Contracting for SaaS

Considerations and caveats

October 2024

Unbalanced and deliberately unclear license terms are increasingly part of software contracts, in particular with SaaS vendors.

If you work with IT contracts, you are likely to encounter them both as license terms and unilateral price increases.

This document outlines some of the challenges, potential ways to address them as well as highlighting scenarios where the advantages of SaaS outweigh the challenges.

RA Advisory

This short paper aims to highlight some of the contractual risks associated with typical SaaS terms. The motivation for the paper is the experience that SaaS software is often adapted uncritically as the “modern” approach and in that process are accepting problematic terms.

The key take-away is threefold. Firstly, it is important to distinguish between the characteristics inherent in the SaaS technology from the contractual terms pushed by SaaS vendors. Secondly, there are a number of situations where the advantages of SaaS like "start small and scale fast" are likely to outweigh the disadvantages. And thirdly that situations where SaaS software supports larger operations with critical functionality should be subject to careful scrutiny as both the technology and the contractual terms can be challenging.

The paper walks through some of the typical challenges and balances them against advantages of SaaS in various scenarios, and finally discusses potential remedies.

1 Motivation

This note is motivated from experiences in numerous engagements on contracting for IT projects, where SaaS vendors participated in the selection process or were incumbent parts of the contemplated solution. The engagements include core business IT systems, enterprise systems, monitoring and elements of mobile networks, thus spanning diverse functionalities.

In these processes, SaaS vendors frequently offered terms that entailed a very unbalanced commercial relationship after implementation. Similarly, for incumbent vendors where their systems were to take a larger role, the terms had the same characteristic.

Further, it is also clear that many customers of SaaS vendors are struggling with some of the challenges discussed in this paper, causing the costs of the use of systems to go significantly beyond originally envisioned levels, and beyond what alternative solutions would cost.

The experiences have been accentuated by the recent price increases from several prominent vendors; while not all SaaS vendors, they apparently have employed similar terms.

The SaaS technology (as introduced below) does carry with it certain operational changes. However, the vendors have, at least in part, managed to convince the industry that the SaaS technology implies a commercial regime that favours the vendors, the "SaaS business model". The basic message, typically, is that you can just use the system and pay for what you use.

The problem from a customer perspective is that what you get and what it costs is pretty much up to the vendor of the system to eventually decide. Anecdotal evidence indicates cost increases of 20% yearly or more even without extraordinary price increases. And the SaaS business model does not follow from the technical model.

A further complication is that many companies today have a "cloud strategy". There may exist "cloud strategies" that are based on a solid cost/benefit analysis, but in most conversations the authors have participated in, the "cloud strategy" is founded on a belief in the technology rather than a specific business case.

Finally, we have seen several recent examples where the "SaaS business model" is having spill-over effects on software that is deployed in classical manner. These vendors take inspiration from the SaaS business model and attempt to gain some of the advantages that it has from their perspective.

Of course, software vendors need to be compensated for their services. Many different license models can secure this. The right price depends on several factors, typically including volume, number of potential vendors and size of deal. The errand here is not to challenge this, but to highlight that as a customer, you should be able to count on that the prices and other terms agreed at contract signature are valid for a reasonable time.

The challenges outlined in this paper pertain mainly to systems that form a core part of the business of the customer. When this is the case, the cost of shifting vendor can be huge, and the implementation of a new system would take years. In such cases, the normal protection offered by classical vendors are important.

If the software is used for a marginal purpose, or it is secured that it can be replaced with relative ease, the problems here are less critical. Similarly, there may be uses in smaller firms where the advantages of SaaS solutions outweigh the commercial challenges.

As a final introductory note, we do not claim to hold the truth of what is good and what is challenging for SaaS solution. But we wish to add a more nuanced perspective to the use of SaaS solution to counter the – in our experience – often uncritical adaptation of the technology.

So, in summary, the motivation of this short paper is the experience that people accept terms that are unreasonable and have potentially significant cost implications. And the purpose, following that, is to highlight some of the typical challenges and discuss methods of countering them.

2 Introduction

SaaS is short for "Software as a Service" and implies a delivery method of a solution characterized by:

1. Centralized hosting of the software; typically, in the "pure SaaS" model, it is not possible to procure the software for local installation.
2. Multi-tenant hosting, where many customers run as users on the same hosting instance.
3. Delivery over the internet, so customers can sign up to the use of the software online and immediately start consuming the service.
4. Centralized and vendor-controlled delivery cycle.

The SaaS vendors typically provide a software functionality like helpdesk, service management, CRM, billing, ERP, or other business functions. This makes SaaS different from general cloud services, like renting a server with different level of software; these are typically termed merely "cloud" or "IaaS" (Infrastructure as a Service) or "PaaS" (Platform as a Service). The general cloud services have similar challenges as SaaS, but the lock-in is typically lighter, making the challenge correspondingly smaller. IaaS, PaaS etc. are not the topic of this paper, but an online search for "why we are leaving the cloud" or similar phrases reveals similar challenges.

Also, SaaS should be distinguished from a full-service hosting. The result from the end-user and operational staff perspective may be almost identical, but the commercial models can be very different.

3 The contractual challenge

This section dives into more detail of the challenge with the contractual terms embedded in the SaaS business model (and, due to the general spill-over, in other pushes from software vendors).

Typically, the SaaS vendors offer contractual terms characterized by:

1. Subscription based on certain measures, some of which are manageable by the customer, e.g., users, some of which are not, e.g., storage consumption or transactions.
2. Comparatively short contract term that are "balanced" in the sense that prices only are known for the committed contract period (see below in section 3.2 for further discussion of this).
3. Unilateral right on the vendor to set prices following expiry of the term.
4. Procurement of the software and implementation as two distinct transactions.
5. Depending on the vendor, limited accommodation for compliance protection, e.g., national autonomy.

These items, typically along with others of similar nature, constitute what in this document is termed the "SaaS business model".

The items are further elaborated below, but the basic issue is that contractual terms as outlined above imply:

1. Limited functional predictability.
2. No or limited price predictability.
3. No or limited control of compliance related issues.
4. Obligation to pay for the software independently of a successful implementation of the software.

For a core system with a replacement time that can reach 3-5 years and with substantial implementation costs, such uncertainty constitute a non-trivial risk.

The rest if this section discusses key concerns related to the SaaS business model in more detail.

3.1 Functional predictability

With a classical, perpetual software license, the support and maintenance procured with the license provides regular updates of the software.

For contracts where proper care is taken, the support and maintenance include (in addition to the normal assistance, patching etc.):

1. Access to all versions of the software, not only smaller releases.
2. Maintained functional support for such functionality used by the customer.
3. Security against additional cost in case of re-packaging, e.g., through moving functionality into new modules that are not procured and discontinuing it in old ones.
4. Known prices, in principle for perpetuity. Or at least for an initial term with change notified with enough time for a replacement to happen.
5. Value preservation, i.e., in case the vendor starts marketing a different product instead of the one procured, the customer has a right to transition to the new product.

Such regulation provides the customer with a good assurance that the software will enable the business of the customer as originally envisioned for the normal life cycle of core systems.

3.2 Balanced duration and termination

In many businesses, IT systems operate the core processes in a manner that makes the business critically dependent on the systems. A replacement timeline of up to five years is not unusual, including decision cycle, sourcing, implementation and stabilization. Further, replacement often runs into tens of millions of Euros.

For these reasons, having known contractual terms, including pricing, for a period corresponding to the 5-year timeline, is important. Further, due to the size of the investment, an initial term of similar length, is important. Hence, to secure a reasonable value from the investment for the customer a contract term of five years following 5-year termination notice from the vendor is required.

Conversely, a committed term of 10+ years on the part of the customer in a rapidly changing world may preclude timely business-critical adaptation.

For the vendor, losing the business of a customer is an inconvenience and can even require adaptation of the internal cost structure. However, such adaptation typically can be done within a matter of months. Hence, while it is convenient for the vendor to have a long contractual term, the business impact of a short termination notice is nowhere near as serious as lack of core functional support is for the customer.

Similarly, the vendors have limited downside of very long contracts (unless they intend to utilize shorter contracts to unilaterally change prices etc.).

The typical vendor position is that "balanced" means equal terms for the parties on termination. From the discussion above, it is clear that equal terms do not address the respective business challenges in a balanced manner.

In most contracts, again assuming a proper process and proper care, it is possible to land a compromise of an initial term of 3-5 years after which termination can be 1-2 years on the customer side and five years on the vendor side.

Assuming that the licenses are not bought independently of the implementation (see section 3.5), this is workable as the initial term is comparable to the typical implementation time.

3.3 Price predictability

Any business case for implementing a new system includes charges to the vendor, both implementation and on-going charges. Such business cases naturally, but typically tacitly, assume that the charges are not impacted by factors external to the business case.

Choice of vender, similarly, typically depends on price as one of the key deciding factors.

Further, business plans and budgets assume known cost levels, dependent on known or controlled drivers like inflation and number of customers.

All these important aspects fail if prices are not predictable.

Predictable pricing requires:

1. A finite set of a priori known parameters that affect prices.
2. Parameters are either directly linked to business outcomes (like customers) or general external (like inflation).
3. Prices include commitment to uphold functionality (see section 3.1).
4. Change of pricing only with same notice as termination (see section 3.2).

To illustrate the concepts, consider the following real-life examples that do not satisfy the price predictability demand.

1. Option for the vendor to increase prices yearly with a high cap, e.g., 10 percent. After seven years, this can mean doubling of prices.
2. Price cap subject to (oftentimes outrageous) standard prices that no-one ever takes. After a

limited duration of an initial term, prices can be anything.

3. Charges per transaction, where transactions are defined as an internal technical concept in the system, potentially dependent on events outside customer control, e.g., end-customer behaviour.
4. Charges based on consumption of data storage with no customer control of evolution.

In classical contracts, hardware costs are still covered by the customer, so transactions and storage have cost impact. This risk is normally acceptable since the direct hardware costs are limited compared to total system costs.

3.4 Compliance

The nature of SaaS systems is that they are hosted on centralized locations. This has historically been challenging with the GDPR directive in EU. The GDPR requirements have caused the SaaS vendors to implement data centres within the EU with guarantee against data exchange outside the EU.

This type of challenge is likely to become more extensive, for example:

1. The NIS2 directive demanding increased security.
2. The broad definition of "critical infrastructure".
3. Demands of "national autonomy" in certain sectors, or at least fallback plans.

While general security is likely to be as good (or better) for SaaS systems than on-prem, it is not within customer control.

Almost by definition of SaaS systems, the national autonomy can be difficult to obtain, at least for smaller countries where the SaaS vendors are unlikely to deploy data centres. Further, the operations of SaaS systems are typically distributed and is not generally fully transparent, e.g., what activities are conducted from which locations and by staff of which nationalities.

These issues are to a large extent due to the SaaS technology. So, from a contractual perspective, fully addressing them almost demands that the SaaS vendors to stop being SaaS vendors.

As this hardly makes sense, it will be a dialogue on what derisking measures the SaaS vendor can manage and, for the customer, what can be an acceptable risk. This assessment should include whether the SaaS technology is suitable given the compliance requirements.

3.5 License vs. implementation

In the SaaS business model, the vendor often wishes for the customer to procure license and implementation independently. Implementation can be sourced with the SaaS vendor, but also from a number of partners or even implemented by the customer internally.

In itself, there is no serious problems with such a model. In most projects, there is some costs associated with development and test environments.

The complication arises in combination with license price negotiation. Virtually all larger projects secure significant discounts on the list prices. However, such discounts frequently come dependent of a purchase commitment in time and license.

The implication is that the customer carries a risk of a significant cost of license that only provides value if the corresponding project succeeds. With the success rate of large IT projects, the risk cannot be discounted.

The challenge is not isolated to SaaS vendors. Other companies selling software only can present similar problems.

The very simple resolution to this is to pay list prices during the initial implementation project and negotiated pricing after the project. This, of course, needs to be agreed before the project starts and some vendors are very reluctant to accept such models.

3.6 Disputes

A frequent contractual discussion relates to the "fix first, settle later" concept. The purpose, from the customer perspective, is that the vendor continues to fix whatever problem is in the system, irrespective of any disputes. And, conversely, that the vendor does not utilize the control of the customer's operations to get an upper hand in negotiations.

The ultimate threat is that a vendor has the power to close down operations in case of a dispute. This is a fairly extreme case, but nonetheless a power that is in the hands of the vendor.

This issue is related to different kinds of outsourcing and is not isolated to SaaS. However, SaaS being a fairly extreme form of outsourcing, it is also a consideration for SaaS contracts.

There is no easy fix for this risk; in part it can be countered by not accepting confidentiality on disputes, seeing that the vendor typically not would want such behaviour widely published.

3.7 Not your change management

The SaaS concept implies that new releases are deployed centrally and with limited customer control. The vendors employ various techniques and processes to limit this problem. But at the end of the day, new releases appear in the customer production environment without the normal internal change management control.

This challenge is inherent to SaaS and the vendors are very reluctant to put contractual measures in place to handle it. Such contractual measures could, for example, be penalties for introducing defects from a centrally controlled release.

Contrary to many of the other items discussed here, the SaaS vendors have good technical reasons to be reluctant to accept such items: in case every customer needs to be verified, some of the scale advantages of SaaS are challenged.

To the extent of the experience of the authors go, this fortunately does not appear to be a major problem in practice. So, for the purpose of contracting, the important part is to make sure that the quality assurance measures are understood, appropriate penalties for defects are in place and that the internal service processes (ITIL or the like) are aligned with the SaaS model.

3.8 Capitalization

In some businesses the ability to capitalize development projects are very important to secure market conform financial statements.

The fact that SaaS systems are inherently rented means that in many situations, the ongoing charges will be treated as operational expenses. In some interpretations of the rules, even the implementation project cannot be capitalized, which in some settings can be almost prohibitive for using SaaS.

A discussion of IFRS interpretation is far beyond the scope of this paper. The point here is merely to highlight that before embarking on implementing SaaS, one should understand the parameters of when capitalization is possible and the importance of capitalizing the expenditures.

3.9 Summary

In summary, the topics discussed in this section are set out in the table below, with indication of whether the SaaS technology implies that the topic is contracted differently from classical software deployment. The column "Inherent SaaS" indicates that the push for non-fulfilment of normal balance for the specific topic is driven by the SaaS technology.

Topic	Section	Inherent SaaS
Functional predictability	3.1	No
Balanced contractual term	3.2	No
Price predictability	3.3	No
Compliance	3.4	Partly
License vs. implementation	3.5	No
Disputes	3.6	Yes
Change management	3.7	Yes
Capitalization	3.8	Partly

As the summary shows, most of the topics discussed are mainly commercial and not related the SaaS technology.

The items that are inherent to the SaaS technology must be dealt with in a combination of contracting and customer adaptation in terms of risk and processes.

4 Advantages of SaaS?

Challenging SaaS as a concept in the present day is, at least by some people, considered heresy at the same level of, say, claiming that the world probably does not need 6G, or that cloud native does not solve all known problems with IT.

So, the main purpose of this paper is not to discuss whether SaaS basically is a good concept, but to highlight typical contractual challenges.

However, having seen that certain contractual terms are inherent to SaaS, we find that it briefly makes sense to touch upon the advantages as well as typical misconceptions.

We would also like to highlight that we ourselves have – and are – using SaaS solutions with very satisfactory outcomes. However, we are also well aware of the challenges and have adapted our use accordingly.

4.1 SaaS positives

The quintessential SaaS use case is the "easy start" of starting quickly and scaling fast. This utilizes core strengths of the SaaS model.

The scenario where a complex existing business is to be transferred to a new solution can also utilize some strengths of SaaS, but the fit is a less obvious one. In contrast to the easy start use case, major systems undertakings have so many complexities that the advantages of SaaS – that still exist – are dwarfed by the overall complexity. Or, in other words, adding infrastructure and software installation on a major IT transformation adds comparatively limited complexity to the overall undertaking.

Some of the key strengths of the SaaS model is outlined below.

4.1.1 *Easy start*

The SaaS model typically permits a customer to go on a website and subscribe to the solution after a credit card payment. This, clearly, is extremely efficient comparted to the classical approach of installing hardware, operating system, database system, software, all within a firewall protected environment. And what is typically even more challenging: to have to maintain the whole thing.

4.1.2 *Fast scaling*

Closely linked to the easy start is that fast scaling. The process is very similar: swipe the credit card to get more users, customers or whatever measure the system in question uses as license driver.

4.1.3 *Mature multitenancy model*

In order for the SaaS model to function, SaaS vendors have implemented robust models to handle many customers on the same platform.

In doing so, the vendors typically have implemented efficient and robust processes for handling new releases, providing online guides for configuration and adaptations.

When implemented well, the customers benefit from this through easy sign-up, seamless upgrades and integrations etc.

4.1.4 *Ease of adaptation*

Many SaaS vendors have implemented internal programming or configuration tools, which can be in the shape of GUI setup, scripting or more classical code.

Adaptation using these tools, particularly for simpler scenarios, tend also to comparatively easy. The reasons include that tools are modern and simple and that many standard use cases are supported or anticipated.

4.1.5 *Security*

With increasing challenges in security, both in terms of heightened risks and increased government focus, building adequate security in a small IT environment can be a daunting challenge.

Part of the SaaS technology is that this is outsourced to the SaaS vendor. As their business depend on people trusting their systems to be safe to subscribe to, they are highly motivated to implement effective security measures. And from the combination of their obvious attractiveness as targets for criminals and the absence of major (publicized) stories, they also appear to be quite good at it.

This point emphasises the "easy start and scale" use case.

4.1.6 *Implementation vendor ecosystem*

While not inherent to the SaaS technology, in practice many SaaS vendors have a combination of size and openness that attracts an ecosystem of implementation vendors.

Hence, it is possible to source implementation and maintenance competence from a broad source of vendors. Similarly, in the absence of sudden surge of demands, it is possible to build inhouse competence to handle development.

4.2 *SaaS misconceptions*

As outlined in section 4.1, there are several advantages to the SaaS model. Some are inherent to the SaaS technology; others are a consequence of the specific situation.

This section addresses some typical arguments for implementing SaaS that is not directly linked to the SaaS technology.

4.2.1 *Price*

A frequently quoted argument is that SaaS is cheaper. While correct in some cases, it is not generally true. This section looks at some of the price components that form part of the typical IT installation.

Implementation for the "start small and fast" use case is typically very much cheaper with a SaaS solution. And also adapted by many smaller companies. For larger IT transformations, the implementation costs for SaaS systems are similar or even higher than classical systems. The friendly salesperson will frequently point to a case where their system has transformed a similar operation in just four months, but experience shows that like for like, time and cost are very similar for SaaS systems compared to classical systems.

Subscription license costs, irrespective of whether they are for a SaaS system, are obviously initially lower than classical procured licenses. Depending on the specific license model, WACC used etc., the lines typically cross after 5-7 years after which subscription based models are more expensive.

For large scale operations, SaaS, IaaS, PaaS etc., tend to be more expensive over time. For IaaS and PaaS it is directly visible, whereas it for SaaS tends to be less transparent as the SaaS fees include multiple elements.

Maintenance cost is more difficult to evaluate as it depends very much on the specific context. In favour of at least some SaaS systems count is argued the restrictions imposed by the technology that may limit the overall complexity. However, based on concrete experiences, this is not universally true.

4.2.2 Lack of customization

In simpler use cases, SaaS systems can be employed with very limited adaptations. For more complex implementations, the number of adaptations is corresponding larger.

SaaS systems often have predefined processes, which is an argument for the limited customizations. But this is countered by a similar development in all major systems: standard processes are emerging and implemented in many standard systems. Hence, while it is true that SaaS systems reduce customization, it is not a characteristic unique to SaaS systems.

From estimates of larger IT transformation efforts, the indications are that implementation costs are not materially dependent on the technology model. This also indicates that for larger efforts, the level of configuration and customization are comparable.

4.2.3 Multi tenancy

The fact that SaaS systems are inherently multitenant leads some to conclude that they are also well suited for use with multiple business units, i.e., providing a good structure for sharing configuration while allowing individual adjustments.

Such multitenancy is related to how customer configuration and customizations are managed and depends on the individual system. The SaaS technology does not inherently allow for such configuration sharing.

Therefore, this must be subject to specific evaluation for the individual system.

4.2.4 Ease of contracting

An argument that is occasionally put forward is that SaaS vendors are easy to work with since they have a standardized contractual framework that all can sign up to. The argument is sometimes voiced on the customer side, but more frequently on the vendor side.

If you think this is a ridiculous argument, you are right and you can skip to the next section.

Firstly, this is not a characteristic by SaaS. They may have an online onboarding process where you sign up to standard terms. But almost all companies have a set of standard terms that you can sign up to if you wish.

Secondly, such standard terms are obviously written to protect the vendor. In some cases, they are reasonably balanced and in others they are just outrageous. They are, however, never protecting the customer to the extent of a contract resulting from a proper sourcing process can.

4.2.5 Freedom from IT

As will be well known, IT departments are instituted by mankind with the purpose of delaying and generally frustrating business development, in particular customer offers. Hence, customer focused business units tend to wish for freedom from the constraints of the IT prioritization process.

From this fairly universal dissatisfaction with IT department follows a frequent, more or less disguised, argument that a given functionality can be handled better if a system is managed directly by business. And SaaS systems, due to the easy start, are well suited for this.

This paper is not about IT governance, but there are definitely good reasons for distributing responsibility for IT. Similarly, there are good reasons for securing that the corresponding complexity is dealt with diligently and competently.

Executing on a decentralization through implementing a specific system instead of through a reasoned IT strategy and governance evaluation process is akin to treating the symptoms instead of the underlying disease.

5 Countering the challenge

Dealing with the challenge encountered with the typical SaaS vendor standard contract depends on the scenario, in particular whether the challenge relates to the simple or complex use case.

5.1 The simple use case

For the simpler use cases, where smaller entities employ SaaS solutions, it is rare to get concessions from the vendor. For small volumes, it is also generally not viable to run a full sourcing process: the vendor will not respond seriously (if at all) and they will not change any standard terms for a small volume. Also, the cost of sourcing can easily exceed the SaaS vendor cost at risk.

Therefore, a smaller entity will need to deal with the associated risks – it is basically “take it or leave it”. Similar considerations can also apply to a limited use within a larger corporation.

Such risk management can for example include:

1. Having a regularly updated plan for transition to alternative systems.
2. Securing that all data is available in a readily readable format.
3. Budget contingency to deal with sudden cost increases.

4. Fallback measures in case of connectivity disruption.

These are all measures that fall on the customer.

5.2 *The complex use case*

In the complex use case where the contemplated system would be part of a larger IT transformation, the volume of business is such that the vendors, including SaaS vendors, may be willing to adjust their standard terms. They reason we write "may" is that quite some of these SaaS vendors have grown so successfully to a dominant market position that they are unwilling to adapt their approach.

A proper sourcing process can secure one of two outcomes:

1. That the potential vendor does in fact change the terms to be viable.
2. That the challenges with the terms are visualized in a manner that makes the business risk clear.

A "proper" sourcing process can be executed in many ways, but generally is a structured approach of setting demands, evaluating responses and selecting vendors in multiple steps. Examples of such a process can be found in other white papers on the web site where this paper originated.

The counter for non-SaaS systems that try to introduce similar terms as the SaaS business model is basically the same: a proper sourcing process.

6 Summary

In summary, the SaaS vendors promote both a SaaS technology and a SaaS business model.

The SaaS technology has a number of advantages but also certain operational implications. For the simple use case, the easy start and scaling can be determining for the choice of a SaaS system, and in many cases the alternative would be significantly more complex and costly.

For more complex use cases, where the dependency to the core business and the compliance requirements typically are higher, the terms of the SaaS business model should be subject to more diligent scrutiny through a proper sourcing process.

In executing such process, it is important to bear in mind that there is nothing in the SaaS technology that implies the SaaS business model. The link is merely a choice made by the vendors in an attempt to increase their revenue, both in absolute amounts and in the control they have over it.

Therefore, in case a potential SaaS vendor insists that the SaaS business model follows, then you should consider whether your "cloud first" strategy has enough benefits to justify the risks.

7 Contact

The authors of this document are working at RA Advisory, a niche consulting firm focusing on procurement and implementation of IT and network equipment.

We can be contacted via: info@ra-advisory.dk or on +45 5070 0070. This and other white papers are available free on www.ra-advisory.dk.