Software-as-a-Service Revenue Models

Arto Ojala, *University of Jyväskylä, Finland*

When should software providers maintain their traditional licensing model versus offering software as a service, and which SaaS revenue model is more profitable: rental or pay per use? For customers, what are the trade-offs between traditional licensing, renting, and paying per use?

Cloud computing is changing the way in which software is sold, delivered, and used. In the software-as-a-service (SaaS) model, customers can access software online as needed instead of permanently installing it on their computers. This ensures they always use the latest version of the software, and they no longer need to worry about their computer's technical specifications or storage capacity.

However, the customer can't be the only beneficiary of this model. The software vendor must ensure a profitable revenue stream when an initial license fee is replaced by a usage-based or rental fee. Also, there are some risks for the customer. SaaS can increase uncertainties related to data security when important data is stored on a public cloud server and when applications require a reliable Internet connection.

The revenue models in SaaS are based on the pay-per-use method or software rentals.[1–3] Pay per use involves charging the customer only for the metered usage of the software. In software rental, the customer pays a negotiated subscription fee for a certain time period. In some cases, a firm might also offer a traditional licensing model in addition to its SaaS offering. In this case, the customer can buy a traditional license for a workstation or use the software in a private cloud.

Here, I give an overview of the three main revenue models, focusing on the advantages and disadvantages for SaaS providers and their customers. I also identify the most effective revenue model for particular situations.

Cloud Computing

In line with Michael Armbrust and his colleagues,[1] I consider cloud computing to cover not only software applications delivered through the Internet but also the hardware and system software that data centers use to provide these services.

 Published by the IEEE Computer Society

Infrastructure as a service (IaaS) provides computation and storage capacity, platform as a service (PaaS) provides software development tools and an application execution environment, and SaaS provides applications on top of a PaaS, IaaS, or private data center.[1]

The data center hardware and software that form a cloud can be categorized as public, private, or hybrid. In a public cloud, software vendors use their own or a third party's cloud infrastructure (data center) to offer SaaS for customers on a pay-as-you-go basis. A private cloud involves the customer's own internal data center, where the software is installed and used in a centralized manner in the organization—in other words, the software isn't made publicly available.[1,4] In the case of a hybrid cloud, a firm using a private cloud can, for example, offload part of its workload to a public cloud to acquire more computing capacity.[4,5]

Revenue Models

Software belongs to the category of information goods that can be delivered and sold either in a material form (as a CD or DVD package, for example) or in an intangible form through the Internet, so it can be used with several revenue models. Software products usually have a high initial cost, due to the substantial human resources required for product design and development. However, for a ready-made software product, the reproduction costs can be practically nonexistent. This makes the pricing strategies for software products somewhat different from those in other industries.[6,7]

Various terms are used to describe how a firm makes money by selling its products or services—including *business, licensing,* or *revenue models* and *revenue* or *earning logic*. Here, I use the term "revenue model" in an operational sense, referring to how a firm collects revenue from its customers. Thus, it relates to the various options that a firm might offer to customers who want to buy its software.[8]

Vendors can also sell software using combinations of different models. Such models might include packaged and server-based licensing, software renting, pay-per-use pricing, effort-based pricing (in software projects), revenue sharing with partners, utility-based charging, freemium, and advertisement-based models.[8,9] Here, I focus on the most common revenue models—namely, software renting and pay per use, which can be provided by SaaS. I also discuss the advantages and disadvantages of traditional software licensing in comparison to pay-per-use and software renting models. An assumption here is that the SaaS providers sell the software using a public cloud, although in some cases, the SaaS providers can rent the software for use in a customer's private cloud.

Pay Per Use

With pay per use, there's a unit with a fixed price, and a customer is periodically charged according to the units used. The metered unit might be based on the length of time that the software is running, the number of times that a key subroutine is called, the number of transactions handled, or some combination of these.[10]

The pay-per-use model lets software vendors diversify their customer base, making their software available to customers who might not have the financial resources to buy a traditional software license. The model is also suitable when the customer needs the software only occasionally or for some specific purpose. The model also promotes the *network externalities effect*, meaning that the increased number of buyers increases knowledge of the product among potential customers, lowers customers' search costs, and makes the product well known in the market.[11] For a software vendor, it's hard to know how many copies of software are made but easy to know how many times it's used. This situation favors the pay-per-use model,[11] because executing software on a cloud server makes software piracy practically impossible.

However, the pay-per-use model has three main disadvantages for software vendors. First, because customers don't sign long-term contracts to use the software, it's inexpensive and easy to change the software provider if alternative applications at a lower price appear. Also, with usage-based pricing, the vendor might need to maintain and, on demand, present an itemized document of each customer's software usage,[12] increasing administrative work and requiring a reliable process for delivering such documents. Finally, in the pay-per-use model, where initial incomes are low and uncertain, recouping the

software development costs is bound to be riskier than with traditional licensing.

From the customer perspective, for smaller companies lacking a budget for the kind of investments required by traditional software licensing, the pay-per-use model is a good option. Customers can purchase the software without making special budgeting arrangements or undertaking long decision-making processes. Thus, the software usage resembles an operational cost rather than a capital investment.

The pay-per-use model also has advantages over other models if customers need the software only occasionally, because this decreases costs and lessens the need for customers to set up their own IT infrastructure. As mentioned earlier, it's easier to change vendors if the software's quality or functionality isn't appropriate. Thus, in this model, it's possible to test and evaluate the software to see if it suits the company's needs. It also frees the customer from the need to install, maintain, and update the software, and it lessens the need to have IT personnel. All in all, this means that there are no hidden costs related to software usage—bearing in mind that hidden costs in traditional software licensing can increase a firm's IT budget by as much as 80 percent.[3]

The disadvantages of the pay-per-use method for customers relate to software pricing, estimation of the actual need to use the software, data security, and the permanence of the software provider in the market. In most cases, the price of software offered by the pay-per-use method is fixed—there aren't any price negotiations with the software vendor. In addition, it's often difficult to estimate in advance how much the software will actually be used. So another model might be more appropriate if the software is available and needed on a continuous basis.

Another concern involves data security. If a firm's data is stored on a public cloud, the firm might not know precisely where the data is being stored or if it's mixed with other data.

Finally, for customers who use the software for their core business, the continued existence of the cloud provider is vital. If a cloud provider disappears from the market, the effect on the customer will be immediate and possibly catastrophic.

Software Rental

In software rental, the customer pays a negotiated subscription fee to use the software license for a certain limited time.[1] Software rental shares several similarities with the pay-per-use model. For the vendor, software renting offers more pricing flexibility than the pay-per-use model. The price of the rental can be based on the length of the agreement, the number of users in the customer organization, the software's functionalities, or the company's size. The software might be cheaper for smaller customers than for larger corporations.

The rental model is also easier to apply than the pay-per-use method, which requires monitoring usage. This also means that if the software vendor uses a third party's IaaS or PaaS service to provide the SaaS offering, the software can be priced independently. The third party's costs related to the IaaS or PaaS use can be added later. In the long run, the rental model might generate more revenue than the other models, as long as the vendor can maintain loyal customer relationships.

The disadvantages of software renting for the vendor are fairly similar to those in the pay-per-use model, except that renting excludes the need to maintain auditable records of software usage.

For customers, software renting offers more possibilities for price negotiations and for varying the usage terms and contract length. The total software costs are also predictable and contractually defined, so there are no hidden costs related to the software, and the customer knows the financial resources that must be allocated during the contract.

The disadvantages are again similar to those for the pay-per-use model, except that in the rental model, the customer pays, irrespective of whether the software is used.

Software Licensing

In traditional software revenue models, software is commonly sold as packaged or server-based licensing. In packaged licensing, a customer buys a single license for a single user or computer, whereas in server-based licensing, the software is bought for a certain number of processors running the software. So, the software can only be run on a limited number of computers. In addition to the initial license fee, customers often

Table 1. Comparison of revenue models from the software provider's viewpoint.

Model	Advantages	Disadvantages
Pay per use	• Diversifies the customer base • Promotes network externalities effect • Makes software piracy impossible	• Carries risk of not recouping development costs • Requires maintaining auditable records of usage • Lowers switching costs for customers
Software rental	• Offers flexible pricing strategies • Diversifies the customer base • Promotes network externalities effect • Makes software piracy impossible • Doesn't require metered usage • Increases profits if customers remain loyal • Offers cumulative profits	• Increases the risk of not recouping development costs • Lowers switching costs for customers
Software licensing	• Makes it easier to recoup development costs • Increases switching costs for customers	• Offers no substantial income after initial purchase • Carries risk of license misuse and software piracy

must pay a maintenance fee, which includes technical support and version updates.[9]

In traditional licensing, the benefits for a software provider come from high license fees. These help to cover the software development costs, and they do so in a shorter time than with other revenue models. All other things being equal, a high license fee also increases switching costs for the customer; thus, if the software is appropriate, it increases customers' loyalty. However, there's little income from the customers afterwards. In addition, because the software is installed in the cu stomer's premises, it makes misuse of the license or direct software piracy more likely.

From the customer's viewpoint, traditional licensing can have advantages over other models if the software is needed for a long period of time and if it's for everyday use in the firm's core business. In these cases, buying the license could be more profitable than rental or pay-per-use models. Buying the license also makes it possible to store and secure the data within the firm's own data center.

The disadvantages for customers in the case of licensing include hidden costs related to various aspects including maintenance fees, installation and configuration, hardware, and employment of IT personnel. Thus, the purchase price is often just a small part of the total software costs.[3] Buying the software can be a major investment for a firm, requiring budgeting and decision-making at top management levels. Time is also needed for planning and implementation. In addition, switching costs might increase, and the model tends to tie the customer to the software selected. In traditional licensing, a company also needs its own IT infrastructure—that is, data storage and computing capacity.

Tables 1 and 2 summarize the advantages and disadvantages of the three revenue models from the software provider and customer viewpoints, respectively.

Choosing a Revenue Model

Table 3 illustrates some factors to consider when selecting the most appropriate revenue model. The company's size is crucial. Larger companies might be more willing to buy a software license and use the software in their own data center, especially if the software is needed for critical business processes. Large companies also have more resources and can secure their data within their own IT infrastructure.

On the other hand, smaller companies with limited financial resources might find it more cost-effective to use rental or pay-per-use models. If the target segment for the software is narrow, traditional licensing or software renting can work well. By contrast, the pay-per-use model is better suited to a large customer group, with further possibilities for positive network externalities.

If there are concerns about software piracy or misuse of the software license, SaaS-based revenue models are appropriate, because customers don't install the software on their own computers.

Most customers don't have unlimited possibilities when choosing a revenue model, but when offered the three main models discussed here, Table 4 shows some criteria to consider.

If software is needed for a firm's critical business processes, software licensing is the best

Table 2. Comparison of revenue models from the software provider's viewpoint.

Model	Advantages	Disadvantages
Pay per use	• Doesn't require high initial investments • Suits occasional usage • Allows shift from capital investment to operational costs • Offers low switching costs • Lets customers test and evaluate software suitability • Doesn't require installation, maintenance, or updates • Lowers the expense for your IT personnel • Doesn't require your own IT infrastructure	• Offers the same price for all customers (non-negotiable) • Carries data security concerns • Requires estimating actual usage, which is difficult
Software rental	• Offers negotiable pricing • Doesn't require initial investments • Makes costs are predictable • Allows shift from capital investment to operational costs • Doesn't require separate budgeting • Works well when product is needed for a fixed period • Lets customers test and evaluate software suitability • Doesn't require installation, maintenance, or updates • Doesn't require owning your own IT infrastructure • Lowers the expense for your own IT personnel • Lowers switching costs	• Keeps costs the same whether or not the software is used • Carries data security concerns • Carries risk of software provi der falling out of the market
Software licensing	• Works well when product is needed for a long period • Works well when the software is used for the firm's core business • Offers negotiable pricing • Keeps data stored and secure within own premises	• Requires separate budgeting and decision-making • Requires data storage and computing capacity • Includes hidden costs • Carries high switching costs

Table 3. Factors and their impact on revenue model decisions.

Factors	Pay per use	Software rental	Software licensing
The customer is a large corporation	Low	Medium	High
The customer is a small or medium-sized firm	High	Medium	Low
The target segment for the software is narrow	Low	Medium	High
The software is targeted at a broad customer group	High	Medium	Low
There is risk of software piracy or misuse of a software license	High	Medium	Low

Table 4. Criteria for selecting the most advantageous model for software purchase and the level of impact.

	Pay per use	Software rental	Software licensing
Software needed for critical business processes	Low	Medium	High
Software needed only for occasional use	High	Medium	Low
Limited resources to purchase the software	High	Medium	Low
Need to customize the software	Low	Medium	High
Full and detailed evaluation of the software use costs	Low	High	Low
Ease of use	High	Medium	Low

choice, because the customer isn't dependent on the Internet connection or the provider's continued existence in the market.

However, if the software is only for occasional use, pay per use is best, because it doesn't need closely defined agreements with a software

provider or separate budgeting to purchase the software.

If the customer needs software customized to special needs, software licensing will be the best option, bearing in mind that in some cases the firms that rent out their software products are willing to make small modifications to the software for a particular customer.

Software renting is the only choice if it's critical to have a fully evaluated cost structure for IT expenses, because, in this case, there will be no hidden costs, and the amount used won't affect the software's price. If the customer prefers ease of use, then pay per use will be the easiest way to buy and use the software, as all activities can be conducted online, and the customer won't need to invest in separate IT infrastructure.

In some cases, traditional software licensing still has advantages over the SaaS model, but software rental can provide a trade-off between traditional licensing and the pay-per-use model. It can suit both large and small customers and both wide and narrow target segments. Furthermore, it helps protect against software piracy and provides a predictable and less risky revenue stream. Nevertheless, it's not the most advantageous model in every case. Altogether, software providers can increase profitability and expand their customer base by providing a traditional license for large customers who make extensive use of the software in their core business, a rental model for mid-class users, and a pay-per-use model for occasional users.

From the customer's viewpoint, software renting can provide a trade-off between traditional licensing and the pay-per-use model. It has the particular advantage of making it possible to estimate the software costs and to buy the software without separate budgeting or complicated decision processes.

Acknowledgments

The research reported here was carried out within the framework of the Cloud Software Program, which was governed by TIVIT Oy (www.tivit.fi/Company). TIVIT was nominated to organize and manage the programs of the Strategic Center for Science, Technology and Innovation in the field of ICT. The program was funded by the Finnish Funding Agency for Technology and Innovation (TEKES). In addition, the Finnish Foundation for Economic Education provided financial support for this study.

References

1. M. Armbrust et al., "A View of Cloud Computing," *Comm. ACM*, vol. 53 no. 4, 2010, pp. 50–58.
2. V. Choudhary, "Comparison of Software Quality Under Perpetual Licensing and Software as a Service," *J. Management Information Systems*, vol. 24, no. 2, 2007, pp. 141–165.
3. B. Waters, "Software as a Service: A Look at the Customer Benefits," *J. Digital Asset Management*, vol. 1, no. 1, 2005, pp. 32–39.
4. P. Louridas, "Up in the Air: Moving Your Applications to the Cloud," *IEEE Software*, vol. 27, no. 4, 2010, pp. 6–11.
5. B. Sotomayor et al., "Virtual Infrastructure Management in Private and Hybrid Clouds," *IEEE Internet Computing*, vol. 13, no. 5, 2009, pp. 14–22.
6. Y. Bakos and E. Brynjolfsson, "Bundling Information Goods: Pricing, Profits, and Efficiency," *Management Science*, vol. 45, no. 12, 1999, pp. 1613–1630.
7. B. Mahadevan, "Business Models for Internet-Based E-Commerce: An Anatomy," *Californian Management Rev.*, vol. 42, no. 4, 2000, pp. 55–69.
8. L-M. Sainio and E. Marjakoski, "The Logic of Revenue Logic? Strategic and Operational Levels of Pricing in the Context of Software Business," *Technovation*, vol. 29, no. 5, 2009, pp. 368–378.
9. D. Ferrante, "Software Licensing Models: What's Out There?," *IT Professional*, vol. 8, no. 6, 2006, pp. 24–29.
10. K.D. Hunter, "Metering System with Remotely Resettable Time Lockout," United States Patent no. 5377268, 1994.
11. V. Choudhary et al., "Economic Benefits of Renting Software," *J. Organizational Computing and Electronic Commerce*, vol. 8, no. 4, 1998, pp. 277–305.
12. A. Sundararajan, "Nonlinear Pricing of Information Goods," *Management Science*, vol. 50, no. 12, 2004, pp. 1660–1673.

Arto Ojala *is working as a postdoctoral researcher in the Cloud Software Program in the Department of Computer Science and Information Systems at the University of Jyväskylä, Finland. He is also Adjunct Professor in Software Business at the Tampere University of Technology. His research interests include software business, cloud computing, and the internationalization of software firms. Ojala has a PhD in economics from the University of Jyväskylä. Contact him at arto.k.ojala@jyu.fi.*