policy of a feature based software product line

With the knowledge into which features a software product line is divided into and how (configuration knowledge) features can be combined into a product variant, the base for setting prices is given. The customer selects his features for a product variant and in the background a price aggregation (the next pricing step under formation, shown in Figure 2) and a multiplication with the classification numbers happens. Example: When selecting *Feature 1* (f1) with a price of 10 units and *Feature 2* (f2) with a price of 10 units (shown in Figure 2), the price equals the sum of f1 and f2, 20 units. When choosing features the classification number (CN) must be multiplied with the aggregation sum of them. In our example CN is 0.8, so the price is (f1+f2)*0.8, 16 units. A notation for CN could be (f1, f2, 0.8). Classification numbers have an influence on the differentiation of a price. Differentiation could be applied to feature groups or to a standalone feature (appointed by the characteristics of the feature, like quality, quantity, etc.). Nevertheless, a price differentiation is the last step in the pricing policy (see Figure 2) to set the price of a product variant with procedures mentioned above.

After the pricing (Figure 2) a vendor knows the price of the customized product variant for a customer. The next step is to choose an accounting for a product variant or for every customer specific feature (see Figure 3). When a vendor has a ready product variant (left hand side of Figure 3), he chooses an accounting for it. This accounting is coupled with a price differentiation. A second price differentiation is required because there is an economical difference between getting the money cash or to get it divided into parts monthly or yearly, therefore a vendor could differentiate the price. The right side of Figure 3 shows the same circumstance for a feature. It is possible that every feature has its own accounting or that ex post a feature will be inserted into a product variant with a dedicated accounting. After an accounting is fixed the customer gets a final price for his selected product variant.

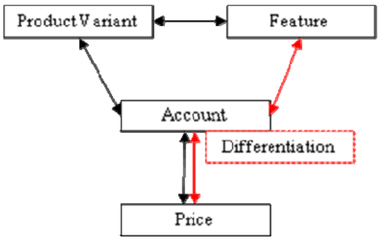

Figure 3: Accounting of a product variant or a feature

But nevertheless there are situations where even this final price could find a third differentiation. Example: When the price is too high for a customer, the vendor could prove and find (with a value based approach) the maximum willingness to pay of the customer and sell the product variant at a price, which is probably lower than the aggregated price. As mentioned, software product lines have special development costs and economies of scale and scope, that's why a return on investment should be reached as soon as possible. The revenue incensement is given nearly with every sold variant.

3 Example

A small example of a web shop SPL is used for illustration purposes. Figure 4 shows a simple web shop in two variants:

- Core system and shopping cart for 1250 € and
- Core system, shopping cart, and interactive sales assistant for 1750 €.

The needs of potential customers might be covered only partially with respect to functional variability. By introducing economic features relevant to the customers, e.g. by means of market analysis, a higher number of customers can be addressed. Figure 5 shows the corresponding enhanced feature diagram. With the additional economic features, each annotated with a price-attribute, there are now six different system variants in the SPL.

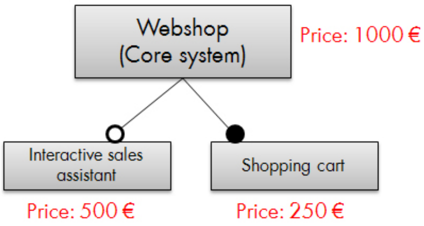

Figure 4: A simple web shop

By applying the conceptual framework of economic variability the vendor can substantially increase the SPL's overall variability model with respect to the strategic goals, competition, and market demands.

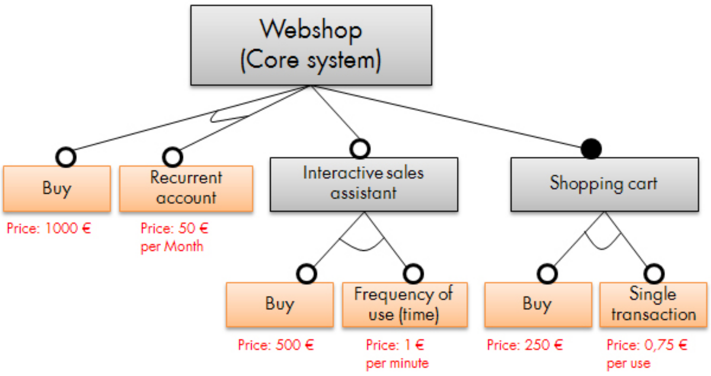

Figure 5: A simple web shop with economical features

By dynamically readjusting the offered economic variability a vendor should now be able to skim the highest revenue and increase his sales with customer self selection over the SPL lifetime.

4 Related Work

A number of related fields are affected with the software economy. On the one hand the classical economy with pricing and accounting procedures and the traditional way of software development. On the other hand we have the economy of digital goods and the description how to develop software products as a part of an SPL.

5 Conclusion

Software product lines are particularly interesting for the further software-economic view because of the given cost structures, the exclusivity and the variability of the products. At the same time the approach renders it possible to create individual software system variants and serve selected market segments. More variability in price policy and accounting of software system variants can be achieved by this software-economic view of software system families. In addition the possible associated effects of variability on revenue and yield of software products are examined. Furthermore we had a look at the consequences which arise from the fact that besides the software part concerning features and the range of information also the price policy and the accounting belong to the variability of a software product line.

References

[Ba06] Baldwin, C.Y.: The Option Value of Software Product Lines, SPLC 2006, Baltimore MD, Harvard Business School.

[BC96] Brownsword, L.; Clements, P.: Product Line Systems Program, A Case Study in Successful Product Line Management, 1996.

[Be02] Bednasch, T., Konzept und Implementierung eines konfigurierbaren Metamodells für die Merkmalmodellierung, Dipl.Arbeit, 2002.

[Bl03] Bliemel, F., Adolphs, K. (2003): Wertorientierte Preisstrategien in Handbuch Preispolitik Strategien – Planung – Organisation – Umsetzung; Diller, H./Herrmann, A. (Hsgb.); Gabler-Verlag; Wiesbaden 2003.

[Bo84] Boehm, B.W., Software Engeneering Economics, IEEE, Transactions on Software Engineering, Vol. SE-10 (1), 1984, S.4-21.

[CE00] Czarnecki, K.; Eisenecker, U.: Generative Programming: Methods, Tools, and Applications. Addison Wesley Publishers, 2000.

[CHE05] Czarnecki, K.; Helsen, S.; Eisenecker, U.: Formalizing Cardinality-based Feature Models and their Specialization, 2005.

[Cl06] Clement, R., Digitale Güter: Marktbesonderheiten und Marktstrategien, In: WISU 2/06 S.238-244 (2006).

[Co96] Cox, B.J., Superdistribution, Addison-Wesley, New York, 1996.

[Cr02] Cressman, G.E., Value Based Pricing Savest he Day!, 2002, CEO Refresher, www.strategicpricinggroup.com.

[Di00] Diller, H., Preispolitik; 3. Auflage; Kohlhammer-Edition-Marketing; Stuttgart; Berlin; Köln, 2000.

[Di03] Diller, H., Herrmann, A., Handbuch Preispolitik, Gabler Verlag, 2003.

[Ei02] Eisenecker, U. W., Generative Programmierung und Software-Ökonomie", www.gp-web.org, 2002, gelesen am 3.09.2006.

[Fa96] Fassnacht, M. Preisdifferenzierung In: Diller, H., Herrmann, A., Handbuch Preispolitik, Gabler Verlag, 2003, S. 483-502.

[Ka90] Kang et. al.: Feature-Oriented Domain Analysis, 1990.

[Ki04] Kittlaus, H.B., Rau, C.,Schulz, J., Software-Produkt-Management, Springer, 2004.

[Ko99] Kotler, P.: Grundlagen des Marketing, Prentice Hall, München, 1999.

[Kr06] Krueger, C.W.: New Methods in Software Product Line Development, SPLC 2006, Baltimore MD, BigLever Software.

[Me00] Meffert, H.: Marketing – Grundlagen marktorientierter Unternehmensführung; Konzepte – Instrumente – Praxisbeispiele; 9. Auflage, Gabler-Verlag; Wiesbaden 2000.

[PB05] Pohl, K., Böckle, G., van der Linden, F.: Software Product Line Engineering – Foundations, Principles, and Techniques, Springer, Berlin, Heidelberg, New York, 2005.

[Pe98] Pepels, W., Einführung in das Preismanagement; Oldenburg-Verlag; München 1998.

[Po00] Porter, M., Wettbewerbsvorteile. Spitzenleistungen erreichen und behaupten, Campus Fachbuch, 2000.

[Pe05] Pechtl, H., Preispolitik; Lucius & Lucius Verlagsgesellschaft mbH; Stuttgart 2005.

[Sc06] Schröter, M., Preisbildung bei Softwaresystemen, Universität Leipzig, Diplomarbeit, (2006).

[Se07] http://www.sei.cmu.edu/productlines/spl_case_studies.html, 2007.

[Si92] Simon, H., Preismanagement: Analyse – Strategie – Umsetzung; Gabler Verlag; Wiesbaden, 1992.

[Sk99] Skiera, B., Preisdifferenzierung In Albers, S., Clement, M., Peters, K. Handbuch Marketing mit Interaktiven Medien. Strategien zum Markterfolg, S.283-296, 1999.

[Sk05] Skiera, B.: Erlösmodelle und Preismodelle für den Business-to-Consumer-Bereich im Internet in: Wirtschaftsinformatik 47 / 4 S.285-293 (2005).

[Sk06] Skiera, B., Lambrecht, A., Erlösmodelle im Internet (2000), in: Albers, S., Herrmann, A. (Hrsg.), Handbuch Produktmanagement, Gabler Verlag, 2006, S. 813-831.

[St00] Stelzer, D., Digitale Güter und ihre Bedeutung in der Internet-Ökonomie, Köln, In: WISU 6/00 S. 835-869 (2000).

[SV99] Shapiro, C., Varian H. R.: Information Rules – A strategic guide to the network economy, Harvard Business School Press, Boston, 1999.

[Wi96] Withey, J.: Investment Analysis of Software Assets for Product Lines, Technical Report CMU/SEI-96-TR-010, Software Engineering Institute, Carnegie Mellon.

[Vo03] Voigt, K.-I., Preisbildung für neue Produkte (2003), in: Diller, H., Herrmann, A., Handbuch Preispolitik, Gabler Verlag, 2003, S. 691 – 717.

[Ze99] Zerdick, A., Picot, A., u.a., Die Internet-Ökonomie: Strategien für die digitale Wirtschaft / European Communication Council Report, 1998, Springer, Berlin, 1999.