Chapter 5
Cloud-Based Development Using Classic Life Cycle Model

Suchitra Ravi Balasubramanyam

Abstract Information technology (IT) today has evolved into a rapidly changing and dynamic science. Timelines have shrunk drastically for technology from being termed cutting edge to becoming obsolete. In such a fast-changing and dynamic world needing customised solutions, cloud computing offers a viable alternative. Cloud can overcome the redundancy factor and evolve over time to suit user needs. It is characterised by a wide array of deployment models and services that are very promising. While the concept of cloud computing has been around for some time now, industry adoption has been rather slow. Due to the sheer possibilities on offer, one remains optimistic of wider acceptance of this technology in future. This chapter takes us through the steps needed to validate the choice of public cloud via risk-based feasibility analysis. The chosen option can be built into needed IT systems based on cloud variants of the classic life cycle model. This chapter discusses the phases and activities of this development. The Wrapper model discussed here will enable better understanding of system control determinants for services opted on the cloud. A case study is discussed to help provide a better insight and understanding of the life cycle model.

Keywords Software life cycle • Cloud provision • Wrapper model

5.1 Introduction

Rapid adoption of the World Wide Web has brought a paradigm shift in business computing. This transformation can be attributed to the robust client-server architecture of the Web and its request-response operation model. The days of using

S.R. Balasubramanyam (✉)
Education and Research Unit, Infosys Limited,
No 350, Hebbal Electronics City, Hootagalli, Mysore 570027, India
e-mail: suchitraravi_b@infosys.com

Z. Mahmood and S. Saeed (eds.), *Software Engineering Frameworks for the Cloud Computing Paradigm*, Computer Communications and Networks,
DOI 10.1007/978-1-4471-5031-2_5,

HTML only for information presentation are past. Web-based applications that augment the computing capabilities using Java, XML and Web Services are the current trend. Such Web-based applications provide both partial and complete business solutions, with the user interfaces being accessible anytime online through the Web. For any business, presence on the Internet implies availability of computing facilities on demand.

But there are costs involved for such anytime access. In addition to computational infrastructure, the software installation, configuration and updates, along with the operating system and upgrades, add to the costs. Involving third-party service providers of such services helps reduce such costs, which cloud computing is best suited for. Cloud offers flexibility in software, platform and infrastructure on the Web that are optimal for individual business needs. The pay-per-use model makes it even more attractive to potential customers.

Cloud computing evolved out of grid computing, which is a collection of distributed computers intended to provide computing power and storage on demand [1]. Grid computing clubbed with virtualisation techniques help to achieve dynamically scalable computing power, storage, platforms and services. In such an environment, a distributed operating system that produces a single system appearance for resources that exist and is available is solicited most [2]. In other words, one can say that cloud computing is a specialised distributed computing paradigm. Cloud differs with its on-demand abilities like scalable computing power – up or down, service levels and dynamic configuration of services (via approaches like virtualisation). It offers resources and services in an abstract fashion that are charged like any other utility, thus bringing in a utility business model for computing. Though virtualisation is not mandatory for cloud, its features like partitioning, isolation and encapsulation [3] and benefits like reduced cost, relatively easy administration, manageability and faster development [4] have made it an essential technique for resource sharing. Virtualisation helps to abstract underlying raw resources like computation, storage and network as one, or encapsulating multiple application environments on one single set or multiple sets of raw resources. Resources being both physical and virtual, distributed computing calls for dynamic load balancing of resources for better utilisation and optimisation [5]. Like any other traditional computing environment, a virtualised environment must be secure and backed up for it to be a cost saving technique [3]. Cloud computing is a transformation of computing by way of service orientation, distributed manageability and economies of scale from virtualisation [3].

The National Institute of Standards and Technology (NIST) defines cloud computing [6] as “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”. According to NIST [6], the essential characteristics of cloud computing are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. We are seeing a paradigm shift in business IT due to these characteristics.

5.1.1 Cloud for Business

The environment, the businesses operate in today, is increasingly getting complex, with rapid changes in markets, products, customers and regulatory demands. Growing businesses in these environments generate vast amounts of data for analysis, which means scaling up of IT infrastructure adding up to huge business costs.

Here, cloud offers a viable, sustainable and scalable alternative to businesses that are both resource and cost-effective. Thus, managing business growth while controlling costs on IT infrastructure is perfectly balanced. For example, resource pooling controls costs while addressing scalability. It also allows for mobility of operations, helping control businesses spread across locations.

With green technology and sustainable business practices gaining ground worldwide, the pooled IT resources under cloud models have an added advantage for businesses. It helps businesses significantly reduce their carbon footprint as they scale up.

Businesses have a choice when deciding on cloud deployment – public, private, hybrid or community based. Public cloud is deployed on the Internet externally. Private cloud resides on an intranet or private network, hybrid models are a combination of public and private cloud, while community cloud is shared by several organisations supporting a specific community. All the options offer software, platform and infrastructure as services. Regardless of the type of cloud deployed, it impacts the entire computational ecosystem. Be it a casual user, software developer, IT manager or hardware manufacturer, all levels of participants experience the impact in varying degrees [7]. The different cloud services, if chosen correctly, support business and its IT needs. The NIST definitions of cloud services, their benefits and the limitations are now given along with case scenarios to better understand their applicability.

Software as a Service (SaaS) offers software with or without customisation and allows changes only to the user-specific configuration settings. NIST definition of SaaS [6] is "The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface such as a web browser (e.g. web-based e-mail) or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings".

Some examples of software offered on cloud are productivity applications like word processor, spreadsheet, slide creators and image manipulators. Major enterprise applications like customer relationship management (CRM) are also part of the cloud offerings. They are largely ready-to-use and users pay per use. Users can access the software through a Web browser instead of installing them on individual computers. Sometimes, software like CRM may require limited customisation but still are very cost-effective tools. Benefits of SaaS include reduced cycle times to market, anywhere access, no licence requirements and automatic version updates, lower operating and maintenance costs and pay as you consume.

Platform as a Service (*PaaS*) offers environment to develop "cloud-ready" applications and deploy them with required configuration settings. The scalability of the application at run time is administered by the service provider as per the deployment settings. NIST definition [6] of PaaS is "The capability provided to the consumer to deploy onto the cloud infrastructure consumer created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application hosting environment".

Enterprise applications need an enabling technology termed as platform (also called application infrastructure or middleware). Operating systems, application servers, databases, business process management (BPM) tools and application integrators are a few examples. Platform services like Google App Engine provide run time environments for Java and Python Web applications. Once deployed, scaling up the application to handle increased traffic and enhance data storage will be handled by the service provider. Heroku, another cloud platform built on an open standard, is a polyglot. It supports several languages like Java and Ruby, multiple frameworks and databases. Other facilities on offer include HTTP caching, logging, memcache and instant scaling.

The above platforms are examples of ready-to-use, on demand services where the user is charged a fee depending on the computational infrastructure used. In many cases, the application development, testing and deployment are constrained by the platform provider specifications, application programming interfaces (API), among other parameters. However, platform services that leverage existing skill sets of developers are more appealing to customers. While recommending PaaS offerings, Gartner Analyst Yefim Natis [8] says "PaaS is still emerging. It is neither mature nor standardised". Among the positives, PaaS has the benefits of inherent dynamic scaling and perfect bundled environment for development, testing and production.

Infrastructure as a Service (*IaaS*) offers hardware like servers, processors and memory obtained on rental. The user has control over the rented resources, configure any operating system on them, install any software and host or run any application on them. NIST definition [6] of IaaS is "The capability provided to the consumer to provision processing, storage, network, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g. host firewalls)".

In the case of start-up enterprises, short-term business campaigns and seasonal businesses, investing on data storage centre may not be the right strategic choice. So would the case be with businesses needing growing infrastructure. In such conditions, outsourcing the building and maintenance of data centre appears more

prudent. With an IaaS service, it is possible to reduce costs, space and management overheads. At the same time, the user can avail on-demand computing capability. Currently, infrastructure services are chosen mostly for non-critical applications. Principal Research Analyst Kyle Hilgendorf at Gartner states in his article titled "Evaluation Criteria for Public Cloud IaaS Providers" [8] that "IaaS is at the crossroads. To host mission critical business applications, IaaS should offer capabilities that convince the user, and it is yet to mature in this direction". However, key benefits include effective utilisation of infrastructure, resource provision on demand and reduced operating costs.

The key benefits of cloud services can be broadly summarised as enhanced business mobility, operating cost reduction, agility, flexibility and enabling green IT. However, key challenges still remain on cloud. These include among others security, data privacy, compliance and absence of standards, performance and availability issues. With these limiting factors, businesses need to make certain compromises when choosing cloud services. A risk analysis-based approach to decision making on availing cloud services will be of immense help, and the methodology is detailed in the following section.

5.2 Cloud-Based Development

5.2.1 Risk-Based Approach for Feasibility Analysis

The current business environment is characterised by only two constants – change and uncertainty. Business applications need to be agile to adapt to such fluidity, but constant changes to software and IT infrastructure is expensive. It is here that the characteristics and capabilities of the cloud, like the distributed model, high network availability and scalability can enable large Web-based applications to cope with constantly changing business demands. For businesses to select a cloud-based deployment, it is these benefits that are the decisive factors.

But, challenges like security, data privacy, regulatory issues and compliance are among the road blocks. The critical factor is security, as the user has no ownership or control over the cloud [3]. A related issue is the integrity of information in the hands of a third party, with current international laws and regulations governing such data misuse hazy. Other issues like governmental enforcement of IT laws and regulations, vendor lock-in that prevent federation of services from different providers, performance consistency and scalability also impact user decisions.

All stakeholders need to relook at all these issues holistically for wider and faster adoption of cloud services. One way to enhance user trust could be to consider international protocols and standards available, to certify cloud applications and Internet security. Certification is a proven technique to help establish identity and trust. Establishing trust is critical for cloud applications as the boundaries are

more logical than physical. In the virtual world, access policies and privileges need to change dynamically depending on the user and workload [9]. Further, user location and device used to access the service are equally important. Standards will allow consistency, portability and interoperability [10]. However, expecting international standards to govern cloud security is quite impossible as of now. Among existing standards, some of them – federated identity across multiple systems and providers, interoperability between different services and context-based protection – would be appropriate, based on the nature of request, data criticality and risk profile of service provider. Establishing a federated credential management system involves a repository of heterogeneous credentials, transfer and translation of those credentials [11].

Businesses need to consider the following factors when decisions are made to move to cloud-based services:

- What data and applications to move to the cloud
- The services that are needed on cloud, based on gathered business requirements
- The risks of service provider integrity, security breach and privacy violations

Presently, there is no single international standard or specification existing that guarantees safeguards to cloud applications and protection against these risks. Service providers can enhance user trust by integrating with currently available international certifications and standards. These steps in combination with the following suggested best practices [12] establish a three-step approach for feasibility analysis. This approach inherently suggests how to choose service providers and the right cloud services.

Step 1: Conduct risk assessment – to minimise risks, assess and choose service providers based on parameters like interoperability, portability and legal compliance. Assess the providers risk profile, ecosystem, supply chain and the quality of their infrastructure and operations.

Step 2: Assessment of one's own security capability in a cloud environment – in any cloud environment, the higher the support from a provider, the narrower the scope and control for the consumer. From SaaS to IaaS, responsibility for security varies for the consumer from least to highest. This increases flexibility for the consumer in implementing security controls. Data and users' interactions with the system can be controlled by encryption, authentication and secure access points. These are achievable irrespective of service provider.

Step 3: Implementation of a governance network – consumers should ensure from the providers logging of event observation and notifications. User should opt for remote monitoring where possible. Contracts and SLAs help establish a robust governance framework [10].

This three-step approach will help potential users evaluate and adopt cloud, capitalise on emerging technologies and be a part of evolving cloud standards. Any cloud-based development must ensure such analysis as part of overall requirements gathering. Next, software system life cycle is presented before exploring classic life cycle model to help better understand cloud in the development context.

5.2.2 Software System Life Cycle

Information systems help organisation to capture and manage data to produce information useful for its employees, customers, partners and suppliers. Each information system has a life of its own. Engineering such a software system involves process, methods and tools that facilitate systematic, disciplined and quantifiable approach to the overall software development. Process is the foundation layer and comprises a framework of activities to be carried out regardless of domain, size and complexity, methods indicate how-to of each activity and tools support process and methods. The generic process framework for software engineering comprises communication phase, planning phase, modelling phase, construction phase and deployment phase. Typical set of activities carried out in each of these phases are listed below [13]:

- Communication phase: Requirements gathering, focus on what requirements, specify the requirements and project initiation.
- Planning phase: Prepare project schedule, estimate of efforts and task duration. Tracking of the schedule happens in parallel with the rest of the subsequent phases.
- Modelling phase: Contains analysis phase and design phase. Activities of analysis phase are model the requirements, build prototype and evaluate alternate options. Similarly design phase activities are translate the requirements into a blueprint for software construction and iterate to a fine grain level details needed for the coding.
- Construction phase: Construct the code, in other words implement the design, test unit wise, after code integration and finally the system.
- Deployment phase: Deliver, support and maintain the deployed software.

5.2.3 Classic Life Cycle Model

The classic life cycle model is also called waterfall model. In this model, the workflow is linear in nature from communication phase to deployment phase and the outputs of each phase act as inputs for the next phase. It is a theoretical and sequential model, not adaptable directly. But other process models in use today are basically iterative in nature, which is more practical. Customer requirements evolve over time resulting in extension of life cycle phases. In particular, the design phase needs to evolve to be in line with these requirements [14]. However, in an ideal scenario the classic model is the simplest to understand and implement and so is used as the reference model in this chapter.

The life cycle of a cloud-based development is discussed by referring to this model. However, different services of cloud result in variants of this model, which are discussed in subsequent sections. Before exploring the variants, it is to be noted that system controls vary depending on the service and is depicted in the Wrapper model that follows.

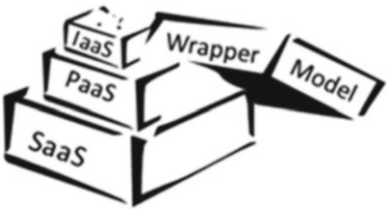

Fig. 5.1 Wrapper model

5.2.4 Cloud Services Wrapper Model

Consider any application software being used. The software needs a platform, which is resident on suitable infrastructure. The functional aspects of the application are dealt with at the software level, with the non-functional aspects spread beyond, reaching up to infrastructure including the platform. Cloud services follow the same pattern. SaaS is existent because it wraps a PaaS, which further wraps an IaaS. At each service level of the cloud, the non-functional facets are dependent on the underlying support. The scope of controls that can be exercised follows the same path and varies from low to high from outermost wrap to innermost. This wrapping of services can be shown as an abstract Wrapper model, represented as in Fig. 5.1. This model would help understand the variants of the classic life cycle phases better.

The model shows an IaaS wrapped inside a PaaS that in turn is wrapped inside SaaS. SaaS alone cannot exist without PaaS; similar is the case with PaaS. It is evident from the figure that IaaS and PaaS are hidden inside SaaS, and SaaS gets it support from PaaS which in turn depends on IaaS. Just dealing with outer wraps ignoring the inner ones will not help understand cloud relationships holistically.

5.2.5 Variants of Classic Life Cycle Model

The discussion in this section is based on the premise that software development is an outsourced activity and is for the public cloud. We also need to understand the relationships between the various stakeholders on the cloud. Given under Fig. 5.2 is the relationship between the customer, service provider and solutions provider in the cloud paradigm.

This model depicts the triangular relationship among Cloud Customer, Cloud Solutions Provider and Cloud Service Provider. The smaller triangles indicate the purpose of interaction between these stakeholders, like the customer interactions with solutions provider are for their IT needs. The Solutions provider offers solution catering to those needs by developing applications for the service provider's cloud. Now the service provider offers customised services to the customer on its cloud.

The customer needs a cloud-based IT system, which is developed by the solutions provider and deployed on the service providers cloud. The solutions provider may or may not utilise the public cloud for development and testing of the solution.

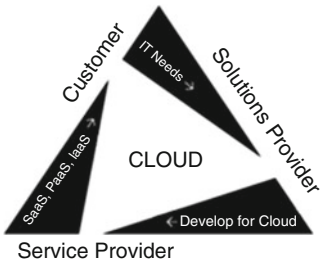

Fig. 5.2 Cloud stakeholders relationship

The focus here is only on the life cycle from the solution providers' perspective. It is assumed that the system may use any one, all or a combination of cloud services. In all the cases, the emphasis is only on the system under development and the activities are mentioned accordingly.

Following are the activities of cloud-based development with reference to classic life cycle model. There are generic activities involved in each of the phases, as well as different activities for each of the cloud services. Table 5.1 gives a general listing of activities irrespective of cloud services. Tables 5.2, 5.3, 5.4, 5.5 and 5.6 sum up phase-wise activities for SaaS, PaaS and IaaS.

The table highlights the common activities that are exclusively carried out for cloud-based development.

One of the important activities of the communication phase is requirements gathering and analysis. For large systems, requirements analysis is the most difficult and uncompromising activity [15]. Whether the system to be built is small or big, there are cases where the requirements have to be visualised and produced. With cloud this becomes more challenging as the choice of the cloud service provider should be made foreseeing the needs of other phases. For the two kinds of requirements – namely, user requirements and a high-level abstraction and system requirements – all relevant details have to be gathered. The requirements of the system further fall under functional, non-functional and domain-related categories. Requirements tell what the system should do and define constraints on its operation and implementation [16]. It gives a lead to choose service providers.

Selecting a suitable service provider is very much influenced by the feasibility analysis for the cloud adoption. This is a critical step as no industry standards exist as of today for choosing service providers' services. This step involves having an architectural description that gives a high-level view of the system, its structure, software elements and relationship among them [15]. Typically descriptions should cover the business domain, applications integration, technology, data and information architecture [14]. Due to lack of standards for cloud-architecture evaluation, the proposed architecture against non-functional quality attributes may not give a right evaluation result. A possible alternative is to follow Web Application Architecture Framework (WAAF) proposed by author David Lowe that categorises the architecture into Structure (what), Behaviour (how), Location (where) and Pattern (in Web applications) [17]. Apart from requirements specification document that contains precisely stated requirements, architecture description is also produced as an output.

Table 5.1 Classic life cycle model for cloud: common activities

Classic life cycle model for cloud	
Phase	Activities common to all services i.e., SaaS, PaaS and IaaS
Communication	Requirements gathering
	WHAT and not HOW of requirements
	Requirements specification
	[a]*Cloud adoption decision – taken based on risk based feasibility analysis*
	[a]*As there are few players in cloud computing arena, choose service provider in line with needs of future phases. This is critical as presently industry standards do not exist*
	Project initiation
Planning	Project schedule
	Estimation of efforts and task duration
	Tracking the activities – happens parallelly in all phases
Modelling	ANALYSIS:
	Model requirements
	Build prototype
	Evaluate alternate options
	[a]*Consider and incorporate non-functional and infrastructure requirements while modelling. Traditionally this is delayed until designing*
	DESIGN:
	Translate requirements into software construction blue print, iterate up to fine grain
	[a]*Include design goals of traditional web applications (irrespective of domain, size and complexity) for end-user interfaces*
Construction	Code and Test
	[a]*Testing to happen in a simulated or real cloud environment; hence testing to be planned accordingly*
Deployment	Deliver
	Support and maintain the deployed software

[a]Activity applied for cloud-based development

In the requirements, barring functional requirements, performance and external interface requirements, design constraints are heavily dependent on the cloud service and the respective provider for the same. This phase gives way to the next phases by a formal project initiation activity.

Planning phase [15] requires both requirements specification and architecture description as inputs for coming up with an executable plan. A plan for the processes to be followed for the entire project needs to be formulated. Based on this, the project schedule includes activities, their timelines and milestones to cross. Estimates of effort and resources can be done by taking expert opinion or through the use of models. As very little expertise exists in the industry today, and models are more suited for traditional development efforts, estimation is a challenging task. A good estimation has as its ingredients – scope of activity being estimated, work environment and usage of tools [18]. One needs to look at increasing the productivity which is an added result of environment, tools and experience. A heuristic approach along with both of the above could be a better option. Quality and configuration management

Table 5.2 Classic life cycle model: communication phase in cloud services

Phase	Services
Communication	*SaaS*
	Software is by third party provider. Important to choose appropriate service provider
	Along with domain specific standard requirements, customization as applicable are specified
	Software service may need customization as per feasibility analysis
	PaaS
	Application is to be custom built, so will be the requirements gathered
	However platform and infrastructure are by third party provider; choose appropriate provider
	Platform service to satisfy non-functional requirements as per feasibility analysis
	IaaS
	Application and platform as per customer needs, only computing environment by third party provider; customer virtually owns infrastructure
	Choose appropriate provider
	Infrastructure configuration is customized and utilization is charged accordingly
	Infrastructure service to satisfy non-functional requirements like traditionally owned infrastructure with focus on security, legal compliance and proper governance

Table 5.3 Classic life cycle model: planning phase in cloud services

Phase	Services
Planning	*SaaS*: Plan for software service customization
	PaaS: Plan for application development for the cloud platform and deployment on that platform
	IaaS: Plan for application development, deployment on specified platform and on the cloud infrastructure

plans are to be made ready. Risk management is managing of the unknown risks in the cloud development arena. To assess a project situation, use a carefully crafted monitoring plan to track activities across the development phases. This helps in comparing actual performance against the plan, and thereby ensuring the right actions at the right time to achieve the project goals. The output is a detailed plan based on these efforts and resources estimation and anticipated risks.

Modelling phase is split into analysis and design phase. Though the high-level architecture is available by now, it is of utmost importance to determine whether the suggested strategy deals with client's constraints. The obvious approach is prototyping [19]. Modular decomposition of modules is performed and prototypes constructed for chosen significant modules. Alternate options are explored, if needed using appropriate techniques to do the analysis. Consideration of non-functional requirements and infrastructure requirements during analysis is a critical step. For a

Table 5.4 Classic life cycle model: modelling phase in cloud services

Phase	Services
Modelling	ANALYSIS
	SaaS
	Based on requirements + customization specifications, identify the necessary custom interfaces for software service
	PaaS
	Based on requirements choose suitable application design architecture
	Devise deployment architecture considering the platform of the provider
	IaaS
	Based on requirements choose suitable applications design and deployment architecture
	Devise the configuration set up of the necessary infrastructure based on the offerings of the provider
	DISIGN
	SaaS: Design the identified interfaces as part of necessary customization
	PaaS and IaaS:
	Based on the architecture design interfaces that are internal as well as external to application
	Design software components as per the requirements model i.e., structured analysis model or object-oriented analysis model or both

Table 5.5 Classic life cycle model: construction phase in cloud services

Phase	Services
Construction	*SaaS*
	Code as per customization needed for the software
	Focus on regression testing of software as frills are added in the name of customization
	PaaS
	Code and Test as per design specified
	Focus on performance testing as non-functional requirements like scalability, availability etc. of web application are essential
	IaaS
	Code and Test as per design specified
	Focus on recovery testing and failover testing as recovery and failover are essential non-functional requirements. Testing becomes more significant as infrastructure is owned virtually and no physical control on them

public cloud as total infrastructure is owned by a third party, this analysis decides the success of the subsequent phases. Detailed design of the system is carved out to the level of individual methods and their interfaces for all the modules. As the application would run on distributed, heterogeneous, virtual computers on the Web, following Web engineering approach helps in successful Web-based system [20]. Web requirements like uniform look and feel, up-to-date, navigability and more have to be fulfilled within the design. For Internet-based applications, additional challenges are scalability and load balancing [20]. In an open environment like the Internet, it is not so easy to understand and predict the workload and user profiles. An unbalanced workload can become a cause for reduced system performance, reliability

Table 5.6 Classic life cycle model: deployment phase in cloud services

Phase	Services
Deployment	*SaaS*
	Deploy as per the configuration guidelines given by the provider
	Support and maintenance in line with the provider's software maintenance strategy
	PaaS
	Deploy like traditional application onto the provider's platform with configuration setting suggested by the provider
	Support and maintenance depends on the platform support by the provider
	IaaS
	Deploy like traditional application except that infrastructure is virtually owned
	Configure for cloud parameters like resource requirements, bounds for elasticity, dynamic provisioning, details for billing and metering etc.
	Support and maintenance happens as with traditional applications, depends maintenance strategy adopted by the provider for the virtual machines

and availability. A gap analysis of scalability and load balancing offerings from the provider would help in properly configuring those parameters, anything additional required must be built in-house through solutions provider.

The major concern in construction phase is vendor lock-in. Hence, the focus should be on writing interoperable, portable applications for the cloud. Applications need to be as flexible as possible and open for changes. This would in turn facilitate the maintenance phase.

Deployment phase is to deliver, deploy, support and maintain the cloud application. Stephen Schach mentions in his book [19] that maintainability should be built into the system from the beginning and not compromised any time during development. Like for any product, maintenance is like after sales service for an application; in case of cloud this is to be taken care of by all providers involved. Establishing a shared maintenance service, switching or replacing service providers are the key complex challenges faced in this phase.

Communication Phase: Apart from the common activities, choice of service provider is most important for SaaS. Unlike other digital products, plug-and-play is not the case with cloud product/software. Hence, it is also essential to specify the required customisations and describe architecture to suit the current needs with a provision to accommodate future requirements. With PaaS, it is the platform that runs on third-party-owned infrastructure that needs to be configured and customised. Application to be built by the solutions provider follows the life cycle of a typical Web application. While choosing the PaaS service providers, bear in mind the non-functional requirements. In case of IaaS, except for the infrastructure, the rest of the Web application cycle is similar to that of typical Web applications. The virtual infrastructure made available as a service is charged based on configuration and utilisation. PaaS and SaaS utilisation will be charged inclusive of the charges for the wrapped services as well. Refer to Fig. 5.1 to know what and how services are wrapped. Along with the other non-functional requirements security, legal

compliance and proper governance are to be looked at without fail while choosing the service providers for all cloud services. Here, the Wrapper model can help in visualising the control that can be exercised by the customer in getting the non-functional requirements satisfied. Outer layers indicate lesser control, while inner layers can lend themselves to better control.

Planning Phase: When compared with a typical Web application that can go along with classic life cycle, the cloud services have variations as per architecture description given below. Accordingly the needed service, application or interface integration should be made part of the plan.

- SaaS – plan for the required customisation for the cloud software
- PaaS – plan for application development for cloud platform and deployment on it
- IaaS – plan for application development and deployment on specified platform and on the cloud infrastructure

Modelling Phase: After planning comes the modelling and designing of the system based on the project requirements. If prototypes are to be built, they are to be analysed and best one chosen. With SaaS, the customisation demands the necessary interfaces to be analysed and designed. In PaaS and IaaS, the application is modelled like any other web application; the deployment architecture is analysed, the internal and external interfaces are designed keeping in mind the platform and infrastructure offerings of the providers.

Construction Phase: For each service, code is written in line with the design. Focus during testing varies in services. Security testing needs to be done compulsorily. Control in SaaS needs regression testing as frills are added as part of customisation. PaaS would need performance testing on the parameters like scalability and availability, as infrastructure is virtual. In IaaS, recovery and failover have to be tested since physical resources are in the hands of third party.

Deployment Phase: SaaS deployment is based on service provider guidelines. Support and maintenance will be controlled by the provider. Depending on the support and maintenance strategy of the provider, the phase can be smooth or a roller coaster ride. In PaaS application, deployment happens on the providers' platform configured as per their guidelines. Support and maintenance is a shared responsibility between the solutions provider and the service provider. Cloud application aspect is dealt by solutions provider and that of platform is with service provider. With IaaS, the deployment activity involves the infrastructure specification in the form of configuration parameters like number of resources needed, namely, processor, operating system, storage capacity, bounds of elasticity, dynamic provisioning, monitoring and metering. Again support and maintenance is a shared responsibility, infrastructure onus is with service provider whereas platform and the application is with the solutions provider.

The complexity of any cloud project is mainly in requirements analysis and mapping those to the capabilities of the cloud environment. The emphasis is mainly on communication, planning and modelling phases of the life cycle. Compared to

traditional development efforts, cloud-based development efforts and costs are lower in the construction phase. The activities listed above make it clear that cloud computing activities need tailored approach to classic life cycle model, especially for deployment and technology architecture. Lack of standards has made cloud-based developments highly platform- and vendor-specific projects. Hence, a high adaptability model is recommended for solution implementation. Other process models possible for cloud have been briefly explored in the next section.

5.2.6 Other Process Models

Process models that are suitable for cloud projects are basically iterative in nature. An incremental model can help in managing the technical risks by way of planned increments to the application. This approach is in general combined with other approaches for realising the application in quick time.

Prototyping is an iterative model wherein core requirements are realised quickly, thereby reducing the time to market. It is a mechanism to define requirements in an iterative manner till requirements are clearly understood and frozen. By itself prototyping is not a cost-effective model for large-scale complex applications. However, it is best applied in the context of other process models.

Spiral model is the best of waterfall and prototyping models. It imbibes the systematic aspect of waterfall and iterative aspect of prototyping. It is a risk-driven process model and reduces the degree of risk through iterations [21]. Though it is suitable for large-scale systems and software, it is not a convincing model for cloud projects at this stage. This is because of lack of standards and lack of expertise in the industry on cloud risks assessment.

Unified process which is driven by use-cases and centred around architecture is incremental and iterative; hence suitable for object-oriented projects. In the cloud the dynamic provision of resources and application components is inherently object oriented. This aspect of cloud can be best utilised by the model in combination with other process models [21].

As indicated earlier in the feasibility analysis section, cloud can adapt to changing business needs. One development approach that accommodates changes is the agile methodology. Some of the process models based on this methodology are Extreme Programming (XP) and SCRUM. Adaptability is the basic principle of these models. They focus on satisfying the customer's requirements as of today with a provision for long-term requirements. In this methodology, system is built over multiple releases by constantly responding and implementing requirements. It is a harmonious collaboration between customer and solutions provider teams for a sustainable application development. It includes improved communication among the working teams and thereby results in faster deployment.

5.3 Case Study

The industry service providers under the cloud platforms have many success stories. The underlying factor in most of these successes is hybrid cloud. This is considered the right choice for IT systems needing Web hosting, content delivery, e-commerce, backup and storage. NIST definition [6] of Hybrid Cloud is "the cloud infrastructure is a composite of two or more distinct cloud infrastructures (private, community or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g. cloud bursting for load balancing between clouds)". Only non-critical businesses, start-ups and one-off Web presence cases choose public cloud. The Web sites of some service providers like Amazon [22], IBM [23] and Google [24] have case studies for cloud that can be referred. A case study presented here helps understand usage of the proposed classic life cycle model.

5.3.1 Background Scenario

Back to Basics (B2B) is a not-for-profit organisation, with the stated goal of providing free education covering all age groups, through content delivered online through the Internet. The organisation has an online learning portal hosted within their country. To realise their stated goal of reaching out to all learners across the globe, they now want to re-establish themselves on the Web. The founding members of the organisation are determined to realise their stated goal and ensure the success of this online learning program. To cater to the needs of non-English-speaking countries, the organisation is planning to create video lessons in different languages and is also trying to subtitle present content videos wherever possible. As the demand for video lessons are increasing by the day, they also need to ensure portal availability 24×7 for the global audience. They also have plans to Webcast (live or recorded) important education conferences especially for university learners. All these plans call for huge infrastructure outlay.

Presently, the site offers lessons in basic science and mathematics. These are video lessons of short duration of 5–8 min. Every learning video hosted on the site goes through a process workflow. Apart from aspiring learners, there are other site users like administrator, editors, authors and reviewers.

The authors and reviewers are a large community of volunteers who contribute to the video lessons bank periodically. The site administrator oversees the need for innovations, simplicity of content and additional practice assignments. He posts artefact generation requirements to the author community after a careful scrutiny of reports. These reports are generated based on the video ratings, assessment of scores of learners and frequency of downloads of a particular artefact.

Whenever there is a post on artefact generation requirement, interested authors can nominate themselves along with a story board for the subject. This is reviewed

and approved by the administrator in consultation with the editor-in-chief. Once approved, the administrator charts a schedule and updates are sent to the selected author (if there are many nominations for a particular artefact) and a corresponding reviewer is assigned. Designated authors produce the videos and upload it onto the site for review. In the first instance when the video is uploaded, both the administrator and assigned reviewer receive e-mail alerts. From now, the review activity between author and reviewer happens on the network, till final reviewer approval. The final call for any improvements or full rejection rests with the editor-in-chief, who is the designated authority to sign off the video for both the story board and hosting. As per set standards, the editor-in-chief awards credit points to both author and reviewer, based on the popularity and usefulness of the video. This ensures special recognition of the volunteers' contribution, a source of satisfaction for them.

Learners can download the video lessons of their subjects and topics of interest, practice assignments and take up assessment tests. The assessment scores are recorded for later analysis. The learners need to rate the videos on different parameters like simplicity of presentation, narrative style, topic coverage, innovative examples and sufficiency of time for a topic. Learners can also provide feedback for any improvements.

For this mammoth mission, the organisation does not want to invest on large CAPEX (capital expenditure), preferring moving to OPEX (operational expenditure) and wants to utilise the available funds more thoughtfully. So they are exploring a viable, cost-effective and scalable solution and have discussed their requirements with a solutions provider. The following section details how the solution was arrived at.

5.3.2 Classic Life Cycle Model: Application

The organisation has started the communication phase with the selected solution provider. Risk-based analysis is carried out along with other activities in this phase, and public cloud is chosen. As the project security is not critical, the whole system resides on public cloud. The solution offered is not just limited to a Web application and its associated database but comprises additional features like e-mail services, calendar service and video production workflow. The video hosting and video streaming are part of content delivery network (CDN) aimed to serve a global audience of learners. The storage needs to be scalable to meet growing future demand. The main drivers for choosing cloud for solution implementation are:

- To spend on OPEX rather than CAPEX
- Affordable and scalable storage
- Access mobility for end users
- Users resident across the globe
- Anytime availability and elasticity
- Non-critical application
- Needs collaboration

Ensuring privacy is important, as log reports, Web access reports, location, country- or continent- wise assessment reports need to be secure due to legality issues. This needs to be incorporated carefully in the contract being entered into with the service provider. It is assumed that this requirement is fully met in the contract signed between customer and service provider. The other option is to hold such critical data on premises, and opt for hybrid cloud covering the rest of the application. We proceed next to the three-step approach for feasibility analysis.

- Step 1: Assess risk involved especially with the breach of legal compliance. Study the risk profile of service provider under consideration. Get assured on the quality of their infrastructure and operations.
- Step 2: Assess own security capabilities. Opt to encrypt data that gets onto the cloud, else it may lead to legal risks. (Here it is assumed that the organisation opts for encryption and contract with service provider is entered into accordingly.)
- Step 3: Establish a governance framework to monitor the events real time, as application is targeted at a global audience. (Here it is assumed that the organisation wants to delay having a monitoring system.)

Life cycle phases follow the project initiation. Table 5.7 gives details of the phases involved. The architecture diagram along with explanation given helps understand how to choose the right cloud service as well as service provider.

In the communication phase, solutions provider selects the right service provider for the proposed high-level architecture (Fig. 5.3). As all of software, platform and infrastructure services are chosen, project schedule and plan are prepared to be run in tandem till the necessary integration points are reached. During the modelling phase, the modules and interfaces are defined; however, prototypes on integration interfaces have to be built and analysed, based on which other alternate interfaces are explored. Especially for the cloud integrator supposed to integrate SaaS, PaaS and IaaS, prototyping is required. One more prototyping needed is to understand the browser compatibility and limitations. With satisfactory prototyping, the high-level architecture is now frozen for designing.

Design phase details out the modules and interaction interfaces with other modules. This being a Web-based system, the attributes like usability, navigability, response time, interaction efficiency, localisation, mobility, accessibility, consistency and compatibility are part of the user interface design goals [25]. Since the solution provider has chosen cloud itself for the development of the system, testing happens in the public cloud environment that becomes a facilitator for this testing activity. Support and maintenance responsibility would be shared among the providers, and accomplishing this federated activity is with the solution provider. For detailed activities in each of the phases, refer to the explanations in Tables 5.2, 5.3, 5.4, 5.5 and 5.6.

Architecture: The system comprises an end-user interfacing application for accessing the video lessons, connected database for data storage, reports and exclusive video storage. Other functions like e-mail, calendar, video production workflow and content delivery network (video hosting, video streaming) are equally essential for a fully operational system. All these services have to be seamlessly integrated on the cloud. Figure 5.3 depicts a high-level architecture diagram.

Table 5.7 Classic life cycle model for case study

Back to basics – case study	
Phase	Activities carried out
Communication	Gather requirements, Specify requirements
	Choose services – email services, calendar service as SaaS video production work-flow as PaaS, Main Web Application and its database as PaaS, content delivery network as SaaS and storage as IaaS. A cloud integrator to integrate all these
	Choose the service providers – Amazon, IBM, We Video, Google Apps
	Initiate project
	Service specific activities carried out; refer Table 5.2
Planning	Schedule is ready along with estimation
	Service specific activities carried out
Modelling	ANALYSIS
	Model the requirements
	Build prototype as necessary, and evaluate alternates
	High level architecture is decided
	Service specific activities carried out; refer Table 5.4
	DISIGN
	Translate requirements to software blueprint with detailed designing
	As it is a web-based application, the design goals like usability, navigability, simplicity, consistency, compatibility etc.
	Service specific activities carried out; refer Table 5.4
Construction	Code and Test as per design
	Testing happens in real cloud environment as solution provider uses cloud for development
	Service specific activities carried out; refer Table 5.5
Deployment	Deliver i.e., deploy application live
	Support
	Service specific activities carried out; refer Table 5.6

The diagram depicts that to produce online content, the authors and reviewers community first need a collaboration medium, which also needs to be provided to the administrator and editor-in-chief. The medium is primarily for communication and planning, e-mail and calendar services. The video creation workflow needs a platform service where all stakeholders interact on work allocation, subsequent submission of the allocated work, followed by the work approval within the defined flow. Saving of the intermediate outcomes and final artefacts of video creation workflow requires for growing storage that can only be offered through infrastructure service. The videos approved are hosted on the provided storage. This huge content residing on the storage is meant for delivery via video streaming. Here a software service that offers content distribution and streaming is a must. A key need for enthusiastic learners is a Web interface that is a one-stop shop for them to access and download the video lessons, practice assignments and take up assessment tests. This Web application is built in Java and hosted on a Web server platform service. Apart from this, learners' data needs to be saved in a structured fashion for future reports generation and subsequent analysis. This calls for a relational database platform. Now with multiple services and providers in the arena, a suitable integrator

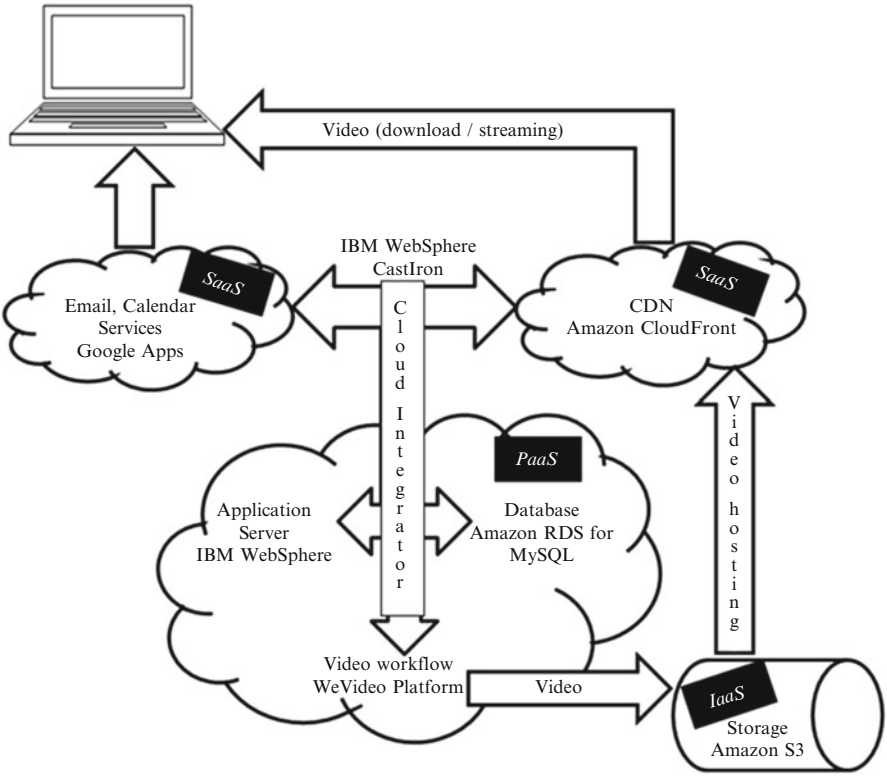

Fig. 5.3 High-level architecture diagram (Abbreviations: *Amazon S3* Amazon Simple Storage Service, *Amazon RDS* Amazon Relational Database Service, *CDN* Content Delivery Network)

that integrates them effortlessly is vital for the success of the system. The choice of providers and their services for the above functions are mentioned in the following section. The impact of the differences using life cycle model for SaaS, PaaS and IaaS is highlighted in conclusion.

- IBM WebSphere Application server [23] that is chosen for hosting the main Web application is a PaaS service. It is opted to achieve the necessary interoperability of different services. The relational database is again a PaaS service, and Amazon RDS for MySQL [22] is chosen for the same.
- The interaction among the users other than learners happens through e-mails and calendars for work scheduling. This implies need for a system that offers collaboration and mobility. Google Apps e-mail and calendar SaaS services [24] are selected for this.
- WeVideo [26], a cloud-based, collaborative video-editing platform, best fits the video creation workflow. The kinds of users, their activities and the complete workflow are catered to by this PaaS. The next functionality is to host and distribute the video content. WeVideo also has an option to export baseline videos to a hosting and distribution environment.

- Amazon CloudFront [22], a SaaS, best fits the need of content distribution network, inclusive of streaming.
- Storage for videos is achieved through IaaS, in particular storage service Amazon S3 [22].
- As there are multiple cloud players offering SaaS, PaaS and IaaS, it is crucial to have a seamless integrator that makes it a unified system. IBM WebSphere Cloud Integrator [23] is the choice here.

5.4 Conclusion

As evident from the illustrated case study, the activities of each phase are closely interconnected to the service chosen. Based on choice of services opted for during the communication phase, the span of activities for each of the functions varies in the subsequent phases. If the planning phase is not carefully thought through, the entire project may become highly risky. This means that the planning phase is extremely crucial to determine successful choice of provider, as also design and deployment on the cloud.

Any SaaS opted for mainly involves customisation that may cut down project resources and project costs in the modelling phase. However, this is not the case if PaaS and IaaS are opted for. Construction phase follows the same pattern with the exception of testing. Irrespective of the service selected, prior to final deployment on the cloud, the application is tested in a simulated environment. The extent of testing can be planned which is based on controls exercised for each service. This can be better understood with reference to the Wrapper model. The deployment with SaaS and PaaS here is as per the service providers' specifications. With IaaS, it is like traditional deployment on a virtual infrastructure. Support and maintenance for all services in deployment phase rely heavily on the cloud providers' strategy.

The development activities of the chosen cloud service follow a linear path under the classic life cycle model. If all the three services, namely, SaaS, PaaS and IaaS, are opted for, then across services within a phase, all activities happen simultaneously with varying time durations but with a lag. For instance, while SaaS is in construction phase, PaaS and IaaS could be in modelling phase.

If factors like cost, design, technology lead to a single service being chosen, IaaS leads to virtual infrastructure being used with traditional development, PaaS leads to virtual platform inclusive of infrastructure used with traditional development, while SaaS leads to using customised virtual software. The core differences among services are highlighted across activities of all the phases in the classic life cycle model.

Acknowledgements I would like to thank my colleague Mr. Krishna Prasad Srinivasa Rao for his assistance in content creation and my managers Mr. Rajagopalan P, Dr. Ramesh Babu S. and Mr. Srikantan Moorthy for their valuable guidance and support. The encouragement and support received from my husband Mr. Ravi Balasubramanyam helped me immensely in chapter presentation.

References

1. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud computing and grid computing 360-degree compared. Paper presented at grid computing environments workshop, Conference Publications, Austin, 12–16 November 2008
2. Dollimore, J., Kindberg, T., Coulouris, G.F.: Distributed Systems: Concepts and Design, 4th edn, pp. 206–207. Pearson Education, Harlow (2005)
3. Hurwitz, J., Bloor, R., Kaufman, M., Halper, F.: Cloud Computing for Dummies, pp. 197–208. Wiley, Hoboken (2009)
4. Velte, T., Velte, A., Elsenpeter, R.: Cloud Computing: A Practical Approach, pp. 253–258. Tata McGraw-Hill, New Delhi (2010)
5. Ramaswamy, R. (ed.): The Art and Technology of Software Engineering: A Mosaic of Models and Methods, pp. 131–139. Tata McGraw-Hill, New Delhi (2002)
6. NIST Cloud Computing Reference Architecture Version 1, March 30, 2011. http://www.nist.gov (2011)
7. Hayes, B.: Cloud computing. Commun. ACM **51**(7): 9–11. http://cacm.acm.org/ (2008)
8. Gartner Information Technology Research and Advisory Company: http://www.gartner.com
9. Mateo, L.A.: IBM IT Consulting Leader, Europe, December 13, 2011, Post named "Migration to cloud: It is all about workloads" at Thoughts on Cloud Blog sponsored by IBM. http://thoughtsoncloud.com/ (2011)
10. SETLabsBriefings, Cloud Computing, Infosys Labs Publications, vol. 7, no. 7 (2009)
11. Chakrabarti, A.: Grid Computing Security. Springer, Berlin/New York (2007)
12. Symantec, White Paper, The Secure Cloud: Best Practices for Cloud Adoption. http://www.symantec.com/
13. Pressman, R.S.: Software Engineering: A Practitioner's Approach, 7th edn. McGraw Hill, New York (2009)
14. Pradhan, A., Nanjappa, S.B., Nallasamy, S., Esakimuthu, V.: Raising Enterprise Applications: A Software Engineering Perspective, pp. 33–39. Wiley India, New Delhi (2010)
15. Jalote, P.: An Integrated Approach to Software Engineering, 3rd edn, pp. 67–211. Narosa Book Distributors, New Delhi (2008)
16. Sommerville, I.: Software Engineering, 8th edn, pp. 131–164. Pearson Education, Harlow (2009)
17. Pressman, R.S., Lowe, D.: Web Engineering: A Practitioner's Approach. Tata McGraw Hill, New Delhi, pp. 115–116, 253–258 (2011)
18. Parthasarathy, M.A.: Practical Software Estimation. Addison-Wesley, Upper Saddle River, pp. 6–22, 206–207 (2007)
19. Schach, S.R.: Software Engineering, 7th edn, pp. 332, 515–520. Tata McGraw Hill, New Delhi (2006)
20. Suh, W.: Web Engineering: Principles and Techniques. Idea Group, Hershey, pp. 1–22, 81–82 (2004)
21. Tsui, F., Karam, O.: Essentials of Software Engineering, 2nd edn. Jones & Bartlett Learning, Burlington (2010)
22. Amazon Web Services: http://aws.amazon.com/
23. IBM: http://www.ibm.com/
24. Google Cloud Platform: http://cloud.google.com/
25. Gerti, K., Siegfried, R., Brigit, P., Werner, R.: Web Engineering, pp. 219–246. Wiley, Hoboken (2010)
26. WeVideo: https://www.wevideo.com/