Electronic Markets (2018) 28:53–75
https://doi.org/10.1007/s12525-018-0284-7

RESEARCH PAPER

Cloud providers viability

How to address it from an IT and legal perspective?

Cesare Bartolini[1] · Donia El Kateb[2] · Yves Le Traon[1] · David Hagen[3]

Received: 12 April 2016 / Accepted: 2 January 2018 / Published online: 22 January 2018

Abstract

A major part of the commercial Internet is moving toward the cloud paradigm. This phenomenon has a drastic impact on the organizational structures of enterprizes and introduces new challenges that must be properly addressed to avoid major setbacks. One such challenge is that of cloud provider viability, that is, the reasonable certainty that the Cloud Service Provider (CSP) will not go out of business, either by filing for bankruptcy or by simply shutting down operations, thus leaving its customers stranded without an infrastructure and, depending on the type of cloud service used, even without their applications or data. This article attempts to address the issue of cloud provider viability, defining a possible way of modeling viability as a non-functional requirement and proposing some approaches that can be used to mitigate the problem, both from a technical and from a legal perspective. By introducing a structured perspective into the topic of cloud viability, describing the risks, factors and possible mitigators, the contribution of this work is twofold: it gives the customer a better understanding to determine when it can rely on the cloud infrastructure on the long term and what precautions it should take in any case, and provides the CSP with means to address some of the viability issues and thus increase its customers' trust.

Keywords Cloud · Viability · Standardization · Service Level Agreement (SLA) · Software escrow

JEL Classification G3 · K

Introduction

In November 2013, an alarm rang. Nirvanix, a major provider of cloud storage services operating in California, suddenly closed down its business. Nirvanix was one of the major stakeholders in cloud storage services, and customers had to recur to insourcing or migrate to other cloud providers. However, customers were notified only two weeks before the final shutdown (Butler 2014). Hosting terabytes of data on Nirvanix services, they had to deal with a considerable interruption time that entailed a halt of their running services. The impact of this incident on some major enterprizes, including IBM (who used Nirvanix's cloud storage technology) and Dell (who had some agreements with Nirvanix) has not been disclosed. The sudden event wroke havoc in the cloud community, and revealed a significant weakness of the cloud-based business model: once an enterprize outsources services and data to a cloud provider, it is no longer in control of them, and can suffer from adverse conditions occurring to the cloud provider.

Nirvanix was not an isolated case, as several Cloud Service Provider (CSP) incidents have been reported over the last years. In 2012, Megaupload, one of the pioneering companies in providing storage services, has been shut down by the US Department of Justice which started an investigation against its employees (Anthony 2012), and its founder Kim Dotcom created a new storage service known as MegaCloud. In late 2013, this service disappeared as well (McKendrick 2013), possibly due to an NSA blockade. Users have suddenly lost access to the areas where they had stored the files.

Responsible Editor: Jörn Altmann

✉ Cesare Bartolini
Cesare.bartolini@uni.lu

[1] Interdisciplinary Center for Security, Reliability and Trust (SnT), Université du Luxembourg, Luxembourg, Luxembourg

[2] LuxTrust, Luxembourg, Luxembourg

[3] Commission de Surveillance du Secteur Financier (CSSF), Luxembourg, Luxembourg

The case of the 2e2 company highlights other interesting consequences. After 2e2 went bankrupt, it first asked its customers (such as Vodafone and Kellog) a large amount of funding to keep the service up and running (Venkatraman 2013). Shortly after, it was discovered that the company had outsourced most of its customers' data to third-party services. After the bankruptcy, the data centers were acquired by a company that guaranteed that it would keep providing the service to customers.

While the outsourcing of services to the cloud is definitely the business model toward which the market is heading, it introduces new and significant challenges. Some of them are simply a new way of addressing well-known issues, such as performance issues, resource availability, service security and reliability, but others are strictly related to the outsourcing of parts of the business to the CSP. Outsourcing assets to a CSP is a critical decision that requires a careful weighing of its costs and benefits (Haile and Altmann 2013).

The risk of the CSP suddenly going out of business, either through a "soft" cessation (i.e., filing for bankruptcy), or in a more abrupt way (simply ceasing the operations and removing the assets) concerns the uncertainty about the stability of the CSP. If the CSP faces a bad financial situation, it might decide to go out of business, and this could happen more or less abruptly. For example, the CSP might file for bankruptcy, leaving its customers to entertain their business with the trustee instead of the regular CSP management board. Even worse, the CSP might not go through the legal shutdown, and instead simply cease to do business, shutting down all operations and leaving its customers stranded and unable to use their required infrastructure anymore. And all this might occur without any forewarning to the CSP's customers.

This is the problem of the so-called *long-term viability* of CSPs, one of the main factors to take into account when moving to a cloud platform. Normally, the migration to a cloud is a one-way door. Unless there is a major change in the size of the enterprize's business (in which case there might be pressing needs to adopt an in-house architecture), the choice to rely on a CSP to run its business, both as a starting choice or after migrating pre-existing applications, is a long-term one. The relationship between the client enterprize and the CSP is likely to last until either one goes out of business. But there is a lot of difference between the client enterprize and the CSP going out of business. In the former case, the CSP would lose a customer, which might have serious repercussions from an economic point of view, but this fact does not per se hinder the CSP's operations. On the other side, the CSP going out of business would seriously hamper the client's operations, regardless of the client's business status. Even if the enterprize's activity were solid and growing, the CSP's end of business could block all of its IT activities, because the underlying infrastructure would not be accessible anymore. Of course, the customer can always switch to another CSP or set up an in-house architecture, but this in turn raises new problems concerning time and money.

This paper addresses the issue of CSP long-term viability. Due to the novelty of the topic, there is no analysis of viability from a requirements engineering point of view. To address the problem, it is necessary to formalize it, also providing metrics for risk assessment. The purpose of this work is to evaluate what solutions (mostly not specifically designed for viability) the CSP and the client can adopt to avoid the consequences of the CSP suddenly going out of business, and what further problems emerge from these solutions. We discuss the issue from two perspectives, namely an IT and a legal point of view, and propose a list of technical and legal measures to mitigate the problem.

In particular, this paper mainly aims at addressing the following questions.

RQ1 What is long-term viability and what effects does it have on enterprizes?

RQ2 What is a possible formal model to express long-term viability?

RQ3 What are the possible means to increase the long-term viability of a CSP?

To answer the first question, an analysis of the (few) sources dealing with the issue of long-term viability will help define the risks and consequences, also on the basis of the nature of the CSP, and separate it from the related, and more popular, topic of reliability. The answer to the second question will leverage on Requirements Engineering (RE) techniques and tools to model long-term viability and find suitable metrics for it. Concerning the third question, the answer lies in an analysis of solutions already adopted by the industry or introduced in literature, with a proposal to classify them and evaluate their potential benefits.

Conversely, a question that this paper does not address is the one concerning the convenience, for a CSP, to enact some of the proposed solutions on the basis of economic factors. This work is focused on legal and technical implications of the proposed solutions, and a detailed cost analysis entail a different approach, as it is extremely complex to assess many of the factors that affect the benefits and costs of implementing them. Without delving into an in-depth quantitative analysis (which will be considered as a future work), the paper only proposes a coarse-grained evaluation of the proposed approaches, in particular by classifying them depending on the type and size of the CSP. Such an assessment can already provide some useful insights that could be used as a starting point for a more detailed analysis.

The problems and solutions presented here, to some extent, can be applied to service providers in general. However, this paper mainly focuses on the cloud for two main reasons. On one hand, the concept of cloud is quite wide and tends to embrace many different types of service providers. For example, the SaaS paradigm represents the business perspective of Service Oriented Architecture (SOA). On the other hand, outsourcing assets to the cloud has a more significant impact on the IT infrastructure than relying on external service providers for minor activities such as hosting or content delivery.

This work is structured as follows. "Problem statement" delves into the concept of cloud computing and illustrates the problem with respect to the various types of cloud. "Related work" is a survey of literature on cloud viability and some approaches that could partially address the problem. "A mathematical model of viability" proposes a mathematical model of viability, to assess the seriousness of the problem and evaluate the improvements that can be achieved by means of specific mitigators. "Addressing the problem" tries to build on existing research to propose solutions or mitigators based on technical ("Discussion of possible IT solutions") or legal (Legal perspective and solutions) approaches, or even mixed solutions (Mixed approaches). The, "Considerations on the proposed solutions" summarizes the proposed solutions, together with a high-level estimate of their potential costs and benefits, and describes a real-life scenario that is recurrent in Luxembourg as a European financial center. A cloud provider is hosting a Web Banking application that represents the main IT service of a large Bank in Luxembourg, with an interface to perform financial transactions for many customers. The techniques used by the CSP that might affect long-term viability are shown. Finally, "Conclusions" tries to wrap up the outcomes of the paper.

Problem statement

The current paper focuses particularly on the problem of service providers offering cloud services. The differentiating criteria between a CSP and other types of services depend upon the definition. Referring to the definition used by the Luxembourgish Commission de Surveillance du Secteur Financier (CSSF) (Surveillance du Secteur Financier (CSSF) 2017), a number of criteria are defined for a service provider, and if not all of them are met, the provider cannot be qualified as a CSP. The definition relies on several features to define the activity of a cloud provider, such as on-demand self-service, broad network access, resources pooling, rapid elasticity and measured service. The outsourcing of services is subject to several security practices that enable to protect the customers' hosted data. However, the model and solutions presented herein could be adapted to other service providers that do not fit into the definition of CSP.

Depending on the amount of the assets that have been outsourced, the effect of the CSP going out of business can be much different (Caplan 2010). When dealing with IaaS or PaaS CSPs, the main assets (application source code and data) are under the control of the customer, who can migrate them (with different technical difficulties between IaaS and PaaS); in a SaaS environment, on the other hand, the customer normally has no access to the application source code or the datacentre where its data are stored, plus it might also lack the technical skills to perform a migration.

In particular, with respect to the most common types of cloud models:

- the loss of IaaS platforms[1] would require the client to find or set up a suitable hardware architecture;
- PaaS clouds would require it to also set up the basic software platform for running its applications. This configuration is very common for those enterprizes which offer web-based services to their end customers. In general, the customers of PaaS clouds[2] are application developers who offer their applications through the cloud;
- finally, in the SaaS paradigm, the client (the end user of the IT service) normally only uses a web browser (thin client) to access the service, so a CSP going out of business completely cuts its clients off their IT applications and data.

The problem might depend not only on the CSP but also on other third-party partners going out of business in case the CSP has in turn outsourced some of its services (Caplan 2010), with a potential cascading effect. Additionally, the size of the problem depends on the degree of interoperability and standardization adopted by the CSP (Gebregiorgis and Altmann 2015; Jeffery et al. 2015).

Long-term viability is one of the major risks that must be taken into account when choosing to rely on a CSP to support one's software applications. The money that can be saved by relying on an external infrastructure (Koch et al. 2012) might be counterweighted by the risk of having the business stalled for some time, or the software and data lost due to the shutdown of the CSP.

The risks related to long-term viability change with the size of the CSP. Relying on major CSPs, such as a large provider with an insignificant risk of abruptly shutting down, mitigates the problem. But the type of service offered by these CSPs might not be suited for the needs

[1] The reference example is Amazon Elastic Compute Cloud (EC2) (http://aws.amazon.com/ec2/).

[2] Such as Google App Engine (https://appengine.google.com/start) or the Salesforce1 platform (http://www.salesforce.com/platform/connect-integrate/).

of the would-be customer. Additionally, while the potential customer enterprize might enjoy some negotiating power against smaller CSPs, so as to adjust the cloud functionality to its own needs, no such power will be available against major corporations, leaving the enterprize the sole options to accept or reject the standard offer. The enterprize might have political reasons to prefer a small CSP over a large one (for example nationality); the reasons might be economic (favorable offers by the small CSP); or the choice might be pushed by the need for interoperability with partners relying on a specific cloud. However, small, start-up cloud providers are more subject to market fluctuations, and might at some time decide to stop providing hosting services to their customers, or to move to another business model.

The negative impact that the bankruptcy of a CSP can have on its customers can vary greatly, depending on the delivery model of CSP (IaaS, PaaS or SaaS), on the nature and amount of the assets outsourced, and on the technical infrastructure. Without carrying out a detailed analysis of all the possible combinations, it is possible to identify some of the main threats.

- The most likely issue is the loss of the outsourced assets, which might be data (especially in the less-pervasive delivery model SaaS) or the technical means required to offer one's services. Lost assets require the customer to rebuild them, which can entail a major investment of time and money, or face the consequences of the loss, which might lead to legal remedies to refund of potential damages.
- Even if the assets are not lost, recovering them might take a significant amount of time through the bankruptcy estate, and this in turn might lead to significant losses in the *interim*.
- In addition to the assets problem, the customer might face some difficulty in finding or setting up a suitable alternative to restart running its operations.
- All of the above situations entail a significant cost in terms of planning, time and money, in addition to the possible loss of revenues and reputation due to prolonged inactivity, and the customer, especially a small one with limited financial resources, might be unable to face such costs. Therefore the CSP going out of business has the potential of dragging its customers into a similar fate.

Over the last years. cloud providers' viability has exposed some criticalities of the cloud-based business model. According to Gartner, one of the major risks posed by cloud adoption is the uncertainty related to providers viability, and one out of four cloud providers were deemed eligible to go out of business in 2015 due to some reasons such as bankruptcy or acquisition (Thibodeau 2013).

Long-term viability is in the interest both of the client enterprize and the CSP: the former has an interest that the cloud infrastructure will be available and able to guarantee its service, whereas the CSP wants to be seen as a reliable entity which enterprizes can rely upon, so both parties might be willing to undertake some measures against the risk of a sudden disappearance of the CSP (bankruptcy, acquisition by third parties and subsequent termination, forced closure by public authorities and so on).

Related work

While the need that a software service is available and functioning for long periods of time is certainly a fundamental requirement, there is little consensus in literature concerning its classification. Most sources (Hennesy 1999; Avizienis et al. 2004; Glinz 2007; Lee et al. 2009) classify it either as reliability (Roman 1985) or availability (ISO 2011). Other sources (Laprie and Kanoun 1996) define availability as the readiness for usage, and reliability as the continuity of service, and both are a subset of dependability.[3] In general, however, this requirement is treated in the perspective of a continuity of business. In other words, it is a requirement that measures the amount of transient situations that prevent the usage of a service.

Some clear overlapping exists between the topics of reliability and long-term viability. The former is seen as a major concern (Hu et al. 2011), and several models, metrics and solutions have been proposed (Bauer and Adams 2012; Franke et al. 2014). In particular, (Glinz 2005) proposes an approach which distinguishes between those requirements that can be assessed with quantitative, qualitative or descriptive metrics. van Moorsel (2001) separates the traditional Quality of Service (QoS) from two other parameters, namely Quality of Experience (QoE) and Quality of Business (QoBiz), which represent, respectively, the quality of the provider as it is seen from the user, and the ability of the provider to make profits. The author relates the various qualities differently in the provider chain, B2C and B2B scenarios.

Concerning reliability in service compositions, Zo et al. (2007) present a method to evaluate the reliability of service compositions. Based on a description of the various possible structures of service compositions, it evaluates the combined reliability for each structure. To model service compositions, Bocciarelli and D'Ambrogio (2011) define a BPMN extension to model reliability requirements.

[3] Today, an official definition of reliability is "the ability of a system or component to perform its required functions under stated conditions for a specified period of time" (ISO 2010, p. 297).

Another perspective for the continuity of IT activities is that of business continuity. Since the interruption of IT infrastructures can seriously disrupt a business, IT continuity is essential to achieve business continuity and should be a part of an effective Business Continuity Management (BCM) (Wan 2009), which in turn is a goal that has been deeply targeted by several international standards, such as BS25999, now superseded by ISO 22301 (ISO 2012).

A framework known as Information Technology Infrastructure Library (ITIL) is a set of best practices for managing IT services, developed over the last decades, that has inspired a number of the more recent international standards. A significant component of ITIL is Information Technology Service Continuity Management (ITSCM) which, in the key area of service delivery, expressly addresses the need of avoiding outages and failures of IT infrastructures. ITSCM represents the IT equivalent of BCM (Fry 2004). To ensure IT continuity, an enterprize must first perform a risk assessment from an IT perspective, identifying the various potential threats and their estimated disruptive effects on the business. Several models exist to perform this assessment (Guo et al. 2012). On the basis of such an analysis, it is possible to define a continuity strategy, which should be both proactive, to avoid incidents as much as possible, and reactive, to mitigate the consequences of unavoided incidents (Roa Martínez 2011).

Other frameworks exist (Sallé 2004; Năstase et al. 2009), such as the ISACA Control Objectives for Information and related Technology (COBIT).[4] Since it is focused on objectives rather than procedures, it is not seen as an alternative to ITIL, but the two frameworks are compatible (Năstase et al. 2009), and the objectives of COBIT can be mapped onto the corresponding ITIL activities (Kozina 2009). One of the COBIT objectives is to ensure continuous service (Sahibudin et al. 2008). Some of the standards in the ISO 27000 series (ISO 2013), also cover the topic of IT continuity, but mostly in the perspective of information security.

However, long-term viability corresponds neither to reliability nor to IT continuity in business. Compared to reliability, viability is different in causes and effects. Reliability issues generally stem from faults in the technical infrastructure, and therefore the topic is more related to short-term problems, in the perspective of making a service continuously available, without interruptions or outages (Lyu 1996). Long-term viability, on the other hand, mostly concerns business or financial problems, and viability problems imply the complete cessation of the service (Mills 2009; Kandukuri et al. 2009; Khajeh-Hosseini et al. 2010). The approaches for short-term failures are not suited for addressing the final termination of the CSP, and similarly, the approaches to achieve long-term viability might be unfit to solve a few hours' downtime.

Similarly, the attention on IT continuity is generally focused on temporary outages, data losses or natural incidents. IT continuity approaches mostly aim at a continuity of business, so they are not applicable to a CSP that undergoes a complete cessation of its activities. To some extent, they might be applicable to the cloud customer, because from its point of view the cessation of the CSP can be partly assimilated to a disaster, requiring a disaster recovery plan. Previous research has made some steps in applying IT continuity approaches in case of changes to the IT infrastructure (Sauvé et al. 2008) or cloud migration (Aubert et al. 2002), in particular designing risk analysis models.

The main reference to the problem of CSP long-term viability resides in a popular Gartner analysis (Brodkin 2008) which highlights seven critical features that a business should evaluate when moving to cloud; among these is long-term viability. The report is considered a milestone concerning cloud risks. A different risk classification is provided in Van Hoboken et al. (2013), highlighting the problem of stored data in the event of bankruptcy.

Other works (Kauffman et al. 2014) include the vendor's financial stability in a survey of the potential factors that need to be taken into account before migrating to a CSP. Also Mills (2009) considers the various risks associated with the migration from an insourced infrastructure to a CSP, and the set of risks includes long-term viability, in the perspective of contractual issues that have to be addressed, but does not provide a significant insight into the subject. A slightly more detailed analysis is offered by Bowen (2011). This work introduces the distinction among several scenarios in which the CSP undergoes a soft shutdown (i.e., filing for bankruptcy), merges or is acquired, and abruptly ceases its business. Apart from a brief overview on software escrows (described in "Mixed approaches"), the book does not explore other solutions.

Perhaps the only work which concretely attempts to address the issue of a CSP going bankrupt or ceasing its business is (Louwers 2013). However, this work exclusively focuses on legal solutions, without discussing their technical implications. Moreover, it puts too much emphasis on the source code of services, essentially addressing SaaS environments only. Some of these solutions will be discussed in this paper.

The main works dealing with viability or topics close to it are summarized in Table 1, along with the main issues that differentiate them from the present work.

[4] http://www.isaca.org/cobit/pages/default.aspx.

Table 1 Overview of related work

Source	Focus	Differences from this work
Lyu (1996)	Reliability, short-term failures	Causes and consequences are different for a final shutdown
Sauvé et al. (2008)	IT continuity in infrastructure change	Not specifically designed for bankruptcy or a final shutdown
Aubert et al. (2002)	IT continuity in cloud migration	
Brodkin (2008), Van Hoboken et al. (2013), Kauffman et al. (2014), Mills (2009)	Problem statement and classification	No solutions proposed
Bowen (2011)	Problem statement	Just a slight consideration on software escrows
Louwers (2013)	Viability	Legal solutions only, mostly SaaS

A mathematical model of viability

The problem of long-term viability is intuitively obvious, meaning that the intrinsic risks in outsourcing assets exposes the outsourcer not only to its own business risks, but also to those of the CSP. However, viability lacks a formalization in a mathematical perspective, metrics that can be used to some extent to measure the viability of a CSP.

Long-term viability can become a safety-critical dimension for the assets of a business. As such, it could eventually be included in the scope of established methodologies and standards for risk assessment, such as the ISO 27000 series of standards[5] to assess information security matters (in a very broad meaning of the term). A mathematical model captures the main parameters that play a role in the measured dimension. It thus translates the intuition into a number. On the basis of such metrics, viability could be approximated in terms of the probability of encountering business continuity issues due to the CSP going bankrupt. A model that represents viability in terms of a probability (therefore on a scale from 0 to 1) is also easy to integrate with metrics representing other types of risk.

An analysis of long-term viability would also be useful for the purposes of selecting a CSP over another. In other words, metrics could serve as a parameter for cloud brokering (Liu et al. 2011; Pettey and van der Meulen 2009).

Due to the lack of a model and metrics to express long-term viability, we introduce a basic model which can be used to assess the impact of the various risk mitigators presented in "Addressing the problem". It is a simple model, based on a number of approximations, and could be subject to refinements in the future.

As a quick reference for the formulas introduced in the present section, Fig. 1 displays a possible visual description of the model, in a popular notation used to model the relations between non-functional requirements (NFRs) (Chung et al. 2000). The diagram shows the relationships between the various goals (the clouds) that contribute to determine the viability of the CSP. The relationships representing the contribution between goals are either alternative (OR relation, represented by the connections with double bars) or aggregated (AND relationships, connections with single bars). The octagonal nodes represent the problems that can contribute negatively to the goals. Elements shown as **bold** clouds or circle in the model are operationalizations, meaning that the goal or problem they denote is based on technical, implementable features. Figure 1 also shows which variables introduced in the rest of this section affect the goals and operationalizations in the NFR model.

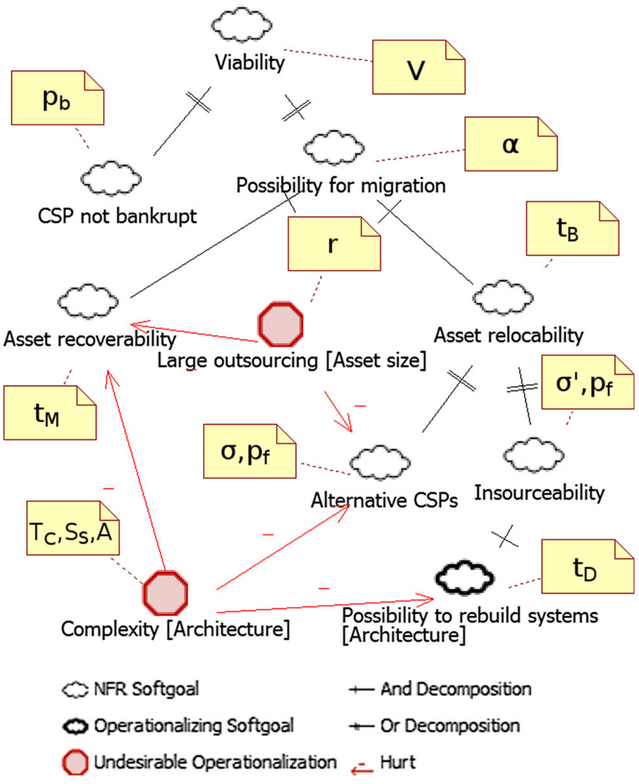

Fig. 1 Visual representation of the NFR model for viability

[5] http://www.27000.org/.

The viability of a CSP can be seen under two perspectives:

1. the CSP *won't* go bankrupt (or other similar disruptive event);
2. the CSP *will* go bankrupt, and its customer will be able to recover business continuity.

The viability of a system corresponds to the probability that the customer of a CSP can maintain business continuity regardless of the financial difficulties of the CSP. If p_b is the probability that the provider goes bankrupt, its viability V can be defined as follows:

$$V = (1 - p_b) + p_b\alpha = 1 + (\alpha - 1)p_b \tag{1}$$

where

$$0 \leqslant \alpha \leqslant 1. \tag{2}$$

To better explain Eq. 1, the probability for a CSP customer to run correctly partly depends on the CSP not ceasing business (a situation with a probability of $1 - p_b$); in this case, no business continuity problems will befall its customers. In the opposite event that the CSP does indeed fail (with probability p_b), continuity problems might or might not occur, depending on a parameter α, which measures the ease of recovery of the assets and migration to in-house hosting or to a different CSP.

p_b is a complex business index. Previous research (Scott 1981; Dichev 1998; Hillegeist et al. 2002) has defined several mathematical models for measuring the probability that a business goes bankrupt, and developing on that topic is out of the scope of the present research.

The problem, at this point, is assessing α, which enters into consideration only when the provider ceases business or files for bankruptcy. In this case, the safest situation occurs when the following conditions apply:

1. the assets (applications and/or data, depending on how much has been outsourced) can be recovered promptly (this corresponds to the "Asset recoverability" goal in Fig. 1);
2. an alternative host is promptly available (related to the "Asset relocability" goal in Fig. 1), either in-house ("Insourceability" and "Possibility to rebuild systems" goals) or another CSP ("Alternative CSPs" goal);
3. the assets do not need any elaboration to be migrated (also related to the "Asset relocability" goal in Fig. 1).

If all conditions are met, then $\alpha = 1$ and $V = 1$. In other words, if a CSP goes out of business, its customers won't have any problems because they can recover their assets promptly and migrate them to an alternative CSP without the need to convert or rewrite anything.

The three directions under which the CSP must be assessed, then, are the following:

1. the amount of recoverable assets r;
2. its fungibility (interchangeability with other CSPs) p_f, representing the probability of finding a suitable alternative, i.e., a CSP that offers a similar service, or can however allow the customer to perform the same operations;
3. the degree of simplicity of the CSP (whether it uses an easily manageable Application Programming Interface (API) or a complex proprietary structure) σ.

The importance of these three factors is not the same. The first one is way more relevant than the others: if the cloud customer can't recover the assets, its previous business is lost. Also, p_f has a greater importance than σ, since if no alternative infrastructure exists then a new one will need to be built, and the adoption of standards by the bankrupt CSP will provide marginal benefits. On the other hand, if an alternative CSP is available, a simple and standardized structure for the bankrupt CSP will allow a faster migration of the assets.

The amount of recoverable assets r is a variable whose value changes from one customer to another. It can be expressed as a number ranging from 0 to 1. The higher it is, the more assets the customer is able to recover, with 0 being a total loss and 1 being a total recovery. r will depend on the actual size of the outsourced assets S_A. As an approximation, let it be assumed that the asset recovery follows a linear function, with ρ expressing the recovery rate, S_A the size of the assets, t_0 the setup time, before which no recovery can occur, and t_M the maximum available time, after which the CSP will shut down, preventing any further recovery. Both t_0 and ρ depend on the degree of standardization adopted by the CSP, so using common and easily-manageable APIs will reduce the initial slack and increase the rate at which the assets can be recovered.

Given unlimited time, the time required to recover all the assets would be

$$t_S = t_0 + \frac{S_A}{\rho}. \tag{3}$$

The recovery function (which is normalized) would be

$$r(t) = \begin{cases} 0, & \text{if } t \leqslant t_0, \\ \frac{\rho}{S_A}(\min(t, t_S) - t_0), & \text{if } t > t_0 \end{cases} \tag{4}$$

which, substituting t_S, becomes

$$r(t) = \begin{cases} 0, & \text{if } t \leqslant t_0, \\ \frac{\rho}{S_A}\min(t - t_0, t_S - t_0) = & \\ = \min\left(\frac{\rho}{S_A}(t - t_0), 1\right), & \text{if } t > t_0, \end{cases} \tag{5}$$

and consequently the amount of recoverable assets is

$$r = r(t_M) = \begin{cases} 0, & \text{if } t_M \leqslant t_0, \\ \min\left(\frac{\rho}{S_A}(t_M - t_0), 1\right), & \text{if } t_M > t_0. \end{cases} \tag{6}$$

Assessing p_f is more complicated. In general, it will be possible to migrate a service from one CSP to another, unless very specific and targeted requirements are involved. However, the customer might have restrictions based on cost, accountability, security, and so on. The availability of alternative CSPs will mostly depend on the type and complexity of the service, on the size of the market sector, and the business requirements of the customer (for example, a CSP offering a similar service might charge more than the customer can afford). A very common service which is needed by many different business (e.g., a service offering remote storage or geographical maps) will have $p_f \lesssim 1$, whereas a very sectorial service might have $p_f \gtrsim 0$. The models introduced in existing research on cloud ranking (Kumar Garg et al. 2013) and brokering (Buyya et al. 2009; Tordssona et al. 2012) can be used to assess this parameter.

Finally, σ determines the ease of having the services up and running again on the new CSP, provided there is one available. Regardless of how this property is measured, in this model σ is assumed to be normalized, therefore defined in the range $0 \leqslant \sigma \leqslant 1$. This variable is hard to assess, depending strongly on the structure and complexity of the service and on the use of standard APIs and protocols. A very simple service might be migrated easily ($\sigma \lesssim 1$) even if it uses proprietary formats and protocols, whereas a complex service will have a high σ only if the APIs have a clear organization that follows common patterns. Also, the degree of outsourcing will strongly impact this parameter: it will be easier to migrate a SaaS service than a IaaS infrastructure. In general, this parameter can be expressed as a function $\sigma(T_c, S_s, A)$, where T_c denotes the type of cloud (SaaS, PaaS or IaaS), S_s represents the size and complexity of the outsourced service structure (from 0 for small and simple services, to arbitrarily large for complex ones), and A qualitatively denotes the degree of standardization of the cloud APIs. T_c cannot be considered as part of S_s, although the type of cloud will clearly impact on the size of the outsourced assets, because it also affects the infrastructural requirements and how the APIs are made. It should be noted that, once the assets have been recovered and a suitable alternative CSP found, the migration is a matter of time, so σ gives an indication of the time needed to set up the new CSP with the recovered assets.

In case an alternative CSP is not found, there is the need to develop a new infrastructure as well. In this case, the σ parameter will still be relevant because there will be the need for migration, but the company must also account for the effort to develop the new infrastructure, which will again depend on the type of cloud that was previously being used T_c and the size of the service structure S_s. Therefore the ease to set up the services will be $\sigma' = \sigma\frac{t_B}{t_B+t_D}$, where t_B denotes the time needed to stage the assets onto the new infrastructure, and t_D the time needed to develop said infrastructure. For the sake of simplicity, t_D is considered a fixed value given the type of cloud and service, although clearly it will depend on the development capabilities of the customer. Several techniques exist for estimating the development time of a software, some of which are surveyed in Boehm et al. (2000).

The efficiency of the reaction to the CSP going out of business can then be expressed as

$$\alpha = rp_f\sigma + r(1-p_f)\sigma\frac{t_B}{t_B+t_D} = r\sigma\frac{t_B+p_ft_D}{t_B+t_D}. \tag{7}$$

The parameters can be increased in many ways. For example, the availability of an internal backup will extend t_M indefinitely and set a very high ρ, at least for those assets that have been backed up. An insurance to cover management expenses of the CSP can increase t_M, while using standard formats can increase ρ and reduce t_0. A cloud or mirroring escrow would set the probability of having alternative services $p_f = 1$, because it would guarantee that there is at least another service identical to the one that has become unavailable.

Some of these values depend on the customer, such as the size of the outsourced assets and the service structure or the type of cloud (degree of outsourcing). Others, in particular the size of the market sector and the availability of alternative CSPs, as well as the time to develop an infrastructure, are objective parameters on which neither the customer nor the CSP can have any influence. The rest are based on choices of the CSP, and can be used as brokering factors. Table 2 summarizes the parameters and which stakeholder they rest upon. **Bold** entries are those that whose impact can be assessed only in a qualitative perspective rather than measured.

In short, the actual viability of the CSP, i.e., the minimization of the effects of its financial problems on the business of its customers, is a function depending on several parameters, which are summed up in the following expression:

$$V = f(p_b, \rho, t_M, t_0, A), \tag{8}$$

with V growing with all the parameters except t_0, which should be as low as possible.

Addressing the problem

This section analyzes the different methods and approaches that can be used to anticipate the needs for building trustworthy cloud platforms. The mitigators that can be used

Table 2 Summary of relevant parameters

Parameter	Meaning	Depends on	Type	Range
p_b	Probability of bankruptcy	CSP	Probability	[0, 1]
ρ	Assets recovery rate	CSP	Bytes/day	$(0, +\infty)$
S_A	Size of outsourced assets	Customer	Bytes	$(0, +\infty)$
t_M	Maximum time before shutdown	CSP	Days	$(0, +\infty)$
t_0	Asset recovery initial onset	CSP	Days	$(0, +\infty)$
p_f	Availability of alternative CSPs	No one	Probability	[0, 1]
$\mathbf{T_c}$	Type of cloud	Customer	Qualitative	{SaaS, PaaS, IaaS}
S_S	Size of the service structure	Customer	Bytes	$(0, +\infty)$
$\mathbf{A}$	Degree of standardization	CSP	Qualitative	
t_B	Asset staging time	No one	Days	$(0, +\infty)$
t_D	Infrastructure development time	No one	Days	$(0, +\infty)$

to tackle the problem of long-term viability can be classified under two different perspectives:

- by *actor*: depending on who can enact each mitigator, whether the CSP, the customer, or the two together;
- by *nature*: some mitigators are based on technical instruments, some on legal instruments, some on a combination of the two.

In what follows, these approaches are presented according to the classification by nature, taking into account an information technology perspective, a purely legal perspective, and finally an approach that combines both the IT and law perspectives. Some of the solutions analyzed in the following sections were selected on the basis of widespread business practices (e.g., backup or insurance), including some used by CSPs in Luxembourg (of which an example is given in "Considerations on the proposed solutions". Others are consolidated technical approaches, often implemented by existing software tools (e.g., Service Level Agreement (SLA)). Finally, others are based on a research survey, even though their interest is more theoretical than practical (e.g., shared rights over the assets). For each of the mitigators in this section, its effect on the viability parameters introduced in "A mathematical model of viability" is described.

Discussion of possible IT solutions

The risk of a CSP failure can be mitigated by means of some preemptive technical solutions that cannot prevent the CSP from ceasing its activity, but can help avoid the consequences of such an event, especially that of preventing the cloud customer to carry on its business. Although approaches based on redundancy and standardization will involve some additional costs, they might offer a viable trade-off between expenses and benefits.

Backing up

As obvious as it may seem, the most immediate solution to protect against the sudden disappearance of a CSP is to regularly back up one's software and data. By means of an in-house duplication of all the assets that are outsourced to the cloud, an enterprize combines the advantages of both solutions: on one side, it does not require to maintain the high-performance, 24/7-reliable infrastructures to deliver its service to its customers; on the other side, it constantly has an up-to-date version of its assets, ready for migration to another CSP (or for insourcing).

Data backup is also advisable if the service delivery is in-house. In particular, it has been suggested (McKendrick 2013) to use a cloud backup when delivering in-house, and to use an in-house backup when the services and data are outsourced.

According to reliability theory (Gnedenko et al. 1969), redundancy can take several shapes:

- **hot standby**, where the backup system is running and constantly up to date with the primary one. Applied to cloud computing, this means a complete redundancy of the systems, i.e., every asset belonging to the customer (only data in a SaaS model; data and source code of the services in a PaaS model; data, source code and infrastructure in IaaS) is constantly replicated;
- **cold standby**, where the backup system exists but needs to be set up, and the data between the two systems are aligned only occasionally. In cloud computing, this would mean that the enterprize has a copy of the applications and data that have been outsourced to the CSP, but since they are not running and maintained, both the applications and the data might be seriously out of date;

- **warm standby**, an intermediate model where data are aligned periodically, so that the secondary system is not up to date but it does not need to be set up from scratch. In cloud computing, this means that at certain intervals in time the cloud customer performs a fresh backup of the outsourced assets, which will almost never be up to date with the CSP, but nonetheless still reasonably recent.

When dealing with cloud computing, a hot standby in-house backup is unlikely, because it would imply that the services are run internally an not only on the CSP, and this is in contrast with the very logic behind the cloud paradigm. Despite that, the application of such a solution would eliminate all viability risks, as the sudden cessation of business by the CSP would not cause the customer to lose any data or to be unable to deliver its services, but merely reduce or lose its redundancy.

Conversely, a hot standby external backup means that the same service runs concurrently on two different CSPs, and the backup one is ready to take over in case the one being delivered fails. This situation corresponds to mirroring solutions, which are described in "Mixed approaches". The rest of this section therefore deals with the cold or warm standby models, where the assets are replicated but not aligned with the CSP.

On the down side, backing up the software and the data has its costs, but these will reasonably be smaller than those required for in-house operation. Also, this approach does not solve every problem, because the CSP might use technologies, protocols and APIs that are not compatible with an in-house operation, or for migration to a different CSP. So, backed-up data might still require a lot of work to restore full business operations. That is, unless the CSPs are based on some standard technology or structure (as described in the next paragraph).

Backups can be performed directly by the customer without the assistance of the CSP. Backups of outsourced services can be either in-house or external (to a different CSP). The effect on the mathematical model introduced in "A mathematical model of viability" is different.

An internal backup reduces S_A. If all the assets were backed up and aligned with those on the CSP (such as in a hot standby or right after a replication in a warm standby), then $S_A = 0$, $t_0 = 0$, and $t_S = 0$. However, it does not affect any of the other parameters.

An external backup won't affect S_A, because the data will still need to be recovered from the backup. However, it can still significantly reduce t_0 (because the customer needs only access the backup service) and increase ρ (assuming that access to the backup is simple). It also increases t_M indefinitely, because the availability of the backed-up assets does not have any correlation with the bankruptcy of the other CSP.

The main drawback with backups is that they need to be updated. However, this problem can be easily overcome using differential backups. Also, recovering the assets is not enough to guarantee business continuity.

Cloud platforms portability and standardization

In the last years, several PaaS platforms have appeared as an initiative to create and manage scalable cloud-based software. When a PaaS provider goes out of business, cloud customers should be able to migrate their application to others sites.

A cloud platform should provide standard connectors, so that applications can be moved to another platform adopting the same standards. In the last few years, several PaaS tools have been proposed by different PaaS market players. Some of these tools are based on standard programming languages, however they still suffer from the lack of standard APIs in terms of interfaces and applications connectors, thus exposing customers to vendor lock-in risks. With the lack of interoperability between platforms, customers' workload migration from one cloud provider to a new one becomes a tedious task, since the customer has to adapt its migrated applications to the proprietary components of the new provider's platform in order to migrate the software and the data. To ensure high interoperability and portability between PaaS, standardization initiatives have to be developed to provide standard application dependencies mechanisms and common interfaces to ease the transfer of applications between heterogeneous PaaS platforms. Previous research (Gebregiorgis and Altmann 2015) provides an assessment of the cost benefits of a high degree of interoperability.

Interoperability measures (Kuyoro et al. 2011) and federated clouds (Rochwerger et al. 2009) are potential mitigators to avoid "vendor locking", as they could ease the transition to a new cloud in case the client enterprize's chosen CSP goes out of business. With federated clouds, two or more CSPs share an agreement according to which the underloaded clouds offer resources to overloaded ones.

Standards normally have a body devoted to their specification and development. While the adoption of some standard tools and interoperability measures (e.g., the XML language as a file format) does not have any formal requirements, there are many standards that, due to their complexity, need auditing and certification by the issuing organization or by a specifically-endorsed enterprize or institution. Although auditing can entail high costs on the CSP, the certification by an authoritative source can increase the customer's trust (Klass 2006; Kaufman 2009).

Unfortunately, there are no such things as standard cloud implementations. Despite some de facto general rules for building a CSP, and some efforts at defining a standard (C-SIG SLA 2014), the results so far are still limited. However, it is still possible to use some interoperability tool to migrate the assets from one CSP to another. It is also possible to singularly adopt standard protocols and formats for the various components of the CSP, which does not achieve total interoperability but still provides some degree of facilitation.

A recent initiative to enact interoperability between platforms and to ease the migration of applications is illustrated by the containers model, whose basic idea is to have an isolated packaging of an application embedded with its dependencies. Using this approach, it becomes easier to move the overall application when its hosting environment encounters some problems. "Containerization" (Kharb 2016) has been promoted by Linux Dockers,[6] which adds an additional layer of abstraction to virtual machines, allowing to have isolated features within the same Linux instance.

Some authors (Machado et al. 2009) have presented some proposed standard interfaces such as the Distributed Management Task Force (DMTF)'s Open Cloud Standards Incubator (OCSI) for resources management, the Open Cloud Computing Interface Working Group (OCCI-WG) for IaaS specification, and Cloud Data Management Interface (CDMI) for the manipulation of data elements. Others (Harsh et al. 2012) have discussed some standardization challenges related to credentials and network. Another initiative to achieve cloud interoperability has been presented in the Levels of Information Systems Interoperability (LISI) Maturity Model (Dowell et al. 2011) published by the Department of Defense (DoD). LISI explores four levels of system interoperability related to procedures, applications, infrastructure, and data.

The various approaches related to cloud standardization mentioned in this section are briefly summarized in Table 3.

By defining standards for the alignment of cloud development tools and platforms, the migration from one cloud to another becomes more flexible. Standardization can have a huge impact on the costs of migration, both in money and in time, when services and data need to be transferred from one cloud provider to another one. This is clearly reflected in the mathematical model for viability.

Implementing the services according to standards facilitates recovery of the assets, and in this sense it reduces t_0 and increases ρ. It also affects the degree of standardization A. Additionally, in case no alternative CSP is available, insourcing the development of the services will be easier, thus lowering t_D.

Table 3 Cloud standardization approaches

Source	Focus
(Kuyoro et al. 2011)	Cloud interoperability
(Rochwerger et al. 2009)	Federated clouds
(C-SIG SLA 2014)	Cloud standard
(Kharb 2016)	Containerization
(Machado et al. 2009)	Standard interfaces
(Harsh et al. 2012)	Credentials and network
(Dowell et al. 2011)	Cloud interoperability levels

Service choreographies

Cloud-based services are intricate by design as they are composed of a number of software services that maintain several features which depend on different stakeholders. To maintain these dynamic interactions between stakeholders, an abstraction layer that models the different flows of communications between cloud-based software entities is needed.

Choreographies (Peltz 2003) have been widely used in the context of web services to specify how services interact with each other. They commonly rely on the orchestration of every entity that can be involved in this choreography. An orchestration specifies from a central view the behavior of collaborating parties and the flow of message exchanges between the different entities. A choreography gives more visibility about the interactions between the different collaborators, since it enables each collaborator in the choreography to observe all message exchanges involved in the different interactions, and thus to have a global view about the system.

Despite being born in the SOA paradigm, choreographies can be revisited in the cloud context to provide a high-level view of the interactions between the different stakeholders involved in maintaining cloud services. Maintaining the status of the different collaborations between the different entities involved in the deployment, delivery and usage of a service eases the tracking of the partners taking part in the management of cloud-based applications when the provider goes out of business.

In the context of this paper, choreographies can be seen under two perspectives. The first is when one of the actors in a choreography is the CSP that could potentially file for bankruptcy. In this situation, the choreography has the same role of a general cloud customer. The second, which is examined in this section, is when the CSP is designed as a choreography (and in turn relying on external services). If it undergoes financial problems, the choreography design can indirectly mitigate cascading problems to its customers.

[6] https://www.docker.com/.

As a choreography is simply a model of the interactions between services, it does not per se favor the continuity of business of the service. However, the viability benefits of using a choreography in a CSP can be found in the modularity of the model and in the clear expression of the requirements, especially when used in conjunction with SLAs (see "Mixed approaches"). For example, in case of financial difficulties, the choreography manager can renegotiate the requirements, in order to enlist cheaper actors and reduce the costs. The modularity of the choreography might allow a progressive shutdown of the services, thus keeping available only those that allow the cloud customers to recover their assets.

Concerning the viability model described in "A mathematical model of viability", the service choreography can affect the parameters in several ways. First off, the fact that actors are generally replaceable reduces the overall probability of bankruptcy p_b, thanks to the possibility of enlisting actors that have a smaller impact on the overall costs. In case of a progressive shutdown in bad financial circumstances, the time before the final shutdown t_M is increased. Finally, the need to have a modular and replaceable structure would require the use of popular standards, consequently raising A.

Using a choreography, however, also has some drawbacks. The complex and distributed operations carried out in a service structured as a choreography might imply a low availability of alternative services p_f. Worse than that, the need to coordinate a number of actors, some of which might become hardly available, could increase the time t_0 it takes before the actual operations to recover the assets can begin.

Legal perspective and solutions

Contractual agreements can help the stakeholders of a cloud environment avoid the worries of the CSP's bankruptcy or cessation of business. If the problem is not addressed beforehand, the customer might incur into serious trouble, because ex post remedies can be quite ineffective, as explained in the following section.

Consequences of a CSP shutdown

Once the customer's assets fall into the bankrupt estate, recovering them might prove uncertain, expensive, and long enough to seriously damage the customer's business. In addition, the liability of a CSP for not being able to provide its service in continuity is rather blurred (whereas provisions exist concerning liability for IP violations, failure in data protection, or negligence in security (Weber and Staiger 2014)); to make things worse, CSPs normally operate by means of non-negotiable contracts that make frequent use of liability waivers. Customers are not guaranteed in case of a prolonged inactivity of the CSP which cascades upon them.

When filing for bankruptcy, the management of the company is transferred from the CEO or board to a trustee, under the surveillance of a tribunal. Licensors or third-party providers cannot take action against the bankrupt CSP without a judicial authorization, so customers have a few days to react. However, this protection can't be relied upon if human intervention is needed, because employees cannot be guaranteed to provide customer support (Caplan 2010). This problem would also impact differently depending on the size of the CSP: the customers of a major provider would have an easier opportunity to recover their data, than those of a small enterprize (Anthony 2012).

If the CSP simply ceases its activity without filing for bankruptcy, the situation can be much more complicated. The customer will simply have the normal legal remedies such as filing a suit for a breach of contract, but this does not aid in recovering the assets, especially when manual intervention would be required on the CSP's side. Contractual remedies are the ordinary situation that occurs when the customer cannot do anything but file against the CSP for damages, so in this situation the viability model is not affected in any way.

It should be noted, however, that a CSP suffering bad financial conditions might not file for bankruptcy or cease its business, but rather be acquired or merged by a financially stronger enterprize. Under these conditions, the customer is relatively safe, because the acquiring enterprize will inherit the existing contracts of the CSP, giving the customer time to renegotiate the terms or migrate to other providers.

Insurance

In the perspective of the customer not losing its assets or its business continuity in case of the CSP shutting down, the CSP can play a significant role by relying on insurances, in several different forms.

An insurance negotiated by the CSP with a third-party insurance company in favor of the customer can reduce the customer's risks, covering the loss of revenues suffered in case of the CSP's cessation of business and giving the customer time to set up the services again. Still, an insurance would per se not aid the customer in recovering the assets. Also, a simpler form of private insurance between customer and CSP (without the insurance company as a third party) might imply the CSP requesting a few months' payment in advance from the customer; this fee would have to be bound to ensuring the temporary continuation of the service (including manual operation) even in case of bankruptcy. This solution, combined with some agreement which would grant the customer access to its assets, would allow the customer to recover the data and prepare for the migration or insourcing of the services, without any loss of data or interruption of business continuity.

A CSPs might use part of its revenues to establish a special guarantee fund, devoted either to cover the damage from loss of revenues of its customers, or to keep its own services up and running for several months and give its customers the opportunity to take appropriate measures. In this sense, a SaaS-guarantee fund has been suggested (van de Zande and Jansen 2011), but it can be applied to other paradigms as well. The fund could also be set up in a collaboration between several CSPs, but this would have the drawback that when an individual CSP in the consortium disappears, the others might feel like they are paying for its expenses, and this would be an inhibitor against setting up such a fund. At any rate, the authors acknowledge that no such fund exists at the moment.

While all these forms of insurance can provide benefits to the customer (either by covering damages or by ensuring business continuity), they will all increase the costs of the CSP. In case of inter-party insurances, the insurance is paid directly by the customer, while third-party insurances or guarantee funds would place some costs on the CSP, which would in turn charge them on the customer in some way. The availability of an insurance, and its details and costs, should be the subject of careful evaluation by an enterprize when selecting a CSP.

Insurance against bankruptcy or cessation of business may or may not affect the viability model from "A mathematical model of viability" in different ways, depending on the type and purpose of the insurance:

- **third-party insurance**, i.e., an insurance between the CSP and an external insurance company, can be contracted to guarantee either the liability of the CSP, or an extended period of operation in case it goes bankrupt. While the former insurance does not provide any benefit to the customer in terms of keeping its its business running (because it only pays the damages caused to the customer), the latter can have a significant effect on the time window t_M during which a customer can recover its assets;
- **CSP guarantee funds**, if they were existent, would affect the model exactly as a third-party insurance, and just as much they could be contracted to guarantee either liability or an extended operational period. The only difference is that, being based on a collective agreement between a number of CSPs, the individual CSP would have less freedom in negotiating the terms and the limitations for the guarantee;
- **inter-party insurance** (meaning an up-front payment by the customer in favor of the CSP to cover a few months of the CSP's operation), on the other hand, can have the sole purpose of extending the operations of the CSP (there is no point in the customer paying an insurance to the CSP to pay damages to itself). While there is no doubt that the effect would be to increase t_M, in this case there are some doubts as to the effectiveness of the insurance, because there is a risk that the CSP uses the extra cash for other purposes.

Policy support

In 2013, the Luxembourgish Parliament tried to address the problem of a CSP going bankrupt. The Parliament introduced provisions[7] that allow the owners of intangible non-fungible goods held by a bankrupt to reclaim those data at their own expenses, provided they are separable from all other assets. The provision, according to the parliamentary discussion, was introduced specifically to address the issue of a CSP going bankrupt, although its application is not limited to cloud environments.

The Luxembourgish approach is definitely a step toward addressing the problem of CSP viability. However, it only allows to recover the assets, but does not help in relocating to some other CSP or insourcing, unless the services have been designed with portability in mind. Moreover, it does not work well with SaaS platforms, because the customer will not be able to recover the software, which is a shared platform and not a proprietary asset of the customer.

Such a capability to recover one's data in case of a CSP going bankrupt has a significant impact on the viability model. In particular, it makes the variable t_M totally irrelevant, since the bankrupt estate will provide the means to recover the assets, independently of the shutdown.

On the other hand, the complex legal procedures in case of bankruptcy can take some time before the right of the customer to recover the assets is recognized and the recovery operations can begin, thus increasing t_0, in contrast with the need for an expedited reenactment of business operations.

Limited rights over the assets

Some imaginative solutions have been suggested in Louwers (2013). However, those solutions place too much emphasis on the source code, and appear to address the legal issues but not the technical implications. A split copyright between the CSP and the customer consists in transferring part of software copyrights to the user, whereas a joint copyright transfers a share of the ownership; in both cases, the copyright of the whole source code does not fall into the bankrupt estate. The software code could be assigned in usufruct to the customer, or IP rights transferred to a

[7] Luxembourgish Code of commerce, Article 567.

foundation or association of customers. Notwithstanding the inefficiency of these ideas, which would hinder the CSP's ability to evolve or improve the services, they place too much emphasis on the source code. They may allow its partial recovery, but not ensure the continuity of the services, as the customer might not have the technical resources to use those assets.

Data ownership, on the other hand, can be detailed in the contract, so that the customer's data do not fall into the bankrupt estate. Again, this does not help if the CSP simply shuts down the operations, as the customer will not be able to recover the data even if it has the right to do so. Therefore, the contract should either guarantee some maintenance even after the business closure, at least for a short period, or provide the customer with some means of accessing the data without the intervention of the CSP personnel.

Despite the practical difficulties that would arise from the establishment of a split or joint copyright, from a merely theoretical perspective such solutions would undoubtedly provide some benefits in terms of business continuity. Considering that the customer would have access to the technologies of the CSP, it would be eased in the recovery of the assets, both increasing the recovery rate ρ and reducing the initial onset t_0. Secondly, the privileged insight into the CSP's internal structure would make it easier to rebuild an architecture which provides a similar service, thus reducing t_D.

However, the economic problems behind a split or joint copyright probably overcome their benefits, effectively making them undesirable solutions.

Mixed approaches

Some mitigators of viability problems consist of legal solutions that can have a direct technical implementation. In particular, the efficiency of these solutions is that they can be implemented, and therefore they operate at a technical level, but their legal nature gives them a binding strength that can be enforced in a court if they are breached. These solutions need to be examined taking both perspectives into account.

Service Level Agreements

A Service Level Agreement (SLA) is a means of expressing the QoS requirements of a service. In its essence, a SLA is "an agreement between the service provider and its customers quantifying the Minimum acceptable service to the customer" (capital not added) (Hiles 2000), or a contract expressed in a semi-formal language.[8] Several SLA languages have been defined, e.g. Andrieux et al. (2007), Ludwig et al. (2003).

SLAs are generally designed to accommodate languages for expressing requirements, but not all requirements are fit to be modeled using formal languages. However, the formal part is well suited for automated processing, including monitoring the system to detect SLA violations (Sahai et al. 2002).

Long-term viability would clearly be one requirement that is not fit for formal specification or monitoring. Nonetheless it could be addressed in a SLA, for example by addressing issues such as disaster recovery, data portability and an exit strategy. In particular, with respect to data portability, the parties might benefit from clauses specifying:

- the use of some interface which would ease migration to other providers, to protect the customer in case of data loss or business failure of the CSP;
- strict conditions and terms under which the customer is entitled to enact the migration, to avoid damages to the CSP due to abuse on part of the customer.

Recently, the European Cloud Select Industry Group on Service Level Agreements (C-SIG SLA)[9] has released a set of guidelines toward a standardization of SLAs. Among the service level objectives that should be covered by the SLA, the guidelines suggest including a termination process (C-SIG SLA 2014), with steps to enable the customer to retrieve their data. The problem with this clause, however, is that it might be difficult to apply in a short time in the case of a sudden bankruptcy of the CSP.

The effect of a SLA on the viability parameters can be highly variable, depending on the exact content of the agreement. However, it can be used to guarantee a minimum forewarning time before the final shutdown, thus increasing t_M; to ensure the assistance of the CSP in recovering the data, reducing t_0; and also to require the use of certain technologies and formats, thus increasing the degree of standardization A.

The drawback is that there is no guarantee that the CSP will fulfill the terms of the agreement, especially in case of serious financial problems. In that case, there will be additional charges on the CSP for breach of contract, but this does not help business continuity at all.

Software escrows and implementations for CSPs

Software escrows (Pappous 1985) are a part of intellectual property licenses that define necessary terms to maintain a product, including software code. A software escrow is based on a relationship between a licensor and licensee,

[8] Other definitions of SLAs exist. According to Wieder et al. (2011), SLAs are "a common way to formally specify the exact conditions (both functional and non-functional) under which services are or should be delivered".

[9] Information about the group can be found at https://ec.europa.eu/digital-agenda/en/cloud-select-industry-group-service–level-agreements.

set up through an escrow agent (a trusted third party). The agent delivers the software to the licensee and manages it according to an agreement. To ensure continuity of the service, software management is delegated to the licensee in case the licensor is not able to guarantee the software service anymore, due to bankruptcy (Conley and Bryan 1985) or disaster. The licensee will then provide the service continuity. In the classification provided by this paper, a software escrow is categorized as a mixed solution because, in addition to being based on a contractual agreement between the licensor and the licensee, the latter must have a technical infrastructure fit for managing the licensor's software services.

Software escrow techniques need to be revisited to be used in cloud-based services. As the hosting of data and/or software is outsourced, the availability of services is not guaranteed by securing a traditional escrow (which only covers source code), but lies in the hands of the CSP, which is a key entity to ensure continuity of the services.

Software escrow solutions raise issue concerning software synchronization and the viability of software escrow agents, which is one of the major factors that reduces the assurance of service continuity in a software escrow service. Additionally, software escrows are mainly able to address a SaaS services model, whereas solutions for IaaS and PaaS providers require mirroring the infrastructure. Specifically, mirroring a service delivered on a PaaS cloud would require mirroring the libraries, databases, and specific technologies used by the PaaS cloud, which normally are proprietary assets of the CSP and not available externally. Additionally, a IaaS cloud also comprises the hardware resources, delivered by the CSP in the form of virtual machines that might again use technology not available to customers, thus making them difficult to replicate.

When escrows are in force, the viability metrics from "A mathematical model of viability" are highly affected. In fact, the software assets are readily available to the customer, even in case of a shutdown of the CSP. Therefore, parameters such as ρ (assets recovery rate), S_A (outsourced amount), t_M (time available to recover the assets) and t_0 (time required to initiate the recovery) are irrelevant. Of course, this does not affect the availability of alternative CSPs, nor the fact that the assets must be transferred to a different CSP or insourced, which still depends on the degree of standardization and the complexity of the architecture. Additionally, software escrows would only cover the assets belonging to the customer, and not the CSP infrastructure, making the benefits mainly relevant for SaaS services and largely ineffective for PaaS and IaaS delivery models.

In recent years, dedicated escrow solutions for cloud-based software applications offer a mirroring of the whole application execution environment (van de Zande and Jansen 2011). To maintain the cloud-based software available when the provider goes out of service, the hardware resources, application platform, dependencies and on-line status execution are continuously mirrored. Companies like Iron Mountain and EscrowTech have developed some dedicated services for SaaS systems, offering business continuity options based on backup sites mirroring the services and the data.

A mirroring escrow combines the strengths of backup solutions and software escrows. Like a backup, it maintains an updated copy of all the assets of the CSP; like a software escrow, it legally enables the licensee to deliver the service in case the licensor becomes unable to. This solution is ideal since it enables to restore the whole provider's context when it goes out of business, assuring the continuity of services. In case of a service failure, the escrow agreement enables these backup sites to ensure the continuity of services. However, it is quite expensive for both the cloud customer and the provider because each service that runs in the cloud environment must be cloned.

Recently, an escrow alliance has described the essential building blocks for a cloud escrow solution:[10]

- the contract that states relevant elements related to the source code of applications and their underlying platforms, the SLAs and the users' context;
- an online deposit as a backup environment to ensure service availability;
- verification mechanisms to control backup and restoration of processes.

Both cloud providers and users can benefit from software escrow techniques, but there is still a lot of effort to be done by cloud stakeholders in the domain of software escrows, particularly with respect to standard agreements.

Concerning the viability model, mirroring escrows provide even greater benefits than software escrows, since they replicate the underlying architecture and the user's context. This means that an alternative platform to run the business is already available, although possibly at higher costs (the escrow agent would need to employ more resources to deliver the service than to simply store a duplicate of the CSP infrastructure and the customer's context). In other words, the customer would still need to find an alternative CSP or insource the infrastructure, but business continuity is guaranteed, at least in the short term.

Summary

A brief summary of the proposed mitigators is shown in Table 4, also emphasizing how they fit into the classification introduced at the beginning of this section.

[10] http://www.escrowalliance.nl/en/escrow-solutions/cloud-escrow-solutions/.

Table 4 Summary of solutions for long-term viability

<table>
<tr><th rowspan="2">Mitigator</th><th rowspan="2">Applicable delivery models</th><th colspan="2">Actor</th><th colspan="2">Nature</th><th rowspan="2">Variables</th><th rowspan="2">Notes</th><th rowspan="2">Economic impact</th></tr>
<tr><th>CSP</th><th>Customer</th><th>Legal</th><th>Technical</th></tr>
<tr><td>Back-ups</td><td>All</td><td></td><td>✓</td><td></td><td>✓</td><td>S_A, t_0, t_S, t_M, ρ</td><td>Some in-house infrastructure is required, but less expensive as the one needed to deliver the service. Ready duplicate of software and data for fast redeployment</td><td>Costs depend on assets' size and are long-running. Benefits only for recovery</td></tr>
<tr><td>Standards and interoperability</td><td>All</td><td>✓</td><td></td><td></td><td>✓</td><td>ρ, A, t_D</td><td>Standards can offer a widespread technology for the services, thus allowing to migrate to other services that adopt similar standards</td><td>Cost depends on time of standard adoption, plus auditing costs based on size. Benefits in for customers in saving in case of migration</td></tr>
<tr><td>Service choreographies</td><td>All</td><td>✓</td><td></td><td></td><td>✓</td><td>p_f, t_0</td><td>Standard interfaces between services make it easier to replace an actor in case of a cessation of business. A governing board can also remove an actor if it doesn't provide enough guarantees</td><td>Design and governance costs. Benefits in modularity of the service</td></tr>
<tr><td>Legal remedies</td><td>All</td><td></td><td>✓</td><td>✓</td><td></td><td></td><td>It might be hard to obtain the fulfillment of the CSP's obligations and damage restoration once it ceases business, and damage restoration does not help the customer reenact its business in a short time</td><td>No costs on benefits before bankruptcy. Then costs (legal and losses due to data and operativity loss) only on customers. No benefits</td></tr>
<tr><td>Third-party insurance</td><td rowspan="3">All</td><td>✓</td><td></td><td rowspan="3">✓</td><td rowspan="3"></td><td rowspan="3">t_M</td><td rowspan="3">Should be combined with a contractual agreement to recover the customer's own assets, without the CSP's cooperation</td><td rowspan="3">Cost increase with the size of the insured assets and potential economic consequences, but can be charged on customers. Benefits in the potentially integral compensation of losses</td></tr>
<tr><td>Inter-party insurance</td><td colspan="2">✓</td></tr>
<tr><td>CSP guarantee fund</td><td>✓</td><td></td></tr>
<tr><td>Recovery of own data</td><td>IaaS, partly PaaS</td><td>✓</td><td></td><td></td><td>✓</td><td>t_M</td><td>Only applicable in Luxembourg so far</td><td>No costs involved. Benefits in a faster asset recovery</td></tr>
<tr><td>Split or joint copyright</td><td>SaaS</td><td>✓</td><td></td><td colspan="2">✓</td><td>ρ, t_0, t_D</td><td>Problems in splitting copyright over multiple customers and about the ownership of the data. Only allows the recovery of the source code which is not the key feature of modern cloud environments</td><td>Costs might scale rapidly if large technological assets need to be modified. Limited benefits in court proceedings, as customers partially own the assets</td></tr>
<tr><td>Service Level Agreement</td><td>All</td><td colspan="2">✓</td><td colspan="2">✓</td><td>t_0, t_M, A</td><td>A SLA can define both functional and non-functional requirements, but metrics are required for enforcement. Such metrics might help assess the CSP's financial stability</td><td>May entail costs for violations. Benefits in the use of standard components, and for customers in case of migration</td></tr>
<tr><td>Software escrow</td><td rowspan="2">All</td><td colspan="2">✓</td><td rowspan="2" colspan="2">✓</td><td rowspan="2">ρ, S_A, t_0, t_M</td><td rowspan="2">The agreement with the original CSP must address the transmission of data to the escrow</td><td rowspan="2">Costs scale rapidly with infrastructure size. Benefits for customers as no assets or operativity is lost</td></tr>
<tr><td>Mirroring escrow</td><td>✓</td><td></td></tr>
</table>

Considerations on the proposed solutions

Given the possible solutions that can help increase the CSP's viability, thus offering it more credibility in the eye of its (potential) customers, the main question that needs to be answered is what solutions a CSP should adopt, and (from an opposite perspective) what features an enterprize should look at when selecting a CSP, to have a guarantee of long-term viability. Some preliminary considerations, including a qualitative evaluation of the costs and benefits involved, will follow, focused specifically on the event of the CSP ceasing business, but a detailed cost analysis of the technical and legal solutions is out of the scope of this paper and will be reserved for a future work. A summary of these considerations can be found also in the "Notes" and "Economic impact" columns of Table 4.

Backing up the assets is a measure that resides solely on the customer, and its costs can increase significantly with the assets' size. Additionally, there will be an initial cost for the backup infrastructure, plus a running cost, independently of the type of backup (internal or external). For an internal backup, the initial costs will concern the acquisition and setup of the storage facilities, and the running costs will involve their maintenance. For an external backup, the initial cost will be represented by the initial transfer of the assets, while the running cost will include the lease of the storage space (an assessment is provided in Vrable et al. 2009). Backed-up assets, on the other hand, do not provide any economic benefit until there is a failure and the backup needs to be recovered. In that case, a backup will save economic backfires due to asset loss or to delays in recovering the assets from a bankrupt CSP, but it still does not provide any benefit with respect to the need of finding or setting up a new suitable infrastructure to replace the bankrupt one.

Interoperability techniques and standards for cloud portability are easy and cheap improvements if adopted early on, while they become more complex and expensive for the CSP as the business size and the number of customer grows. In particular, the early-on adoption of a specific standard is almost inexpensive. As it is unlikely that a large CSP disappears in a short time, redesigning a non-interoperable platform to allow easy portability of the services might be a appealing solution than it would be for small and medium CSPs. Weitzel et al. (2006) proposes a cost analysis model for the adoption of standards, although designed for network protocols. Once a standard is in use, there might be additional costs for audits for standard compliance and certification, both internal to the enterprize and external from certification bodies. These recurring costs increase with the size of the audits, but this size only depends on the amount of services offered by the CSP and its infrastructure, while it is unaffected by the number of customers and the amount of their outsourced assets. On the other hand, the benefits of well-designed interoperability measures can greatly outweigh their costs, as pointed out by Tassey (2000), for example by allowing to use existing specifications and tools designed in conformity with the standard. In case of bankruptcy, assuming that the assets can be recovered and that suitable alternatives exist, customers would have an easier time in relocating their assets.

Adopting a service choreography model, which generally also relies on SLAs, can improve the quality of the single actor and of the overall service composition. Although a service choreography consists of a number of actors providing interconnected services, there is normally a governing board that manages the choreography as a whole. The governing board has the power to take decisions concerning the assignment of the roles, and a proactive attitude of the board can help anticipate financial problems of the actors, thus avoiding a cascading effect on the whole choreography. On the other hand, choreographies require complex planning, design and governance that can weigh somewhat on the costs, especially a large plethora of services are provided. Little literature seems to address an assessment of their costs and benefits, although some model to analyze the cost of a choreography role is provided by Pandey and Chaudhary (2008). In case of bankruptcy of the CSP, such formal models may provide some benefits in migrating to available alternatives. Additionally, a carefully-designed choreography can save significant expenses on the CSP's side, allowing to disconnect those roles that provide non-critical services or renegotiating the roles with cheaper actors.

From a legal standpoint, an insurance that guarantees that the services will keep running for some months after the CSP goes out of business would help medium and large providers, while for small ones it might have too high a cost against the benefits it offers. The insurance cost would be charged back on the customers, and this might be possible only in an economy of scale. Therefore, the benefits of insurance are more easily seen in large-scale CSPs, where otherwise the financial losses of a cessation of business could be worldwide devastating. Additionally, as an insurance only provides a financial support but no means to recover the assets that are stored on the CSP (in particular in the case of the CSP simply ceasing business), it should be coupled with some other solution that addresses that need.

The Luxembourgish law on data asset recovery suffers the limitation of being applicable only Luxembourg-based CSPs. It does not require any cost on the CSP's or the customer's side, since it is a legal provision. On the downside, its economic benefits are somewhat limited, as it simply allows a preferential lane to recover the outsourced assets that therefore don't fall into the bankrupt estate. Such a benefit would be easily outweighed by a backup or an escrow, which eliminate the need to recover the assets.

SLAs are a practical solution when standardization is not a viable option (e.g., because non-standard systems are already in place and a complete redesign would be too expensive), because they do not involve redesigning the services but only providing a guaranteed quality of service and adequate interfaces. They are fit for both small and large CSPs. To be effectively implemented, however, a SLA needs to be supported by adequate software tools and appropriate metrics to measure the quality requirements. Concerning costs and benefits, these instruments are similar to standards, although more flexible in that their requirements may be negotiated and are not established by an external regulator. However, SLAs are also legally-binding contracts, and in case of a breach the non-obliging partner may incur into penalties, even in the absence of damages. Cost models for SLAs takes into account the adoption of the SLA (Maurer et al. 2012) and the penalties in case of violation (Abrahao et al. 2006).

Finally, software escrows and mirroring need adequate planning to be set up, and in particular they must clearly define the terms of the transfer of the assets from the original CSP to the escrow. Such solutions provide huge benefits to viability, but their cost might be prohibitive. Small enterprizes might not be able to afford the costs of maintaining a duplicate of their services and data at an escrow provider, whereas large CSPs might have such a technological asset that doubling the resources might be a non-feasible task. Additional costs stem from the risk of failure or bankruptcy by the escrow service (Mezrich 2001). In short, escrow services seem to provide a positive benefit-cost ratio only for medium-sized CSPs.

Summary depending on CSP type and size

On the basis of the above considerations, it would be useful to sum up the analysis carried out so far, trying to identify what are the preferable solutions to increase the long-term

Table 5 Possible solutions depending on size and delivery model

<table>
<tr><th>Type
Size</th><th>IaaS</th><th>PaaS</th><th>SaaS</th></tr>
<tr><td>Small</td><td colspan="3">Backup
Standards
Inter-party insurance</td></tr>
<tr><td rowspan="3">Medium</td><td colspan="3">Backup
Standards</td></tr>
<tr><td>SLA
Choreography</td><td>SLA
Choreography</td><td></td></tr>
<tr><td colspan="3">Third-party insurance
Escrow</td></tr>
<tr><td rowspan="3">Large</td><td colspan="3">Standards</td></tr>
<tr><td>SLA
Choreography</td><td>SLA
Choreography</td><td></td></tr>
<tr><td colspan="3">Third-party insurance</td></tr>
</table>

viability of a CSP. The answer to this question depends on a huge number of factors and cannot be straightforward, but it is possible to highlight a few general guidelines.

The main variable to take into account is necessarily the size of the CSP, both in terms of the amount and variety of services provided, and of the number of customers. Additionally, the delivery model (IaaS, PaaS or SaaS) is relevant.

As shown in Table 5 and better explained *supra*, the only mitigator that a CSP is encouraged to adopt in any situation is to use standard tools and technologies (provided they are available for the desired purpose) instead of developing proprietary ones. SLAs and choreographies can be of little use in a small scale, where the complexity of the system is limited (both in the size and in the delivery model), but become more relevant as its size and complexity grow. Escrows might be an excessive solution for small businesses, as its cost might outweigh the benefits, and might entail overwhelming costs for large enterprizes with huge amounts of assets. Concerning insurance, small CSPs might benefit from an anticipation of a few months' worth of pricing by their customers (thus endowing the CSP with the ability to keep the systems running after the cessation of business), but such a solution might become impossible as the infrastructure size (and maintenance costs) increases, so an insurance against data loss from a licensed insurance company might be more useful. Finally, the customer is always encouraged to have a backup of its assets, independently of the size and type of the CSP that it uses as an infrastructure.

Best practices in an actual example

The objective of this section is to describe best practices and possible techniques actually used to ensure cloud providers viability. Such techniques can be shown through the example of a cloud provider located in Luxembourg that hosts a Web Banking application. A practical representation of the techniques can be shown with models used in requirements engineering, and particularly to the Socio-Technical Security modeling language (STS-ml) language (Dalpiaz et al. 2011) and the STS-Tool (Paja et al. 2014), derived from the i* notation (Yu 1997). The requirements model for the example is displayed in Fig. 2. In particular, the diagram shows the activities put into place by the Luxembourgish CSP that have an impact on long-term viability.

In the diagram, the gray boxes represent the documents (in a very broad sense) produced by each actor toward the fulfillment of a given goal. "Darker" boxes represent documents that are produced by externally and transmitted from one actor to another.

On a regular basis, the CSP performs a backup of all the assets and services that support the deployment and the execution of the Web Banking application. To ensure portability and standardization, it recurs to XML and web services for the implementation of the application. This enables the use of standard connectors, and allows to easily redeploy the application to another environment in case the cloud provider stops its services. All business flows that describe the customers' connection to the Web Banking application, the financial transactions and services that can be performed once the end users log into their web banking session are modeled through choreographies. A legal framework defines the terms of services between the bank and the cloud provider. In this legal framework, objectives and penalties are defined by means of SLAs, and a termination plan defines the different actions required to ensure the continuity of services. Additionally, as the CSP is located in Luxembourg, in case of bankruptcy its customers can retrieve their assets, according to the Luxembourg law.

The banking application runs through a load-balancing mode in three sites, two distant sites in two different centers in Luxembourg and one site in Germany. This load-balancing mode enables continuous service delivery even in case of disaster occurring in one of those sites. From the point of view of the mathematical model described in "A mathematical model of viability", it also contributes to increasing the fungibility p_f of the system, because, in case of bankruptcy of the Luxembourgish CSP, it is more likely that other companies acquire the assets and enact the services again, as it would be possible to acquire not the whole CSP but just one of the sites. The backup of data and critical services is performed in all three sites.

A certified auditor controls all the services according to the ISO 27001 standard twice a year. ISO 27001 specifies an Information Security Management System (ISMS) that is used to identify potential risks. It focuses on ensuring confidentiality, integrity and availability of providers supported services. It also requires an exhaustive listing and asset management with criticality level classification. This classification allows to identify and set up the appropriate measures to increase the amount of recoverable assets r

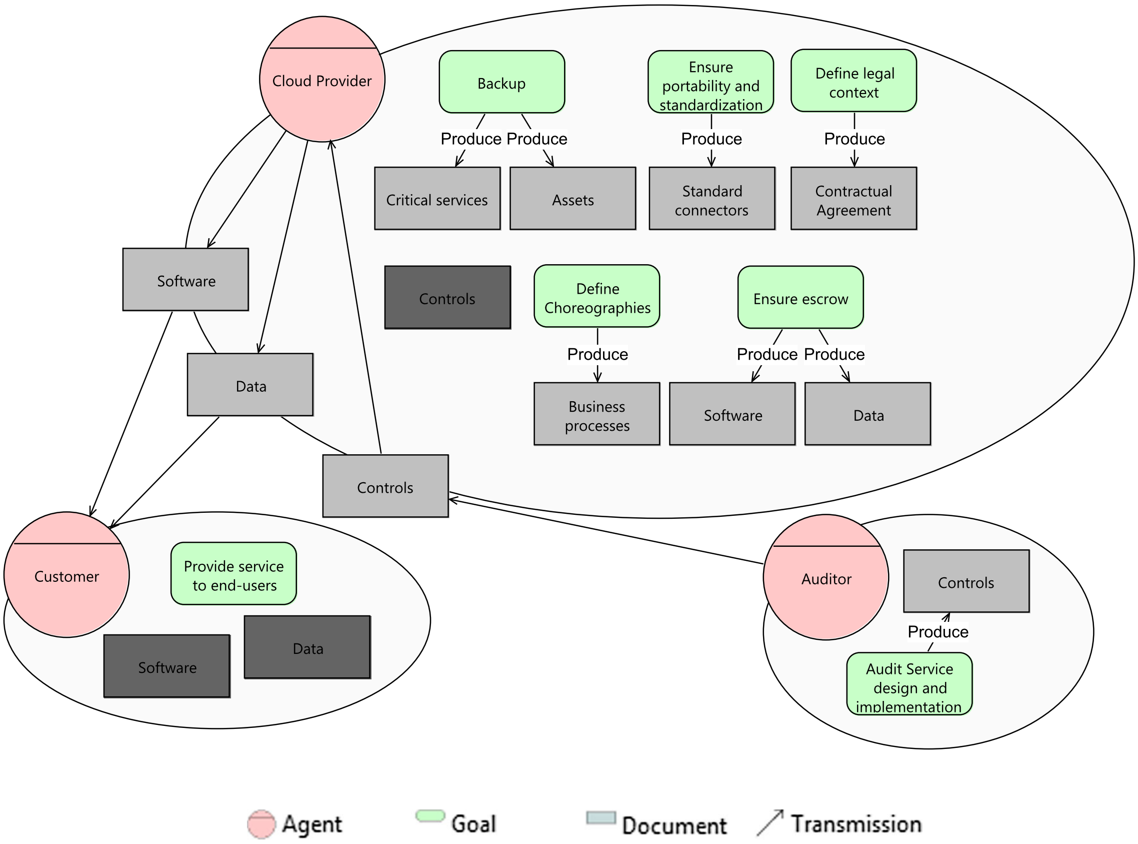

Fig. 2 i* requirements model

defined in the viability model. The auditors also verify that Business Continuity Plan (BCPs) are identified as formal procedures in the enterprize, and that BCP tests are performed regularly to assess the recovery rate and the setup time required before recovery. The outcome of such procedures ensures that appropriate actions are taken following these tests. The impact on the parameters in the viability model is to improve the recovery rate ρ and the setup time value t_0.

The cloud provider acts as a Professionals of the Financial Sector (PSF) entity (the wide set of operators who exercise in a professional financial activity in Luxembourg), and transmits all reports to the CSSF, which regulates the financial sector in Luxembourgish institutions, to verify that the cloud provider is compliant with all the prudential regulation when providing its services.

What emerges from the scenario presented above is that, although it is impossible to completely wipe away all risks for the customer in the case of bankruptcy of the CSP, the combination of several of the above techniques (including backup, standard interfaces and connectors, choreographies, SLAs, and policy support) creates a protective cushion that can offer guarantees in such a situation. If the guarantees offered are considered sufficient, the CSP will create a high degree of trust that will attract customers. It should also be noted that some of the above measures, such as backing up the assets and the use of certain standards, is mandatory in such a delicate domain as finance, but other, less critical domains would still benefit from a voluntary adoption of such measures.

Conclusions

This work analyzes the current cloud landscape by focusing on the challenges related to CSPs' long-term viability, which is largely unexplored. One of the main risks associated with the cloud computing model has been pointed out, analyzing the current challenges behind the continuity of services when the provider goes out of business. It is a minimal risk for big corporations, as they are generally financially strong and able to withstand financial difficulties for extended lengths of time; on the other hand, when outsourcing to a small enterprize, appropriate measures are needed in case the CSP vanishes from one day to the next one, because even a short period of bad financial results might prevent it from continuing its business. The problem of cloud providers' long-term viability has no easy solution, since it is unpredictable if and when a CSP will go out of business. Unfortunately, it is the object of very little investigation, despite the huge consequences that it can have on modern businesses.

The paper starts out by framing the problem of long-term viability, using some real-world examples to highlight its potential impact. Then, the first topic that this work tries to address is that of classifying viability in relationship with other requirements, particularly reliability.

Following an approach widely used in requirements engineering, a model of long-term viability is then proposed, destructuring it into its basic factors and defining metrics to measure them according to this partitioning.

Using that model as a reference, the paper then surveys the most relevant approaches that can be undertaken to increase the viability of a CSP and mitigate the problems into which customers can incur in case it actually (and maybe suddenly) goes out of business. These approaches can ensure that the service keeps running steadily, albeit under a different legal entity (software escrow); keep the services running for some time, to allow the customer to retrieve its assets (insurances and advance payments); allow to quickly migrate the outsourced assets (standards and interoperability measures); or allow recovery of the assets in a bankruptcy procedure (under Luxembourgish law). Among these, the first category offers the customer a long-term guarantee, as the escrow can deliver the service in a manner which minimizes the inconvenience to customers, while the approaches in all other categories are more limited, as customers are required to undertake a prompt reaction in order not to lose their assets and business continuity. It should be noted that the approaches described in the context of this paper were not originally designed to solve the problem of long-term viability, but they can be used for this purpose as well.

The choice of a CSP can also be based on the degree of viability offered (such as specific contractual clauses). For this reason, the above measures serve not only to protect the customer against the loss of the means of doing its business, but also to increase the trust inspired by the CSP. The present research offers a means of evaluating the measures adopted by the CSP, and perform a quick analysis of its long-term viability. The proposed model allows to use viability as one of the parameters that will drive the choice of outsourcing assets to the cloud, and in the selection of which CSP to adopt. With some refinement, it would also be possible to implement the measures in a cloud-brokering tool. An implementation of the proposed model using a formal language to express non-functional requirements, such as SysML or one of the formalisms used for SLAs, might allow it to be integrated into existing tools.

The proposed measures differ significantly, both in their costs and in the benefits they provide. A CSP should find the right trade-off between a desired degree of long-term viability and the resources that it should invest toward that goal. The last result of the current research is an analysis of the proposed solutions in the perspective of their prospective costs and benefits. It is not a full-fledged cost analysis, but at least it provides an estimate of what measures should be preferred, on the basis of the main features (delivery model and size) of the CSP.

Acknowledgements The present work is an invited extension of (Bartolini et al. 2015).

References

Abrahao, B., Almeida, V., Almeida, J., Zhang, A., Beyer, D., Safai, F. (2006). Self-adaptive SLA-driven capacity management for internet services. In *Proceedings of the 10th IEEE/IFIP network operations and management symposium (NOMS)* (pp. 557–568). IEEE. ISBN: 1-4244-0142-9. https://doi.org/10.1109/NOMS.2006.1687584.

Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M. (2007). Web services agreement specification (WS-Agreement). Open Grid Forum (OGF). http://www.ogf.org/documents/GFD.107.pdf (Accessed 3 Nov 2016).

Anthony, S. (2012). Megaupload's demise: what happens to your files when a cloud service dies? http://www.extremetech.com/computing/114803-megauploads-demise-what-happens-toyour-files-when-a-cloud-service-dies (Accessed 3 Nov 2016).

Aubert, B.A., Patry, M., Rivard, S. (2002). Managing IT outsourcing risk: lessons learned. In R. Hirschheim, A. Heinzl, J. Dibbern III (Eds.) *Information systems outsourcing. Enduring themes, emergent patterns and future directions* (Vol. 155 p. 176). Berlin: Springer. ISBN: 978-3-662-04756-9. https://doi.org/10.1007/978-3-662-04754-5_7.

Avizienis, A., Laprie, J.-C., Randell, B., Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, *1*(1), 11–33.

Bauer, E., & Adams, R. (2012). *Reliability and availability of cloud computing*, 1st Edn. Wiley-IEEE Press.

Bartolini, C., El Kateb, D., Le Traon, Y., Hagen, D. (2015). Cloud providers viability: how to address it from an IT and legal perspective? In *Proceedings of the 12th international conference on economics of grids, clouds, systems and services (GECON).* Springer.

Bocciarelli, P., & D'Ambrogio, A. (2011). A BPMN extension for modeling non functional properties of business processes. In *Proceedings of the symposium on theory of modeling & simulation (TMS/DEVS)* (pp. 160–168). Society for Computer Simulation International.

Boehm, B., Abts, C., Chulani, S. (2000). Software development cost estimation approaches—a survey. *Annals of Software Engineering*, *10*(1–4), 177–205.

Bowen, J.A. (2011). Legal issues in cloud computing. In R. Buyya, J. Broberg, A.M. Goscinski (Eds.) *Cloud computing: principles and paradigms*, 1st Edn, Chap. 24 (pp. 593–613). Hoboken: Wiley.

Brodkin, J. (2008). *Gartner: seven cloud-computing security risks.* Tech. rep. Gartner.

Butler, B. (2014). The best time to prepare for getting data out of the cloud is before you put it in there. http://www.networkworld.com/article/2173255/cloud-computing/cloud-s-worst-case-scenario-what-to-do-if-your-provider-goes-belly-up.html (Accessed 3 Nov 2016).

Buyya, R., Pandey, S., Vecchiola, C. (2009). Cloudbus toolkit for market-oriented cloud computing. In M.G. Jaatun, G. Zhao, C. Rong (Eds.) *Cloud computing. Lecture notes in computer science* (Vol. 5931, pp. 24–44). Berlin: Springer.

C-SIG SLA (2014). *Cloud service level agreement standardisation guidelines*. Cloud Select Industry Group on Service Level Agreements (C-SIG SLA). Brussels. http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?action=display&doc_id=6138 (Accessed 3 Nov 2016).

Caplan, D.S. (2010). *Bankruptcy in the cloud: effects of bankruptcy by a cloud services provider*. Tech. rep. 1289. Chapel Hill: Law Offices of David S. Caplan. http://ftp.documation.com/references/ABA10a/PDfs/3_3.pdf (Accessed 3 Nov 2016).

Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J. (2000). *Non-functional requirements in software engineering* (Vol. 5). US: Springer.

Conley, J.M., & Bryan, R.M. (1985). Software escrow in bankruptcy: an international perspective. *North Carolina Journal of International Law and Commercial Regulation*, *10*(3), 579–607. ISSN: 0743-1759.

Dalpiaz, F., Paja, E., Giorgini, P. (2011). Security requirements engineering via commitments. In *Proceedings of the 1st socio-technical aspects in security and trust (STAST)*. IEEE.

Dichev, I.D. (1998). Is the risk of bankruptcy a systematic risk? *The Journal of Finance*, *53*(3), 1131–1147.

Dowell, S., Barreto, A. .I, Michael, J.B., Shing, M.-T. (2011). Cloud to cloud interoperability. In *Proceedings of the 6th international conference on system of systems engineering (SoSE). Albuquerque: IEEE* (pp. 258–263).

Franke, U., Johnson, P., König, J. (2014). An architecture framework for enterprise IT service availability analysis. *Software & Systems Modeling*, *13*(4), 1417–1445. ISSN: 1619-1366. https://doi.org/10.1007/s10270-012-0307-3.

Fry, M. (2004). Service-continuity goals important. *Communications News*, *41*(10), –48.

Gebregiorgis, S.A., & Altmann, J. (2015). IT service platforms: their value creation model and the impact of their level of openness on their adoption. In K. Jeffery, D. Kyriazis (Eds.), *Procedia computer science. 1st international conference on cloud forward: from distributed to complete computing* (Vol. 68, pp. 173–187). ISSN: 1877-0509. https://doi.org/10.1016/j.procs.2015.09.233.

Glinz, M. (2005). Rethinking the notion of non-functional requirements. In *Proceedings of the 3rd world congress for software quality (WSCQ)* (pp. II–55–II–64).

Glinz, M. (2007). On non-functional requirements. In *Proceedings of the 15th IEEE international requirements engineering conference (RE)* (pp. 21–26). IEEE.

Gnedenko, B.V., Belyayev, Y.K., Solovyev, A.D. (1969). Mathematical methods of reliability theory. In Z.W. Birnbaum, E. Lukacs (Eds.), *Probability and mathematical statistics: a series of monographs and textbooks* (518 pp.). Academic Press. ISBN: 978-1-4832-3053-5.

Guo, Q., Zhan, Z., Wang, T., Zhao, X. (2012). Risk assessment and optimal proactive measure selection for IT service continuity management. In *Proceedings of the network operations and management symposium (NOMS)* (pp. 1386–1391). ISBN: 978-1-4673-0267-8. https://doi.org/10.1109/NOMS.2012.6212080.

Haile, N., & Altmann, J. (2013). Estimating the value obtained from using a software service platform. In K. Vanmechelen, J. Altmann, O. F. Rana (Eds.) *Economics of grids, clouds, systems, and services. 10th international conference, GECON 2013, Zaragoza, Spain, September 18–20, 2013. Proceedings. Lecture notes in computer science* (Vol. 8193, pp. 244–255). Berlin: Springer International Publishing. ISBN: 978-3-319-02413-4. https://doi.org/10.1007/978-3-319-02414-1_18.

Harsh, P., Dudouet, F., Cascella, R.G., Jegou, Y., Morin, C. (2012). Using open standards for interoperability - issues, solutions, and challenges facing cloud computing. In *Proceedings of the 8th international conference on network and service management (CNSM) and 6th international DMTF academic alliance workshop on systems and virtualization management: standards and the cloud (SVM)* (pp. 435–440). Las Vegas: IEEE.

Hennesy, J. (1999). The future of systems research. *Computer*, *32*(8), 27–33.

Hiles, A. (2000). *Service level agreements: winning a competitive edge for support & supply services*, 2nd Edn. Rothstein Catalog on Service Level Books. Brookfield: Rothstein Associates Inc.

Hillegeist, S.A., Keating, E.K., Cram, D.P., Lundstedt, K.G. (2002). Assessing the probability of bankruptcy. *Review of Accounting Studies*, *9*(1), 5–34.

Hu, F., Qiu, M., Li, J., Grant, T., Tylor, D., McCaleb, S., Butler, L., Hamner, R. (2011). A review on cloud computing: design challenges in architecture and security. *Journal of Computing and Information Technology*, *19*(1), 25–55.

ISO (2010). *Systems and software engineering – Vocabulary*. Tech. rep. International Organization for Standardization. https://doi.org/10.1109/IEEESTD.2010.5733835.

ISO (2011). *Systems and software engineering– systems and software Quality Requirements and Evaluation (SQuaRE) – system and software quality models*. Tech. rep. International Organization for Standardization.

ISO (2012). *Societal security – business continuity management systems – requirements*. Tech. rep. International Organization for Standardization.

ISO (2013). *Information technology – security techniques – Code of practice for information security controls*. Tech. rep. International Organization for Standardization.

Jeffery, K., Kousiouris, G., Kyriazis, D., Altmann, J., Ciuffoletti, A., Maglogiannis, I., Nesi, P., Suzic, B., Zhao, Z. (2015). Challenges emerging from future cloud application scenarios. In K. Jeffery, D. Kyriazis (Eds.), *Procedia computer science. 1st international conference on cloud forward: from distributed to complete computing* (Vol. 68, pp. 227–237). ISSN: 1877-0509. https://doi.org/10.1016/j.procs.2015.09.238.

Kandukuri, B.R., R. Paturi, V., Rakshit, A. (2009). Cloud security issues. In *IEEE international conference on services computing (SCC)* (pp. 517–520). IEEE. ISBN: 978-1-4244-5183-8. https://doi.org/10.1109/SCC.2009.84.

Kaufman, L.M. (2009). Data security in the world of cloud computing. *IEEE Security & Privacy*, *7*(4), 61–64. ISSN: 1540-7993. https://doi.org/10.1109/MSP.2009.87.

Kauffman, R.J., Ma, D., Yu, M. (2014). A metrics suite for firm-level cloud computing adoption readiness. In K. Vanmechelen, J. Altmann, and O.F. Rana (Eds.), *Economics of grids, clouds, systems, and services. 11th international conference, GECON 2014, Cardiff, UK, September 16–18, 2014. Revised Selected Papers* (Vol. 8914, pp. 19–35). Lecture Notes in Computer Science. Springer International Publishing. ISBN: 978-3-319-14608-9. https://doi.org/10.1007/978-3-319-14609-6_2.

Khajeh-Hosseini, A., Greenwood, D., Sommerville, I. (2010). Cloud migration: a case study of migrating an enterprise IT system to IaaS. In *IEEE 3rd international conference on cloud computing (CLOUD)* (pp. 450–457). IEEE. ISBN: 978-1-4244-8207-8. https://doi.org/10.1109/CLOUD.2010.37.

Kharb, L. (2016). Automated deployment of software containers using dockers. *International Journal of Emerging Technologies in Engineering Research (IJETER)*, *4*(10), 1–3. ISSN: 2454-6410.

Klass, A.B. (2006). Modern public trust principles: recognizing rights and integrating standards. *Notre Dame Law Review*, *82*(2), 699–754. ISSN: 0745-3515.

Koch, F., de Assunção, M.D., Netto, M.A.S. (2012). A cost analysis of cloud computing for education. In K. Vanmechelen, J. Altmann, O.F. Rana (Eds.) *International conference on grid economic and business models. 9th international conference, GECON 2012, Berlin, Germany, November 27–28, 2012. Proceedings. Lecture notes in computer science* (Vol. 7714, pp. 182–196). Berlin: Springer. ISBN: 978-3-642-35193-8. https://doi.org/10.1007/978-3-642-35194-5_14.

Kozina, M. (2009). COBIT - ITIL mapping for business process continuity management. In B. Auer, M. Bača, K. Rabuzin (Eds.) *Proceedings of the 20th central European conference on information and intelligent systems* (pp. 113–119). Varaždin: University of Zagreb, Faculty of Organization and Informatics.

Kumar Garg, S., Versteeg, S., Buyya, R. (2013). A framework for ranking of cloud computing services. *Future Generation Computer Systems*, *29*(4), 1012–1023.

Kuyoro, S.O., Ibikunle, F.A., Awodele, O. (2011). Cloud computing security issues and challenges. *International Journal of Computer Networks*, *3*(5), 247–255.

Laprie, J.-C., & Kanoun, K. (1996). Software reliability and system reliability. In M.R. Lyu (Ed.) *Handbook of software reliability engineering*, Chap. 2 (pp. 27–69). New York: McGraw-Hill.

Lee, J.Y., Lee, J.W., Cheun, D.W., Kim, S.D. (2009). A quality model for evaluating software-as-a-service in cloud computing. In *Proceedings of the 7th ACIS international conference on software engineering research, management and applications (SERA)* (pp. 261–266). IEEE.

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., Leaf, D. (2011). *NIST cloud computing reference architecture*. Recommendations of the National Institute of Standards and Technology SP 500-292. Gaithersburg: National Institute of Standards and Technology.

Louwers, E.-J. (2013). Continuity in the Cloud: new practical solutions required. ITechLaw 2013 European Conference.

Ludwig, H., Keller, A., Dan, A., King, R.P., Franck, R. (2003). *Web Service Level Agreement (WSLA) language specification*. 1.0. IBM Corporation, New York. http://www.research.ibm.com/people/a/akeller/Data/WSLASpecV1-20030128.pdf (Accessed 3 Nov 2016).

Lyu, M.R. (Ed.) (1996). *Handbook of software reliability engineering*. Hightstown: McGraw-Hill. ISBN: 0-07-039400-8.

Machado, G.S., Hausheer, D., Stiller, B. (2009). Considerations on the interoperability of and between cloud computing standards. In: *27th open grid forum (OGF27), G2C-Net workshop: from grid to cloud networks*. Banff: OGF.

Maurer, M., Emeakaroha, V.C., Brandic, I., Altmann, J. (2012). Cost-benefit analysis of an SLA mapping approach for defining standardized Cloud computing goods. *Future Generation Computer Systems*, *28*(1), 39–47. ISSN: 0167-739X. https://doi.org/10.1016/j.future.2011.05.023.

McKendrick, J. (2013). What to do in case your cloud provider falls off the grid. http://www.forbes.com/sites/joemckendrick/2013/11/04/what-to-do-in-case-your-cloud-provider-fallsoff-the-grid/#6c2fb6da3c53 (Accessed 3 Nov 2016).

Mezrich, J.L. (2001). Source code escrow: an exercise in futility? In *Marquette intellectual propetry law review 5* (pp. 117–131). ISSN: 1092-5899.

Mills, L.H. (2009). Legal issues associated with cloud computing. http://www.secureit.com/resources/Cloud%5C%20Computing%5C%20Mills%5C%20Nixon%5C%20Peabody%5C%205-09.pdf (Accessed 3 Nov 2016).

Năstase, P., Năstase, F., Ionescu, C. (2009). Challenges generated by the implementation of the IT standards COBIT 4.1, ITIL V3 and ISO/IEC 27002 in Enterprises. In *Economic computation & economic cybernetics studies & research* (Vol. 3, pp. 5–20). ISSN: 1842-3264.

Paja, E., Dalpiaz, F., Giorgini, P. (2014). STS-Tool: security requirements engineering for socio-technical systems. In M. Heisel, W. Joosen, J. Lopez, F. Martinelli (Eds.) *Engineering secure future internet services and systems. Lecture Notes in Computer Science* (Vol. 8431, pp. 65–96). Berlin: Springer International Publishing.

Pandey, R.S., & Chaudhary, B. (2008). A cost model for participating roles based on choreography semantics. In *Proceedings of the IEEE Asia-pacific services computing conference (APSCC)* (pp. 277–283). IEEE. ISBN: 978-0-7695-3473-2. https://doi.org/10.1109/APSCC.2008.117.

Pappous, P.A. (1985). The software escrow: the court favorite and bankruptcy law. *Santa Clara High Technology Law Journal*, *1*(2), 309–326.

Peltz, C. (2003). Web services orchestration and choreography. *Computer*, *36*(10), 46–52.

Pettey, C., & van der Meulen, R. (2009). Gartner says cloud consumers need brokerages to unlock the potential of cloud services. http://www.gartner.com/newsroom/id/1064712 (Accessed 3 November 2016).

Roa Martínez, N.A. (2011). Viernes Negro. *Estudios Gerenciales*, *27*(120), 227–249. ISSN: 0123-5923. https://doi.org/10.1016/S0123-5923(11)70177-2.

Rochwerger, B., Breitgand, D., Levy, E., Galis, A., Nagin, K., Llorente, I.M., Montero, R., Wolfsthal, Y., Elmroth, E., Cáceres, J., Ben-Yehuda, M., Emmerich, W., Galán, F. (2009). The reservoir model and architecture for open federated cloud computing. *IBM Journal of Research and Development*, *53* (4), X:1–X:11.

Roman, G.-C. (1985). A taxonomy of current issues in requirements engineering. *Computer*, *18*(4), 14–23.

Sahai, A., Machiraju, V., Sayal, M., vanMoorsel, A., Casati, F. (2002). Automated SLA monitoring for web services. In M. Feridun, P. Kropf, G. Babin (Eds.) *Management technologies for E-commerce and E-business applications. Lecture notes in computer science* (Vol. 2506, pp. 28–41). Berlin: Springer.

Sahibudin, S., Sharifi, M., Ayat, M. (2008). Combining ITIL, COBIT and ISO/IEC 27002 in order to design a comprehensive IT framework in organizations. In *Proceedings of the second asia international conference on modeling & simulation (AICMS)* (pp. 749–753). IEEE. ISBN: 978-0-7695-3136-6. https://doi.org/10.1109/AMS.2008.145.

Sallé, M. (2004). *IT service management and IT governance: review, comparative analysis and their impact on utility computing*. Tech. rep. HPL-2004-98. Palo Alto: HP Laboratories.

Sauvé, J., Santos, R., Rebouças, R., Moura, A.A., Bartolini, C. (2008). Change priority determination in IT service management based on risk exposure. *IEEE Transactions on Network and Service Management*, *5*(3), 178–187. ISSN: 1932-4537. https://doi.org/10.1109/TNSM.2009.031105.

Scott, J. (1981). The probability of bankruptcy: a comparison of empirical predictions and theoretical models. *Journal of Banking & Finance*, *5*(3), 317–344.

Secteur Financier (CSSF), & C. de Surveillance du (2017). Circular CSSF 17/654. http://www.cssf.lu/fileadmin/files/Lois_reglements/Circulaires/Hors_blanchiment_terrorisme/cssf17_654eng.pdf (Accessed 10 June 2017).

Tassey, G. (2000). Standardization in technology-based markets. *Research Policy*, *29*(4–5), 587–602. ISSN: 0048-7333. https://doi.org/10.1016/S0048-7333(99)00091-8.

Thibodeau, P. (2013). One in four cloud providers will be gone by 2015. http://www.computerworld.com/article/2486691/cloudcomputing/one-in-four-cloud-providers-will-be-gone-by-2015.html (Accessed 3 Nov 2016).

Tordssona, J., Montero, R.S., Moreno-Vozmediano, R., Llorente, R. (2012). Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Generation Computer Systems*, *28*(2), 358–367.

van de Zande, T., & Jansen, S. (2011). Business continuity solutions for SaaS customers. In B. Regnell, I. van de Weerd, O. De Troyer (Eds.) *Software business. Lecture notes in business information processing* (Vol. 80, pp. 17–31). Berlin: Springer.

Van Hoboken, J., Arnbak, A., Van Eijk, N. (2013). Obscured by clouds or how to address governmental access to cloud data from abroad. In *Proceedings of the 6th annual privacy law scholars conference (PLSC)*.

van Moorsel, A. (2001). Metrics for the internet age: quality of experience and quality of business. In *Proceedings of the 5th international workshop on performability modeling of computer and communication systems (PMCCS)*.

Venkatraman, A. (2013). *2e2 datacentre administrators hold customers' data to £1m ransom*. http://www.computerweekly.com/news/2240177744/2e2-datacentre-administrators-hold-customers-data-to-1m-ransom (Accessed 3 Nov 2016).

Vrable, M., Savage, S., Voelker, G.M. (2009). Cumulus: filesystem backup to the cloud. *ACM Transactions on Storage*, *5*(4), 14:1–14:28. ISSN: 1553-3077. https://doi.org/10.1145/1629080.1629084.

Wan, S. (2009). Service impact analysis using business continuity planning processes. *Campus-Wide Information Systems*, *26*(1), 20–42. ISSN: 1065-0741. https://doi.org/10.1108/10650740910921546.

Weber, R.H., & Staiger, D.N. (2014). Cloud computing: a cluster of complex liability issues. *Web Journal of Current Legal Issues*, *20*(1).

Weitzel, T., Beimborn, D., König, W. (2006). A unified economic model of standard diffusion: the impact of standardization cost, network effects, and network topology. *MIS Quarterly (Special Issue on Standard Making)*, *30*, 489–514. ISSN: 2162–9730.

Wieder, P., Butler, J.M., Theilmann, W., Yahyapour, R. (Eds.) (2011). *Service level agreements for cloud computing*. New York: Springer Science+Business Media, LLC.

Yu, E.S. (1997). Towards modelling and reasoning support for earlyphase requirements engineering. In *Proceedings of the 3rd IEEE international symposium on requirements engineering (RE)* (pp. 226–235). IEEE.

Zo, H., Nazareth, D.L., Jain, H.K. (2007). Measuring reliability of applications composed of web services. In *Proceedings of the 40th Hawaii international conference on system sciences (HICSS)*. IEEE.