**Software Product Line Architecture for**

**Cloud-based applications**

**Team Members:** 1. Mayur Mehta (A20405901)

2. Suyog Kharage (A20402686)

3. Yash Kotadiya (A20402140)

# Introduction

Microsoft defines cloud computing as “It is the delivery of computing services resources — servers, storage, databases, networking, software, analytics, and more— over the Internet (“cloud”). Companies offering these computing services are called cloud providers and typically charge for cloud computing services based on usage, similar to how you’re billed for water or electricity at home. Cloud computing includes five essential characteristics:

**On-demand self-service:** Consumers can request and receive computing resources as and when required without human interaction with a provider (mostly done though a web-based self-service portal)

**Resource pooling:** To serve multiple consumers using a multi-tenant model, the provider's computing resources are pooled with physical and virtual resources dynamically assigned depending on consumer’s demand. Storage, processing, memory, and network bandwidth are some examples of resources.

**Ubiquitous network access:** Cloud computing resources are accessible over the network, supporting heterogeneous client platforms such as mobile devices and workstations. (more example, tablets and laptops).

**Measured service:** Resource usage are measured, monitored, and billed transparently depending on utilization. In short, pay for use. For example, storage, processing, bandwidth, and active user accounts.

**Elasticity(Scalability):** Resources are provisioned and released on-demand. Hence it makes sure application will have exactly the capacity it needs at any point of time.

There are multiple perspectives (view) of Software architecture for application designs. We can say, the same application architecture consists of multiple views, and this multiple view variability model explains architectural concerns. There are multiple levels of variability like application variability, business profile variability, platform variability etc.

In this paper we will review incidence where Software Product Lines (SPL) is used to tackel the problem.

# SOFTWARE PRODUCT LINES

According to Software Engineering institute (SEI) Software Product Line is a “set of software-intensive systems sharing a common, managed set of features that satisfy specific needs of a particular market or mission, and that are developed from a common set of core assets in a prescribed way”. Software Product Line is strategy for producing software-intensive products. The strategy encompasses organizational anagement, technical management, and software engineering aspects of product production. Object technology can make an important contribution to the success of a product line organization. The three main goals of a software product line are to reduce cost, improve delivery time, and improve quality. A product line has a set of core assets which include shared components, infrastructure, tools, process, documentation, and above all else, shared architecture.

Range of assets such as architectures, specifications and implementations are provided from core asset developers to product developers to produce the product. This work of two group is coordinated and managed by product line managers.  There are some benefits of software product lines which include:

* Increased market agility
* Increased product agility
* Decreased product risk
* Increased mass customization
* Ability to maintain market presence
* Decreased time to market
* Increased customer satisfaction
* Efficient use of human resource

# CLOUD COMPUTING SERVICE MODELS

**Infrastructure as a service (IaaS)** is a form of cloud computing that provides virtualized computing resources over the internet. In an IaaS model, a cloud provider hosts the infrastructure components traditionally present in an on-premises data center, including servers, storage and networking hardware, as well as the virtualization or hypervisor layer.



The IaaS provider also supplies a range of services to accompany those infrastructure components. These can include detailed billing, monitoring, log access, security, load balancing and clustering, as well as storage resiliency, such as backup, replication and recovery. These services are increasingly policy-driven, enabling IaaS users to implement greater levels of automation and orchestration for important infrastructure tasks. IaaS customers access resources and services through a wide area network (WAN), such as the internet, and can use the cloud provider's services to install the remaining elements of an application stack. For example, the user can log in to the IaaS platform to create virtual machines (VMs); install operating systems in each VM; deploy middleware, such as databases; create storage buckets for workloads and backups; and install the enterprise workload into that VM. Customers can then use the provider's services to track costs, monitor performance, balance network traffic, troubleshoot application issues, and manage disaster recovery.

Any cloud computing model requires the participation of a provider. The provider is often a third-party organization that specializes in selling IaaS. Amazon Web Services (AWS) and Google Cloud Platform (GCP) are examples of independent IaaS providers. A business might also opt to deploy a private cloud, becoming its own provider of infrastructure services.

**Platform as a Service (PaaS):** is a cloud computing model in which a third-party provider delivers hardware and software tools -- usually those needed for application development -- to users over the internet. A PaaS provider hosts the hardware and software on its own infrastructure. As a result, PaaS frees users from having to install in-house hardware and software to develop or run a new application. PaaS does not typically replace a business's entire IT infrastructure. Instead, a business relies on PaaS providers for key services, such as application hosting or Java development.



A PaaS provider builds and supplies a resilient and optimized environment on which users can install applications and data sets. Users can focus on creating and running applications rather than constructing and maintaining the underlying infrastructure and services.

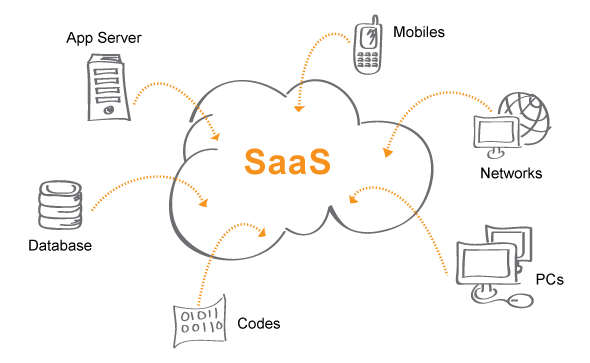
Many PaaS products are geared toward software development. These platforms offer compute and storage infrastructure, as well as text editing, version management; compiling and testing services that help developers create new software more quickly and efficiently. A PaaS product can also enable development teams to collaborate and work together, regardless of their physical location

The principal benefit of PaaS is simplicity and convenience for users -- the PaaS provider supplies much of the infrastructure and other IT services, which users can access anywhere via a web browser. PaaS providers then charge for that access on a per-use basis -- a model that many enterprises prefer, as it eliminates the capital expenses they traditionally have for on-premises hardware and software. Some PaaS providers charge a flat monthly fee to access their service, as well as the apps hosted within it.

Service availability or resilience, however, can be a concern with PaaS. If a provider experiences a service outage or other infrastructure disruption, this can adversely affect customers and result in costly lapses of productivity. Provider lock-in is another common concern, since users cannot easily migrate many of the services and much of the data produced through one PaaS product to another competing product. Users must evaluate the business risks of service downtime and lock-in before they commit to a PaaS provider.

Internal changes to a PaaS product are also a potential issue. For example, if a PaaS provider stops supporting a certain programming language or opts to use a different set of development tools, the impact on users can be difficult and disruptive. Users must follow the PaaS provider's service roadmap to understand how the provider's plans will affect its environment and capabilities

**Software as a Service (SaaS):** Software as a service (SaaS) is a way of delivering centrally hosted applications over the Internet—as a service.



SaaS applications run on a SaaS’s provider server. They can be considered as web-based software, hosted or on demand software also. SaaS applications can be accessed via internet instead of installing and maintaining it in local systems. It avoids the difficult both software and hardware management. SaaS provider provides access, security and availability of application. This also eliminates the cost of hardware acquisition and provisioning, as well as software licensing, installation and support.

We have listed some other benefits of the SaaS model below:

**Lower up-front cost -** SaaS is subscription-based model and has no license fees which results in minimum initial costs. The SaaS provider handles the infrastructure that is running the software, which lowers the fees for hardware and software maintenance.

**Easy upgrades** – SaaS removes the workload and responsibilities from you by handling hardware and software updates, deploying centrally upgrade to the hosted applications.

**Flexible payments:** Instead of purchasing software to install, or additional hardware to support it, customers subscribe to a SaaS services. It is a pay-a-you-go model i.e. they pay for this service on a monthly basis. Users can stop SaaS services at any time.

**Scalable usage:** Cloud services like SaaS offer high scalability, which gives customers the option to access more, or fewer, services or features on-demand

**Accessibility and persistence:** SaaS applications are internet based deliverable services and hence users can access them from any Internet-enabled device and location.

SaaS also have some disadvantages. Local software applications are better as it they provide higher control than that of third party software services. Access and privacy is major issue when it comes about cloud and hosted services. As SaaS is completely dependent on internet, service will be unavailable if connection to internet fails.

Some real time examples of SaaS are Outlook, Hotmail, or Yahoo! Mail. With these services, we log into your account over the Internet, and access it through a web browser. The email software is located on the service provider’s network, and our messages are stored there as well. Emails and stored messages can be accessed from a web browser on any computer or Internet-connected device.

The examples above are free services for personal use. But for organizational use, users have to rent productivity apps, such as email and storage services and some professional services like customer relationship management (CRM), enterprise resource planning (ERP), and document management. Users have to pay to use these services on the basis of subscription.

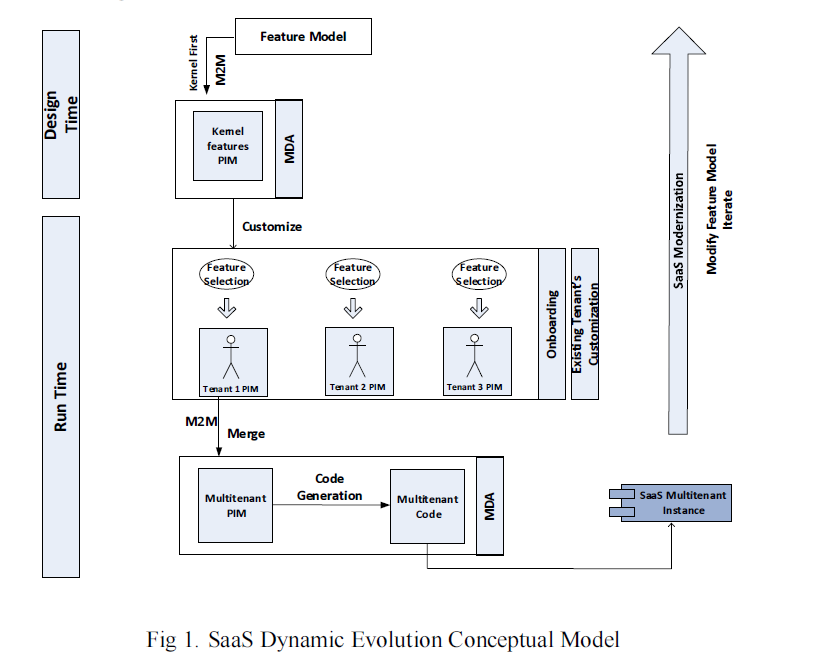
# Usage of SPL in Cloud Based Technologies

## Model driven architecture in SaaS

Problem statement: In a conventional SaaS Multi-tenant application every tenant who is subscribed has the same set of services. Build a Multi-Tenant Single Instance application that specifically aid each tenant based on the requirement. To accommodate evolving tenant requirements, the SaaS instance should evolve systematically

Existing System: A conventional Multi-tenant SaaS model provides same set of services one-size fits all model for all tenants

Recommended System: A multi-tenant single instance SaaS evolution platform based on Software Product Lines (SPLs). The platform specifies set of evolution rules based on feature modeling that govern evolution decisions.



❖ In simple words, providing a tenant specific application rather than providing same set of services for all the tenants.

❖ Customizable versions of SaaS application can be delivered using SPL principles.

❖ SPL is a family of software products with same set of features while exhibiting some variations distinguishing each product.

❖Feature models are the most commonly used technique to capture SPL’s commonality and variability at the requirements and design stages.

❖ Feature diagram is a graphical representation of nodes and associations that represents the common and variable features of an SPL.

❖ MDA is an approach adopted by the Object Management Group (OMG) to provide reusable solutions addressing the whole development lifecycle.

❖ MDA captures the system requirements independently from any platform specific details using a platform-independent model (PIM). Then, it integrates the platform-specifications to a PIM to generate a platform-specific model (PSM). Model-to-Model (M2M) transformations are used to transform one model to another based on a set of predefined rules

## Flexible cloud configuration for evolving workloads in IaaS

Problem statement: One of the main goals in cloud computing is to provide computing resources (e.g. networks, servers, storage, applications and services) on demand at low costs and in a seemingly unlimited amount. This is especially true at the Infrastructure as a Service (IaaS) level offered by the cloud paradigm, which focuses on provisioning computational resources such as processing power, network capacity and/or storage. The focus of the IaaS model makes it suitable for distributed applications (e.g., multi-tier applications), which have the required flexibility to achieve performance and cost goals, among others. Among the flexibility characteristics of cloud environments are the adaptive capabilities that deal with changes in the environment. For example, cloud applications, such as Web applications, can suffer from load fluctuations depending on the variations of the user demand. To achieve this, cloud configurations should be able to deal with the dynamism of the applications resource demand at runtime by reallocating resources while minimizing the monetary costs of the extra resources acquired

Existing system: Current approaches to perform dynamic adaptation of cloud environments have two main drawbacks:

* 1. They are platform-dependent, and
  2. They present manual ad-hoc solutions (as opposed to complete systematic methods) resulting in complex and error-prone approaches

Recommended system: The Software Product Line based framework is used to create and dynamically and proactively adapt cloud configurations according to the behavior exposed by the running applications. The systematization is achieved by the use of Software Product Lines (SPL), which in conjunction with MDD (Model Driven Development) techniques automate the tasks of coding cloud deployment and resource reconfiguration scripts.

The main motivation behind the design is to provide:

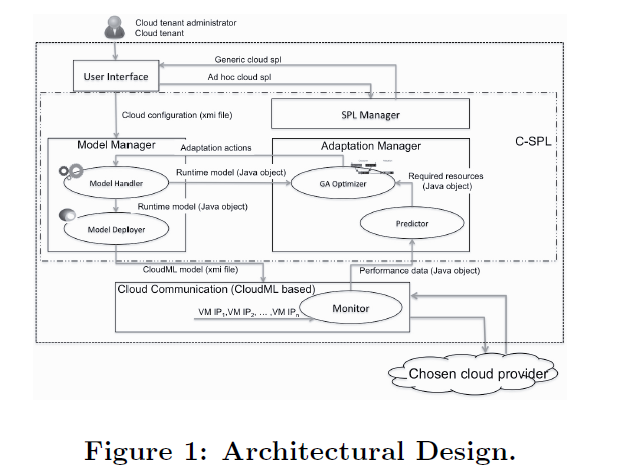
• Cloud platform independence,

• Process automation to reduce complexity,

• A systematic approach to adapt applications in the cloud

• Proactive adaptation to avoid degradation of quality of service (QoS), and

• Optimal resource allocation to reduce monetary costs



The overall architecture of SPL based framework is depicted in the figure 1. It consists of three layers:

* User interface layer
* C-SPL or core layer
* Cloud communication layer

The framework acts as a cloud configuration manager helping the final user with tasks related to creation, deployment, and adaptation of cloud environments at the IaaS level.

The User Interface allows the user to define the selections from a feature tree model. The Model Manager receives the feature tree and the selections and transforms them to a Cloud Machine Learning (CloudML) model. The model is then received and interpreted by the Cloud Communication module, which carries out the configuration deployment on a specific cloud platform. Monitoring information about resource usage, sent by the Monitor, is received by the Adaptation Manager. The Predictor predicts the amount of additional resources that will be required by an application. Then, the GA Optimizer finds an optimal resource reconfiguration and passes it to the Model Manager, which in turns generates a new CloudML model. Finally, the cloud resource reconfiguration is carried out by the Cloud Communication module. In addition, the SPL Manager is in charge of providing available configuration features, defined by the administrator, to the user.

## Dynamic adaptation of cloud environments for SaaS

Lets consider a senario where user roaming with the mobile device and he is not constantly connected to a network because a mobile device application cannot access network in every region. Hence, the user should be aware of these changes in context and informed about the quality of services provided to him. This problem can be solved by using context-aware system which sense and use context information to provide appropriate information to user.

In today’s world, mobile devices are main sources of information for user. But there are many constraints for mobile devices. Almost all applications usually demand more resources than the device capacity. Hence, it need to get resources for external resource i.e., cloud computing platform.

An access to cloud computing resources is too costly and it is not always guaranteed to be available. But, mobile users are increasingly demanding lot of services that are needed by their particular context. Here, context is any information provided by user implicitly or explicitly. And information can be location, user mobile activity, time etc.

Context-aware computing is the process of computing devices to detect and sense, interpret and respond to aspects of a user’s local context and computing devices.

We have a architecture that tries to solve context variability and configuration using Dynamic Software Product Line (DSPL). DSPL can be dynamic and adapt with context changing. This model uses 3-tier architecture which consists of Cloud User, Broker and Cloud Service provider.



1. Context Brokers – It provides the handling between cloud providers and cloud consumer which assist organizations in choosing the right and needy services and offerings. It is an intermediate service based on feature model to show this variability in the context.
2. Consumers – It is service-oriented product line approach by using feature model which changes with the changing condition of the context such as context data. When client uses the device, location, time, activity, preferences and security, it changes.
3. Services – It is the interfaces for the consumer to select context information. The services are represented using feature model where. Each service has its parameters like its cost, response time, resources, quality of service and service context.

# Conclusion

Cloud computing is a new technological development which has the potential to have a great mark on the world. There are many benefits that it provides to its organizations and users. For example, some of organizational benefits are, it reduces operating cost by spending less on maintenance and software upgrades and focus more on the businesses only. Users are still getting used to with this technology and a diverted from conventional processing to cloud computing. With this technology, knowledgeable developers with great ideas about internet services do not need to spend large money in building and expanding their software and hardware infrastructural capabilities and instead of it, they can focus on effective utilization of cloud services. Cloud computing offers various challenges as well. In this paper, we have reviewed Software Product Line approach for tackling the issues related to application development in cloud and exploring the issue related to variability.

1. The Software Product Line variability models are employed for modeling SaaS applications for adoption of customer’s needs.
2. Software Product Lines based framework is employed for IaaS service model. It makes use of Model-driven deployment is being used with runtime implementation issues of IaaS configurations.
3. Feature based model is employed for PaaS applications to keep an eye on behavioral changes of context.

# References

[1] IEEE paper: ‘Towards a Software Product Line-based approach to adapt IaaS cloud configurations’

[2] IEEE paper: ‘Review of Software Product Line Models Used to Model Cloud applications’

[3] IEEE paper: ‘Towards Software Product Lines Based Cloud Architecture’

[4] ACM paper: ‘Cloud Based Software Product Lines’

[5] Software Engineering Institute (SEI)

<https://www.sei.cmu.edu/productlines/>