Cloud Computing and Technological Lock-In
Literature Review

Robert Viseur[1,2], Etienne Charlier[1] and Michael Van de Borne[1]
[1] *CETIC, Rue des Frères Wright, 29/3, 6041 Charleroi, Belgium*
[2] *UMONS Faculty of Engineering, Rue de Houdain, 9, B-7000 Mons, Belgium*

Keywords: Cloud Computing, Lock-in, Iaas, Paas, Saas, Big Data, Standardization, Open Source.

Abstract: The increasing use of cloud computing services results in an increased risk of lock-in that is source of anxiety for users facing the risk of having their data to be hosted online without the possibility of migrating them on their own IT resources or on competitors' platforms. In this preliminary research, we deal with the problem of the management of lock-in in case of use of cloud computing services. We aim to answer six questions: (1) *What is the lock-in?* (2) *Is the lock-in perceived as a major problem?* (3) *What are the causes of lock-in?* (4) *What is the impact of lock-in on the users?* (5) *How can the users avoid lock-in?* and (6) *Is the general public concerned with the problem of lock-in?* Our paper is organized in three sections. The first section presents the methodology used for this study. The second section details the results. This section identifies in particular six mechanisms to reduce the risk of lock-in. The third section discusses the results and suggests further work.

1 INTRODUCTION

The cloud computing has its roots in the Application Service Providers (ASP) model that emerged in the early 2000s. The phenomenon "*represents a fundamental change in the way information technology (IT) services are invented, developed, deployed, scaled, updated, maintained and paid for*" (Marston *et al.*, 2011). Cloud computing can be defined as "*an information technology service model where computing services (both hardware and software) are delivered on demand to customers over a network in a self-service fashion, independent of device and location*" (Marston *et al.,* 2011).

From the point of view of a user, cloud computing comes in three distinct models (CIGREF, 2013; Marston *et al.*, 2011). The Software as a Service model (SaaS) provides the user with an application hosted in the cloud (e.g. Google Mail, Google Documents). The Platform as a Service (PaaS) model provides an environment to develop and deploy applications (e.g. Microsoft Azure, Google App Engine). The Infrastructure as a Service (IaaS) model provides storage and computation capacities (e.g. Amazon S3, Amazon EC2). These models can be deployed in a corporate network or an external platform. The first case is known as private cloud, the second, as public cloud. The latter benefits on an important marketing from companies and the growing notoriety of providers such as Amazon (EC2) and Microsoft (Azure). The cloud also affects the general public. The latter indeed faced through online services such as SaaS, for example, Facebook messaging services, Google Mail email services or Google Documents online productivity tool.

Whereas previously the user had its applications and its data on its own computer (or on a network of computers under its control), the cloud computing outsources infrastructure, applications and/or data. This results in a lock-in that is increased (or perceived as such). It is a cause of concern for users facing the risk of having their online data without the possibility of migrating them on their own IT resources or on competing service provider.

In this preliminary research, we deal with the problem of the management of lock-in in case of use of cloud services. We aim to provide an initial response to the following six questions: (1) *What is the lock-in?* (2) *Is the lock-in perceived as a major problem?* (3) *What are the causes of lock-in?* (4) *What is the impact of lock-in for the users?* (5) *How*

Viseur R., Charlier E. and Van de Borne M..
Cloud Computing and Technological Lock-In - Literature Review.
DOI: 10.5220/0005109903050313
In *Proceedings of 3rd International Conference on Data Management Technologies and Applications* (DATA-2014), pages 305-313
ISBN: 978-989-758-035-2

can users avoid lock-in? and (6) *Is the general public concerned with the problem of lock-in?* Our paper is organized into three sections. The first section presents the methodology used for this study. The second section develops the results that are relative to the six research questions. The third section discusses the results and proposes further work.

2 METHODOLOGY

Our research consists of a literature review. The latter is based on two types of sources. On the one hand we used articles from scientific literature that substantially address the issue of lock-in. These papers were found mainly by querying the Google Scholar search engine (scholar.google.fr). On the other hand we relied on a set of articles from the trade press that specifically deal with the issue of lock-in in the cloud. The Google search engine was used to identify these items. In practice, the next four queries were used: `cloud lock-in`, `(saas OR paas OR iaas) lock-in`, `(amazon OR rackspace OR azure OR "app engine" OR smartcloud OR salesforce) lock-in`, and `(facebook OR gmail) lock-in`. The links from the first page of search results were filtered in order to keep only the substantive articles (for the third query, the number of off-topic links were obliged to go on the third page of results). At the end, twenty-four articles were identified with this method. They were complemented by a serie of five articles pointed in the first series of articles. The articles from the professional literature can be identified in the references by the presence of the URL. Note that the notion of lock-in seems to be used very little for public services, perhaps because the term is more familiar to business users.

The name of the suppliers that we used in queries was determined on the basis of the emphasis in the professional and scientific literature (Crochet-Damais, 2013; Darrow, 2012; Harsh *et al.*, 2012; Nachmani, 2012; ZDNet, 2013; Zhang *et al.*, 2013). In particular, the publication of market shares enables to objectify this choice. Amazon Web Services (AWS), Microsoft Azure, Google App Engine, Rackspace, Salesforce and IBM SmartCloud for professionals were therefore chosen as the focus in this paper. Facebook and Gmail were taken into account for services that are more oriented towards the general public. Each article from the professional or scientific literature was processed to identify items that meet the six research questions. These elements were then categorized.

3 RESULTS

3.1 What Is the Lock-In?

According to Germain (2013), the lock-in appears when the cost of changing technology from one vendor to another is so expensive that the client is unable to leave the vendor's offers. The lock-in is not a new concept (Linthicum, 2012). Juengst (2012) gives also well-known examples including blocked mobile phones and ink cartridges. Linthicum (2012) adds that the principle of the use of resources external to the company is not new. According to Germain (2013), the vendor lock-in has been part of life in the IT business for many years. In practice, the topic of lock-in also implies migration costs.

Chatzakis (2012) examines the issue of lock-in in AWS, and identifies three levels of lock-in. The light lock-in corresponds in practice to an absence of lock-in: for example the platform uses industry standards. The medium lock-in occurs when the platform provides non-standard services whose blocking character may be limited by rules relative to development and architecture. The hard lock-in requires parts of the source code to be rewritten but it is associated with the provision of innovative features that are sources of opportunities for the users.

According to Malhotra (2013), "lock-in" is another way of saying "risk". There is a trade-off between the lock-in (risk) and the value (profit). Less lock-in means less functionality and more source code to write.

Zhang *et al.* (2013) combines the concept of lock-in with three other concepts: interoperability, compatibility and portability. The interoperability in the cloud is the ability of multiple vendors working together. The compatibility in the cloud means that data and applications can operate in the same manner regardless of the cloud provider. The portability in the cloud means that data and applications can be easily moved and reused whatever the choice of cloud provider, operating system, storage format or API. In practice, by improving the interoperability, the compatibility and the portability help to reduce the lock-in.

Note that the term "lock-in" is also found in the literature relevant to Increasing Returns to Adoption

(in french: "*Rendements Croissants d'Adoption*", RCA). Translated as inflexibility (in French: "*inflexibilité*"), the lock-in is a locking mechanism in the adoption process of a technology that competes with other technologies (Foray, 1989).

3.2 Is the Lock-In Perceived as a Major Problem?

The perceived importance of lock-in is widely discussed.

According to Gruman (2007), the issue of lock-in is one of the concerns for the users of enterprise management solutions, especially as the sector has experienced a significant consolidation in 2002. This concern is reinforced by the exit costs that are particularly high for management applications (Messerschmitt and Szyperski, 2001). For Nachmani (2012), the lock-in has an important impact on the decision to use or not to use the cloud.

In practice, however, the results are more mixed. Smets, a French entrepreneur active in free software edition and SaaS solutions based on free software, believes that the protection of privacy, commercial confidentiality and freedom to migrate are irrelevant differentiators for the vast majority of the market (Viseur, 2013b). Kash (2013) noted that only 15% of customers are concerned about the vendor lock-in. This criterion appears in eighth position, a result to be compared to security vulnerabilities that appear in the first position and concern 51% of customers.

3.3 What are the Causes of Lock-In?

The first cause of lock-in is the pace of innovation and the search for differentiation between competitors. The cloud service providers seek to differentiate themselves by placing innovations on the market. These innovations take the form of advanced features to customize the services provided by the cloud computing suppliers (Kash, 2013). These features increase the risk of lock-in. They are particularly established in the form of vendor-specific API (Germain, 2013; Juengst, 2012; Kash, 2013). These APIs allow the management of the platform to be automated or customized, or access to innovative services to be provided, for example in the field of storage (Harris, 2013; Kash, 2013). The vendor-specific API are currently used by 23% of customers, a figure expected to increase (Kash, 2013).

The second cause of lock-in is the search for Increasing Returns to Adoption. Babcock (2013) estimates that the lock-in occurs when a company becomes a dominant seller for a technology and develops products that make progress with proprietary elements. This strategy prevents customers from leaving. The providers keep their technologies proprietary for as long as possible, because this blocks the customers in their environment (McKendrick, 2011). The lock-in is a good thing for the vendor because it reduces customer turnover ("churn") (Harsh *et al.*, 2012). However, the authors challenge this view because they believe that brand loyalty must be obtained by service quality and attractive prices. For Zhang et al. (2013), the incompatibility between cloud products and services providers may temporarily protect the interests of each supplier. However, this strategy will prove counterproductive in the long term as the market will become more mature. In addition, this strategy goes against the new modes of cooperative definition of open standards that favor a wide dissemination of the standard rather than control (Adatto, 2013).

The third cause of lock-in is the use of proprietary data formats by providers (Chow *et al.*, 2009). In SaaS, the interoperability problem arises especially in terms of data (Zhang *et al.*, 2013). Regardless of any desire to curb the exit of data, the volume of data can itself hamper migration and block the user with a service provider. The outsourced Big Data applications pose specific problems in the case of migration, given the volume of data. The contracts do not always specify the terms of migration of user data when they wish to change supplier (Kash, 2013).

The fourth cause of lock-in is the PaaS type cloud platforms. The PaaS services are frequently highlighted for their significant risk of lock-in (Harris, 2013; Germain, 2013; Juengst, 2012; Nachmani, 2012). Zhang *et al.* (2013) justify this point by the fact that, from IaaS to SaaS, the automation increases; therefore, from IaaS to SaaS, the portability decreases. More portability means more work for the management and the deployment of the software. Various factors explain the importance of lock-in for PaaS (Coté, 2008; Germain, 2013; Juengst, 2012): the use of a proprietary programming language, the use of open source languages extended by proprietary APIs, the provision of proprietary infrastructure services, the use of proprietary databases, etc.

However, even for PaaS, the degree of lock-in is variable (Nachmani, 2012). For example, Force.com, the PaaS of Salesforce.com, has a maximum degree of lock-in, because the Apex language and the database are proprietary. Conversely, Heroku, acquired by Salesforce.com, supports JSON and XML Web services, and widely used languages such as Java, Ruby or PHP, and open source databases such as PostgreSQL and MySQL.

3.4 What Is the Impact of Lock-In for the Users?

The first impact for the users is the blocking of user data and the longer periods of migration. The user data can be difficult to process because of the impossibility of technical means to access, the use of proprietary formats, or the volume of data to be fetched (Chatzakis, 2012). The general public also faces this problem with online services like Facebook or Flickr (Weinberger, 2012).

The extension of the migration duration that results is not without potential consequences for the company. The supplier is not immune to an acquisition by another company (McKendrick, 2011). The new owner can change its policies, and cause problems, including legal problems (e.g. location of data). The customers have no control over the evolution of a commercial cloud (Chow et al., 2009). They may therefore find themselves in trouble due to the closure of a service. The Coghead users faced this situation due to the closure of the company and the purchase of assets by SAP.

The issue of lock-in may be related to privacy, through the issue of life-cycle data, whether for business users or the general public (Pearson, 2009). The last phase of the life cycle is the decommission that provides secure deletion and removal of personal and sensitive data.

The consequences of lock-in can be masked by the pricing practices of companies. Thus, the customers can be attracted by the price war on the upload of data (Darrow, 2013). It is possible that, once these data are online, the providers try to be remunerated otherwise, and therefore hamper the data exit.

The second impact for the users is the price increase for the use of cloud service. Zhang et al. (2013) argue that there is a risk of ridiculously high costs due to lock-in. The company can increase its prices. The user may encounter difficulties to migrate to new more attractive platforms (Coté, 2008). A PaaS vendor can increase the prices once customers are blocked (Juengst, 2012). However, this last statement applies only to PaaS that are not based on a published full open source implementation.

The third impact for the users is the slower pace of innovation. According to Germain (2013), the lock-in creates monopolies for the seller to the detriment of customers and limits the pressure on the supplier to innovate.

The fourth impact for the users is the reduction of the development life cycle time. Against all expectations, some authors present the lock-in as a positive element for the users. This assessment is associated with the vision of lock-in as a result of an innovation process that allows the user to benefit from the service. This view usually comes from people close to cloud providers.

Coffee (2012) prefers the term “leverage” rather than “lock-in”. According to the author, the applications developed on Force.com have a shorter life cycle. Therefore, they can be quickly adapted to market realities. This results in a significant competitive advantage for the users of cloud service that are able to adapt their applications to rapidly changing markets, such as mobile applications market (“time -to-market advantage”).

Magnusson (2013) supports this view: the lock-in is the price to pay for an innovative platform. The platform takes on more work and allows the user to gain time (e.g. Google Datastore). The exit out of the Google App Engine (GAE) needs 3-4 months of work for a large application (Magnusson, 2013). However, the benefits due to GAE are numerous: management of firewall, protection against DDoS, protection against viruses, applying patches and updates, load balancing, etc.

3.5 How can Users Avoid Lock-In?

The first way to avoid lock-in is the use of standards. The standardization improves the interoperability between clouds and reduces lock-in (Chow *et al.*, 2009; Messerschmitt and Szyperski, 2001; Zhang *et al.*, 2013). The standards are numerous. Pahl *et al.* (2013) list a set of standards for the service modelling (e.g. Open-SCA, USDL/SoaML/ CloudML, EMML), service interfaces (e.g. OCCI, CMMI, EC2, TOSCA, CDMI) and infrastructure (OVF).

However, Wolpe (2013) believes that the cloud industry still suffers from a lack of standardization. According to this author, the standardization would happen after an initial innovation phase because it appears as an obstacle to progress (*sic*). However, this vision of standardization contradicts Adatto (2013). The latter analyzes the emergence of new modes of cooperative definition of standards, together with FLOSS implementations, and the development of strategies of competition between industrial players. The development of the FLOSS application concurrent with specification work brings the standardization in the heart of the innovation process. Kash (2013) also believes that the use of open source technologies benefits from a rapid pace of innovation.

Other authors highlight the current lack of standardization and interoperability in the cloud. Kash (2013) regrets the lack of mature standards. If they consolidate on IaaS services they would, however, be virtually non-existent for PaaS services. However, even for IaaS, some technologies are missing for interoperability (Harsh *et al.*, 2012). In IaaS, there is a set of technical issues to be resolved (Zhang et al., 2013). They are partially covered by standards (e.g. OVF, CDMI and OCCI) that are offered by organizations such as the Open Grid Forum (OGF), the Distributed Management Task Force (DTMF) or the Storage Networking Institute Association (SNIA). These standards are progressively implemented in open source solutions such as OpenStack, OpenNebula and Eucalyptus. The progress of standardization may however be variable. For example, network virtualization and security procedures are important issues currently processed at a minimum, unlike the issue of virtualization formats that is well covered by OVF (Harsh *et al.*, 2012; Zhang *et al.*, 2013). In terms of IaaS , there is a taxonomy of interoperability in the IaaS, to point more quickly to the technical problems to be solved, that distinguishes access mechanisms, virtualization, storage, networking, security and service-level agreement (SLAs) (Zhang *et al.*, 2013).

The second way to avoid lock-in is the use of FLOSS (Free Libre Open Source Software). According to Weinberger (2012), a solution to the problem of lock-in is the use of FLOSS implementations such as Apache CloudStack, OpenStack and Eucalyptus. The open source software is increasingly accompanied by standardization initiatives (Adatto, 2013). They often appear at the forefront in the field of standardization. OpenStack is distinguished for example by its ability to describe network via software (Zhang *et al.*, 2013).

The open source implementations may also provide an answer to the concerns of lock-in for PaaS-type services. Juengst (2012) recommends the use of PaaS that offer support for multiple programming languages, are built on open source blocks, are open source themselves and do not offer proprietary APIs. The cases of OpenShift (www.openshift.com) and Google App Engine (cloud.google.com/AppEngine) are illustrative.

The Google App Engine has two open source implementations (Magnusson, 2013). The first is App Scale (www.appscale.com). Google is working with Red Hat for the second that is integrated to OpenShift software via CapeDwarf (www.jboss.org/capedwarf). Google is also working to create a Technology Compatibility Kit (TCK) for Google App Engine (GAE) API (www.appengine-tck.org) that allows the editors of alternative implementations to perform compatibility tests. Red Hat (redhat.com) supports the functionality of the Datastore storage service via Infinispan (infinispan.org) open source software.

Red Hat with OpenShift uses in practice the absence of lock-in as a commercial argument (Juengst, 2012). The promise is to provide a public platform (PaaS OpenShift Online) that the companies can implement in their network, or on a chosen public IaaS provider (via OpenShift Enterprise) that is based on a FLOSS implementation of the OpenShift technology (OpenShift Origin). In practice, Google Trends (www.google.com/trends/) shows a strong takeoff of OpenShift.

The third way to avoid lock-in is the development of applications based on a generic functional base. The developments on cloud platforms can be addressed by requiring the use of middlewares (or frameworks) to avoid the dependency to the differences between IaaS or PaaS-type cloud platforms (Kash, 2013; Zhang *et al.*, 2013). In terms of IaaS, Zhang et al. (2013) cite the existence of libraries that facilitate interoperability, such as Libcloud (libcloud.apache.org) and Deltacloud (deltacloud.apache.org). These libraries allow only the common features in the different supported platforms to be used. In terms of PaaS, the Simple Cloud API, built by Zend (www.zend.com) for its PHP framework (framework.zend.com) that

supports storage services including the Amazon and Microsoft offers, can be cited.

The lock-in can also be limited by developing key algorithms in a widely supported development language, such as Java, and by exploiting the rest the facilities offered by cloud computing providers (Coffee, 2012).

The fourth way to avoid lock-in is the use of specialized technical operators. Some specialized IT suppliers, called "technical cloud brokers", can help agencies avoid lock-in and operate several cloud services concurrently (Kash, 2013). They are similar to the "enablers" in Marston *et al.* (2011).

The fifth way to avoid lock-in is the trust in "open cloud" labels. The Foundation for a Free Information Infrastructure (www.ffii.org) proposed a definition of open cloud. This definition has three degrees of freedom: TIO (Total Information Outsourcing) Free / Open / Loyal (Scoffoni *et al.*, 2012). The TIO Loyal level "provides a framework to reach the same level of trade secret and operational transparency as with their own staff" (tio.ffii.org). The TIO Open level provides freedom of information and structuring of data in a clearly specified format. The TIO Libre level provides freedom of information, freedom of software and freedom of competition. However, the TIO label has a limited promotion at this stage (Viseur, 2013b). In addition, the guidelines for implementing and ensuring compliance with the conditions remain difficult to identify. Other similar proposals exist, such as the Open Cloud Principles of the Open Cloud Initiative (www.opencloudinitiative.org) or the Open Cloud Manifesto (www.opencloudmanifesto.org) (Jean, 2013).

The sixth way to avoid lock-in is the implementation of an exit strategy (McKendrick, 2011). The cost of the latter must incorporate the calculation of the costs for implementing the solution. The data migration should however be tested, not just discussed with vendors (Kash, 2013).

The support of open standards or their FLOSS implementations facilitates the implementation of an exit strategy. The latter may also rely on the existence of migration tools. Generally, these tools support the most popular solutions for virtualization and cloud computing such as VMWare, Amazon and OpenStack (Kash, 2013). With Amazon, for example, the importance of lock-in depends on the existence of conversion softwares (e.g. VM translations for EC2), the support for standard APIs (e.g. RDS compatible with Oracle or MySQL), alternative implementations (e.g. Elastisearch compatible with Memcache) or the existence of frameworks that provide a layer of abstraction (e.g. Zend Framework and Django for S3) (Chatzakis, 2012). Although migration solutions exist, the DynamoDB and SimpleDB NoSQL database services are a source of hard lock-in.

Note that the development of a realistic exit strategy assumes that the business has not been affected by the loss of skills that can result from outsourcing initiatives (Quélin, 2003). This loss can cause a handicap for a reversal or even the evaluation of alternative solutions.

3.6 Is the General Public Concerned with the Problem of Lock-In?

The portability of data encoded by Facebook users is a well-known problem. Facebook hinders the ability for users to export the list of friends to Google Plus, a rival social networks (Asay, 2011). It is possible to export the personal data in a downloadable archive (Protalinski, 2011). The latter is especially useful as a personal backup. Facebook improved its export system for developers by enriching data with microformats (hAtom, hmedia and hCard) (Protalinski, 2011). Exporting contacts is, however, the subject of numerous articles on the Web, which is proof, if any was needed, that the interoperability between social networks is limited.

Weinberger (2012) shows that the difficulty of data exit also arises for other consumer services. Flickr, for example, does not offer an officially supported method for migration. Flickr was further highlighted in 2006 by blocking the migration of photos submitted by their users to Zoomr competitor service. Its API, although functional, was rendered inoperable for contractual reasons: "*We choose not to support use of the API for sites that are a straight alternative to Flickr*" (Ozerman, 2006).

However, the consumer pressure led to the emergence of initiatives to facilitate the migration of data, such as Google Takeout Initiative (Weinberger, 2012). The latter is integrated to the Google Data Liberation Front (www.dataliberation.org) and allows the export of data managed by Google services towards standard data formats (*de facto* standards such as DOCX or open standards such as ODF). For example, a mail box can be exported to MBOX, one format that is supported by Mozilla Thunderbird or Microsoft Outlook.

4 DISCUSSIONS AND PERSPECTIVES

This preliminary research enabled the four different causes of lock-in to be highlighted, four impacts of lock-in on the users to be identified and six mechanisms to reduce the risk of lock-in to be proposed. Moreover we showed that the problem of lock-in also had many similarities for professionals and individuals.

Our research focused on a set of dominant worldwide providers. It therefore deserves an extension to niche suppliers (e.g. Ikoula). Their positioning face to dominant players could be highly instructive.

The issue of lock-in is as old as the existence of computers. The issue of data formats in the field of productivity software is a well-known illustration (Adatto, 2013). However, the problem has an additional dimension in the case of cloud computing. Indeed, the lock-in in the cloud not only causes difficulties in terms of evolution of the service (for example, if the pace of innovation offered by the supplier decreases) but also has an increased risk in terms of continuity of service. In practice, a software solution that is installed on a local network by a publisher in bankruptcy can be used for quite some time by the company. A bankrupt cloud provider or a cloud provider that bases its service on a bankrupt cloud infrastructure provider causes greater difficulties to its customers due to the inaccessibility of the service. This point justifies the high visibility of the topic in the IT press.

This research allowed us to identify a vocabulary associated with the issue of (vendor) lock-in. This includes, in particular, the following expressions: (1) portability, compatibility and especially interoperability, (2) exit strategy (3) increasing returns to adoption, and (4) outsourcing. Research about the studies associated with these expressions could shed additional light on the issue of lock-in in the cloud services.

The standardization that guarantees the interoperability between platforms of cloud computing emerges as a major track to avoid lock-in. However, the existence of standards does not solve all problems. The issues of functional coverage of standards, industrial support of standards, coverage of standards by implementations and low success “open cloud” labels remain open questions.

Table 1: Support of *de facto* and open standards in open source projects.

	OpenStack	Eucalyptus	OpenNebula	CloudStack
OVF	Yes	N/A	Yes	N/A
CDMI	Yes	N/A	Yes	N/A
OCCI	Yes	Yes	Yes	Yes
AMI	No	Yes	N/A	N/A
S3 API	Yes	Yes	No	Yes
EC2 API	Yes	Yes	Yes	Yes

Zhang *et al.* (2013) provide an initial response in the case of IaaS regarding the functional coverage of standard and the coverage of the standards by implementations. A complementary study, based on Zhang *et al.* (2013), snia.org, dmtf.org, occi-wg.org, openstack.org, opennebula.org and cloudstack. apache.org, is proposed in Table 1. Further work is necessary, for example, to estimate the actual level of software compatibility with the listed standards. Given the subtlety of some causes of lock-in (e.g. possibility or not to migrate IP addresses, to change the DNS, etc.), the granularity of the specifications should also be deepened. More investigation on the issue of industrial standards support is also needed. For example, it could be estimated through hit counts of search engine results (webometrics), as is already done for the estimation of market shares (Viseur, 2012; Viseur, 2013a). The thinking should be extended to PaaS, for which standardization initiatives have been emerging.

The particularization of the issue of lock-in based on the model of provision (IaaS, PaaS or SaaS) seems difficult at this stage. On the one hand, although the literature demonstrates a tendency to assign a higher risk of dependence in the case of PaaS, multiple initiatives for standardization and availability of FLOSS implementations (e.g. OpenShift) have been considerably changing the situation. On the other hand, it became apparent that the decomposition between IaaS, PaaS and SaaS providers, if it facilitated the understanding and the analysis, was sometimes artificial. Indeed, IaaS labeled platforms like Amazon make services that are typically associated with PaaS platforms available (e.g. NoSQL database). In addition, a platform can be clearly straddling two modes of provision. For example, Facebook will be considered a SaaS provider with instant messaging available to users, or PaaS provider if the developer who contributes to the application store is taken into consideration. Idem with Saleforce.com (SaaS) and its complement Force.com (PaaS).

REFERENCES

Scientific References

L. Adatto, “Standards ouverts et implémentations FLOSS (Free Libre Open Source Software) : vers un nouveau modèle synergique de standardisation promu par l’industrie du logiciel,” in Terminal : Technologie de l'Information, Culture, Société, n°113-114, pp. 137-170, 2013.

R. Chow, P. Golle, M. Jakobsson, et al. , “Controlling data in the cloud: outsourcing computation without outsourcing control,” in Proceedings of the 2009 ACM workshop on Cloud computing security, ACM, pp. 85-90, 2009.

D. Foray, “Les modèles de compétition technologique. Une revue de la littérature”, in Revue d'économie industrielle, Vol. 48, n° 1, 1989, pp. 16-34.

P. Harsh, F. Dudouet, R.G. Cascella, Y. Jegou and C. Morin, “Using open standards for interoperability: issues, solutions, and challenges facing cloud computing,” in 8th International Conference on Network and Service Management (CNSM), IEEE, 2012, pp. 435-440.

B. Jean, “Impact of cloud computing on FOSS users,” in Conférence EOLE (European Open source & free software Law Event), Bruxelles, 6 décembre 2013.

S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, “Cloud computing - The business perspective”, in Decision Support Systems, 51(1), pp. 176-189, 2011.

D. G. Messerscmitt, C. Szyperski, “Industrial and Economic Properties of Software: technology, processes and value,“ in Computer Science Division, University of California, 2001.

C. Pahl, L. Zhang and F. Fowley, “Interoperability standards for cloud architecture,” in Closer 2013, Germany, May 2013.

S. Pearsin, “Taking account of privacy when designing cloud computing services,” in ICSE Workshop on Software Engineering Challenges of Cloud Computing (CLOUD'09), IEEE, pp. 44-52.

B. Quélin, “Externalisation stratégique et partenariat: de la firme patrimoniale à la firme contractuelle ?,” in Revue française de gestion, n° 2, 2003, pp. 13-26.

P . Scoffoni, E. Ogez, L. Dubost, “L’open cloud : garder la maîtrise de son système d’information,” in Méthodes, techniques et outils, Documentaliste-Sciences de l'Information, 2012/2 (Vol. 49), pp. 8-15.

R. Viseur, “Les moteurs de recherche commerciaux sont-ils des outils de webométrie fiables ?,”, in 30ème congrès InforSID, Montpellier (France), 29-31 mai 2012.

R. Viseur, “Estimate the Market Share from the Search Engine Hit Counts,” in Proceedings of International Conference on Data Technologies and Applications 2013, Iceland, July 29-31, 2013.

R. Viseur, “Evolution des stratégies et modèles d’affaires des éditeurs Open Source face au Cloud computing,” in Terminal : Technologie de l'Information, Culture, Société, n°113-114, 2013, pp. 173-193.

Professional References

M. Asay, “Facebook, Google, and the war to lock you in,” in The Register, 5 juillet 2011 [Online]. Available: http://www.theregister.co.uk

C. Babcock, “10 Tools To Prevent Cloud Vendor Lock-in”, in InformationWeek, 14 février 2013 [Online]. Available: http://www.informationweek.com/

A. Chatzakis, “How I Learned not to fear Amazon Cloud Lock-in,” in Newvem, 28 mai 2012 [Online]. Available: http://www.newvem.com

CIGREF (2013), “Fondamentaux du cloud computing,” Cigref, 2013 [Online]. Available: http://www.cigref.fr

P. Coffee, “Refuting Cloud 'Lock-In': Zero, One, Too,” in Blog Network Salesforce.com, 18 janvier 2012 [Online]. Available: http://cloudblog.salesforce.com

M. Coté, “SalesForce-to-Google and the Force.com PaaS Lock-in Question,” in Coté's People Over Process / RedMonk, 24 juin 2008 [Online]. Available: http://redmonk.com/

A. Crochet-Damais, “Marché du CRM : Salesforce passe devant SAP,” in JDN, 29 avril 2013 [Online]. Available: http://www.journaldunet.com/

B. Darrow, “Fear of lock-in dampens cloud adoption,” in GigaOM, 26 février 2013 [Online]. Available: http://gigaom.com

J.M. Germain, “How to Avoid Cloud Vendor Lock-In,” in LinuxInsider, 13 novembre 2013 [Online]. Available: http://www.linuxinsider.com

G. Gruman, “Is Open Source the Answer to ERP ?,” in CIO, February, 2007, Vol. 20, pp.23-28.

D. Harris, “Google defends, quantifies App Engine lock-in concerns,” in GigaOM, 9 juillet 2013 [Online]. Available: http://gigaom.com

D. Juengst, “How to Avoid Cloud Vendor Lock-In when Evaluating a PaaS,”, in OpenShift, 3 août 2012 [Online]. Available: https://www.openshift.com

W. Kash, “Avoiding Cloud Lock-in,” in Information Week Government, Novembre 2013 [Online]. Available: http://www.informationweek.com

D. Linthicum, “2 more cloud myths busted: Lock-in and locked up.,” in InfoWorld, 27 avril 2012 [Online]. Available: http://www.infoworld.com

P. Magnusson, “Google App Engine: Lock in, what lock in ?,” in VB VentureBeat, 25 juillet 2013 [Online]. Available: http://venturebeat.com

R. Malhotra, “The Truth About Lock In,” in CloudAve, 13 février 2013 [Online]. Available: http://www.cloudave.com

J. Mckendrick, “Cloud Computing's Vendor Lock-In Problem: Why the Industry is Taking a Step Backward,” in Forbes, 20 novembre 2011 [Online]. Available: http://www.forbes.com

O. Nachmani, “The Great PaaS Lock-In,” in DZone, 29 mai 2012 [Online]. Available: http://cloud.dzone.com

R. Ozerman , “Why is Flickr afraid of Zooomr?,” in Techcrunch, 16 juin 2006, [Online]. Available: http://techcrunch.com

E. Protalinski, “Facebook finally makes your exported data useful,” in ZDNet, 8 septembre 2011 [Online]. Available: http://www.zdnet.com

M. Szynaka, “Ask the Expert: Is PaaS vendor lock-in unavoidable ?,” in SearchCloudComputing, 04 avril 2013 [Online]. Available: http://searchcloud computing.techtarget.com