Information and Software Technology 55 (2013) 690–698

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Vertical software industry evolution: The impact of software costs and limited customer base

Oleksiy Mazhelis *, Pasi Tyrväinen, Lauri Frank

Department of Computer Science and Information Systems, University of Jyväskylä, P.O. Box 35, FI-40014 Jyväskylä, Finland

ARTICLE INFO

Article history:
Received 28 February 2011
Received in revised form 22 October 2012
Accepted 29 October 2012
Available online 10 November 2012

Keywords:
Software industry
Customer base
Software development costs
Telecommunications OSS/BSS software

ABSTRACT

Context: Software systems are commonly used in a variety of industries as a means of automating organizational business processes. Initially, such software is often developed in-house by the vertical organizations possibly with the support of professional IT service providers; however, in many cases, internally developed software is eventually replaced with the software products provided by independent software vendors. These vendors often use license fees to recover their software development investments, as well as to gain some margin. However, if the vendor's customer base for a specific type of software is limited, then either the license fees are too high and hence the customers may prefer to develop the software internally, or the margin has to be decreased. As a result, the market for software products of that type may not materialize.

Objective: The paper introduces an analytical model that defines the minimum number of customers that the software vendor should have for its software to be less expensive as compared to the in-house software.

Method: Following a conceptual-analytical approach, a model is constructed wherein the minimum number of a vendor's customers is represented as a function of other factors affecting software development costs. This model is verified by applying it to estimate the minimum customer base in the segment of telecommunications billing mediation software.

Results: Using the proposed analytical model, the minimum number of customers and the maximum number of software vendors in this segment are evaluated. The obtained results are found to be in line with the information available from a telecommunications software market database.

Conclusions: Based on the model, a preliminary conclusion is made that in industries with high software development costs, heterogeneous legacy systems to integrate with, and a limited pool of potential customers, the number of software vendors is unlikely to be significant, and hence the in-house or custom-made software is unlikely to be superseded by the software products.

1. Introduction

Software systems are widely used in virtually all industries as a means of automating business processes. Organizations deploy them in order to provide innovative and value-adding services, as well as to reduce their operational costs [1,2]. In some industries, such as banking or telecommunications, both business processes and the software systems automating them are highly complex and heterogeneous; as a result, software systems need to be tailored to the peculiar needs of each organization. Depending on the costs associated with software development, the organization may decide to invest in the internal in-house development and maintenance of the required software thereby making the software vertically integrated, or to acquire tailored software or a software product from independent software vendors [3,4].

The evolution of the software industry has been recently studied by Tyrväinen et al. [5] who, based on the analysis of 20 industries, have elicited three phases through which the evolution of the software industry proceeds: (i) first, software is produced within the industry using that software, often by company internal divisions that may rely on the support of professional IT service providers; (ii) in the second phase, software products serving the industry's needs emerge on the market, e.g., made by software spin-off firms; and (iii) in the third phase, software products by external producers dominate the market, and are complemented with external consultation services, tailor-made software, and integrated software products, while the industry's internal software production is low.

According to Tyrväinen et al. [5], software is vertically integrated in the beginning of its lifecycle; often both software and

* Corresponding author. Tel.: +358 40 805 4293.
E-mail addresses: mazhelis@jyu.fi (O. Mazhelis), pasi.t.tyrvainen@jyu.fi (P. Tyrväinen), lauri.frank@jyu.fi (L. Frank).

0950-5849/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.infsof.2012.10.006

hardware are produced by the same vendor. However, as the organizations need to focus their efforts on a specific part in their business, parts of the software are outsourced to subcontractors or otherwise acquired from third parties. As a result, in many cases, the initially vertically integrated software eventually becomes split into a set of horizontal software layers with standardized interfaces among them, and with the software at each layer being provided by independent software vendors (ISVs).

Several barriers can be envisioned on the general pattern of transforming vertically integrated software into horizontal layers. For instance, an extensive set of interfaces to be supported may result in immense interface implementation efforts, thereby limiting the number of competing software vendors [6]. Another barrier, which is the focus of this paper, may be insufficient pool of the customers, given the high costs of developing and deploying software products.

Indeed, previous research has found that the expected cost savings have the greatest effect on the organization's decision to outsource software development to an ISV [7]. If the costs of acquiring and integrating the software exceed the costs of developing the software in-house, the transition from internally developed software to acquired software products is less likely, and, as a result, the market of software products of that type may not materialize. In other words, for a software product to appear and be sustained in the market, the costs of software acquisition and integration should not exceed the costs of internal development and maintenance of the software. Furthermore, for the above condition to be met, the software vendor should have a sufficient number of customers and a sufficient degree of software reuse across the customers.

This paper introduces an analytical model that considers the minimum number of customers and the minimum degree of software reuse across customers that the software vendor should have in order for its software to be less expensive as compared to the software developed in-house. Combined with the overall pool of customers, the minimum number of customers is used as a means to estimate the maximum number of software vendors that can coexist in the market.

The proposed model is based on the assumptions about the costs incurred by the customer and software vendor in the process of software development, configuration, and integration. Similar to the software reuse models [8–10], the costs of developing for reuse and the costs saving due to the reuse of software are taken into account when assessing the efforts.

Various economic aspects of software development and use have been studied. A significant portion of these studies focus on the problem of software effort estimation; an overview of available estimation approaches, practices, and models can be found in [11–14]. The majority of the estimation approaches are based on the expert judgment [15,16] or on simple data-driven methods, such as size- and effort-based regression [14]. A number of more elaborate models have also been introduced including fixed models exemplified by Putnam SLIM [17], COCOMO [18], and KnowledgePlan [19,13], machine learning tools such as decision trees and neural networks [20,21], as well as the models relying on system dynamics [22], though these are rarely used in practice [14].

Besides software effort estimation, studies on software economics have addressed numerous decision making issues present in software development, in particular the choice of the products or components to develop [23–25], risks monetization [26,27], the cost-efficient investments in quality assurance and testing [28–30], market positioning of the customized software Rokonuzzaman and Choudhury [31], etc. Similar to our work, Rokonuzzaman and Choudhury [31] consider the software reuse in the context of customized software solutions. However, the customization in [31] is assumed to be done on a per segment rather than on the per-customer basis, and the objective is to maximize the firm's profits by proper market segmentation and positioning. The authors are not aware of any public research studying the effect of the costs and customer-specificity of the software on the feasibility of the software product emergence.

The elaborated model is applied in the paper to the telecommunications mediation software that interfaces the network elements and billing systems of communications service providers (CSPs) and is responsible for gathering the data describing service usage, aggregating this data, and converting it into billing statistics. The minimum number of customers (CSPs) of such mediation software and the maximum number of software vendors in this market are estimated. Based on comparing the results against the real-world market situation, the plausibility of the proposed model is assessed. A preliminary version of the model has been presented in Mazhelis et al. [32] and in a condensed form in Mazhelis and Tyrväinen [33]. In this paper, the model is extended by taking into account the effect of maturing market, as well as the maintenance and service fees based on Mazhelis and Viitala [34].

The remainder of the paper is organized as follows. In the next section, the details of the analytical model of boundary conditions for the software development and integration costs are given. Special cases of the model, including open and closed interfaces in a matured market, as well as the effect of maintenance and service fees, are analyzed in Section 3. In Section 4, the proposed model is applied in the context of the telecommunication billing mediation software. Section 5 discusses the application of the model to assess the market concentration. Finally, conclusions to the paper are provided in Section 6.

2. Modeling software development and integration costs

ISVs usually charge their customers a license fee for the right to use their software.[1] Using the cost-based pricing approach common in tailor-made software development, the value of the license is set so as to recover the costs of the software development and configuration, plus gain some profit margin on top of these costs.

If the software solution is based on a core reused across multiple customers, the ISV can distribute the core-related costs among these customers. Thus, the expected number of customers affects the license fee: the greater the expected customer base, the lower the fee. On the other hand, when the number of ISV's customers is limited, then the license fees are too high and hence the customer may prefer to develop the software internally. In such a case, the ISV may either decrease the margin, or reduce the per-customer customization, e.g., by widening the core. Below, the minimum number of customers needed for the recovery of software development and configuration costs is estimated, and the limiting values for other parameters affecting the costs of the software development are determined.

2.1. Conditions of profitable software products

Let us consider the customer's costs associated with (i) acquiring the software from an ISV and (ii) developing the software in-house. In the former case, the customer's costs include:

- the cost of acquiring a license for the software product (C_{prod});
- the cost of integrating the acquired product with the other deployed systems (C_{integr}).

We assume that the costs of buying and integrating a product

[1] The software renting and software-as-a-service models of charging are also possible; however, for the sake simplicity these are not considered in this paper.