LUT UNIVERSITY

LUT School of Energy Systems

LUT Software Engineering

Master's Thesis
2021

Anukesh Singh Chauhan

DEPENDENCIES OF PRICING MODELS AND SAAS ARCHITECTURES: A LITERATURE STUDY

Examiner(s): Kari Smolander, LUT

Supervisor(s): Andrey A. Saltan, LUT

ABSTRACT

LUT University

LUT School of Engineering Science

LUT Software Engineering

Anukesh Singh Chauhan

Dependencies of Pricing Models and SaaS Architectures: A Literature Study

Master's thesis
Year of Completion 2022

54 pages, 23 figures, 8 tables

Examiners: Professor Kari Smolander, LUT

Supervisors: Andrey A. Saltan, LUT

In these modern times, SaaS software might have many different pricing models depending on the characteristics of the architecture. Despite the fact that SaaS is a success which is due to the combination of these two factors, architecture and pricing models. Not much study has been found regarding the relation between architecture and pricing models.

Since there is a lack of answers in this area. The objective of this study is to figure out whether SaaS architecture and pricing structures are related. Explaining how SaaS architecture affects pricing models and how pricing models effect SaaS architecture.

A multivocal literature review (MLR) was used in this thesis as a research method by exploring "grey" literature and "white" literature as well. A total of 70 bibliography was collected, of which 28 were grey, and 42 were white literature items. All the information

from these items was collected and this tried to answer the relation between architecture and pricing models.

SaaS architecture and pricing models are found to be related closely to each other as they both have their own importance. A rock-solid proof was not found that proves that proves that these are always related to each other and are dependent on one another. They are somewhat important to each other as having a well-designed architecture helps in choosing the right pricing models and when the pricing models are chosen are already, that affects the determining of the SaaS architecture.

Table of Contents

LIST OF SYMBOLS AND ABBREVIATIONS

SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a Service
SISaaS	System Infrastructure Software as a Service
CC	Cloud computing
NIST	The National Institute of Standards and Technology
MTA	Multi-tenancy architecture
ERP	Enterprise resource planning
CRM	Customer relationship management
STV	Shared Technology Vulnerability
HPC	High Performance Computing
WL	While literature
GL	Grey literature
MLR	Multi-vocal literature review
SLR	Systematic literature review

1 INTRODUCTION

Software-as-a-Service (SaaS) is a business and delivery model whose characteristics are specified by software architectural and business model characteristics (Marston *et al.*, 2011). Along with SaaS (Software as a Service), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS), are the other two core ideas of cloud computing (CC).

SaaS is a delivery model which hosts the software off-premises and delivered via the internet, with a subscription model as the payment method (Nitu, 2009). In three ways, SaaS varies from typical traditional software. First, software as a service means that software vendors handle both technical infrastructure and management, including hardware, software, and professional services. Second, SaaS is remotely hosted, which means that software is distributed as hosted services for clients to access (Campbell-Kelly, 2009; Shiliang Wu, Wortmann, and Chee-wee Tan, 2014). Third, SaaS has a single version, which means that each client has access to only one version of the software at any one time (Lehmann and Buxmann, 2009).

In contrast to typical licensed software, the SaaS product is delivered through a different revenue mechanism, which are based on subscription and/or usage. The product development is being affected by the growing variety of SaaS pricing models which includes both technological architecture and design of the product. Like other traditional software products in which pricing is done at the end of the development process, SaaS pricing must be considered early in the design process (Choudhary, 2007). There many success firms in the world of SaaS, Salesforce is considered as one of them. It is a company that offers services like customer relationship management. As a example, let's assume a seller who is going from one place to another, the software can be accessed by him using a laptop while he is mobbing and this helps in minimizing task like infrastructure, software management and upgrade complications, and data synchronization from many sources for the company. Along with Salesforce, Google is one of those big names which is offering emails and software which helps in the productivity via its own CC platform (Campbell-Kelly, 2009).

As there is development in the field of internet and web-based application, SaaS has become an extremely popular model in CC. Over the last few years, it has become one of the most popular topics in IT news. It is becoming a more favorable alternative for customers to adopt

SaaS solutions rather than investing in a brand new on-premises software (Spruit and Abdat, 2012; Kaplan, 2008).

According to the International Data Corporation, the market of public cloud services which consists of System Infrastructure Software as a Service (SISaaS), IaaS, PaaS and SaaS has grown by 24.1 percent year after year to $312 billion in the year 2020. In which $148 billion in sales is estimated in the year of 2020, SaaS applications are considered as the largest and most mature category of public cloud. Data-driven, intuitive, and perfectly suited for more distributed cloud infrastructures, organizations across industries have accelerated the replacement of legacy business systems with a new generation of SaaS applications (International Data Corporation, 2021).

Before launching their SaaS product, SaaS suppliers must define their pricing plan. They must be able to comprehend all possible income streams, deployment and distribution expenses associated with their solution, their capacity to offer them at pricing that maximize profit, and guarantee that the approach is sustainable over time to assure the vendor's continued success (Spruit and Abdat, 2012, p. 4).

SaaS architecture is responsible for limiting or permitting the usage of multiple pricing model options, in addition to the impact of pricing on the design (Laatikainen and Ojala, 2014,p.298). Companies that are a new start-up, they focus usually on the product development in the first place, not on the pricing and on the contrary, some studies prove that they are dependent as some fixed priced projects lead to poorer design while in some cases if SaaS architecture is good, it does not limit the pricing and a well-planned SaaS architecture gives the idea on how to price products (Patrick Campbell, 2016) Furthermore, in many circumstances these two business units do not communicate on a regular basis; as a result, the weakness of the software's pricing model and design may be recognized too late, resulting in unnecessary losses. As a result, in circumstances where the software's architecture and pricing are strongly linked, both the technical lead and the company's business managers need to be aware of these interrelationships (Laatikainen and Ojala, 2014, p.298).

Finding the relation between the two might help companies in achieving their goals at a faster rate as once they know the pricing models they are going to implement for their product and will help them in planning better architecture (Laatikainen and Ojala, 2014).

1.1 Objectives of the study

The aim of the study is to determine the relationship between SaaS architecture and pricing models. Explaining how SaaS architecture might affect pricing models and vice versa. Some businesses rate their product as a one-time purchase, while others opt for a subscription model so, it is critical for the organization to understand the product's value as well as the client's demand. If planning is better from the beginning of the implementation, it will lead to better growth and success of the company.

As the aim of this thesis is to find the relation between SaaS architecture and pricing models which will be achieved through this study by a literature review to understand the current stage of how pricing models are related to SaaS architecture. It will provide insights into the relationship between them and getting information that can help in forming the basis of the framework.

Publications will be collected that are available in search engines, digital libraries, and databases. Analysis of these publication will be done based on which publication is white literature (WL) and which one is grey literature (GL), as explained in the Chapter 3 below. More focus has been put on the WL in the master's study. After differentiating them based on the type, they will be further analyzed depending on the year they were published, by whom and their type like journal articles, books, thesis, etc. The objective of doing all the above explained steps is to find the information which is most closely related to the topic of this thesis.

1.2 Research Questions

There has not been any clear evidence that if there is relation between them or not. So, keeping these in my mind, this study aims to answer the following research questions:

1. How do changes made in SaaS architecture affect pricing models?
2. Does SaaS architecture limit pricing models?
3. What is the impact of pricing models on SaaS architecture?

2 SAAS ARCHITECTURE AND PRICING IN LITERATURE

This section starts with a small introduction of Cloud Computing (CC), it's service models and deployment models, followed by explaining the concepts of SaaS architecture, SaaS maturity model, pricing models and the relation between architecture and pricing based on the previous studies.

2.1 Cloud Computing

Cloud computing (CC), is the long-held concept of computing as a utility, has the potential to change a large segment of the IT sector, increasing the attractiveness of software as a service and influencing how IT hardware is developed and acquired. With the concept of CC, new internet services can be deployed by the developers at a cheap price with fewer people and less money spent on hardware for running the businesses. It also helps them in knowing how much time and money should be spent on planning about the services depending on their popularity in such a way that they are not doing more than required or vice-versa (Armbrust *et al.*, 2009).

CC offers a wide range of services including computers, databases, storage, virtual machines, servers, analytics, and machine intelligence. Cloud computing delivers these services over the internet, making them scalable and allowing businesses to avoid capital expenditures on hardware purchases (Mukundha, 2017).

CC has many different definitions by different authors in the Table 1 below.

Table 1. Definitions of CC

Definitions	References
CC is an internet-based system that allows us to access software, data, and resources from any location with an internet connection.	(Alzakholi *et al.*, 2020)
CC is a virtualization-based IT deployment strategy in which resources are distributed on the internet in form of services	(Böhm *et al.*, 2014)

by the service providers. These services can be provided according to the demand and pricing can be done based on the number of users.	
CC could change the deployment method of these applications and computing resources and giving opportunity to new business models	(Diaby and Rad, 2017)

The National Institute of Standards and Technology (NIST) which is a non-regulated government body that creates technology, measurement, and standards in order to help US-based businesses in the field of science and technology. NIST has published a definition of CC which states that *"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"* (Mell and Grance, 2011, p. 2).

The CC services contain three different categories of models which are IaaS that is a type of service that includes storage, networking, and visualization on a pay-as-you-go basis, PaaS in which hardware and software tools are provided over the internet like internet, database management, and SaaS which delivers application that is accessible online and managed third-party vendors (Akande, April and Van Belle, 2013; Tony Hou, 2018). Figure 3 and Figure 4 depicts an example of a few companies that are using these service models and explaining the roles of each service simultaneously. The tip of the pyramid denotes the least degree of control, as well as the least level of responsibility whereas the bottom of the cloud pyramid shows the highest level of resource control, which also leads to the greatest level of responsibility (Cheshire, 2019).

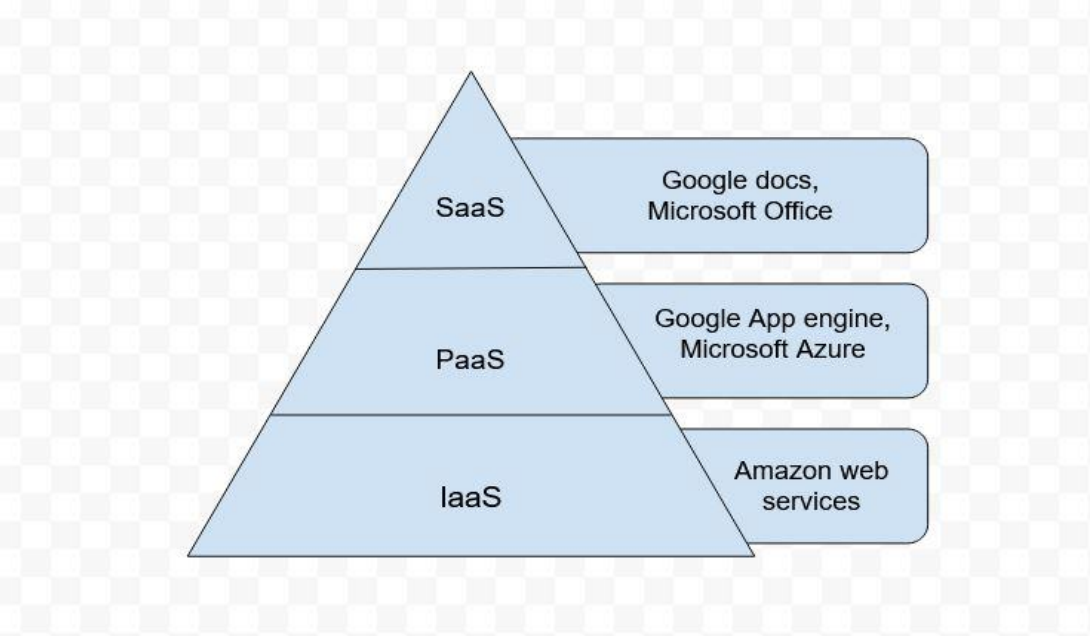

*Figure 1. Cloud service model (*Everything as a Service (XaaS) - the evolution of cloud-based Services, *2021).*

When we talk about the solutions that are cloud-based, choosing the correct model is very important. To select the best service model or a mix of service models, one must first understand what each service model comprises, as well as the obligations that are faced by cloud service providers and consumers (Michael Kavis, 2014).

2.1.1 Cloud Computing Service Models

As we talked earlier the 3 service models are IaaS, PaaS, and SaaS. Figure 4 illustrates more about the roles and responsibilities in each model. As in On-premises, the users have to manage everything on their own and as they go more towards cloud computing models. The picture clearly depicts that the factors like virtualization, server, storage, and networking are handled by a service provider in IaaS, whereas PaaS except for application and data, everything else is handled by the provider and finally in SaaS user just uses the application without worrying about anything as all the aspects are managed by the provider.

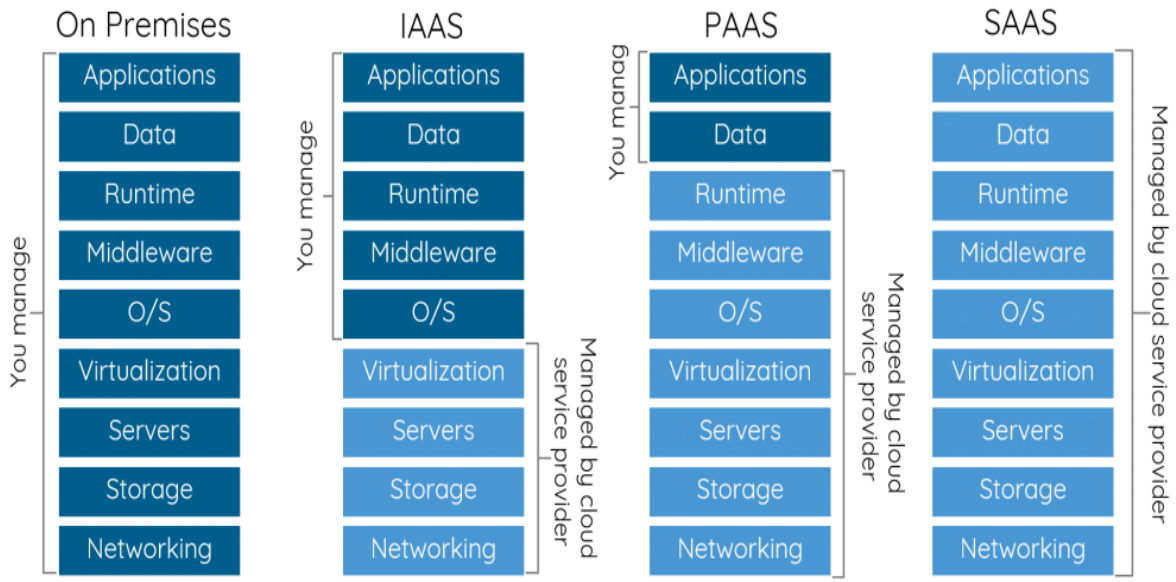

Figure 2. IaaS vs Paas vs Saas (Michael Kavis, 2014).

These three services are discussed below based on the prior findings and studies, and we will be focusing more on the third model which is SaaS, its architecture, and pricing models.

IaaS

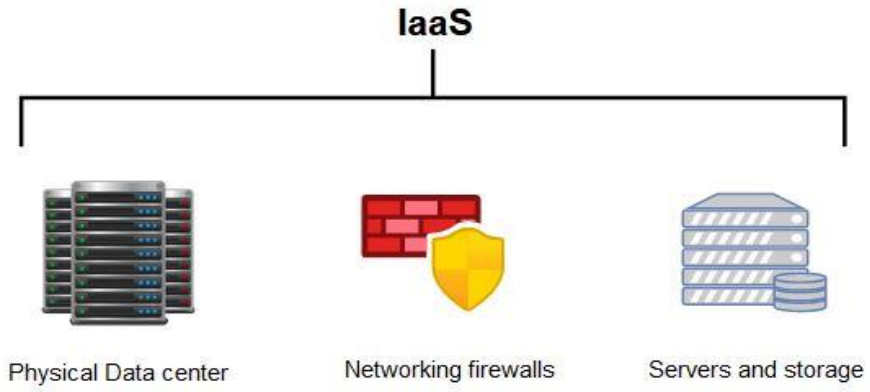

Figure 5. IaaS

In IaaS, necessary amenities like storage, networks, processing, and all other basic resources of computing are provided by the supplier, where the user can run and deploy any software he/she wants, that includes operating systems and applications (Khan, 2012). Although the cloud infrastructure is managed by the service provider but the customers might be given the authority to have limited control on operating systems, storage, and programs, and also over some certain networking components e.g. firewalls (Miyachi, 2018). It offers users a platform in the form of a computer environment or infrastructure (including hardware and software). IaaS provides companies with alternatives that are based on the cloud in order to

avoid the on-site resources which costs a lot more than on-premise infrastructure (James Ng, 2020). Being a service provider, IaaS has a server that is provided virtually consisting of one or more central operating units. The server has various running packages like centralized and fully automated. Also, users can request to be provided with facilities like networking and storage along with computing services which are there by default (Mohan *et al.*, 2017).

Dynamic scaling, desktop virtualization, and approach-based administrations are some of the characteristics of IaaS circumstances. Clients of IaaS pay on a per-use basis, typically by the hour, week, or month, depending on how much time they spend using the service. Some virtual machine space providers also charge consumers based on the amount of virtual machine space that they consume each month. Pay as you go models eliminate the need for capital investments in equipment and programming for internal communications (J. Beschi Raja and K. Vivek Rabinson, 2016, p.100). According to (Ankita Sharma, Sonia Vatta, 2013), IaaS service model is not recommended for the organizations that have efficiency and bandwidth on high priority and also where the storage of data and its processing is an issue itself.

PaaS

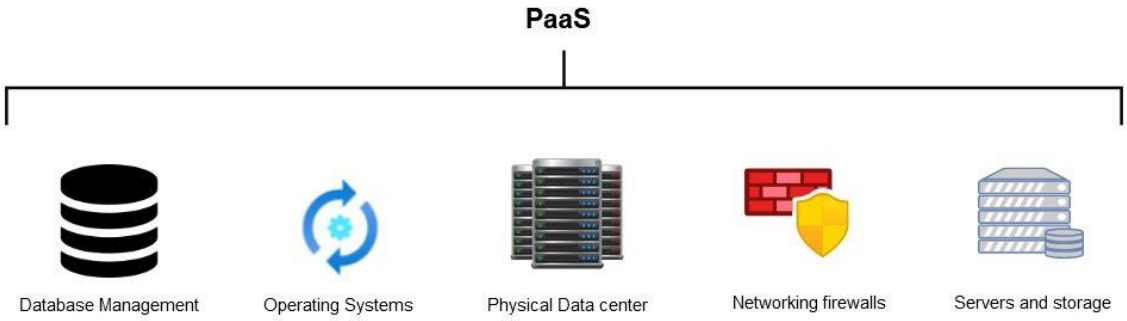

Figure 6. PaaS

PaaS has the central position in the hierarchy of CC. In this, a framework is provided to the developers in order to build apps and programs in a way that these programs and apps can be allocated without the need of installing the production environment. Servers and resources that are software defined can be rented by clients to run the software (Mohammed, Chnar Mustafa and Zeebaree, Subhi R. M., 2021). In PaaS, customers do not have the access to control hardware like servers and network but on the other hand they can control application and configure them according to their need (Bukhari *et al*., 2016) as shown in Figure 3. Users or customers can make their own system by using the tools available and keeping the other

service running like management of the software (Diaby and Rad, 2017). The current example of PaaS is Azure from Microsoft where the main focus is on cloud-based software that developers use and deploy (Mohammed, Chnar Mustafa, and Zeebaree, Subhi R. M., 2021).

PaaS has many attributes like self-manageable and maintainable, it enables many options, for instance, a user can make any number of customizations to the database and allows them to fully customize the user interface. It enables the deployment of numerous copies in the same or various clouds for situations that may require isolation from other business operations. This is critical for apps that must address regulatory requirements or for applications that have an internal vs an external interface. Each of these scenarios allows the developer to continue using standard tools and recommended practices while operating in another, safe environment. Additionally, businesses may leverage PaaS to mix native resources and data to create customized mix for a range of online services (Intel, 2014, p.5). PaaS is also beneficial in case of the applications that require mobility across the environment where they are hosted. When it comes to performance, both hardware and software can be changes in order to meet requirements (Almubaddel and Elmogy, 2016, p.2).

SaaS

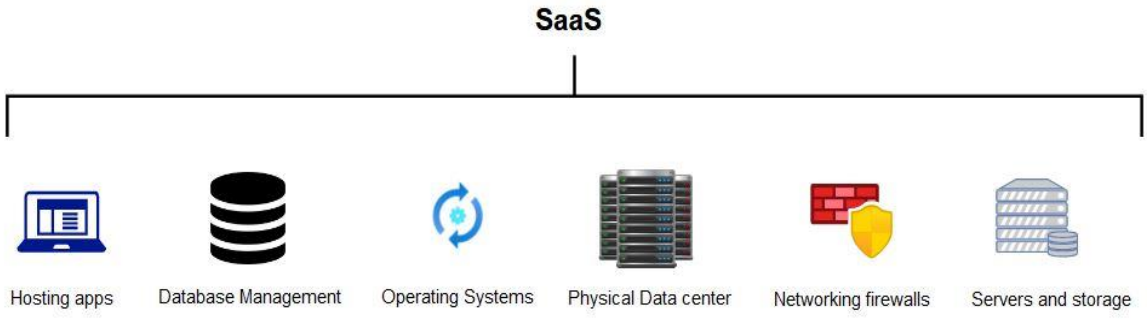

Figure 7. SaaS

SaaS model came to existence after evolving from Application Service Provisioning (ASP). ASP we introduced as an alternative for on-premise software as it allowed tenants to outsource the hosting and maintenance to ASP (Laatikainen and Ojala, 2014, p. 598). Later in 1999, Pearl Brereton, a member of Pennine Group, put forward an idea of turning software into services, by arguing the idea of developing new architecture which is based on constructional forms like objects or components as the future, but in delivering software functionality to users in a radically different way (Turner, Budgen and Brereton, 2003).

SaaS is multi-tenant service which is available on-demand which is suitable for cloud software. It is not required to install software in the machine as a result it is easily accessed via internet using web browser which results in availability of this in many people in short span of time (Liu *et al.*, 2010, p. 1). There is no need to control this by the user except the configuration setting of the application (Miyachi, 2018, p. 7). According to the definition given by National Institute of Standards and Technology (NIST) SaaS is *"...a capability which is available to the users by providers' to use their application which is running on cloud infrastructure. These applications can be accessed via different devices of users like web-browsers."* (Miyachi, 2018, p. 7). Moreover, SaaS is a multi-tenant single-instance software that allow users to share the resources available to them without interrupting one another. With this function, all the upgrades and deployed patches are transparent to the users (Liu *et al.*, 2010, p. 402).

2.1.2 Deployment models of Cloud Computing

There are four deployment models in CC named as public cloud, private cloud, community cloud and hybrid cloud. All these categories of cloud are explained briefly below.

Public Cloud

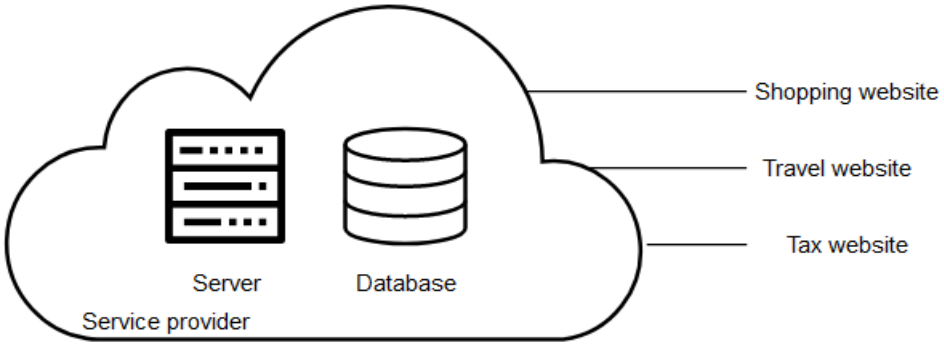

Figure 8. Public cloud (Atul Kumar, 2020).

Public cloud computing makes clouds accessible to the entire public, and all data is stored and generated via third-party services (Cruz, 2021, p. 1). The term "public" does not always mean free, even though it can be free or inexpensive to use (Venkata Rao J. and D. Bhargava Reddy, 2011, p. 99). This infrastructure is owned by the cloud services provides and is made available to the public or a group of industries (Savu, 2011, p. 2).

This deployment is very economical for the users as they have to pay for the services they use and don't need to invest is hardware or infrastructure. Also, it is made sure that the

infrastructure is available to the users 24 x 7. Drawback of the public cloud is that there are some data privacy issues as the data is prone to public theft and the customers or users have control over who has access to their data (Cruz, 2021, p. 1).

Private cloud

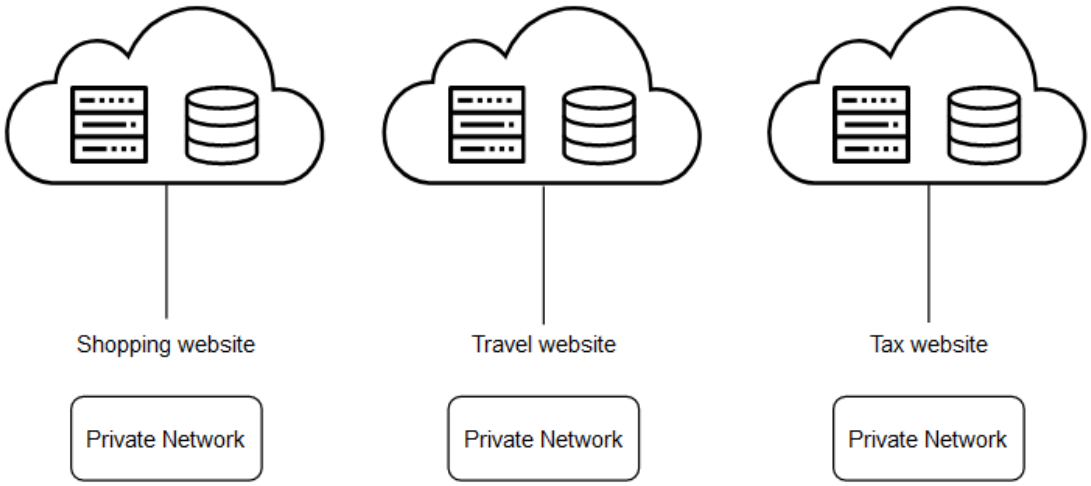

Figure 9. Private cloud (Atul Kumar, 2020).

The difference between public and private cloud services is that the public cloud is available to public on the other hand private cloud is limited to companies. Additionally, private cloud services give the provider and user greater control over the cloud infrastructure, enhancing security and reliability using restricted and designated user access and networks. Virtualization and distributed computing advancements have enabled corporate network and datacenter managers to transform themselves into effective service providers, meeting the needs of their "clients" within the organization (Venkata Rao J. and D. Bhargava Reddy, 2011, p. 99).

It has flexible deployment as the resources can be customized and is most secure as the resources can be accessed only by the person with authorization. The only drawback of this cloud is that the cost is very high as compared to public cloud (Cruz, 2021, p. 1).

Community cloud

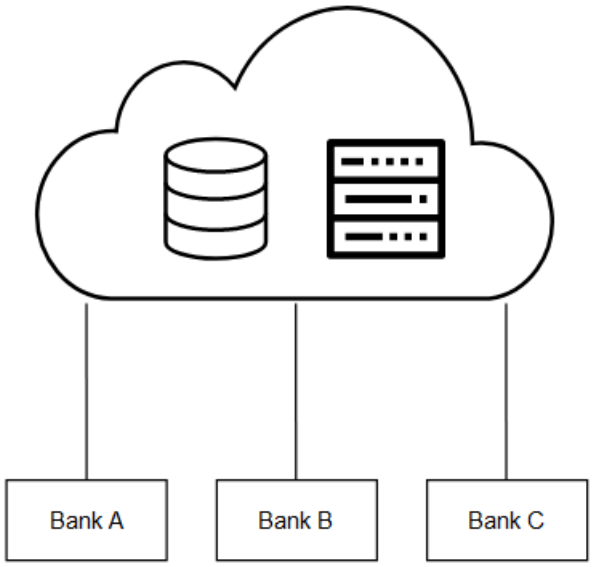

Figure 10. Community cloud (Atul Kumar, 2020).

This deployment model is simila1r to private cloud but with different set of users. Multiple businesses share the cloud infrastructure, which serves a specific community with common issues (Savu, 2011, p. 2). Another difference is that the private cloud is utilized by one business whereas community enables the usage of cloud resources by many companies with same background (Cruz, 2021, p. 1). It can be located on or off site and is sometimes handled by the organization themselves (Savu, 2011, p. 2).

Pros of this deployment model is that it had better security, privacy and has easy data collaboration and resources sharing. Even though having better security and privacy, this model is not that commonly used and is expensive than the public cloud (Cruz, 2021, p. 1).

Hybrid cloud

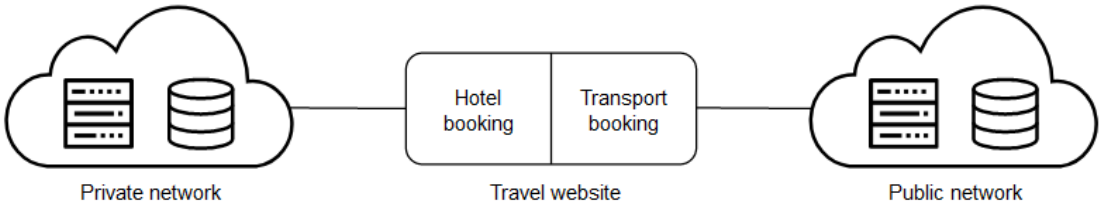

Figure 11. Hybrid cloud (Atul Kumar, 2020).

This deployment model is the combination of private and public cloud that operates within (Savu, 2011, p. 2). This deployment technique is critical when a company must expand and

link on-premises infrastructure and resources to cloud-hosted apps. The non-critical information is outsourced in order to control critical services and data (Venkata Rao J. and D. Bhargava Reddy, 2011, p. 99).

While cloud computing is frequently portrayed as the industry's future, the hybrid approach is more common for a variety of reasons. Large companies frequently have already made significant investments in the infrastructure necessary to offer resources internally. Additionally, many companies would like to maintain ownership of sensitive data in order to assure its security (Venkata Rao J. and D. Bhargava Reddy, 2011, p. 99). It has more reasonable prices and better scalability, security and privacy than others (Cruz, 2021, p. 1).

2.2 Software as a Service Architecture

This sub-chapter focuses on the SaaS architecture which covers one part of this thesis. Architecture is something that outlines the in-depth structure, logic of a system and describes how different parts of a system interact with each other. In a way, it acts as a guideline for the developers by explaining the principle and rules for developing a software using the provided architecture (Hyrynsalmi, Rauti and Kaila, 2019, p. 1763). SaaS applications are architecturally very close to the other application that are built based on the principles of service-oriented design (Ju *et al.*, 2010, p. 385). The development of SaaS software requires a different approach than the traditional software. In the process of making a SaaS product, architectural design is very important. It should be designed in a way that all the requirements are fulfilled (Aleem *et al.*, 2019, p.1). SaaS architecture is now common in businesses tasks like ERP, CRM, service desk management, and more (Bhardwaj, Jain and Jain, 2010, p. 41).

According to (Laatikainen and Ojala, 2014, p. 598) for an architecture to be well defined, it should possess characteristics like configurability, multi-tenancy, and scalability. (Guo *et al.*, 2007, p. 1) considered multi-tenancy as the key facility of SaaS network. (Aleem *et al.*, 2019, pp. 2–4) explains about customization, scalability, integration, MTA, security, and fault tolerance in his study. Customization, MTA, and scalability are considered as the most important characteristics of SaaS by (Tsyganov, 2018, pp. 45–46).

Considering all the studies that describes different characteristics as important one for SaaS, it was concluded that customization, scalability, redundancy, security, virtualization, integration, fault tolerance and MTA (multi-tenancy architecture), all are important for an

architecture to be successful and thus are explained briefly below. (Tsai, Bai and Huang, 2014, pp. 3–4; Aleem *et al.*, 2019, p. 6).

2.2.1 Customization

The concept of customization also known as configuration is being studied for around 45 years (Tsai, Bai and Huang, 2014, p. 9). It aims to fulfill the need of every customer. This characteristic of architecture comes up with a facility for the user in which they can make changes in the limited parts of SaaS application with the help of an interface that is specially designed for that (Tsyganov, 2018, p. 26). SaaS being a single instance multi-tenant application model. Vendors cannot customize the software for any single customer as it will change for all others too, configuration allows each tenant to customize their software uniquely (Nitu, 2009, p. 21). There are 4 different areas in customization which are data field, process, service and interface (Luan, Shi and Wang, 2009, pp. 337–338).

(Wei-Tek Tsai and Xin Sun, 2013, pp. 1–2) explained in his study that who are the people that can make these customizations, how it is done and the level of easiness. It is briefed in the Table 2.

Table 2. Who, how and who else?

Who	**Developers/designers**	**Consultants**	**Users**
	In the development stage according to the demand of the tenant.	People are hired by tenants like in CRM, ERP.	Based on their personal requirement though it is external and limited
How	**Source Code**	**Composition**	**Configuration**
	Source code is added by predefining the interface	Making the workflows again according to their demand.	Providing different configuration parameters.
	Manual	**Automated**	**Guided**

How easy	Decisions are taken by tenants manually.	Done automatically based on the tenant's requirement.	Guided via automated customization and manually reviewed by tenants.

2.2.2 Scalability

When a program has high scalability, it means that it can perform at same level on both small and big data sets (Gao *et al.*, 2011, p. 62). In other word a system is called a scalable system if it can give stable performance even when the quantity of the work increases. Take an example of online portal for shopping clothes where they have a lot of users visiting because of the discount period, this the time where scalability will make sure that the response time is still justifiable even after heavy traffic (Aleem *et al.*, 2019, p. 2). Tenants can have any number of users at any time, so it is the responsibility of the tenant to make sure that the system is still performing well under the heavy workload (Gao *et al.*, 2011, pp. 62–63).
It can be classified into two categories which are scale-up which is also called vertical scaling and scale-out which is also known as horizontal scaling, shown in Table 3.

Table 3. Categories of scalability

Categories	Explanation
Scale-up or vertical scaling	When the application is running on a machine which has better configuration like more memory, storage space and higher band width (Gao *et al.*, 2011, p. 62; Tsai, Bai and Huang, 2014, p. 10).
Scale-out or horizontal scaling	When the application is distributed on different machines with same configurations (Gao *et al.*, 2011, p. 62; Tsai, Bai and Huang, 2014, p. 10).

As a single machine can't be upgraded after some extent, so it recommend in most of the CC scenarios to use scale-out rather than scale-up (Gao *et al.*, 2011, p. 62).

2.2.3 Redundancy

It means that making one more than one duplicates of the data or some elements of the system and makes sure that data can be accessed at any time. It can be done by adding more servers and load balancer than required (Christopher Wray, 2016; Tsyganov, 2018, p. 27).

2.2.4 Security

Security has been a major concern not in the field of SaaS only but also in general as well. According to study done by (Akande, April and Van Belle, 2013, p. 120) it is the biggest issue in CC as the applications are hosted on internet and security can be breached at any moment. The data is visible to others as well, as it is available online. So, the questionable one is the service provider if there is any breaching. They divided security in three different parameters as shown in Table 4 below.

Table 4. Security aspects.

Aspects	Explanation	References
Identity breach and confidentiality	Personal information should not be shared, and not unauthorized person can have access to the data.	(Zissis and Lekkas, 2012, p. 586; Akande, April and Van Belle, 2013, p. 120; A. Kofahi, 2018, p. 4; *Top Threats to Cloud Computing: Egregious Eleven*, 2019, p. 9)
Integrity	Protecting data from getting modified, deleted, or moved without permission.	(Zissis and Lekkas, 2012, p. 586; Akande, April and Van Belle, 2013, p. 120; A. Kofahi, 2018, p. 6)

Availability	The owns has access to data whenever needed even in the case of any infringement.	(Zissis and Lekkas, 2012, p. 586; Akande, April and Van Belle, 2013, p. 120; A. Kofahi, 2018, p. 4)

2.2.5 Virtualization

The idea of virtualization is to design a virtual version of the system (Tsyganov, 2018, p. 27). A virtual machine is used that helps in deploying the software dynamically rather than installing it on the specified environment (Zhong *et al.*, 2010, p. 145; AlMutair and Zaghloul, no date, p. 2). According to (Venkata Rao J. and D. Bhargava Reddy, 2011, p. 100) when a single system is used along with sharing the resources to run many operating system is known as virtualization.

A virtual machine is like an emulator that behaves like the original machine and controls all the virtual systems installed on the physical one (AlMutair and Zaghloul, no date, p. 2). In this, system space is altered using a load balancing system which helps tenants in providing the computer power according to the requirement (Kang *et al.*, 2010, p. 344). With the addition of virtualization as a characteristic in SaaS architecture, the virtual desktop is combined with local and makes that makes it look like the application is running on the local desktop (Zhong *et al.*, 2010, p. 145). This approach act as a pillar for multi-tenancy architecture.

2.2.6 Integration

With the rising use of SaaS, companies are increasingly requesting integration of their SaaS apps with their backend systems as every tenant has different business requirements (Liu *et al.*, 2010, p. 402) and to satisfy business requirements, a business application should combine both on-premises and SaaS applications in order to gain the full benefits of SaaS technology (Aleem *et al.*, 2019, p. 5). SaaS applications, on the other hand, must integrate with other services and applications in order to be valuable, but most of these services are provided by corporate on-premise systems (Luan, Shi and Wang, 2009, p. 1).

As there are 3 major layers in SaaS application, integration is taking place in all of them and is depicted as user interface, process, and data integration. The user frequently changes

between many interfaces that require various credentials for SaaS and on-premises applications. In user interface integration, single sign-on is necessary, in which the user needs to sign in just once and can access all essential SaaS services and apps (Sun *et al.*, 2007, p. 560; Aleem *et al.*, 2019, p. 5).

2.2.7 Fault Tolerance

Finding a fault in any system before it happens or taking care of it once it has happened, is very important. Fault tolerance is a type of characteristic that deals with fast replacement and repair of any fault to hold onto the system (Ganesh, Sandhya and Shankar, 2014, p. 845). In a way it keeps the system in function even if there is a fault (Aleem *et al.*, 2019, p. 5). It uses a dummy of the server to cover the failure of other dummies. There can be many reasons for a system to fail like software or network fault etc. (Ganesh, Sandhya and Shankar, 2014, p. 845).

Cloud has real-time High-Performance Computing (HPC) activities that need a high level of fault tolerance. For real-time applications, there are two types of policies in fault tolerance one is proactive and the other is reactive fault tolerance (Ganga and Karthik, 2013, p. 387). The aim of proactive is to take care of errors and faults and predicting them before that happens by replacing them with working ones (Ganga and Karthik, 2013, p. 387; Ganesh, Sandhya and Shankar, 2014, p. 846; Aleem *et al.*, 2019, p. 6).

2.2.8 MTA (Multi-Tenancy Architecture)

Multi-tenancy is the characteristic of a SaaS application with which multiple tenants can use the application over a single server (Aleem *et al.*, 2019, p. 3). When a software instance operates on an infrastructure of service provider, it may be accessed by many tenants who share components like database layer, etc. which is an architectural concept known as multi-tenancy which can also be seen in the Figure 12 below.(Kang, Kang and Hur, 2011, pp. 462–463).

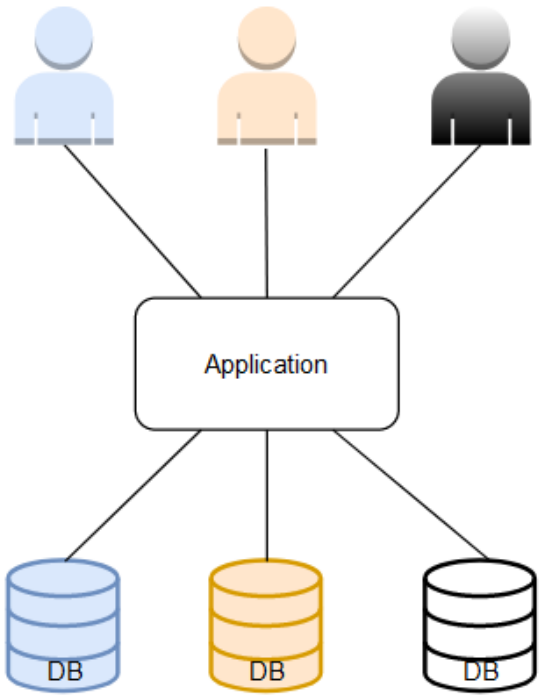

Figure 12. MTA (multi-tenancy architecture)

(Aleem *et al.*, 2019, p. 3) described in his study that are 3 different levels of multi-tenancy which are application, data-model and full multi-tenancy. Data-model is the most basic one in which same database is being shared, in application same instance along with same database is being shared and in full multi-tenancy same database and instance is shared but with their own version of the application if needed.

According to (Tsyganov, 2018, p. 25) this characteristic is closely related to the properties likes customization and scalability. MTA provides a number of benefits, including improved utilization of hardware resources, cost savings associated with application maintenance, and new options for data aggregation (Laatikainen and Ojala, 2014, p. 598; Pinto *et al.*, 2016, p. 1). After all these benefits it also have some drawbacks as the resources are being shared, so any problem caused by one have direct impacts on the other tenants and with more requirement of scalability and customization, the code might need more effort (Laatikainen and Ojala, 2014, p. 598).

2.3 Maturity level of SaaS software

Based on the studies conducted by many organizations, a mature SaaS model may be achieved progressively, and the maturity level is dependent on the maturity level of SaaS architectural features or architectural and business characteristics (Ju *et al.*, 2010, p. 386;

Laatikainen and Ojala, 2014, p. 598). In terms of architectural features, these maturity models classify multi-instance, customer-specific ASP designs as being the least cloud mature, while scalable, customizable, and multi-tenant-efficient applications are the most mature (Laatikainen and Ojala, 2014, p. 598).

The maturity level of SaaS can be described in 4 different levels as shown Figure 4. In the first level, a customized version that is running on the server provided by the host is delivered to each customer and that version has its own instance of the application. As described in the figure these instances are completely isolated from one another. Maintaining the application in this model takes time and is a little complicated (Ju *et al.*, 2010, p. 386; 'SaaS Maturity Levels', 2019, p. 1).

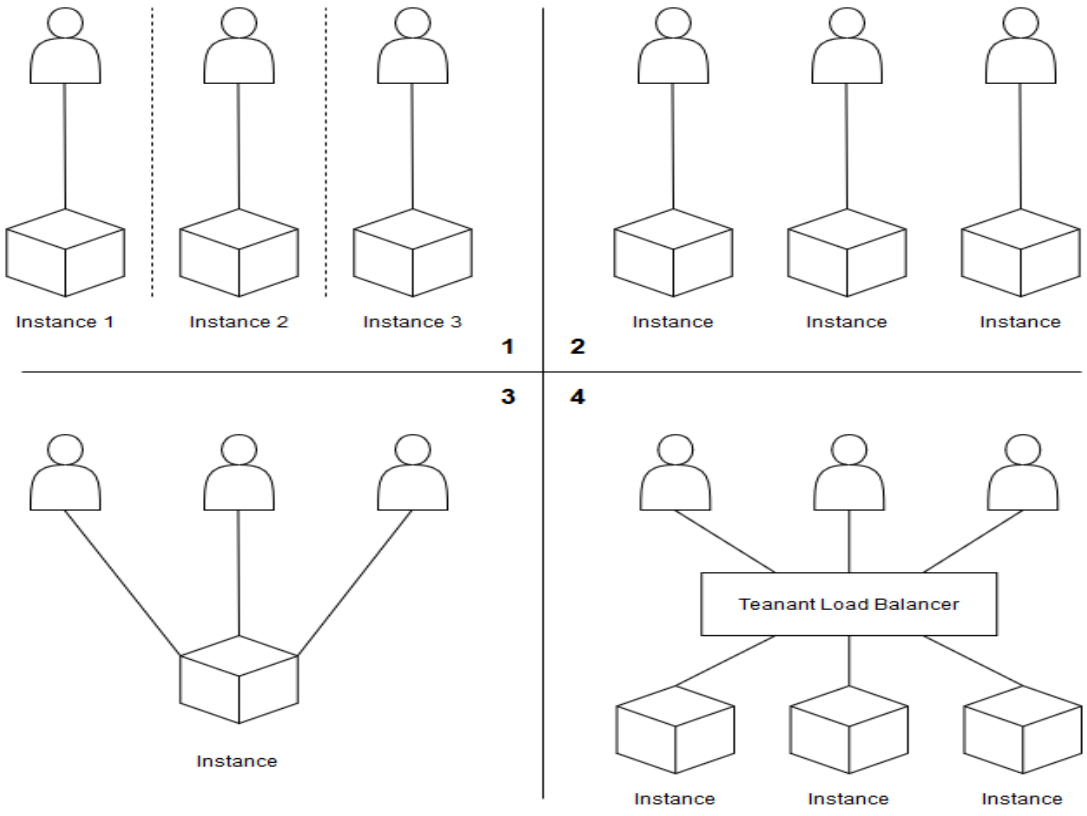

Figure 13. SaaS maturity model.

As can be seen in Figure 13 different customized instance was delivered in the first level whereas in the second level, customers are also delivered with different instance, but the only difference is that these instances are similar to each other which means that the code used for implementation is similar however they are provided with the option of configuration in which the appearance and behavior. This helps in updating the app easily

as whenever they are updated, changes will be automatic on all instances Even though the same code is being used in all instances, they are still fully apart from each other (Ju *et al.*, 2010, p. 386; 'SaaS Maturity Levels', 2019, p. 1).

In third level, singnal is delivered to all the customers which include different set to feature and user experience for each one of them. They make sure that the data from each tenant is kept separate. After accessing the application no one can tell that these instances are being shared among tenants. This leads to the efficient use of resources available to them and saves money. This shows that features like customization and multi-tenancy are added (Ju *et al.*, 2010, p. 386; 'SaaS Maturity Levels', 2019, p. 2).

In the last level, multiple tenants are hosted on different instances of the application. A load balancer is used in this level to make sure that if there is any increase or decrease in the number of instances or serves, it is easily manageable. This makes the application at this level, more configurable, scalable, and multi-tenant (Ju *et al.*, 2010, p. 386; 'SaaS Maturity Levels', 2019, p. 2).

In summary, it is highlighted by many authors that achieving a maturity model is not always the best solution for every vendor. The components to be shared and the level of service should be decided by themselves because to reach maturity level, many aspects are to be kept in mind like who are the targeted customers, architectural properties, and business requirements. Sometimes it is not feasible for the system to reach a higher maturity level as they might face issues like security, migration of the application, and providing their characteristics that are not cost-effective. Improving maturity models in a way is an improvement in the SaaS architecture and according to (Laatikainen and Ojala, 2014, p. 699) improving the maturity level of the application can also lead to change in pricing models as well.

2.4 SaaS Pricing models

Pricing is a very important element of a business and product strategy in every firm, yet it is frequently overlooked. As a fundamental part of your business, pricing has the power to significantly boost your firm's basic figures (Patrick Campbell, 2016, p. 6). Before launching their SaaS service, SaaS companies must define their pricing plan. It is important for the SaaS vendors to comprehend all possible income streams, deployment, all the expenses for

distribution which are associated with their solution, their capacity to offer them at pricing that maximizes profit and guarantee that the approach is sustainable over time to assure the accomplishment of all the objectives and the vendor's continued success (Spruit and Abdat, 2012, p. 4). After investing endless hours and resources in developing a sound business architecture, creating a higher-quality product, and attracting customers, they are still unable to choose appropriate pricing models for the products, which is critical for their business to continue operating and growing exponentially (Patrick Campbell, 2016, p. 2).

Pricing is described as a process of making decisions that helps in taking monetary actions which are related to the quality of the resources and services provided to the customer. It acts as a vital link between the business unit which deals with R&D, sales, etc., and business function which focuses on product management, revenue management, etc. (Saltan and Smolander, 2021, p. 3)

Authors describe pricing models in SaaS differently. (Mazrekaj, Shabani and Sejdiu, 2016) proposed pricing in 3 different types, first is fixed pricing which consists of subscription, pay-per-user, and pricing menu, second is dynamic pricing in which the price is computed using a pricing mechanism after a request is received and third which is market-dependent pricing which consists of yield management that is based on real-time modeling and predicting the demand, bargain which depends on the relationship between parties, the auction includes negotiating so that both parties can agree to a price and dynamic market.

(Shiliang Wu, Wortmann, and Chee-wee Tan, 2014, pp. 153–154) adapted the framework by (Lehmann and Buxmann, 2009, pp. 458–460), it was explained and expanded it in their study. They divided pricing into two different categories as value-based and cost-based. Further, they divided them into sub-categories as described below (Laatikainen, Ojala and Mazhelis, 2013, pp. 120–121; Shiliang Wu, Wortmann, and Chee-wee Tan, 2014, p. 153).

Value-Based	**Payment-Based:** Montlhy subscription
	Product-Based: Based on the requirement of the clients
	Usage-dependent: Based on the usage
Cost-Based	**Usage-independent:** Not on the usage but more on number or clients
	Service-Based: Based on the services they use like backup, integration and security.

Figure 14. Pricing Categories (Shiliang Wu, Wortmann, and Chee-wee Tan, 2014, p. 153).

They described 5 different categories of pricing as shown in the Figure 14 above, the first is Payment-based. In this, vendors want their customers to pay their subscription upfront in order to use their software (Cusumano, 2008, p. 22). They also offer discounts sometimes to the customers that make large payments.

In user Product-based, the users are charged based on their consideration. For example, customers are charged based on the plan they are choosing, as some plans might contain fewer features than other and so on. And sometimes product is divided into sub parts and is sold in the form of a bundle according to the requirements of clients. With this technique, users have the independence to choose any plans or bundles they need (Spruit and Abdat, 2012, p. 154).

In usage-dependent, the users have to pay according to the usage. Usage can have many different aspects that show the usage of services like the number of transactions, memory used meanwhile usage-independent, focuses more on the number of users rather than the usage of services (Lehmann and Buxmann, 2009, p. 455). This category benefits the vendors in uplifting their revenue as the users have to pay for the services which they might not be even using whereas usage-dependent benefits the users more as they can choose the services they want to use, and will only pay for those, which affects the income the of the SaaS vendors (Shiliang Wu, Wortmann, and Chee-wee Tan, 2014, p. 154).

In Service-based pricing category, the vendors sell services like security and backup that once were their responsibility. So, the users have to pay in order to use these services. This is described as cost-based as service providers can be in control of the pricing (Shiliang Wu, Wortmann, and Chee-wee Tan, 2014, p. 154).

2.4.1 SBIFT Model

A model was introduced by (Iveroth *et al.*, 2013, pp. 116–120) showing different aspects on which the above mention revenue models can be based on. This model was named as SBIFT model, which describes pricing models as "system price-related feature of a buyer seller relationship" (Laatikainen, Ojala and Mazhelis, 2013, p. 118).

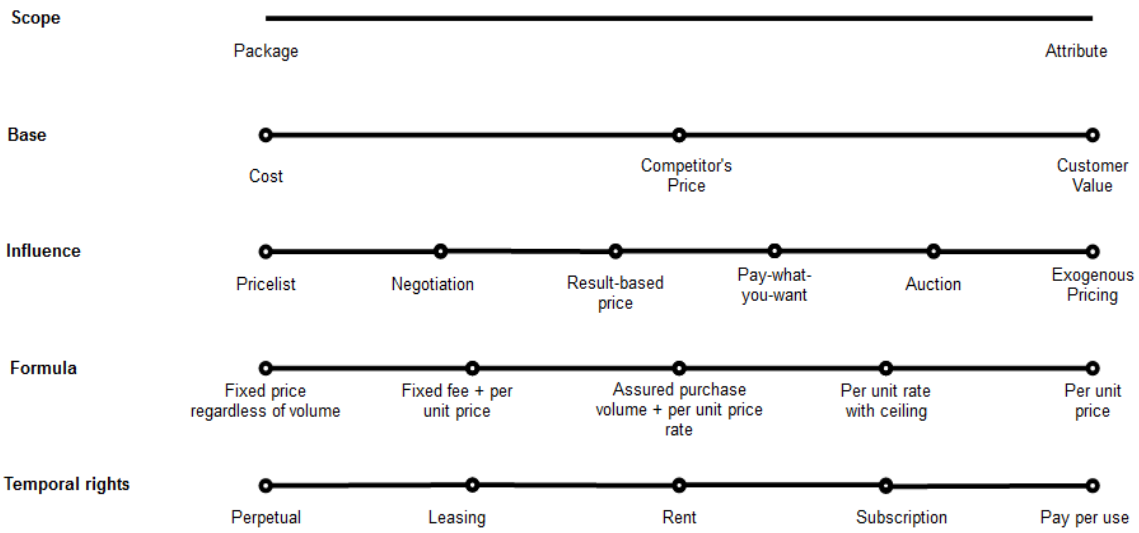

Figure 13. SBIFT model (Iveroth et al.*, 2013, p. 113)*

The dimensions in Figure 15 are described in the Table 5 below (Laatikainen, Ojala and Mazhelis, 2013, p. 118; Laatikainen and Luoma, 2014, p. 246; Laatikainen and Ojala, 2014, p. 599):

Table 5. SBIFT dimensions explained.

Dimensions	Description
Scope	Tells where the product should be offered in bundle or separately with distinct features.

Base	Gives the information about the basis on which pricing is set like performance or value of customer.
Influence	Shows how much influence does buyer or seller has on the pricing of the product.
Formula	Shows the relation between the volume and price.
Temporal Rights	Represents for how the service is available like subscription, perpetual, or pay-per-use.

It can be noted that all these dimensions are different from each other. Base is the one in which decisions are taken by higher management whereas in the rest modifications can be done easily (Laatikainen and Luoma, 2014, p. 247). From the previous studies based on pricing decisions depicts that there are many different types of models described by different authors and any model can be chosen according to the requirement for product pricing.

2.5 Relation between SaaS architecture and pricing models

Very limited amount of information can be found on the internet or in the books that focuses on addressing the relation between SaaS architecture and pricing models.

(Hyrynsalmi, Rauti and Kaila, 2019, pp. 1764–1767) addressed to fill the gap between architecture and business model. They used systematic literature method to collect the material related to this topic. 10 papers were selected and summarized and was concluded that this area is still undressed, although a few studies have been published, their focus has been on either specialized markets or on limited solutions. (Hyrynsalmi, Rauti and Kaila, 2019, p. 1766).

In (Laatikainen and Ojala, 2014, p. 599) they chose 5 different firms. The criteria they used to select the firms was that the selected companies should be software developing firms, both old and new firms, that included traditional, and SaaS based different maturity level. Interviews were done with these firms including face to face meetings, emails, and web pages were used as a means of colleting the required information.

The study's findings indicate that the design and price are intrinsically tied in instances when a firm's value proposition is cloud-native, and hosting is done on public cloud. In this situation, architecture plays a significant role in price, and pricing establishes unique architectural needs. However, in the case of startups or smaller businesses focus is more on the development, architecture and price have little or no influence on one another (Laatikainen and Ojala, 2014, p. 602).

2.6 Summary of Literature

In short, CC is a way of delivering service to the customers or users through internet. It is divided into three service models which are IaaS, PaaS, and SaaS. Furthermore, there are four different ways of deploying these cloud services which are public, private, community and hybrid cloud.

Our focus in this thesis is on SaaS architecture and pricing models. Different architecture attributes have been defined by many researchers. To have an architecture which is well-designed, the architecture should have properties like customization, scalability, redundancy, security, virtualization, integration, fault tolerance and multi-tenancy. To achieve these properties, there is a possibility that we need to choose the pricing models accordingly. There are different types of pricing SaaS application. It includes usage - based and value - based pricing technique. They are explored and explained further in order to see through the characteristics they possess that are scope, base, influence, formula, and temporal rights. To choose a maturity model all the factors like targeted customers, architectural properties and business requirements should be kept in mind.

A lot of focus has been put on the architecture and pricing models individually but not enough on the relation between them. Only two studies were found regarding the relation between SaaS architecture and pricing models. In which they tried to fill in the gap by doing interviews with the companies and the other one used systematic literature review.

3 METHODOLOGY

The focus of this study is on SaaS architecture and pricing models based on different research areas and types of studies. Our main objective is to find relation between SaaS architecture and pricing models. In this thesis, the extent of research is not limited to academic paper i.e., white literature only, rather it includes both white literature (WL) and grey literature (GL) which means multi-vocal literature review (MLR) for the analysis.

3.1 Multi-vocal literature review (MLR)

Systematic literature review (SLR) and systematic mapping (SM) are becoming very famous in the field of software engineering (SE). There is a lot of GL which is being produced by many researchers, but they are not included in SLR or SM. The main difference between the two is that SLR or SM uses academic peer-reviewed articles are as an input. Whereas SLR, in addition to GL it also includes blogs, government reports, conference proceedings, etc. (Garousi, Felderer and Mäntylä, 2016, p. 1; *Grey literature*, 2021, p. 1).

GL are the studies that are not published or the studies that has been published in a non-commercial way e.g., government reports, research reports, dissertations, conference proceedings, newsletters and bulletins (*Grey literature*, 2021, p. 1). Certain types of grey literature can be classified as research since they are the product of prolonged and methodical investigation by academics or other researchers (Lawrence *et al.*, 2014, p. 6).

It can be defined in both limited and broad manner (Adams, Smart and Huff, 2017, pp. 2–3). According to (Levin, 2014, p.1) something that is not published or, lack bibliographical control which makes it difficult to find. This involves raw data, dissertations, thesis or governmental or institutional reports but, fortunately all these can be found easily on internet these days. (Schöpfel, 2011, p.3) studied about grey literatures and described the definition as *"Grey literature stands for manifold document types produced on all levels of government, academics, business and industry in print and electronic formats that are protected by intellectual property rights, of sufficient quality to be collected and preserved by library holdings or institutional repositories, but not controlled by commercial publishers i.e., where publishing is not the primary activity of the producing body."*

It has been a barrier to include different types of grey literature in the reviews, as it hasn't been added into academic discipline-specific databases. However, because of digitalization, the volume and importance of this form of writing has risen (Adams, Smart and Huff, 2017, p. 3). Since grey literature is not limited by the same publication norms as white literature and takes on several formats, data management, extraction, and synthesis are complicated. For example, because grey literature often lacks an abstract, it is frequently impossible to assess the document's relevance or other inclusion requirements without first reading the full text (Benzies *et al.*, 2006, p. 59).

3.1.1 MLR in general

In other disciplines, such as educational research, MLR was established in the early 1990s as SLR that comprises of both academic (formal) and grey (informal) literature (Garousi, Felderer and Hacaloğlu, 2017, p. 5). While the terms "MLR" and "multivocal" are being used by researchers, many sources continue to refer to "grey" literature and how to integrate it in SLRs. For example: (Benzies *et al.*, 2006) addresses the benefits and drawbacks of including grey literature into systematic assessments of the evidence available in the context of evidence-based nursing addresses the benefits and drawbacks of including grey literature into systematic assessments of the available evidence in the context of evidence-based nursing (Garousi, Felderer and Mäntylä, 2016, p. 1; Garousi, Felderer and Hacaloğlu, 2017, p. 5).

(Hopewell *et al.*, 2007, pp. 6–7) performed a study of five research in the field of evidence-based medicine to compare the effect of including or excluding 'grey' literature of some random medical trial using meta-analysis. Results of study showed that there are more participants on an average in a literature that is published formally and also highlighted that there is very limited proof to prove that if the trials published in grey have poor quality methods than in formally published (Garousi, Felderer and Mäntylä, 2016, p. 2; Garousi, Felderer and Hacaloğlu, 2017, p. 5).

3.1.2 Why use MLR

As explained before an MLR is another form of SLR which includes both peer reviewed and non-peer reviewed (Islam, Babar and Nepal, 2019, p. 2). This work encapsulates the view or viewpoint of a broad group of researchers (academics, practitioners, journalists, independent research, and development firms and others). This type of writing takes on a number of

different formats. They reflect a range of objectives, views, and data sources, covers many aspects of a subject and employ a variety of logics (Ogawa and Malen, 1991, p. 265).

(SLR) has become the most often used approach for doing a literature review in Software Engineering (SE). SLR is limited to scientific contributions and excludes grey literature. SLR cannot always give an established body of knowledge because it disregards a substantial quantity of information generated by software engineering (SE) practitioners (Islam, Babar and Nepal, 2019, p. 2). For a practitioner-oriented domain such as SE, it is necessary to synthesize and combine both state-of-the-art and –practice. However, the fact that the vast majority of practitioners in SE do not write in academic journals which implies that their voices are muted when review studies do not include grey literature in addition to academic research. MLRs too have begun to arise in recent years in SE. Additionally, the necessity for additional MLRs in software engineering has been highlighted and experimentally examined recently, which is of significant importance to both research and practice (Garousi, Felderer and Hacaloğlu, 2017, p. 3,5). In this thesis, MLR has been used as a method to collect the information.

3.1.3 Search technique and selecting source

To have a clear, holistic, and unbiased opinion on the relationship between SaaS architecture and pricing models. Three research question have been defined which are demonstrated in the section 1 as well.

RQ1: How do changes made in SaaS architecture affect pricing models?

RQ2: Does SaaS architecture limit pricing models?

RQ3: What is the impact of pricing models on SaaS architecture?

The overall purpose of choosing all these questions is to try and find the relation between SaaS architecture and pricing models. Not much information or publication can be found that gives clear idea on this. RQ1 asses if there is a change in SaaS architecture such as the characteristics that the architecture possesses or if the architecture is altered based on the requirement of the firm or the client. RQ2 investigates that if the SaaS architecture is chosen beforehand, will it restrict the pricing model by going through the information provided by

practitioners in the previous publications. RQ3 identifies the impact of pricing on architecture via the current knowledge of pricing models that is provided by the scholars and practitioners and see if the information is reliable or not. Finding answers to all these research questions is dependent of the previous studies whether enough is available on the internet or not.

Defining the research questions is followed by dictating the sources and defining the search strings that are used to collected information for this study. Considering the large amount of previous work that has been done on the topic of this study, the procedure for collecting the data was based on running automated searches across several scientific databases and digital libraries. A large amount of literature can be found on the internet regarding this subject, the

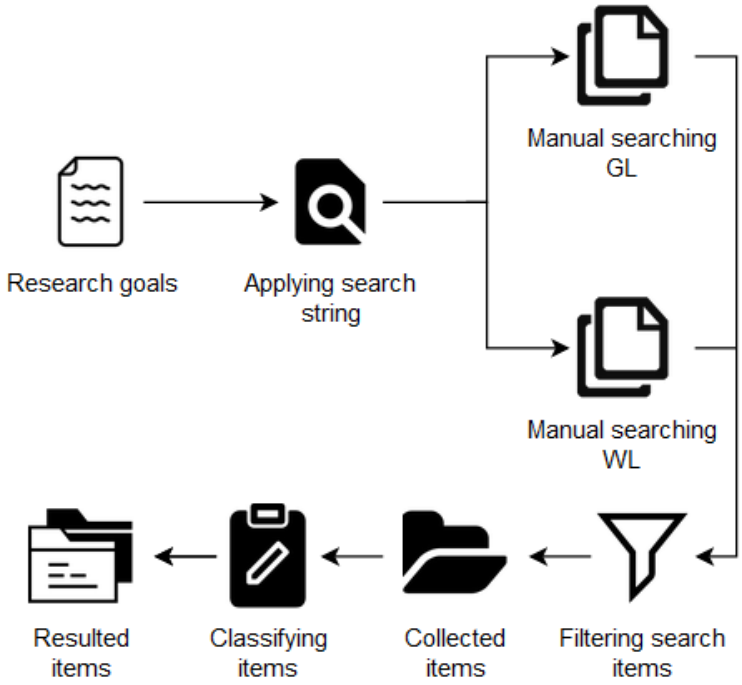

Figure 16. Research procedure.

method of collecting the data used many different database and libraries that are digitally available and search engine which includes LUT primo, Scopus, ScienceDirect, Xplore and google scholar. The above-mentioned databases, libraires and search engine were chosen because they cover all the crucial journal article, books, conference proceedings, blogs, personal blogs, and web pages. Google search engine was used for the items that belonged to GL whereas other libraries and databases focused on WL.

Since the objective of this study is to find if there is any relation between SaaS pricing and architecture. Three different search strings were used to collect the data to solve the purpose of this study that are explained below. The purpose of choosing three different search string rather than a single string was to collect all the appropriate literature there is and, also the idea was to cover all the literature available starting from the SaaS architecture to SaaS architecture and pricing models and as we moved to the third string it was clearly seen that items filtered automatically, and the number of items decreased which is shown below after applying these search terms. The GL was collected using the same protocol and using the same search engine that have been mentioned above.

At both stages, the same search query strings were deployed. They were developed in order to get the most pertinent findings on concerns concerning SaaS architecture and pricing models. Their construction was done for this purpose. The searched were performed using the following search string, which is a mix of keywords and operators that are specified above. These searches were performed in various digital libraries, search engines, and databases.

The first string used was **("SaaS" OR "software as a service") AND ("architecture")** at all the libraries, databases, and search engine as mentioned above. This first search term was defined based on keywords combined with operators which are most relevant to this study. After applying this search string, we found data large amount of data as shown in Figure 17 below. The limit of search was set to first 150 items but when the relevant data was looked at, search stopped showing after first 80-100 items. While doing the research, it was also noticed that most of the journal article, books, conference proceedings, and web pages were getting repeated in all the libraries and databases including google scholar which was mostly considered for GL which is explained later. This search term showed 661 items in LUT primo. The number of items increase successively as the search string was applied in other libraries and databases like Scopus showed 1,594 items, ScienceDirect showed 5,302 items, Xplore showed 27,526, google scholar showed 71,900 items whereas google showed 82400 items, most of which are considered under GL.

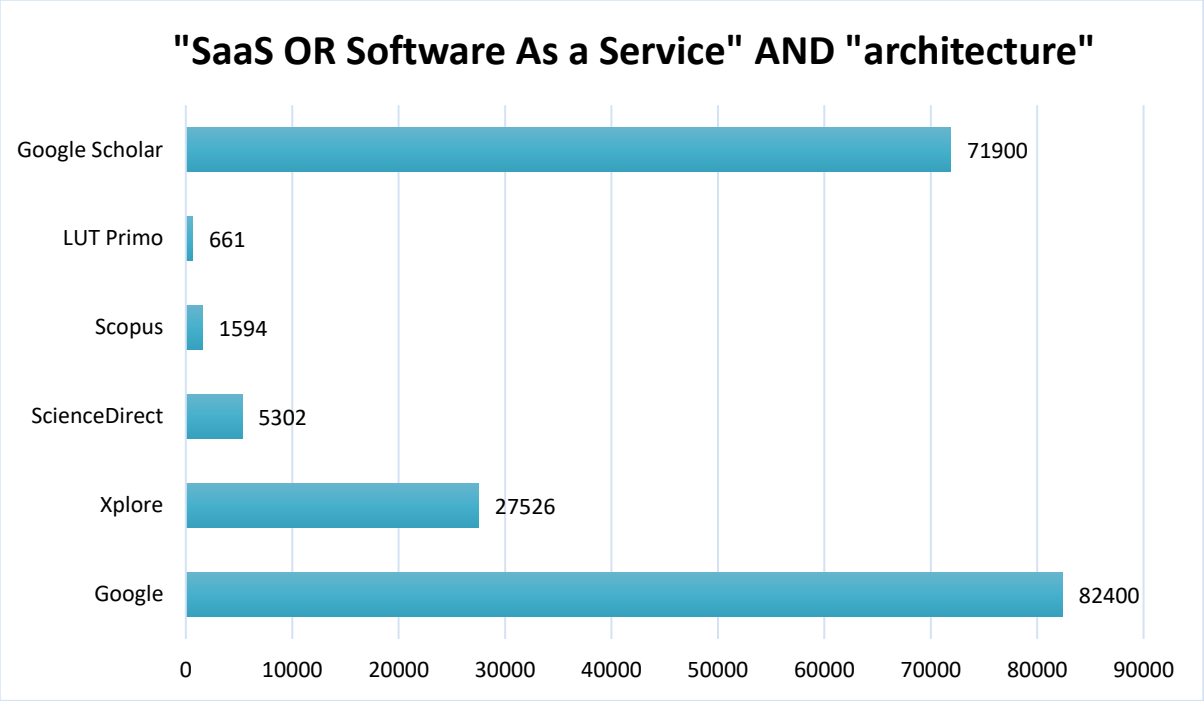

Figure 17. First search term

The second search string used was **(“SaaS” OR “software as a service”) AND (“pricing” OR “"pricing models”)** which showed less items as shown in Figure 18 if compared to the first string in which “architecture” was used as the second keyword. In this search string

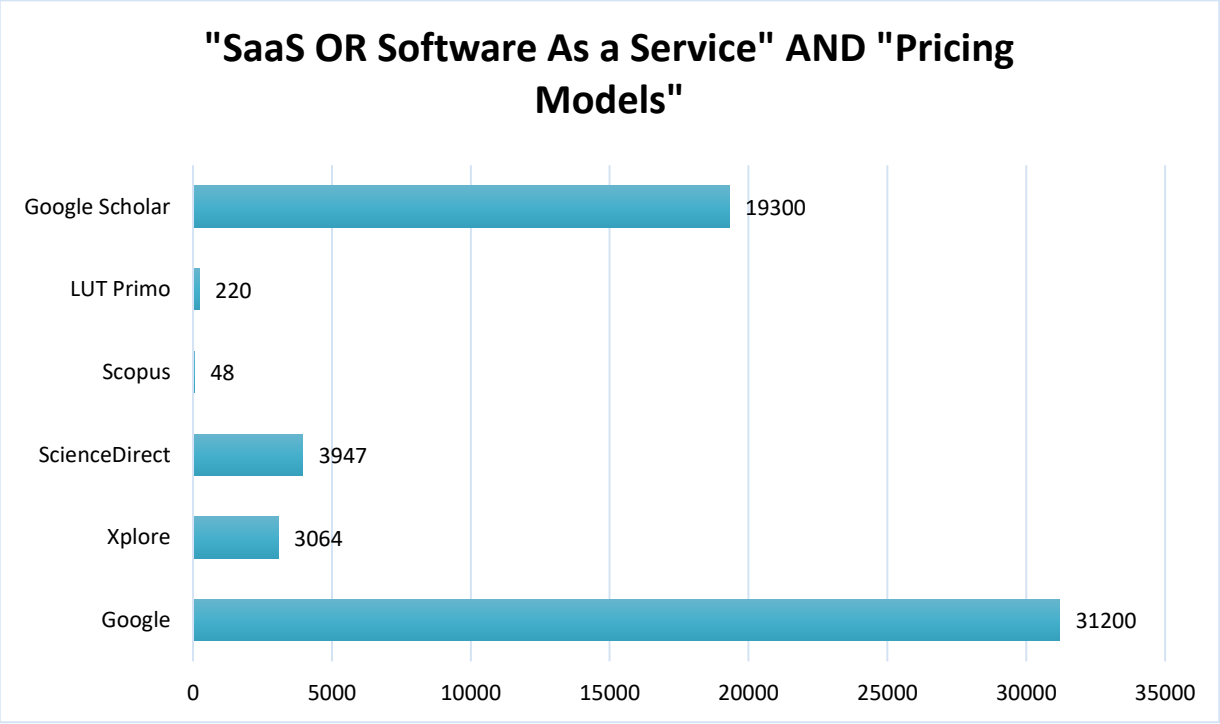

Figure 18. Second search term.

LUT primo showed almost 3 times less items as compared to the first string i.e., 220, Scopus showed only 48 items which is least number as compared to other digital libraries and databases. ScienceDirect showed 3,947 items whereas Xplore showed 3,064, google scholar showed 19,300 and google showed 31200.

The third string which is most important for this study was **(“SaaS” OR “software as a service”) AND (“architecture”) AND (“pricing” OR “"pricing models”)**. After looking at the results, it was perceived that this string showed most relevant item as compared to the two above which the numbers told as seen in the Figure 19 below. Most of the duplicate items were filtered in the search term if we compare it with the first 2.

As the figure 19 depicts the number of literatures decreased in the third search string when compared to the first two terms. LUT primo and Scopus library only showed 74 and 6 items respectively. Each item was considered from these two libraries to collect the literature about this study whereas in others search was stopped after 150 items out of which around first 80-100 were considered.

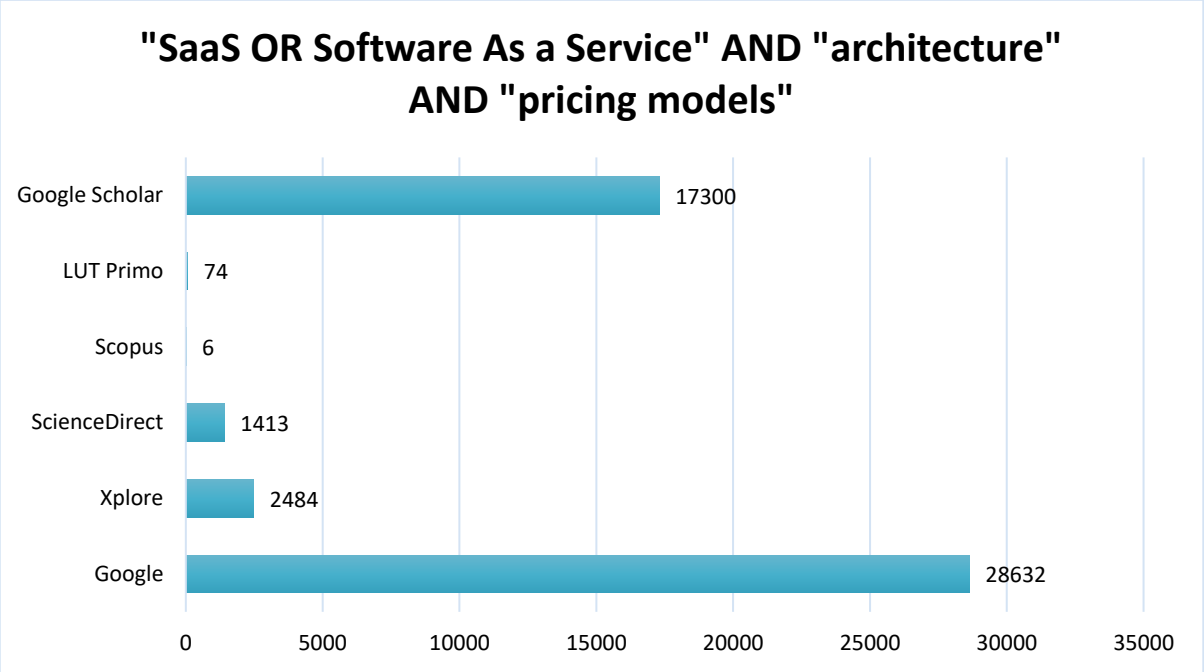

Figure 19. Third search term.

It is clear from the Figure 18 that the number of publications in the WL sector has decreased over time while on the other hand the number has risen in the GL sector. Publication found above were divided in the group of GL and WL. Most of the literature found in google was

GL, a total of 119 items were selected whereas all the WL was found in digital libraries that are for this study with a total of around 130 publications. Out of those WL that

Table 6. Collected items.

Library/database	*URL*	*Number of publications at initial stage*	*Final number.*	
			WL	***GL***
Google	https://www.google.com/	28632		80
Xlpore	https://ieeexplore.ieee.org/Xplore/home.jsp	2484	34	-
ScienceDirect	https://www.sciencedirect.com/	1413	28	-
Scopus	https://www.scopus.com/home.uri	6	4	-
LUT primo	https://lut.primo.exlibrisgroup.com/discovery/search?vid=358FIN_LUT:PUBLIC	74	15	-
Google Scholar	https://scholar.google.com/	17300	50	39
TOTAL (selected items and removing duplicates)			42	28

were mostly searched in digital libraries, 34 were found in Xplore, 28 of them were from ScienceDirect, only 4 were from Scopus and 15 publications in LUT primo. Once the duplicate items were removed from the selected publications 42 WL and 28 GL were chosen to collect the information in this study. Distribution of the collected items has been shown in Table 6.

An automatic search across scientific databases and digital libraries with a so-called backward and forward search was done on the publications that were collected, which was based on the number of citations in Google Scholar, to ensure that nothing is missed during the study's findings. As a result of this approach, it was possible able to identify papers that either did not specifically address SaaS architecture and pricing models but did cover certain aspects and provide valuable insights, or papers that used various synonyms for the term SaaS such as cloud computing or online services and talking about the issues that are close

to SaaS architecture and pricing models or providing any relation between them and talking about the issues that are relevant to this topic.

4 ANALYSIS OF COLLECTED LITERATURE

The gathered material of WL and GL is examined in this section from a variety of angles, as described by the research questions that were posed in one of the previous sections.

The searched items were distinguishing them based on the year. For that, the data collected after using the third search term (“SaaS” OR “software as a service”) AND (“architecture”) AND (“pricing” OR “"pricing models”) was used because this search term showed the most relevant data to this study. After the process as explained above, the number of items were very less from the year 1995 to 2005. All the digital libraries and databases showed less than 50 items each and in Scopus no publishes were found which are related to this thesis whereas 43, 34 and 3 journal, articles, books were found in other directories Xplore, ScienceDirect and LUT primo respectively. In the year gap of 2006-10, the publishing of papers, journal articles, blogs, etc. increased large amount as the number reached to 429, 100, 3, and 40 in the directories Xplore, ScienceDirect, Scopus and Primo respectively whereas in Scholar the number went from 229 to 2400 in this year gap. Most of the publication found in Scholar were categorized as GL. The number kept on rising with almost same amount in the following year gap from 2011-15. But when the year gap was moved 2016-21, the number

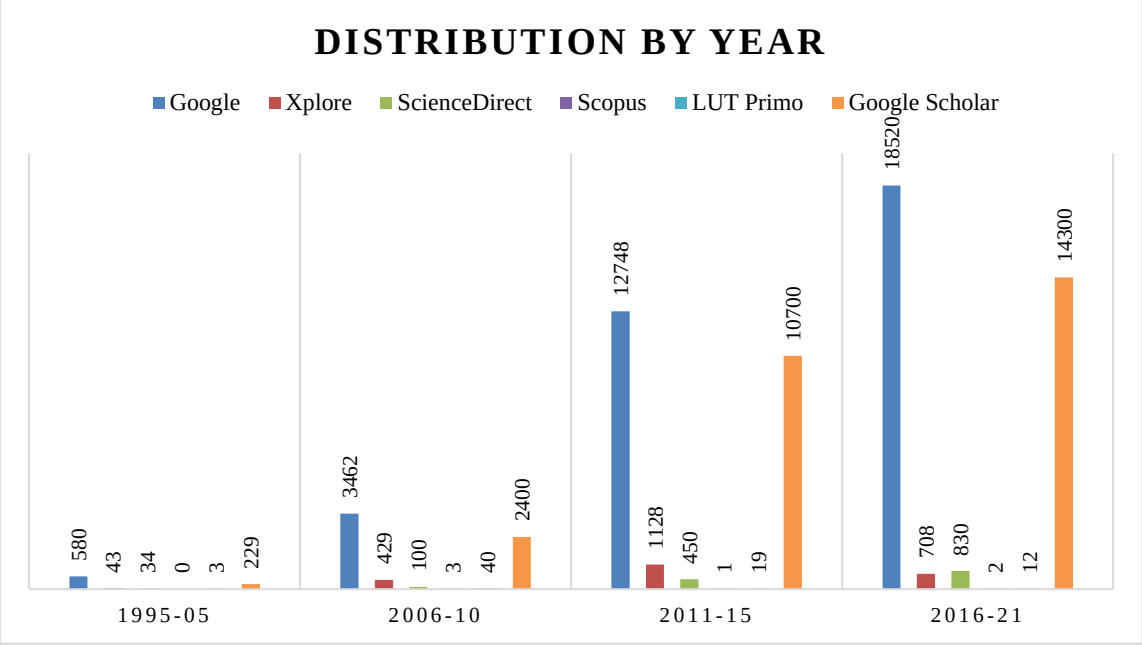

Figure 20. Distribution by year.

of WL publication decreased in number but the number of GL publications for which Google scholar was used kept on rising. The number of publications in this year gap were 708, 830,

2 and 12 which was less when compared to the years 2011-15 and Google scholar showed 14300 items during the initial research which was more in number when compared with previous years as shown in Figure 20.

In this master thesis, search was done for both GL and WL but as it can be seen from the collected number of publications in the methodology section that more focus is put on WL. A total of 28 GL was picked after removing duplicates. When the number of publications was compared, there are 70 items in total which includes both WL and GL. Out of those items, 42 publications were WL and 28 were GL as described above. A total of 60% of WL and 40% of GL as the pie chart depicts in the Figure 21 below were chosen to collect the information to cover the objectives of this master thesis.

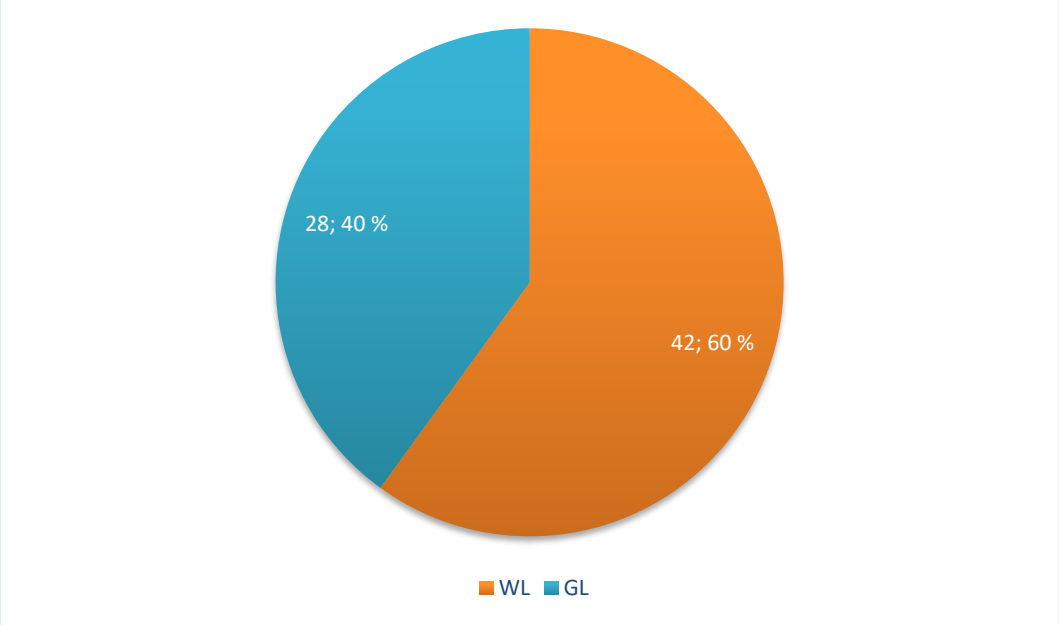

Figure 21. Distribution by type.

Once the publications were selected, a list was made which consisted of the names of all the practitioners and scholars by whom the collected it items were published. Table shows the names and the number of papers, journal, etc., were published by whom in the increasing order. The list consists of scholars and practitioners from both WL and GL. This list does not talk about too much information, just the names of the field in which were working. With the help of this list, it also came to knowledge that a few of these experts were co-author in other publications. This table does not show all the names but covers all the famous authors who has contributed in most of the publication that is used to answer the questions of this master's study.

From the Table 7 below, one of the authors of WL was a co-author in 5 publications, there were few experts who were co-author is 3 and some in 2 publications while the rest of them were author in 1 publication only. These experts did not focus completely on this topic but had the information which has been used to in the master's thesis. 2 publications were there that were close to what the objective of this thesis is. Ojala, A. and Laatikianen, G are the authors of one of these two publications.

Table 7. List of experts.

WL Name of Author	Number of publications	GL Name of Author	Number of publications
Tsai, W.	5	Campbell, P.	9
Bai, X.	3	Lemkin, J.	6
Ojala, A	3	Balaji, S.	4
Laatikainen, G	3		
Huang. Y	3		
Sun, W.	2		
Liu, F.	2		
Chou, W.	2		
Kang, S.	2		
Aleem, S.	1		
Lehmann, S.	1		
Chen, H,	1		
Baumann, P.	1		
Zhang, Z.	1		
Sami, H.	1		

The list of authors with their number of publications was a step forward to move towards the objective of this thesis. With the help of Table 7, the publications were distributed in different types which were journal article, conference proceedings, thesis, and book. A total of 42 publications that were collected are journal article which makes a total of 76% from the whole collection of WL. Only 7 conference proceedings were which in relation to this topic which makes it 17% whereas only 2 thesis and 1 book which makes 5% and 2% simultaneously were found that had the required information needed for this thesis in order to complete the objective. Figure 22 below gives the overview of the data with their percentage.

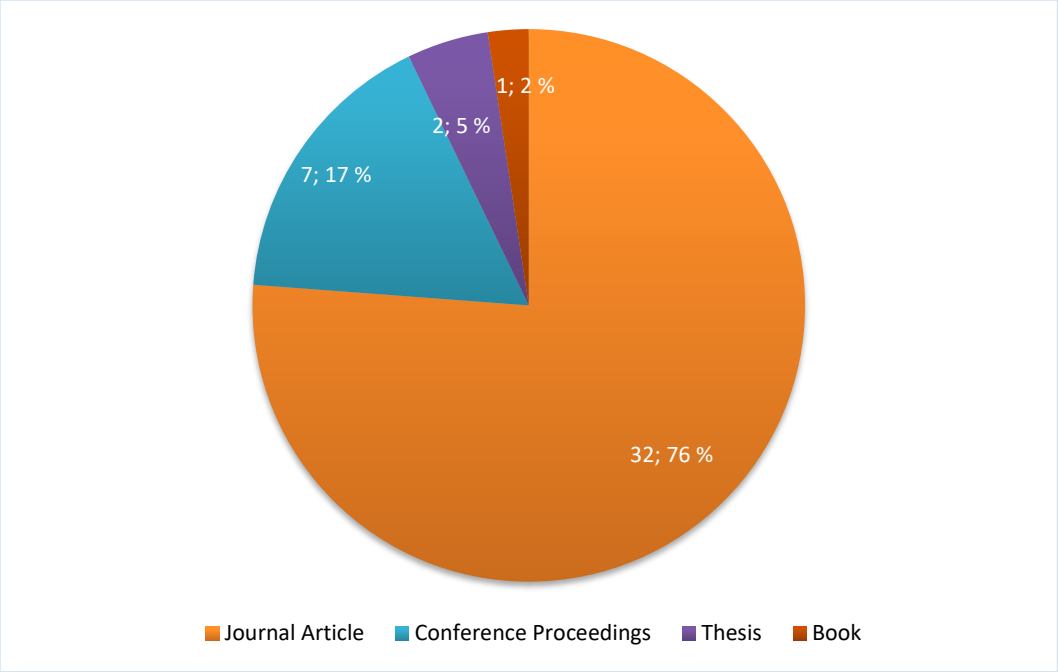

Figure 22. Types of WL publications.

A total of 28 GL was chosen with the assistance of the above explained methodology. Imparting from these publications 19 were labelled as webpages/blog which was 68% of the final selected ones. Narrowing does these items, 6 book sections (21%) and 3 company reports (11%) resulting in the total outcome. Figure 23 below summarizes this data.

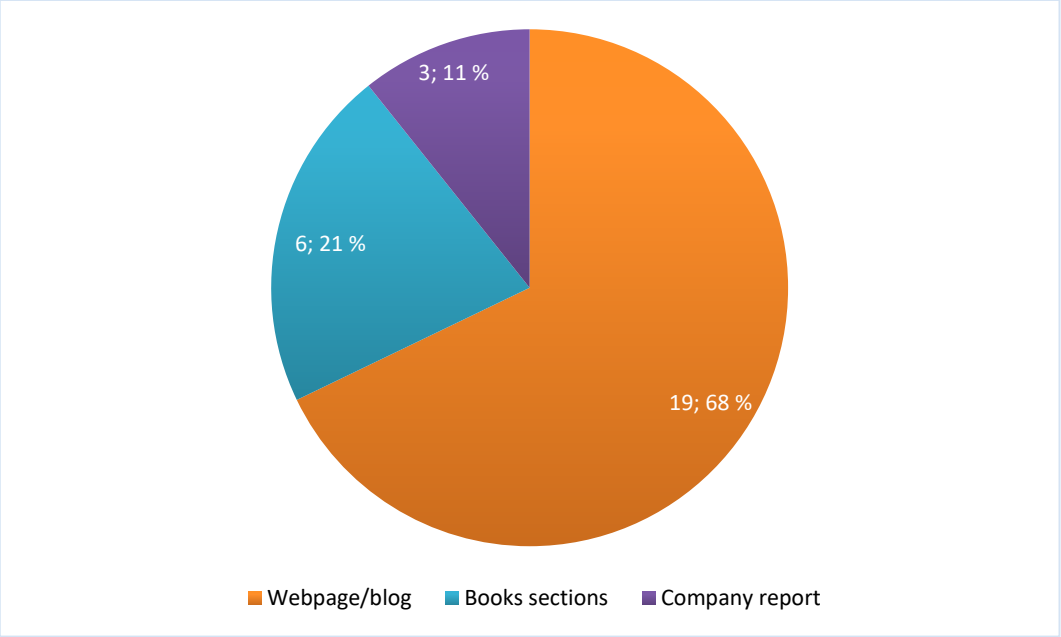

Figure 23. Type of GL publications

To sum up, after collecting all the items using digital libraries and databases and applying filters on them, the analysis was done. Starting from the distribution of the publications based on the year it was noted that the number of WL publication is declining with time starting from 1995 where the number of GL has increased.

A total of 70 items were chosen in which 28 are GL and 72 are WL that makes 40% and 60% simultaneously of the total collected publications. After that, a list of was made which has the name of all the publishers along with the number of papers published by them starting from the ones who had the most number of publications to the less, that are related to this master's thesis topic on the basis of the search terms used that is explained in the section 3 above. It is clearly notable that an author named Tsai, W. has the most amount of WL and Campbell, P. has the most WL i.e., 5 and 9 each.

These publications were further distributed individually based on the type. Their types have been categorized as journal article, conference proceedings, thesis, and books. It is clearly recognized that the maximum number of publications in WL is journal article which is 32 in number or 76% in terms of percentages. On the other that in GL webpage/blog is at the top with 68% or 19 in number.

4.1 Classification

Information gathered from the WL and GL collected in this thesis helped in classifying the point out the relation and effect of SaaS architecture and pricing models on one another. It is explained earlier the architecture needs to have some set of characteristics in order to have a well-designed architecture and how that will affect the decision of choosing pricing models (Saltan and Smolander, 2021, p. 14).

Table 8 (Wei-Tek Tsai and Xin Sun, 2013, p. 1; Laatikainen and Luoma, 2014, p. 246; Laatikainen and Ojala, 2014, p. 600; Hyrynsalmi, Rauti and Kaila, 2019, p. 1765) explains the relation between SaaS architecture and pricing models and the reasons of effect on each other. The first and section research question addresses the relationship between architecture and pricing models and how does it limit the pricing models, and the first two columns provide answers to that topic. The impact that price has had on the design of the building is discussed in the last column. It is clear from looking at the table that the link between price and architecture shifts in different ways based on the information that has been accumulated up to this point. This does not prove any direct relation between SaaS architecture and pricing

models rather it shows that there might or might not be a possible relation between the two depending upon how important to the companies.

Table 8. Relation between SaaS architecture and pricing models.

SaaS architecture effect on pricing models	**Architecture limits pricing**	**Pricing models affects the architecture**
There are small firms that focus on the development of the product that creates a weak connection between the architecture and pricing models.	An architecture that is well-designed has all the above explained characteristics which might lead to a fixed pricing model i.e., no pricing negotiations.	Different pricing models needs variety of characteristics in the architecture. Like, method of delivery or any additional components.
An architecture design that is scalable and highly modularized design makes it possible to implement a variety of price models, including subscription-based models, licenses, a variety of bundling possibilities, and pricing that is dependent on consumption.	The architecture gives the idea on how the pricing models can be chosen rather than limiting it.	To use a particular pricing for e.g., usage based. The pricing must go down which might lead to a new architecture with fewer characteristics.
With the help of public cloud, setting up a new pricing model or remove one makes it easier and simpler.	Using a multi-tenant architecture leads to higher margins in profits. So, if the profit is not there, it will limit the pricing.	The pricing does not have any impact at all on the architecture if the focus of a company is on architecture.
A set fee rather than one dependent on the amount of data used is required for migration to a SaaS	In examples of small firms, interaction between the architecture and pricing is not there which leads to	If the pricing is fixed it will automatically lead to an architecture with poor design

architecture and utilization of public cloud. Any change in architecture makes it possible to adjust the cost. The prices are reduced because of the introduction of SaaS architecture.	limiting the pricing once the architecture is decided.	because of the limited resources.
Architecture that possesses characteristics of being flexible and configurable, it helps in enabling different pricing models.	It is difficult to import the desktop applications. Such as changing from license to subscription.	A new pricing model will require a new architecture to fulfill the needs of the customer and the provider.

4.2 Observations

Most of the scholar and publishers did not talk about the relation between the architecture and pricing models. A larger number of publications focused on the architecture and pricing models separately. Existing studies showed that, to have a well-designed architecture properties like customization, scalability, redundancy, security, virtualization, integration, fault-tolerance, and multi-tenancy are needed. Success in SaaS architecture is attributed to the capacity to customize and scale (Hyrynsalmi, Rauti and Kaila, 2019, p. 1765). Availability is a value proposition and a success component, and redundancy helps accomplish this. MTA is supported by virtualization. Mature SaaS models are intertwined with the architecture of SaaS solutions. In accordance with the maturity model that was established. If an application has Customization, Scalability, or Multi-tenancy, it is considered mature. If it does not, it is considered immature. The most advanced SaaS application has virtualization as an architectural feature (Laatikainen, Ojala and Mazhelis, 2013, p. 124).

Although a wide number of characteristics have been mentioned but not all have received same importance. It was also learnt for these publications that all these properties are not important in every SaaS architecture. It depends on the requirement of the SaaS product, the firm and also on the budget. According to the most practitioner's customization, multi-tenancy and scalability are the most important characteristics in every SaaS architecture

(Gao *et al.*, 2011, p. 61). Though virtualization was discussed in the publications but was not discovered in the case study, a source code inspection of individual services could reveal whether or not it was used. Both in the literature research and in the case study, it was discovered that there was redundancy too.

Pricing models can be chosen based on many distinct aspects. Many practitioners have explained different frameworks for pricing models. Most common way to choose suitable pricing is dividing it into value-based and usage-based. This technique has been adopted in by most of the publishers (Saltan and Smolander, 2021, p. 16). Another most common technique to choose pricing is by acquiring SBIFT model which is a highly effective model according to many scholars. Additionally, not many explanatory examples are provided by the practitioners in their studies.

Even though scholars and practitioners developed a number of ways targeted at structuring and A number of ways are developed by scholars and practitioners that are targeted at structuring and dissecting SaaS pricing from a variety of viewpoints and perspectives (Saltan and Smolander, 2021, p. 12). The recommended alternatives, on the other hand, are incompatible with one another, and no information has been provided on how they are really being used by SaaS providers.

Only two publications were found that focuses on explaining the relation between SaaS architecture and pricing models explicitly. One of those two concluded that the architecture and pricing has been tightly intertwined. Some relation between the two has been found in their case study. Few companies in their case study are migrating its product to a SaaS architecture, the pricing model will become more straightforward, and costs will be reduced overall. Other firms are considering architectural and pricing characteristics are kept in mind during the decisions related to technical and pricing details where it is believed that an architecture that is well-designed makes it possible to a variety of pricing strategies in addition these selected pricing structures necessitate the use of high-quality software. They also concluded that while a flexible and well-planned architecture permits a variety of price strategies, a badly built architecture also constrains pricing. Scalability and a high degree of modularity are critical qualities since they enable a diverse range of pricing structures (Laatikainen and Ojala, 2014, pp. 599–600).

The other used the same criteria as this master's but with less publications. They concluded that, there is a complete absence of practitioner-friendly tools, procedures, and models (Hyrynsalmi, Rauti and Kaila, 2019, p. 1764). Furthermore, while numerous studies have emphasized the importance of company flexibility and the necessity for change, none have provided practitioners with practical advice.

5 DISCUSSION

In this part, reviewing and analyzing of the research findings has been done, as well as recommendations for future are study in the field of SaaS architecture and pricing models, with the goal of filling any gaps that might have been missed in this research.

SaaS architecture and SaaS pricings models are very wide subjects individually. These two are the topics that has been studied from a variety of angles by researchers in a variety of fields. This study talks about the dependency and the relation between SaaS architecture and pricing models. These papers have used a variety of approaches and addressed a wide range of topics related to SaaS architecture and pricing models. It became evident that academia has failed to provide a cohesive body of knowledge on SaaS architecture and pricing models, and a number of research options remain open for further exploration.

It has been acclaimed after doing the analysis that the number of WL publications in the field of SaaS architecture and pricing models has decreased in the recent years whereas in GL the number of publications is always on the rise through time. The analysis also shows that the number of WL has increased until the year gap 2010-2015 but there is a decline in number from the year 2015 till date. There is no concrete reason behind the decreasing of these number. Possibly the publishers do not find this topic encouraging enough in the field of research.

Till date, the study on this topic shows a lack of connection between theory and practical research. While the frameworks and models provided to aid with SaaS pricing decision-making were helpful, there was a lack of guidance in the models and frameworks. Although there were a few indications that researchers were using their results in the real world, there was little evidence that researchers expected this to happen as they rarely assured the further use of their studies. The analysis of this GL and WL publications shows that the information was clearer and more systematic in WL as compared to GL. These collected publications

have revealed that there is no common method to defining and implementing the architecture and pricing models.

After understanding and explaining about the architecture and pricing models, it is understood from this thesis that SaaS architecture has many characteristics. Whenever a SaaS vendor make a SaaS application must keep these characteristics in mind based on their requirement. The pricing models chosen to help these vendors in understanding the expenses that can be there during the various stages of the development process or even after that. Many different strategies and characteristics regarding these two topics have been described in this research. Based on these strategies and characteristics, this master's thesis answers the research questions.

5.1 Limitations and further scope of the study

The fact that restricts this study is that it is based on theoretical research i.e., the best possible information that could be found which is available on the internet in the form of GL and WL has been used. Moreover, this study collects the information from these publications and is limited to the publications that have been published after 1995. Even though a few of the publishers have tried to explain and solve this dependency issue by collecting the information from different firms based on the age of the company and number of employees e.g., startups, old companies, etc. but is limited because they have tried to gather information from 4-5 companies in total.

The databases that were employed in this study had certain limitations. Examples include articles that were only discovered in certain databases, such as some scientific journals. As a result, the selection of the databases to be employed in a systematic literature analysis is critical to its success. This study was able to overcome this constraint by utilizing databases and digital libraries. However, it is possible that some work was overlooked as a result of the selection process.

If the study goes further, a clearer answer to the questions of this thesis can be found if the methodology used was more practical than theoretical. For example, if companies were given a questionnaire or a google from that has the question regarding their company and they were to be answered by someone who has knowledge in this area, might have helped the study in a better way. From the firm's point of view, this study will help the firms in

deciding their architecture and pricing models carefully. As it somewhat clears the concept of their dependencies on one another.

6 CONCLUSION

A clear answer to all these questions was not found in this study but few things were concluded during the observation. To achieve a well-designed architecture, different pricing models have been explained. There are many different characteristics of SaaS architecture. It is not important to have all these characteristics in an architecture. It can be based on the initial planning of the software, on the budget or the requirement so it does have an impact on the pricing models. More the characteristics added to the architecture, the pricing models must be decided or should be changed if decided already accordingly.

The pricing models is decided based on the budget and the requirement of the firm and that leads to the planning of the architecture properly. Because if the architecture has been planned and there is no scope of modification, there might be some difficulties in achieving the target as there is a chance that pricing models does not allow to have all the characteristics in the SaaS product at the same time. To achieve a multi-tenancy architecture, there are 4 stages as explained in this thesis. It is upon the firm or SaaS vendor whether to achieve all the stages or not. Pricing models play a big role in that too as it will restrict in achieving MTA if it was not planned during the pricing models planning.

REFERENCES

A. Kofahi, N. (2018) ‘Identifying the Top Threats in Cloud Computing and Its Suggested Solutions: A Survey’, *Advances in Networks*, 6(1), p. 1. doi:10.11648/j.net.20180601.11.

Adams, R.J., Smart, P. and Huff, A.S. (2017) ‘Shades of Grey: Guidelines for Working with the Grey Literature in Systematic Reviews for Management and Organizational Studies: Shades of Grey’, *International Journal of Management Reviews*, 19(4), pp. 432–454. doi:10.1111/ijmr.12102.

Akande, A.O., April, N.A. and Van Belle, J.-P. (2013) ‘Management Issues with Cloud Computing’, in *Proceedings of the Second International Conference on Innovative Computing and Cloud Computing - ICCC ’13*. *the Second International Conference*, Wuhan, China: ACM Press, pp. 119–124. doi:10.1145/2556871.2556899.

Aleem, S. *et al.* (2019) ‘Empirical Investigation of Key Factors for SaaS Architecture’, p. 14.

Almubaddel, M. and Elmogy, A.M. (2016) ‘Cloud Computing Antecedents, Challenges, and Directions’, in *Proceedings of the International Conference on Internet of things and Cloud Computing*. *ICC ’16: International Conference on Internet of things and Cloud Computing*, Cambridge United Kingdom: ACM, pp. 1–5. doi:10.1145/2896387.2896401.

AlMutair, L. and Zaghloul, S.S. (no date) ‘A NEW VIRTUALIZATION-BASED SECURITY ARCHITECTURE IN A CLOUD COMPUTING ENVIRONMENT’, p. 11.

Alzakholi, O. *et al.* (2020) ‘Comparison Among Cloud Technologies and Cloud Performance’, *Journal of Applied Science and Technology Trends*, 1(2), pp. 40–47. doi:10.38094/jastt1219.

Ankita Sharma, Sonia Vatta (2013) ‘Cloud Computing: Concepts and Architecture’, *International Journal of Advanced Research in Computer Science and Software Engineering*, 3.

Armbrust, M. *et al.* (2009) ‘Above the Clouds: A Berkeley View of Cloud Computing’, p. 25.

Atul Kumar (2020) *Cloud Deployment Models | Public, Private and Hybrid, Cloud Training Program*. Available at: https://k21academy.com/cloud-blogs/cloud-computing-deployment-models/ (Accessed: 21 December 2021).

Benzies, K.M. *et al.* (2006) ‘State-of-the-Evidence Reviews: Advantages and Challenges of Including Grey Literature’, *Worldviews on Evidence-Based Nursing*, 3(2), pp. 55–61. doi:10.1111/j.1741-6787.2006.00051.x.

Bhardwaj, S., Jain, L. and Jain, S. (2010) ‘An Approach for Investigating Perspective of Cloud Software-as-a-Service (SaaS)’, *International Journal of Computer Applications*, 10(2), pp. 44–47. doi:10.5120/1450-1962.

Böhm, M. *et al.* (2014) ‘Cloud Computing and Computing Evolution’, p. 29.

Campbell-Kelly, M. (2009) ‘Historical reflectionsThe rise, fall, and resurrection of software as a service’, *Communications of the ACM*, 52(5), pp. 28–30. doi:10.1145/1506409.1506419.

Cheshire, J. (2019) *Exam ref az-900 microsoft azure fundamentals*. 1st edition. San Francisco, CA: Microsoft Press.

Choudhary, V. (2007) ‘Comparison of Software Quality Under Perpetual Licensing and Software as a Service’, *Journal of Management Information Systems*, 24(2), pp. 141–165. doi:10.2753/MIS0742-1222240206.

Christopher Wray (2016) ‘What Does Redundancy Mean In The Cloud?’, *RSAWEB*, 8 November. Available at: https://www.rsaweb.co.za/what-does-redundancy-mean-in-the-cloud/ (Accessed: 9 September 2021).

Cruz, V. (2021) ‘Cloud Computing Deployment Models: An Overview of Different Types’, *Market Business News*, 21 June. Available at: https://marketbusinessnews.com/cloud-computing-deployment-models-an-overview-of-different-types/268425/ (Accessed: 23 September 2021).

Cusumano, M.A. (2008) ‘The Changing Software Business: Moving from Products to Services’, *Computer*, 41(1), pp. 20–27. doi:10.1109/MC.2008.29.

Diaby, T. and Rad, B.B. (2017) ‘Cloud Computing: A review of the Concepts and Deployment Models’, *International Journal of Information Technology and Computer Science*, 9(6), pp. 50–58. doi:10.5815/ijitcs.2017.06.07.

Everything as a Service (XaaS) - the evolution of cloud-based Services (2021) *oneclick*^{TM}. Available at: https://oneclick-cloud.com/en/blog/trends-en/everything-as-a-service/ (Accessed: 30 August 2021).

Ganesh, A., Sandhya, M. and Shankar, S. (2014) ‘A study on fault tolerance methods in Cloud Computing’, in *2014 IEEE International Advance Computing Conference (IACC)*. *2014 IEEE International Advance Computing Conference (IACC)*, Gurgaon, India: IEEE, pp. 844–849. doi:10.1109/IAdCC.2014.6779432.

Ganga, K. and Karthik, S. (2013) ‘A fault tolerent approach in scientific workflow systems based on cloud computing’, in *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*. *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME)*, Salem: IEEE, pp. 387–390. doi:10.1109/ICPRIME.2013.6496507.

Gao, J. *et al.* (2011) ‘SaaS performance and scalability evaluation in clouds’, in *Proceedings of 2011 IEEE 6th International Symposium on Service Oriented System (SOSE)*. *2011 IEEE 6th International Symposium on Service Oriented System Engineering (SOSE)*, Irvine, CA, USA: IEEE, pp. 61–71. doi:10.1109/SOSE.2011.6139093.

Garousi, V., Felderer, M. and Hacaloğlu, T. (2017) 'Software test maturity assessment and test process improvement: A multivocal literature review', *Information and Software Technology*, 85, pp. 16–42. doi:10.1016/j.infsof.2017.01.001.

Garousi, V., Felderer, M. and Mäntylä, M.V. (2016) 'The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature', in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*. *EASE '16: 20th International Conference on Evaluation and Assessment in Software Engineering*, Limerick Ireland: ACM, pp. 1–6. doi:10.1145/2915970.2916008.

Grey literature (2021). Available at: https://www.une.edu.au/library/services/support/eskills-plus/research-skills/grey-literature (Accessed: 28 September 2021).

Guo, C.J. *et al.* (2007) 'A Framework for Native Multi-Tenancy Application Development and Management', in *The 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC-EEE 2007)*. *The 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC-EEE 2007)*, Tokyo, Japan: IEEE, pp. 551–558. doi:10.1109/CEC-EEE.2007.4.

Hopewell, S. *et al.* (2007) 'Grey literature in meta-analyses of randomized trials of health care interventions', *Cochrane Database of Systematic Reviews* [Preprint]. Edited by Cochrane Methodology Review Group. doi:10.1002/14651858.MR000010.pub3.

Hyrynsalmi, S., Rauti, S. and Kaila, E. (2019) 'Bridging the gap between software architecture and business model development: A literature study', in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia: IEEE, pp. 1519–1524. doi:10.23919/MIPRO.2019.8756974.

Intel (2014) 'What Is PaaS ? How Offering Platform as a Service Can Increase Cloud Adoption'. Intel Corporation.

International Data Corporation (IDC) (no date) *Worldwide Public Cloud Services Market Totaled $312 Billion in 2020 with Amazon Web Services and Microsoft Vying for the Top Position Overall, According to IDC, IDC: The premier global market intelligence company*. Available at: https://www.idc.com/getdoc.jsp?containerId=prUS47685521 (Accessed: 17 August 2021).

Islam, C., Babar, M.A. and Nepal, S. (2019) 'A Multi-Vocal Review of Security Orchestration', *ACM Computing Surveys*, 52(2), pp. 1–45. doi:10.1145/3305268.

Iveroth, E. *et al.* (2013) 'How to differentiate by price: Proposal for a five-dimensional model', *European Management Journal*, 31(2), pp. 109–123. doi:10.1016/j.emj.2012.06.007.

J. Beschi Raja and K. Vivek Rabinson (2016) ‘Iaas for Private and Public Cloud using Openstack’, *International Journal of Engineering Research and*, V5(04), p. IJERTV5IS040191. doi:10.17577/IJERTV5IS040191.

James Ng (2020) ‘IaaS vs PaaS vs SaaS’, *The Startup*, 20 August. Available at: https://medium.com/swlh/iaas-vs-paas-vs-saas-dfece8fd6ca (Accessed: 23 August 2021).

Ju, J. *et al.* (2010) ‘Research on Key Technology in SaaS’, in *2010 International Conference on Intelligent Computing and Cognitive Informatics*. *2010 International Conference on Intelligent Computing and Cognitive Informatics (ICICCI)*, Kuala Lumpur, Malaysia: IEEE, pp. 384–387. doi:10.1109/ICICCI.2010.120.

Kang, S. *et al.* (2010) ‘A General Maturity Model and Reference Architecture for SaaS Service’, in Kitagawa, H. et al. (eds) *Database Systems for Advanced Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 337–346. doi:10.1007/978-3-642-12098-5_28.

Kang, Sungjoo, Kang, Sungwon and Hur, S. (2011) ‘A Design of the Conceptual Architecture for a Multitenant SaaS Application Platform’, in *2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering*. *2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering (CNSI)*, Jeju, Korea (South): IEEE, pp. 462–467. doi:10.1109/CNSI.2011.56.

Khan, A.W. (2012) ‘A Literature Survey on Data Privacy/ Protection Issues and Challenges in Cloud Computing’, *IOSR Journal of Computer Engineering*, 1(3), pp. 28–36. doi:10.9790/0661-0132836.

Laatikainen, G. and Luoma, E. (2014) ‘Impact of Cloud Computing Technologies on Pricing Models of Software Firms – Insights from Finland’, in Lassenius, C. and Smolander, K. (eds) *Software Business. Towards Continuous Value Delivery*. Cham: Springer International Publishing (Lecture Notes in Business Information Processing), pp. 243–257. doi:10.1007/978-3-319-08738-2_17.

Laatikainen, G. and Ojala, A. (2014) ‘SaaS Architecture and Pricing Models’, in *2014 IEEE International Conference on Services Computing*. *2014 IEEE International Conference on Services Computing (SCC)*, Anchorage, AK, USA: IEEE, pp. 597–604. doi:10.1109/SCC.2014.84.

Laatikainen, G., Ojala, A. and Mazhelis, O. (2013) ‘Cloud Services Pricing Models’, in Herzwurm, G. and Margaria, T. (eds) *Software Business. From Physical Products to Software Services and Solutions*. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Business Information Processing), pp. 117–129. doi:10.1007/978-3-642-39336-5_12.

Lawrence, A. *et al.* (2014) ‘Where Is the Evidence? Realising the Value of Grey Literature for Public Policy & Practice, A Discussion Paper’, *Discussion paper*, p. 30.

Lehmann, S. and Buxmann, P. (2009) ‘Pricing Strategies of Software Vendors’, *Business & Information Systems Engineering*, 1(6), pp. 452–462. doi:10.1007/s12599-009-0075-y.

Levin, L.L. (2014) ‘Literature Search Strategy Week: Len Levin on Understanding and Finding Grey Literature’, p. 3.

Liu, F. *et al.* (2010) ‘SaaS Integration for Software Cloud’, in *2010 IEEE 3rd International Conference on Cloud Computing*. *2010 IEEE International Conference on Cloud Computing (CLOUD)*, Miami, FL, USA: IEEE, pp. 402–409. doi:10.1109/CLOUD.2010.67.

Luan, S., Shi, Y. and Wang, H. (2009) ‘A Mechanism of Modeling and Verification for SaaS Customization Based on TLA’, in Liu, W. et al. (eds) *Web Information Systems and Mining*. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 337–344. doi:10.1007/978-3-642-05250-7_36.

Mazrekaj, A., Shabani, I. and Sejdiu, B. (2016) ‘Pricing Schemes in Cloud Computing: An Overview’, *International Journal of Advanced Computer Science and Applications*, 7(2). doi:10.14569/IJACSA.2016.070211.

Mell, P. and Grance, T. (2011) ‘The NIST Definition of Cloud Computing’, *NIST Special Publication 800-145*, p. 7.

Michael Kavis (2014) *Architecting the Cloud - Design Decisions for Cloud Computing Service Models*. Hoboken, New Jersey.: John Wiley & Sons, Inc.

Miyachi, C. (2018) ‘What is “Cloud”? It is time to update the NIST definition?’, *IEEE Cloud Computing*, 5(3), pp. 6–11. doi:10.1109/MCC.2018.032591611.

Mohammed, Chnar Mustafa and Zeebaree, Subhi R. M. (2021) ‘Sufficient Comparison Among Cloud Computing Services: IaaS, PaaS, and SaaS: A Review’. doi:10.5281/ZENODO.4450129.

Mohan, L. *et al.* (2017) ‘A Comparative Study on SaaS, PaaS and IaaS Cloud Delivery Models in Cloud Computing’, p. 3.

Mukundha, D.C. (2017) ‘Cloud Computing Models : A Survey’, © *Research India Publications*, p. 16.

Nitu (2009) ‘Configurability in SaaS (software as a service) applications’, in *Proceeding of the 2nd annual conference on India software engineering conference - ISEC ’09*. *Proceeding of the 2nd annual conference*, Pune, India: ACM Press, p. 19. doi:10.1145/1506216.1506221.

Ogawa, R.T. and Malen, B. (1991) ‘Towards Rigor in Reviews of Multivocal Literatures: Applying the Exploratory Case Study Method’, *Review of Educational Research*, 61(3), pp. 265–286. doi:10.3102/00346543061003265.

Patrick Campbell (2016) *Campbell - 2016 - The Anatomy of SaaS Pricing Strategy.pdf*. 2020th edn. Price Intelligently.

Pinto, V.H.S.C. *et al.* (2016) ‘A Systematic Mapping Study on the Multi-tenant Architecture of SaaS Systems’, in. *The 28th International Conference on Software Engineering and Knowledge Engineering*, pp. 396–401. doi:10.18293/SEKE2016-068.

‘SaaS Maturity Levels’ (2019) *Francesco Arcieri*, 30 October. Available at: https://francescoarcieri.it/saas/saas-maturity-levels/ (Accessed: 23 September 2021).

Saltan, A. and Smolander, K. (2021) ‘Bridging the state-of-the-art and the state-of-the-practice of SaaS pricing: A multivocal literature review’, *Information and Software Technology*, 133, p. 106510. doi:10.1016/j.infsof.2021.106510.

Savu, L. (2011) ‘Deployment models, delivery models, risks and research challanges’, 2, p. 4.

Schöpfel, J. (2011) ‘Towards a Prague Definition of Grey Literature’, p. 24.

Shiliang Wu, Wortmann, H., and Chee-wee Tan (2014) ‘A pricing framework for software-as-a-service’, in *Fourth edition of the International Conference on the Innovative Computing Technology (INTECH 2014)*. *2014 Fourth International Conference on Innovative Computing Technology (INTECH)*, Luton, UK: IEEE, pp. 152–157. doi:10.1109/INTECH.2014.6927738.

Spruit, M. and Abdat, N. (2012) ‘The Pricing Strategy Guideline Framework for SaaS Vendors’:, *International Journal of Strategic Information Technology and Applications*, 3(1), pp. 38–53. doi:10.4018/jsita.2012010103.

Sun, W. *et al.* (2007) ‘Software as a Service: An Integration Perspective’, in Krämer, B.J., Lin, K.-J., and Narasimhan, P. (eds) *Service-Oriented Computing – ICSOC 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 558–569. doi:10.1007/978-3-540-74974-5_52.

Tony Hou (2018) *IaaS vs PaaS vs SaaS: What You Need to Know*, *The BigCommerce Blog*. Available at: https://www.bigcommerce.com/blog/saas-vs-paas-vs-iaas/ (Accessed: 19 August 2021).

Top Threats to Cloud Computing: Egregious Eleven (2019). Available at: https://cloudsecurityalliance.org/press-releases/2019/08/09/csa-releases-new-research-top-threats-to-cloud-computing-egregious-eleven/ (Accessed: 9 September 2021).

Tsai, W., Bai, X. and Huang, Y. (2014) ‘Software-as-a-service (SaaS): perspectives and challenges’, *Science China Information Sciences*, 57(5), pp. 1–15. doi:10.1007/s11432-013-5050-z.

Tsyganov, D. (2018) ‘FUNDAMENTAL PROPERTIES OF SAAS ARCHITECTURE: LITERATURE REVIEW AND ANALYSIS OF DEVELOPMENT PRACTICES’, p. 60.

Turner, M., Budgen, D. and Brereton, P. (2003) ‘Turning software into a service’, *Computer*, 36(10), pp. 38–44. doi:10.1109/MC.2003.1236470.

Venkata Rao J. and D. Bhargava Reddy (2011) ‘Implementation of SaaS in a Cloud Computing Environment’, 2(8).

Wei-Tek Tsai and Xin Sun (2013) ‘SaaS Multi-tenant Application Customization’, in *2013 IEEE Seventh International Symposium on Service-Oriented System Engineering*. *2013*

IEEE 7th International Symposium on Service Oriented System Engineering (SOSE 2013), Redwood City: IEEE, pp. 1–12. doi:10.1109/SOSE.2013.44.

Zhong, L. *et al.* (2010) 'A Virtualization-Based SaaS Enabling Architecture for Cloud Computing', in *2010 Sixth International Conference on Autonomic and Autonomous Systems*. *2010 Sixth International Conference on Autonomic and Autonomous Systems (ICAS)*, Cancun, Mexico: IEEE, pp. 144–149. doi:10.1109/ICAS.2010.28.

Zissis, D. and Lekkas, D. (2012) 'Addressing cloud computing security issues', *Future Generation Computer Systems*, 28(3), pp. 583–592. doi:10.1016/j.future.2010.12.006.