Cloud Computing: A Multi-tenant Case Study

Anindya Hossain and Farid Shirazi(✉)

Ryerson University, Toronto, Canada
{anindya.hossain,f2shiraz}@ryerson.ca

Abstract. Cloud computing has enabled businesses to infinitely scale services based on demand while reducing the total cost of ownership. Software as a service (SaaS) vendors capitalized on the scalable nature of Infrastructure as a Service (IaaS) to deploy applications without having the need for heavy upfront capital investment. This study uses a real case study from a Canadian SaaS vendor migrating from a single-tenant applications system to a single-tenant applications (MTA) system. The results of this empirical study show a decrease of a factor of 3 in setup times and a reduction in number of bugs reported and the amount of time required to fix these bugs. Despite the fact that migration from a single-tenant applications system to a multi-tenant system requires some re-engineering efforts, but the benefits of MTA far outweigh these re-engineering costs. Furthermore, migrating to MTA enables firms to focus on their core competences. The empirical results of this study show that in the long run, MTA can enable SaaS vendors to increase the quality of service, performance, service level agreement adherence, re-focus on creating innovative products, lower operational costs and earn higher profits.

Keywords: Cloud computing · Multi-tenant application · SaaS · IaaS · PaaS · ANOVA

1 Introduction

Cloud computing has truly revolutionized the IT industry in the last decade. The internet has evolved from a mere medium of communication to a medium of delivery for software, middleware platforms and hardware infrastructure. Enterprises no longer need to own their hardware to host applications, or manage and maintain software on client computers. The recent advent of cloud computing has made computing into a utility. Cloud computing has enabled businesses to infinitely scale services based on demand while reducing the total cost of ownership. Software as a service (SaaS) vendors capitalized on the scalable nature of Infrastructure as a Service (IaaS) to deploy applications without having the need for heavy upfront capital investment.

Most SaaS applications such as email and CRM applications are very standard, i.e. they all perform very similar tasks regardless of the organization using the system. Some SaaS applications need to be somewhat customized for each client. Initially, SaaS vendors started running independent application instances for each client (single-tenant applications). However, when these SaaS vendors started to scale, they ran into problems maintaining and setting up a separate instance for each client. They started developing multi-tenant applications (MTA), which can handle multiple tenants

© Springer International Publishing Switzerland 2015
M. Kurosu (Ed.): Human-Computer Interaction, Part III, HCII 2015, LNCS 9171, pp. 178–189, 2015.
DOI: 10.1007/978-3-319-21006-3_18

within the same application. Earlier studies confirmed that an MTA would enable firms to reduce their operating costs and reduce the number of hours spent doing maintenance. However, there was no concrete industry data to prove these theories.

Online implementation and delivery of software, more commonly known as Software as a Service (SaaS), has become more popular than ever before. Dubey and Wagle (2007) claim that SaaS vendors are significantly less profitable than traditional software vendors [1]. They speculate that this might be due to the lack of scale for SaaS. For example, WebEx, one of the largest online meeting and collaboration SaaS, has a profit margin that is almost double the SaaS industry average [1].One strategy to increase scale is to adopt a multi-tenant architecture for SaaS. In addition, multi-tenant systems are often set up in a single database to reduce the total cost of ownership [2].

Just like any other business, SaaS providers need to focus on their core competencies and continually innovate in order to be competitive in their industries. These firms have a finite set of resources and these resources need to be properly managed to maximize dynamic capabilities. In a multi-tenant environment, the software service provider runs a single instance of the application that is configured for each client (tenant). It typically runs on a single database and shares hardware resources. The goal of this research will be to demonstrate that multi-tenant setups can enable firms to focus more on their core competencies and not spend most of their resources on implementations. By focusing on their core competencies, SaaS vendors can continue to innovate and offer better products that are able to return more profit. The aim of this empirical study was to analyze the extent to which multi-tenant systems can be effective in delivering software services on-demand to various clients. To this end, we investigated two issues: a) does the setup of the multi-tenant system require less time to implement than the single-tenant system and b) is a multi-tenant setup more robust and does it require less maintenance time?

GrantStream Inc. a SaaS vendor in Canada made a strategic decision last year to move away from custom single-tenant applications towards a multi-tenant application (MTA) with a single instance of the application running on a shared database and hardware setup. Data was collected from 9 setups in legacy single-tenant applications, 9 new setups in MTA and 9 migration setups in MTA.

This case study will enable us to use real industry data to validate our research questions. The benefits and challenges of migrating to an MTA have been discussed by many researchers. However, none explored the tangible or monetary benefits of this migration. Momm and Krebbs were the first to suggest a simple cost model to evaluate MTA migration for a SaaS vendor [4]. These research findings needed to be validated using real-world data, and this case-study will enable us to validate claims made by other researchers.

2 Cloud Computing Multi-tenant Architecture

An evolution in Internet technology, cloud computing is an advancement providing users with the means to access a wide range of computing power, software and platform as a service, as well infrastructure anytime, anywhere [5]. Cloud computing enables on-demand network access to a shared pool of configurable computing

resources, including servers, storage applications and services. Cloud computing services encompass the following three main layers: the hardware infrastructure (IaaS), middleware services (PaaS) and application services (SaaS). Cloud computing capabilities have enabled businesses to offer services that seem to be infinitely scalable and elastic. Subscribers of cloud services can keep their upfront costs low by adjusting their level of service based on demand. It allows companies to start small and increase incrementally with demand. Cloud computing has made IT resources into a utility that is enabling businesses to enter the markets without the need for heavy initial capital investment [6].

SaaS is a web-based software application associated with business software applications deployed and operated as a hosted service [7]. These applications typically run in the browser of a client (tenant) and can be accessed from anywhere. SaaS applications can range from a simple web mail application to a more complex CRM application [8]. They explain that SaaS vendors started allocating multiple clients on the same application to increase efficiency and reduce Total Cost of Ownership (TCO). They define multi-tenancy as: "an approach to share an application instance between multiple tenants by every tenant a dedicated 'share' of the instance, which is isolated from other shares with regard to performance and data privacy" [10:2]. SaaS providers are usually required to adhere to certain standards set out in their service level agreement (SLA). To be successful, providers need to ensure that the services are scalable on-demand, comply with SLA terms, enable customers to achieve low TCO and have low incremental costs [4]. They mention that for current SaaS providers enabling multi-tenancy is the next big evolution step; it can help them achieve a lower TCO by reducing setup and maintenance costs [4]. Bezemer and Zaidman write that multi-tenancy architecture enables service providers to reach economies of scale by sharing the same instance of the application and database. The application can be configured for each tenant, and it may appear as a custom solution to the client. In addition, multi-tenancy can enable SaaS providers to increase hardware utilization and reduce costs [9].

Multi-tenant setups can have various configurations. Pervez, Lee and Lee categorize SaaS into four maturity levels. The first level is "*custom/ad hoc",* where the SaaS application is fully customized or built for a specific client [10]. At the second level, "*Configurable"*, the application is configured for a specific client, and an independent instance of the application is used for each tenant. At the third level, "*Configurable & multi-tenant efficient"*, a single instance of the application is used by multiple clients and it is somewhat configured for each tenant. At the fourth level, "*Scalable, Configurable, multi-tenant-efficient*", a single instance of the application is used as in the third level, but the services are fully configurable to handle a specific business workflow. In this last level, services are also fully scalable and the focus is to meet or exceed all requirements mentioned in the SLA [10].

3 The Key Characteristics of a Multi-tenant Platform

3.1 Sharing Hardware Resources

Sharing hardware resources among multiple tenants reduces costs for the SaaS provider. Bezemer and Zaidman mention that even though server efficiency can be improved in single-tenant setups through virtualization, it imposes however, a high memory requirement for each virtual setup [11].

They explain about the different types of hardware, software combinations for a multi-tenant setup:

1. Single instance of application shared among tenants each with a separate database
2. Single instance of application with shared database, but separate Tables
3. Single instance of application with shared database and shared tables (pure multi-tenancy)

Requires High Degree of Configurability. Unlike single-tenant SaaS, multi-tenant setups cannot be completely customized since they are shared among different tenants. Therefore, it is absolutely necessary for multi-tenant software to be highly configurable. It needs to accommodate each client's settings, workflows among other [9]. They explain that in a typical single-tenant setup, updates are usually made by creating branches in the development tree, but this does not work in multi-tenant setups and configuration needs to be part of the product design itself. Other researchers propose building a workflow engine on top of a multi-tenant application instance [12]. SaaS like Email and CRM has the same workflow irrespective of which company is using the service. The service is standard across the board. However, for most SaaS, the workflow changes from client to client (tenant). As mentioned earlier, a key challenge in a multi-tenant environment was to provide a configurable setup for each tenant. Pathirage et al. propose a configurable "Workflow as a Service" (WFaaS) that will be added onto the existing multi-tenant SaaS and will allow the application to be configured for each tenant [13].

Easier Deployments. In multi-tenant setups deployments and updates can be pushed out easily as there is ideally only one instance of the program running. In some cases, a provider might have multiple instance of the application running, but it will almost always be lower than any single-tenant setups [11] Short setup times are a key requirement for SaaS clients. They expect SaaS vendors to be able to set them up in days instead of weeks or months [4]. SaaS vendors are required to agree to a Service Level Agreement which stipulates the expected setup times. If the SaaS provider cannot meet these tight deadlines, they might lose potential clients. Therefore, if multi-tenant setups can enable SaaS vendors to reduce setup time, it can lead to a competitive advantage.

3.2 Challenges of a Multi-tenant Platform

Multi-tenant setups face some challenges that are more complex than similar challenges faced in single-tenant setups. Multi-tenant setups share hardware infrastructures, databases, middleware, and the application itself. When issues arise, they cannot be

contained within a certain client; they affect all clients. Bezemer and Zaidman write that the main concerns in multi-tenant setups are performance, scalability, security, downtime and maintenance. Before making the migration from a single-tenant setup to a multi-tenant setup, SaaS providers need to fully understand these complexities [11].

3.3 Cost of Migration from Single-tenant to Multi-tenant

The cost of migrating from a single-tenant to a multi-tenant setup can vary depending on the complexity of the software and its underlying architecture. Momm and Krebs developed a cost model that includes two major components: *Initial reengineering costs* and *Continuous operating costs*. They explain that any savings in operating costs will amortize the initial reengineering costs over a certain period of time, the break even period. Continuous operating costs can be calculated by evaluating fixed costs of infrastructure, middleware, maintenance, etc. They argue that introducing an additional shared resource or component in the stack saves Operating costs of (n-1) times the base costs for the shared resource [4].

Momm and Krebbs propose a simple cost model to evaluate migration to a multi-tenant architecture [4].

Months to break-even = Initial re-engineering costs / Savings in operating costs.

4 Research Approach and Theoretical Framework

Most research in this field of using single-tenant vs. multi-tenant SaaS setups is still very theoretical. There is a body of research that suggests a multi-tenant setup might be better for a large number of clients, but it does come with its own set of challenges [12]. However, none of these studies have looked at the existing SaaS industry to see how they have handled these challenges. Migrating from a single-tenant to multi-tenant setup will always require some re-engineering work as mentioned earlier, but the benefits of migration can outweigh the costs.

This research will enable SaaS vendors to decide which type of architecture better suits their strategies. Earlier studies have suggested that multi-tenant setups enable SaaS vendors to quickly set up and maintain new tenants.

GrantStream Inc. is a Canadian SaaS vendor specialized in grant management software. The company was founded in 2001 and is one of the earliest providers of SaaS in the grant management software industry. The company focuses mainly on providing services to small and medium size enterprises (SMEs). Tehrani and Shirazi argue that despite the fact that adopting new technologies helps SMEs gain a competitive advantage, it usually involves high costs. Cloud computing, as a new computing paradigm, offers many advantages to companies, especially smaller ones. Flexibility, scalability, and reduced cost are among many advantages that cloud computing offers to SMEs [14].

Before the migration to the multi-tenant setup, GrantStream was one of the few providers who provided custom SaaS applications in the grant management industry. The company felt that their ability to customize their application would give them

competitive sustainability. However, the management discovered that as they spent more and more time on client setups, their core product did not receive the updates and innovations it required to differentiate it from the competition. They discovered that their product was falling behind, and to remain competitive they would need to redirect their resources towards product development rather than tenant setups.

Looking at the impact of this change in strategy within this organization will help us confirm the different theories that have been presented by earlier papers. Data from this case study might not be representative of the population, but can confirm some of the theories presented by other researchers. We will do a quantitative data analysis on tenant setup times for:

1. Single-tenant setup
2. Migrate legacy client to multi-tenant
3. New setup in multi-tenant platform

The number of hours spent are recorded by the firm and are presented during the post-project review. In addition, we will also explore the number of reported bugs in the last 12 months for that specific tenant and how many hours were spent fixing these bugs. The data analysis will help us understand how the migration from single-tenant to multi-tenant architecture affected the setup times and bug resolution time. As mentioned earlier, these are key factors in SaaS vendors' SLA terms.

4.1 Single-tenant Setup at GrantStream

As shown in Fig. 1, GrantStream's legacy setup (single-tenant) used a shared database and tables, and ran on a shared hardware setup, as well. By sharing resources, GrantStream could keep its costs fairly low since the company was founded in the early 2000 s. However, having multiple instances of the application running for each tenant made maintenance very hard. In addition, since setting up a new tenant included a custom setup, it was very time-consuming and cumbersome.

4.2 Multi-tenant Setup at GrantStream

GrantStream's multi-tenant architecture uses a single instance of the application, with an underlying shared database and tables, which runs on a shared infrastructure (see Fig. 2). GrantStream uses an IaaS service provider data center to host its SaaS system. By subscribing to this provider GrantStream can minimize costs and increase scalability without adding to operating costs. GrantStream's application needed to be re-engineered to enable it to handle multiple tenant configurations. GrantStream Inc.'s multi-tenant application has reached level 3 of the SaaS maturity model as mentioned earlier [10].

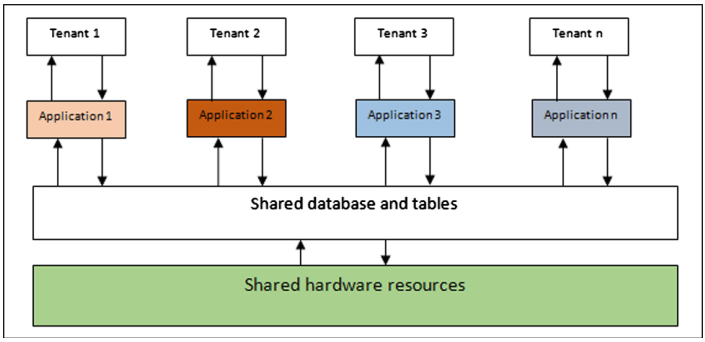

Fig. 1. Single-tenant setup at GrantStream Inc

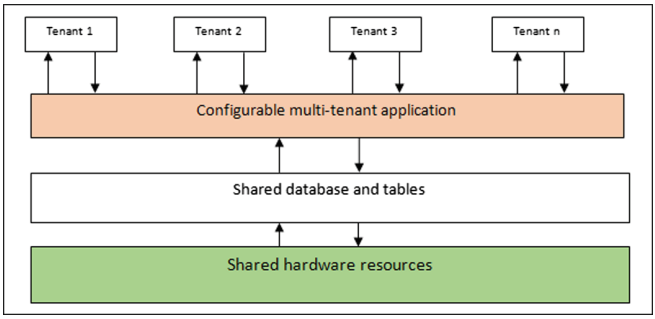

Fig. 2. Multi-tenant setup at GrantStream Inc

4.3 The Research Methodology

Data was collected for 27 client setups during the post-project reviews, and through the logged service tickets. As shown in Table 1 below, we collected number of hours to set up a tenant from post-project reviews (variable HourToSetup). We also looked at logged tickets in the last 12 months for each tenant (variable BugTickets) and the amount of time required to fix each bug (variable HoursFixBug).

Of these 27 setups, 9 were during the single-tenant phase (legacy), 9 were new setups using the multi-tenant application, and 9 were client migrations from the single-tenant to multi-tenant platform (as shown in the column, TypeOfSetup in Table 1). Setup times and hours spent on bug fixes are recorded by GrantStream during a new client implementation project. They shared the data with us for this research, but to protect the confidentiality of the clients, the names of the company's clients were not disclosed. This does not affect the results of the study.

Table 1. Data collected from GrantStream Inc.

Client name	TypeOfSetup	HourToSetup	BugTickets	HoursFixBug
BPE	1	98.58	0	0
CAN	1	79.38	1	1.5
GSC	1	117	4	2.36
SHL	1	102.7	2	1.76
MAL	1	27.65	1	0
SHU	1	77.38	0	0
STA	1	67.92	2	0
TAL	1	59	2	0.42
SLF	1	51.08	5	3.61
SH	2	57	2	1.42
AFF	2	41.72	4	0.8
ALL	2	211	4	6.5
AME	2	269	4	3.9
CPC	2	18	3	5.75
GE	2	37	1	0.33
LDC	2	63.57	4	1.2
PMV	2	128.68	0	0
SHW	2	125.77	2	3.16
MSC	3	240.6	12	20.8
BNC	3	467.45	2	2.25
PHR	3	347.83	1	1.6
UFA	3	110	0	0
IOXM	3	113	3	0.7
IOXMR	3	178	3	1.9
ENB	3	268	2	4.5
VLE	3	195	4	3.6
HDE	3	164.65	5	9.5

Type of setup: 1 = Migration to multi-tenant, 2 = New setup in multi-tenant, 3 = New setup in single-tenant

5 Results

We completed a statistical data analysis with the data provided by GrantStream Inc. As presented earlier, researchers like [4, 11] wrote about the many benefits of migrating from the single-tenant architecture to a multi-tenant structure. However, these findings were never confirmed using empirical data. This research will try to answer two hypotheses previously mentioned: Multi-tenant clients need less time to setup and Multi-tenant setups require less time to maintain. To test these hypotheses, we collected data from 27 setups. Data was collected from 9 setups when the company was using a single-tenant setup. After the migration, we collected data from 9 new setups in a multi-tenant environment and 9 migration setups in the new platform.

Hypothesis 1: Multi-tenant Setups Require Less Time to Implement Than Single Tenant. GrantStream Inc. spent on an average almost 232 h implementing a new tenant in the single-tenant architecture of their application. After they migrated to a multi-tenant architecture, setup times dropped significantly. For new setups, they started spending close to 106 h, and for migrating old clients they spent around 76 h. However, both new setups in the legacy and multi-tenant environments have high standard deviations of 116 and 86, respectively. Compared to new setups, migration setups have a lower standard deviation of 28. New setups typically always have unknown customizations and idiosyncrasies which can cause the setup times to fluctuate more. For migration setups, the client and their setup are already known to the company and setup times are more consistent. As shown in Table 3 below, a one-way ANOVA was performed on the dataset with the *TypeOfSetup* as the independent variable. We have discovered that the *HourToSetup* has a high F of 8.5. *HourToSetup* has a high variance between groups and low variance within the group, which means that the *HourToSetup* is highly correlated with the independent variable, *TypeOfSetup*. Therefore, we can conclude that by migrating from a single-tenant to a multi-tenant architecture did reduce the number of hours required to setup a client.

As indicated in Tables 2 and 3 below, the cloud clients who migrated to a multi-tenant system experience the highest possible system performance and efficiency as measured by the number of hours spent on bugs to be fixed annually. In fact, this performance, as shown in the table, is more than 4 times higher than that for a single-tenant setup. Another important finding is that setting up a new multi-tenant system is more than 1.9 times more efficient than a single-tenant structure. The findings indicate that the cloud multi-tenant system is more efficient than the traditional single-tenant setup formats. When GrantStream Inc. was using a single-tenant application, the system needed to be customized for each client. The company could not capitalize on its learning from previous setups. After migrating to a multi-tenant architecture, there was no need to create a new custom application for each tenant. New setups were more efficient in handling different workflows of different clients. Implementation became standardized, and as a result, setup times dropped significantly.

Hypothesis 2: Multi-tenant Application Requires Less Maintenance Time. As shown in Table 2, we collected data from 27 tenants during the last 12 months to compare the number of bugs reported and the amount of time spent fixing these bugs. As previously mentioned, out of the 27 tenants, 9 are in the legacy single-tenant application, 9 were new tenants in the multi-tenant system and 9 were migrated from the single-tenant to the multi-tenant system as depicted in Table 3. Table 3 indicates also that tenants migrated to multi-tenant system had the lowest number of bugs reported with an average of 1.9 for the year. As for the two other groups, new setups in the multi-tenant system had an average of 2.7 per year and clients in the legacy platform averaged about 3.6 per year. However, for all three groups the variance ranged from 1.5 to 3.7 (shown in Table 2), which means that the number of bugs reported varied significantly from client to client and there was no significant difference between the tenants in one group compared to another. In addition, ANOVA with SetupTypes set as the independent variable indicated that the number of BugsReported had an F of 1.080 (see Table 3), which means that the number of bugs reported is not correlated

with the type of architecture being used by the SaaS vendor. However, when we analyzed the amount of time it took to fix these bugs, FixBugHours, the results were significantly different. Only 1 to 2.5 h per client were spent in the group where the multi-tenant setup was used. In the legacy group, the company was spending close to 5 h per tenant per year to fix bugs reported. FixBugHours had F of 2.085 when analyzed with SetupType as the independent variable. This reduction in the number of hours spent fixing bugs can be attributed to the standard setup of a multi-tenant application instance. Bug fixes are rolled out universally and any fix is applied for all clients simultaneously. Whereas in the past each application for a client needed to be fixed individually, in multi-tenancy fixes are rolled out once and apply to all tenants. Therefore, by migrating from single-tenant architecture to a multi-tenant one, the company is saving almost 2.5 to 4 h per client per year. If a SaaS vendor has 100 clients, they would be saving almost 250 to 400 h from their operating expenses.

Table 2. Statistical Report

SetupType		HourToSetup	BugTickets	HoursFixBugs
1	Mean	75.6322	1.89	1.0722
	N	9	9	9
	Std. Deviation	27.90357	1.691	1.31238
2	Mean	105.7489	2.67	2.5622
	N	9	9	9
	Std. Deviation	86.12265	1.500	2.38821
3	Mean	231.6144	3.56	4.9833
	N	9	9	9
	Std. Deviation	116.23679	3.504	6.56049
Total	Mean	137.6652	2.70	2.8726
	N	27	27	27
	Std. Deviation	106.85945	2.415	4.26913

Table 3. Analysis of variance (ANOVA)

		Sum of Squares	df	Mean Square	F	Sig.
HourToSetup	Between Groups	123238.816	2	61619.408	8.516	.002
	Within Groups	173653.700	24	7235.571		
	Total	296892.516	26			
BugTickets	Between Groups	12.519	2	6.259	1.080	.356
	Within Groups	139.111	24	5.796		
	Total	151.630	26			
HoursFixBugs	Between Groups	70.136	2	35.068	2.085	.146
	Within Groups	403.727	24	16.822		
	Total	473.863	26			

5.1 Cost Model of Migrating to a MTA

As mentioned earlier, Momm and Krebbs proposed a simple cost model to evaluate migration to a multi-tenant architecture [4].

Months to break-even = Initial re-engineering costs/Savings in operating costs

Using the data provided above we can estimate the number of months required for a SaaS vendor to break even given these assumptions:

- Re-engineering efforts for the application cost the company $100,000
- The vendor has approximately 100 tenants
- The vendor completes 1 new tenant setup every month
- Internal development and setup cost the company $100/hour

Cost savings in doing one setup = 232 h – 76 h * $100 = $15,600

Cost savings from bug tickets = [(5 h * 100) – (3.25 h * 100 clients)]* $100 = $17,500/year

Months to break even = $100,000/[($15,600) + ($17,500/12) = 5.86 months

As shown above, a SaaS vendor can migrate to a MTA and break even fairly quickly. After the break-even period, the company will continue benefiting from costs savings. Instead of allocating much of its resources developing and maintaining customized applications, vendors can now redirect their resources towards their core competencies and focus more on developing a technological lead that will enable them to sustain a competitive advantage. As mentioned earlier, Wernerfelt recommends that firms engage their employees in stimulating jobs that create more value for the firm's products [3]. Therefore, by migrating to a multi-tenant application, SaaS vendors are not only saving on operating costs, they also have the opportunity to free up resources that can be focused on creating more innovative products and helping the firm earn higher returns.

6 Limitations and Conclusions

The software industry is constantly changing. Software vendors might find a better way to deliver their SaaS that can handle the simplicity of implementing a multi-tenant setup while maintaining the customization of single-tenant setup. The goal of this study was to investigate claims made in previous research through empirical data. This does not mean that the findings will be universal.

The data collected in this research might not be representative of the population. There have been other studies that made certain claims about multi-tenant setups, but they were never proven with empirical data. This research has tried to investigate some of those claims through data collected from a real organization. This organization has been running a single-tenant application for almost 10 years, but recently changed strategies and developed a multi-tenant application. The findings of this research might not apply to all SaaS providers. As with all technological innovations, the software industry is always changing.

In conclusion, cloud computing has truly revolutionized the IT industry. The software industry dreamed of commoditizing computing for a long time, but it did not

become a reality until the last decade. The internet has enabled firms to deliver software, middleware platforms and hardware infrastructure. Enterprises no longer need to own their hardware to host applications, or manage and maintain software on client computers. Cloud computing has enabled businesses to infinitely scale services based on demand while reducing the total cost of ownership. SaaS vendors capitalized on the scalable nature of IaaS to deploy applications without having the need for heavy upfront capital investment.

References

1. Dubey, A., Wagle, D.: Delivering software as a service. The McKinsey Quarterly **6**, 1–12 (2007)
2. Aulbach, S., Grust, T., Jacobs, D., Kemper, A., Rittinger, J.: Multi-tenant databases for software as a service: schema-mapping techniques. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1195–1206. ACM, June, 2008
3. Wernerfelt, B.: A resource-based view of the firm. Strateg. Manag. J. **5**(2), 171–180 (1984)
4. Momm, C., Krebs, R.: A qualitative discussion of different approaches for implementing multi-tenant SaaS offerings. In: Software Engineering (Workshops), vol. 11 (2011)
5. Pallis, G.: Cloud computing – the new frontier of internet computing. IEEE Internet Comput. **14**(5), 70–73 (2010)
6. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Zaharia, M.: A view of cloud computing. Commun. ACM **53**(4), 50–58 (2010)
7. Kwok, T., Thao, N., Linh, L.: A Software as a service with multi-tenancy support for an electronic contract management application. IEEE Int. Conf. Serv. Comput. SCC **2008**, 179–186 (2008)
8. Krebs, R., Momm, C., Kounev, S.: Architectural concerns in multi-tenant SaaS applications. In: CLOSER, pp. 426–431, April, 2012
9. Bezemer, C. P., Zaidman, A., Platzbeecker, B., Hurkmans, T., & t Hart, A. (2010, September). Enabling multi-tenancy: An industrial experience report. In*Software Maintenance (ICSM), 2010 IEEE International Conference on* (pp. 1–8). IEEE
10. Pervez, Z., Lee, S., Lee, Y.K.: Multi-tenant, secure, load disseminated SaaS architecture. In: The 12th International Conference On Advanced Communication Technology (Icact), 2010, vol. 1, pp. 214–219). IEEE February, 2010
11. Bezemer, C.P., Zaidman, A.: Multi-tenant SaaS applications: maintenance dream or nightmare? In: Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), pp. 88–92. ACM, September 2010
12. Pathirage, M., Perera, S., Kumara, I., Weerasiri, D., Sanjiva Weerawarana, S.: A scalable multi-tenant architecture for business process executions. Web Serv. Res. **9**(2), 21–41 (2012)
13. Hay, B., Nance, K., & Bishop, M. (2011, January). Storm clouds rising: security challenges for IaaS cloud computing. In System Sciences (HICSS), 2011 44th Hawaii International Conference on (pp. 1–7). IEEE
14. Tehrani, S.R., Shirazi, F.: Factors influencing the adoption of cloud computing by small and medium size enterprises (SMEs). In: Yamamoto, S. (ed.) HCI 2014, Part II. LNCS, vol. 8522, pp. 631–642. Springer, Heidelberg (2014)