Supported by tetfund TERTIARY EDUCATION TRUST FUND

CURRENT DYNAMICS OF SOFTWARE PRICE DETERMINATION

Odey John Adinya* and Agaba Francis
Department of Computer Science, University of Calabar, Cross River State, Nigeria
*Corresponding author: johnodey@unical.edu.ng

Received: May 25, 2021 **Accepted:** July 27, 2021

Abstract: To determine the price of a product that will not only yield the highest possible benefits for vendors but also yield expected value for customers is usually not an easy task. Software, being a digital product, has characteristics that make its marketing more complicated than the marketing of physical goods. This paper presents a review of the various methods used by software producers and vendors in the determination of software prices. This review is done by extensively examining academic literature and research on this topic for over a decade, and summarizing the evolving trends on the subject. The fairness of the software prices arrived at by the various price determination models was also discussed.

Keywords: Software pricing, price discrimination, price bundling, software sizing

Introduction

Pricing is simply the procedure where a trade sets the rate to vend its goods and services. In setting prices, vendors usually take into account manufacturing cost, the marketplace, competition, market condition, brand, and quality of the product. In the early years of marketing, setting product costs was all art and no science. Vendors set their prices based on their desired profit margins and manual surveys of competitor prices. Achieving positive results was time-consuming and unreliable (George, 2019). Charging the wrong price for goods usually leads to a poor reputation and low income for the company concerned. Price determination should not be a chaotic process. Vendors should have and use pricing policies that guide them in setting the right price for products at any point in time. The right price here is understood to be the price that will yield the highest possible profit to the vendor and value to the customer.

New technologies are making pricing almost entirely scientific. Modern pricing intelligent tools are combining big data, cloud technology, and artificial intelligence (AI) to set prices. These technologies use automatic and real-time monitoring of market prices for products and powerful analytic tools to help forecast optimum prices as asserted by the author in (George, 2019). While not all packages include AI or big data analytics, those that do not are more limited in what they can do. They are more suitable for smaller companies with simpler needs. After the abstract and introduction, this paper is divided into four sections: the motivation for review, the state of the art, a summary of the major pricing methods practiced today, and a conclusion.

When we take a critical look at the value that can be gotten from software products, two situations usually arise. If the software yields a low value, the user will complain that the software is not worth the cost. If it yields a high value, the user may ask if users can pay for that value if charged the full value by the producer. Determining the ideal pricing point is one of the toughest portions when emancipating a software product as stated in Eran (2017). Pricing is significant as it describes the worth of a product for the producer and its customers. It is the palpable price point that informs customers to know whether the product is worth their time and investment as opined in (Patrick, 2019). Software producers take time, energy, creativity, cash, and capital to realize a product.

There are several components in a trade, from the auctions and advertising, through goods and services, to the employees that empty the trash. These entire components sum up to providing the values users are willing to pay various prices for. The customers neither know nor care what you went through to arrive at the product. The price customers are willing to pay depends on their perception of the value they will get from the product. Knowing how to arrive at an optimal price is thus of utmost importance to software producers. It is also important for customers to have a good idea of how software is priced because this will help them have an idea as to whether the prices are worth the value.

The paper, “A systematic review of software development cost estimation studies” (Magne & Martin, 2007), is a review of three hundred and four software cost estimation papers from seventy-six periodicals and categorizes the articles based on the investigation topic, estimation method, investigation method, schoolwork background, and data set used. A Web-based library of price valuation articles is delivered to influence future recognition of pertinent software price assessment investigation. The main software cost estimation approaches identified in these papers were: Regression-based methods like COCOMO (Constructive Cost Model), an analogy with other similar software with known costs, expert judgment or estimation, function point measurements, theory-derived estimation models, like SLIM, ANN-based estimation approaches and findings on the combination of estimates from different methods. More details on how each of these methods proceeds to determine a fair price for software will appear in the papers below.

Masateru *et al.* (2012) on the “analysis of attributes relating to custom software cost” emphasized the importance of the end-users knowing the fair price to pay for software they need. They feel that the available data is not enough to enable users to determine the fairness of the traditional software cost. So, their major target was to build a cost assessment model and show its reliability for the users to review the fairness of convention software charge. The dataset used in the analysis (custom software used and prices) were gathered from 114 establishments in Japan and included 163 software enhancement protrudes.

Sonja & Peter (2009) researched the pricing strategies of software vendors. They provided a summary of pricing models for software. Pricing models are developed taking into consideration the overall circumstances and features of the software product. For instance, the software is a digital product; the fiscal concept of numerical goods delivers an initial preliminary fact for the improvement of valuing policies for software suppliers. Indestructibility, transmutability, and reproducibility are the essential characteristics of digital properties (Choi *et al.*, 2007). According to Zhang & Seidmann (2013), indestructibility turns out to be ostensible by the unascertainable variance amid novel and secondhand digital products as there is no loss of excellence in the cause of utilizing the goods even if it does not omit a potential loss of worth over period. The transmutability feature means digital properties can be altered with slight exertion permitting economical creation of variations as its usefulness in cost discrimination policies cannot be overemphasized. Reproducibility of digital goods is the characteristic that duplication of digital properties is done with no loss of at tributes and at an affordable price (Choi *et al.,* 2007). The internet and its characteristics can also influence software pricing since the internet is useful in the delivery of digital goods as stated by Gensler *et al.* (2007). The potential clientele is also significantly increased, flexible pricing model, easier interaction between the customer, and easy price change implementation (Skiera, 2010). Software as a solution (SaaS) resolutions lead to reduced provider dependence. Lock-in influences and swapping prices generally can never be evaded even for SaaS resolutions in the arena of initiative software (Lehmann & Buxmann, 2009).

Pricing models for software products: Pricing models have evolved from payment according to computing power in the days of mainframes through licensing of copies of the software (Bontis & Chung, 2010) to usage-based pricing in cloud services today. Software pricing involves the foundation of price determination and the degree of communication from customers. According to Homburg & Krohmer (2006), three types of pricing models are feasible: cost-based, demand-driven, and competition-oriented. It also been argued that the cost-based pricing model does not function well as one need to assess the material goods based on customer worth, and not on the manufacturing cost, thereby making it unsuitable for software licenses, but can only make a virtuous logic for SaaS. The demand-driven pricing model is fundamentally grounded on the request for the manufactured goods, whereby the gratitude of the product's consumers is of prominence rather than the product's price. For competitive price determination, the creation of charges is allied to the bills of competitors (Shapiro & Varian, 2009).

Payment models: Pricing models are basically of two types: Single recompense to have full rights of use for software or recurrent client payments and both variations are feasible as opined by the authors in (Kittlaus & Clough, 2009). An example of a combination of the two models is where clienteles and suppliers settle on a once- a-month or yearly payment cost for two years as a pricing model that is specifically recurrent in SaaS resolutions (Cusumano, 2007). The hybrid of the two is also practiced once a software license is connected to a software upkeep agreement, say 20 percent. The merit of this model is that payments will run comparatively progressively.

Price discrimination: This is the offering of similar products to diverse buyers at diverse costs. The supplier's target is to improve the enthrallment of customer excess. It is specifically vital for suppliers of digital properties as it permits vending to clienteles with a lesser readiness to pay. The easy and low-cost adjustment of digital properties stimulates the use of price demarcation tactics. Cusumano *et al.* (2012) differentiate three kinds of price discernment which are first, second, and third-degrees. In the first-degree, every client obtains a price proposal in the quantity of their readiness to pay. The second-degree is grounded on the criterion of person-choice; that is, the customer pays based on the quantity being bought, the time the purchase is done, and the performance of the product concerning the buyer's needs. The third-degree is grounded on the provider's market place dissection (Patrick, 2019).

Price bundling: This is generally comprehended as the assembling of numerous recognizable sub-services (goods, facilities, or privileges) of one or more suppliers to a platform of proposals with a requirement of a complete cost (Patrick, 2019). It can likewise be seen as a distinctive instance of price discernment.

Pricing strategies: The penetration strategy, the follow-the-free strategy, and the skimming strategy are of particular significance. The penetration tactic has the target to use low costs to make the best use of market place infiltration. The vendor can increase the costs at a future phase, once it reaches a perilous quantity. In the follow-the-free tactic, clienteles obtain manufactured goods free of charge. The vendor's target is to generate a lock-in influence on the users' side to produce incomes later on utilizing complimentary products or premium versions. Anh (2017) conducted a study on the impact of software complexity on cost and quality. Measurement of software complexity improves our awareness of the disposition of software and obliquely measures and forecasts the ultimate value of the product. Complexity metrics are meaningful if they can indicate the project's software quality and effort put into producing it. However, the selection of appropriate metrics that can predict given software attributes is not yet clear in the literature. The investigation concluded that the complexity of software influences its cost but not necessarily quality. Hareton & Zhang (2013) presented a paper on Software Cost Estimation. The article offers an overall summary of software price assessment approaches within the past thirty years with the most current developments in the meadow. As most models depend on a software size estimate such as input, they provided a synopsis of mutual size metrics. Thereafter, they highlighted the price assessment models that have been suggested and utilized effectively. A key factor for them to select a cost estimation model was the accuracy of its estimates. The majority of the price of software improvement is because of human exertion, and most price assessment approaches emphasize this feature and give approximations in terms of person-months. Software cost estimation comprises the willpower of at least one of the following estimations:

i) Effort (person-months)
ii) Project duration (calendar time)
iii) Cost (cash)

In the past three decades, numerous measurable software price estimation models have been developed. They range from empirical models like Boehm's COCOMO models to analytical models. Empirical models use data from preceding tasks to assess the present task and derive the key formula from the examination of the specific data bank obtainable. Analytical models, conversely, use a formula based on universal expectations, like the degree to which developers resolve glitches and the number of glitches obtainable.

i) Software sizing: The software size is the utmost vital feature that influences the price of the software. Here, the most vital metrics for software size are discussed, which include line-of-code and function-point;

(a) Line-of-Code (LOC): Simply implies the number of lines of the provided source program of the software, exclusive of remarks, and vacant lines. Though LOC is the peak extensively utilized metric for software magnitude, it is also a programming language reliant on. However, precise LOC can only be attained after the task is finished.

(b) Function-points: This is a metric grounded on the workability of the code. The whole quantity of function-points hinge on the tallies of discrete (based on setup or dispensation logic) kinds of user-input, user-output, inquiry, internal file, and external file categories.

Cost estimation: The two main approaches are algorithmic and non-algorithmic.

(1) Algorithmic models: They differ extensively in scientific complexity. More or less are grounded on uncomplicated mathematics formulations utilizing swift statistics like averages and standard deviations. Others are grounded on differential equations and retrogression prototypes. They produce price estimation as a function of several variables that are contemplated to be the main price features. The current algorithmic approaches vary in two

facets: the assortment of price factors and the usage of the function f.

Cost factors: Besides the size of the software, there are other cost features. The most complete sets of cost features are suggested and utilized by Beata (2010) in the COCOMO II model. The algorithmic models covered by this paper are linear, Multiplicative models, Power-function models, COCOMO models, Putnam's model and SLIM, linear regression, and discrete models.

(2) Non-algorithmic models: These models comprise the analogy estimate, expert verdict, price-to-win, bottom-up and top-down approaches.

a) Analogy estimate: This technique needs at least one finished task that is comparable to the novel task and originates the approximation through cognitive by analogy utilizing the real prices of preceding asks.

b) Expert verdict: This technique encompasses referring to one or more specialists. The specialists deliver approximations utilizing their approaches and experiences. Expert-consensus types of machinery like the Delphi method or PERT will be employed to solve the discrepancies in the estimations.

c) Price-to-win: The software price is projected to be the paramount cost to win the project. The estimate is grounded on the client's financial plan rather than the software workability. For instance, if a sensible estimate for a project costs 110 person-months but the customer can only afford 70 person-months, it is normal that the estimator is requested to adjust the estimate to fit 70 person-months exertions to win the project.

d) Bottom-up: In this method, every constituent of the software is distinctly projected and the outcomes accumulated to create estimation for the complete scheme.

e) Top-down: This is the opposite of the bottom-up method. Many other models exist each with its strengths and weaknesses. A comparison of the methods shows that;

i) There is no one technique that is optimal for every task;

ii) Parkinson and Price-to-win approaches are unsuitable for establishments that target to win extra businesses; and

iii) Employing a mixture of methods can deliver a superlative estimate. For instance, service able merging top-down estimates with the expert verdict and analogy approaches can deliver an excellent outcome.

Other pricing approaches: Mark *et al.* (2013) published a paper in which they felt that the state of pricing in the software industry was chaotic. They said salespersons have rating rules and tactics, but frequently miscarry to implement them. Generally, they alter pricing methods from one contract to another, grounded on a mixture of factors. Only a small but growing number of vendors are attentive to evolving models, comprising payment SaaS through cloud deployment, pay-as-you-go. The remainders comp rising mainly customer and initiative software establishments are uniting both methods in a crossbreed industry model.

Rashid & Jun, 2017) published a paper on Software vigorous valuing by an optimization deterministic model in an anticompetitive market place. The paper formulates an optimization model for assessing anticompetitive application software in the occurrence of piracy. The obtainability of software at low prices for clienteles reduces piracy, as the high price of lawful software is the main reason for pirating software (Cheng *et al.*, 2017). Allie (2019) in his work titled; "The Ultimate Guide to Pricing Strategies" defines a pricing tactic as a technique employed to create the superlative cost for produce or facility. Pricing strategies enable one to select costs that make the best use of incomes and stockholder value while bearing in mind the customer and market place request. The most widely used pricing strategies are competition-based, Cost-Plus, Vibrant, Freemium, High-Low, Hourly, Scanning, Infiltration, Premium, Project-Based, and Value-Based. The cost-plus pricing tactic emphasizes exclusively the price of creating the manufactured goods or provision plus the profit the producer wants to make. Dynamic pricing is otherwise called flow valuing, demand valuing, or time-based valuing. It is an adaptable pricing tactic where costs vary grounded on marketplace and client requests. A mixture of the words "free" and "premium" freemium pricing is once businesses offer a rudimentary form of their produce expecting that consumers will finally pay to advance or entree more topographies. A high-low valuing approach is when a firm primarily vends a product at a high cost but depresses that cost when the product drips in originality or significance.

Hourly pricing is also called rate-based pricing and it is universally utilized by specialists, service providers, freelancers, and other individuals or labourers who deliver commercial services. The scanning costing approach is when businesses charge the uppermost conceivable cost for the novel product and at that point depresses the cost overtime as the product becomes less widespread. This approach differs from high-low pricing in that costs dropped steadily over pre calculated time intervals.

Penetration pricing approach is once businesses arrive in the marketplace with a very low cost, efficiently pulling consideration (and revenue) away from higher-priced contestants. This approach is not maintainable in the extended track, though, it is characteristically useful for a short time. Prestige pricing tactic is once businesses cost their goods great to present the appearance that their products are of great worth. A project-based pricing tactic costs a level payment per project rather than a direct interchange of cash for time. A value-based pricing approach is when firms cost their products or services grounded on what the client is eager to recompense. Not all pricing strategy applies to each trade. Several approaches are more suitable for corporal goods while others function superlative for digital products.

Discussions

This paper confirms that it is an absolute necessity for software vendors to use standardized methods in setting prices for their software. This does not only helps them reap the highest possible benefits but also gives customers the possibility to verify if prices are fair. The difficulties involved in setting optimum prices for software stem mostly from the indestructibility, transmutability, and reproducibility characteristics of software. Cost estimation methods are essentially cost-driven, demand/value-driven, and competition-driven.

The most essential factors to consider when setting a software price are efforts (normally person-months), protrude period (calendar time), cost (cash), and software size (measured in terms of lines of code or function points).

Previously, the software cost estimation methods that have been used are:

a) Empirical methods where data from past projects is used to decide the cost of current projects.

b) Analytic methods where formulae are based on global assumptions are used.

c) Algorithmic models produce cost estimates as a function of the number of variables. They are usually of the form:

Effort = $f(x_1, x_2, ..., x_n)$ **Where** $\{x_1, x_2, ..., x_n\}$ are price features. Variants of algorithmic models are linear regression, multiplicative, power function, COCOMO, SLIM, and discrete models.

d) Non-algorithmic models: Analogy costing where actual costs of at least one finished project comparable to the novel project is applied to assess the price of the new project; Expert judgment where one or more expert cost estimators are used to do the estimate; Price-to-win is where the price is simply approximated to match what the majority of customers are deemed ready to pay for the software; Bottom-up where the cost of every constituent of the software scheme is estimated distinctly and outcomes combined to generate overall approximation and Top-down which is the opposite of Bottom-up.

It is interesting to note that none of these methods is best for all situations and that usually, two or more need to be combined to obtain good results. Even after the cost of software has been determined, the various methods by which to retrieve this cost and get profit from customers are called Pricing models. Pricing models have evolved from single payment or licensing through recurring payments to price discernment, price hustling, vigorous pricing, SaaS pricing, price skimming, package offers, penetration pricing, freemium pricing, or a combination of two or more of these pricing models.

Conclusion

Despite all the research efforts that have already been put into software cost estimation; it lingers a complicated problem in the software production process that persists to fascinate substantial research concentration. Even with the most recent estimation models based on artificial intelligence, there is still no model that can evaluate the price of the software with an extraordinary level of correctness because of the following issues:

i) There are an enormous amount of interconnected issues and project attributes that affect the software development procedure and whose effect on the project is not mastered by cost estimators;

ii) The development environment is evolving continuously, making it possible to develop the same product by radically different methods and thus radically different costs; and

iii) The deficiency of dimension metric that mirrors the intricacy of a software scheme.

To generate improved estimates, we should enhance our comprehension of all project characteristics and their causative associations; model the effect of the developing environment, and design efficient methods of evaluating software intricacy. Despite these shortcomings, doing software cost estimation by standardized approaches rather than chaotic methods is an absolute necessity because the standardized methods do not only lead to optimal costs that yield the highest profits for developers but also give customers the possibility to know if software costs are fair. For software products, this paper recommended competition-based, freemium, and value-based pricing approaches.

References

Allie D 2019. The Ultimate Guide to Pricing Strategies.https://blog.hubspot.com/sales/pricing-strategy, Retrieved: 26/10/2019.

Anh ND 2017. The impact of software complexity on cost and quality: A comparative analysis between open source and proprietary software. *Int. J. Software Engr. and Applic.*, 8(2): 52-68.

Beata C 2010. Rational pricing of business software systems based on functional size measurement: A case study from Poland; Software Measurement European Forum (SMEF).

Bontis N & Chung H 2010. The evolution of software pricing: from box licenses to application service provider models. *Electronic Networking Applications and Policy,* 10(3): 246–255.

Cheng HK, Sims RR & Teegen H 2017. To purchase or to pirate software: An empirical study. *J. Mgt. Infor. Syst.* https://doi.org/10.1080/07421222.1997.11518142. Accessed 26/03/2020.

Choi SY, Stahl DO & Whinston AB 2007. The Economics of Electronic Commerce: The Essential of Doing Business in the Electronic Marketplace. Macmillan: Indianapolis, USA.

Cusumano MA 2007. The changing labyrinth of software pricing. *Communications of the ACM*, 50(7): 19–22.

Cusumano M, Iyer B & Venkatraman N 2012. Software business. Proceedings of Third International Conference, ICSOB 2012, June 18-20, 2012, Cambridge, USA. Springer: Berlin, Germany.

Eran G 2017. The Science of Software Pricing. https://www.binpress.com/software-pricing/ Accessed 2nd November, 2019.

Gensler S, Dekimpe M & Skiera B 2007. Evaluating channel performance in multi-channel environments; *J. Retailing and Consumer Serv.*, 14(1): 17-23.

George S 2019. Price Analysis in Determining the Right Price Point for Your Products https://www.matrixmarketinggroup.com. Accessed 31/10/2019.

Hareton L & Zhang F 2013. Software Cost Estimation.cshleung, csfzhang@comp.polyu.edu.hk.

Homburg C & Krohmer H 2006. Marketing Management Strategy; Https://www.amazon.com. Accessed 06/04/2020.

Kittlaus HB & Clough PN 2009. Software product management and pricing. Springer-Verlag Berlin Heidelberg, Germany.

Lehmann S & Buxmann P 2009. Pricing strategies of software vendors. *Business Information Systems Engr.*, 1: 452.

Mark M, Pierre M, Grey U, Tom A & Kayvan S 2013. The Future of Software Pricing Excellence: www.pwc.com. Accessed 26/10/2019.

Magne J & Martin J 2007. A systematic review of software development cost estimation studies. *IEEE Transactions on Software Engr.* https://www.researchgate.net/publication/.

Masateru T, Akito M, Kenichi M, Sawako O & Tomoki O 2012. Analysis of Attributes Relating to Custom Software Price. Fourth International Workshop on Empirical Software Engineering in Practice. https://www.researchgate.net/publication/

Patrick C 2019. Importance of pricing, why pricing is important for SaaS and beyond. https://www.priceintelligently.com/ Accessed 02/11/2019.

Rashid M & Jun G 2017. Software dynamic pricing by an optimization deterministic model in a monopolistic market. *Cogent Engineering*, 4(1): 345-359.

Shapiro C & Varian HR 2009. Information rules: A strategic guide to the network economy. Harvard Business School, Boston.

Skiera M 2010. The Impact of New Media on Customer Relationships; Journal of Service Research.

Sonja L & Peter B 2009. Pricing Strategies of Software Vendors. http://www.wirtschafts.inform atik.de.

Zhang J & Seidmann A 2013. The Optimal Software Licensing Policy under quality uncertainty. ACM Press: Proceedings of the 5th International Conference on Electronic Commerce. New York, pp. 276–286.