ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Deadline, Budget and Mobile Context Platform for Mobile Cloud Apps

Gadige Radha[1], G Divya Zion[2]

M.Tech, CSE, Ravindra College of Engineering for Women, Kurnool, India[1]

Asst. Professor (M.Tech), CSE, Ravindra College of Engineering for Women, Kurnool, India[2]

Abstract: Scientific applications require large computing power, traditionally exceeding the amount that is available within the premises of a single institution. Therefore, clouds can be used to provide extra resources whenever required. For this vision to be achieved, however, requires both policies defining when and how cloud resources are allocated to applications and a platform implementing not only these policies but also the whole software stack supporting management of applications and resources. Aneka is a cloud application platform capable of provisioning resources obtained from a variety of sources, including private and public clouds, clusters, grids, and desktops grids. In this paper, we present Aneka's deadline driven provisioning mechanism, which is responsible for supporting quality of service (QoS)-aware execution of scientific applications in hybrid clouds composed of resources obtained from a variety of sources. Experimental results evaluating such a mechanism show that Aneka is able to efficiently allocate resources from different sources in order to reduce application execution times.

Keywords: Mobile Cloud Apps, Cloud Computing, IaaS, PaaS, SaaS, Mobile Computing, Cloud Providers, Cloud Services, Aneka Cloud Platform.

I. INTRODUCTION

1.1 Cloud computing

Cloud computing is a multi-tenant internet based computing, that relies on sharing computing resources for handling the applications, rather running on local servers or personal devices. In cloud computing, the word cloud (also phrased as "the cloud") is used as a metaphor for "the Internet," so the phrase cloud computing means "a type of Internet-based computing," where different services — such as servers, storage and applications —are delivered to an organization's computers and devices using the network as backbone. This is a model, for enabling convenient and on demand network access of a shared pool of configurable computing resources (e.g. Network, server, storage, applications and services), which are be dynamically scalable and rapidly provisioned without having the service provider interaction.
The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second. In consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power consumption, and offer services like online computer games etc.
One of the key characteristics of cloud computing is the scalability, which refers, the ability of a system to adopt the dynamically growing needs. Cloud technology allows for the automatic provision and de provision of resource as and when it is necessary, thus ensuring that the level of resource available is closely matched with the current demand as possible, as differentiating itself from conventional models, where resources are delivered in blocks (e.g., individual servers, downloaded software applications), usually with fixed capacities and upfront costs. With cloud computing, the end user usually pays only for the resource they use and so avoids the inefficiencies and expense of any unused capacity.
However, the advantages of cloud computing are not only limited for its flexibility, but also benefit (in varying degrees) from the economies of scale created by setting up services en masse with the same computing environments, and the reliability of physically hosting services across multiple servers where individual system failures do not affect the continuity of the service.

1.1.1 Architecture

Cloud computing architecture refers to the components and subcomponents required for cloud computing. These components typically consist of a front end platform (fat client, thin client, mobile device), back end platforms (servers, storage), a cloud based delivery, and a network (Internet, Intranet, Intercloud). Combined, these components make up cloud computing architecture.
Cloud services means services made available to users on demand via the Internet from a cloud computing provider's servers as opposed to being provided from a company's own on-premises servers. Cloud services are designed to provide easy, scalable access to applications, resources and services, and are fully managed by a cloud services provider.
A cloud service can dynamically scale to meet the needs of its users, and because the service provider supplies the hardware and software necessary for the service, there's no need for a company to provision or deploy its own resources or allocate IT staff to manage. The examples of cloud services include online data storage and backup

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

solutions, Web-based e-mail services, hosted office suites and document collaboration services, database processing, managed technical support services and more.

Cloud computing providers offer their services according to several fundamental models. Cloud computing architecture is shown in Figure 1.1. These services are broadly divided into three categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS).

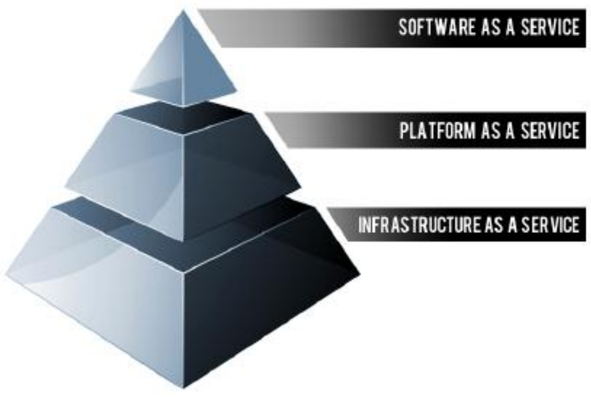


Fig1.1 Cloud computing architecture

Infrastructure as a Service (IaaS)

IaaS is one of three main categories of cloud computing services, alongside Software as a Service (SaaS) and Platform as a Service (PaaS).IaaS is a way of delivering Cloud Computing infrastructure – servers, storage, network and operating systems – as an on-demand service. Rather than purchasing servers, software, datacenter space or network equipment, clients instead buy those resources as a fully outsourced service on demand. IaaS providers also host users' applications and handle tasks including system maintenance, backup and resiliency planning. It also provides virtualized computing resources over the Internet. IaaS platforms offer highly scalable resources that can be adjusted on-demand. This makes IaaS well-suited for workloads that are temporary, experimental or change unexpectedly.

Other characteristics of IaaS environments include the automation of administrative tasks, dynamic scaling, desktop virtualization and policy-based services. IaaS customers pay on a per-use basis, typically by the hour, week or month. Some providers also charge customers based on the amount of virtual machine space they use. This pay-as-you-go model eliminates the capital expense of deploying in-house hardware and software. However, users should monitor their IaaS environments closely to avoid being charged for unauthorized services.

Characteristics of IaaS:

As with the two previous sections, SaaS and PaaS, IaaS is a rapidly developing field. That said there are some core characteristics which describe what IaaS is. IaaS is generally accepted to comply with the following

- Resources are distributed as a service.
- Allows for dynamic scaling.
- Has a variable cost, utility pricing model.
- Generally includes multiple users on a single piece of hardware.

Platform as a service (PaaS):

In the PaaS models, cloud providers deliver a computing platform, typically including operating system, programming language execution environment, database, and web server. Application developers can develop and run their software solutions on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers.

PaaS is analogous to SaaS except that, rather than being software delivered over the web, it is a platform for the creation of software, delivered over the web. PaaS does not typically replace a business' entire infrastructure. Instead, a business relies on PaaS providers for key services, such as Java development or application hosting. For example, deploying a typical business tool locally might require an IT team to buy and install hardware, operating systems, middleware (such as databases, Web servers and so on) the actual application, define user access or security, and then add the application to existing systems management or application performance monitoring (APM) tools. IT teams must then maintain all of these resources over time. A PaaS provider, however, supports all the underlying computing and software; users only need to log in and start using the platform – usually through a Web browser interface.

Most PaaS platforms are geared toward software development, and they offer developers several advantages. For example, PaaS allows developers to frequently change or upgrade operating system features. It also helps development teams collaborate on projects.

Characteristics of PaaS:

There are a number of different takes on what constitutes PaaS but some basic characteristics include

- Services to develop, test, deploy, host and maintain applications in the same integrated development environment. All the varying services needed to fulfill the application development process.
- Web based user interface creation tools help to create, modify, test and deploy different UI scenarios.
- Multi-tenant architecture where multiple concurrent users utilize the same development application.
- Built in scalability of deployed software including load balancing and failover.
- Integration with web services and databases via common standards.
- Supports for development team collaboration – some PaaS solutions include project planning and communication tools.
- Tools to handle billing and subscription management.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Software as a service (SaaS):

Software as a service (SaaS), users are provided access to application software and databases. Cloud providers manage the infrastructure and platforms that run the applications. SaaS is sometimes referred to as "on-demand software" and is usually priced on a pay-per-use basis or using a subscription fee.

In the SaaS model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. Cloud users do not manage the cloud infrastructure and platform where the application runs. This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support. Cloud applications are different from other applications in their scalability—which can be achieved by cloning tasks onto multiple virtual machines at run-time to meet changing work demand. Load balancers distribute the work over the set of virtual machines. This process is transparent to the cloud user, who sees only a single access point. To accommodate a large number of cloud users, cloud applications can be multitenant, that is, any machine serves more than one cloud user organization. The pricing model for SaaS applications is typically a monthly or yearly flat fee per user, so price is scalable and adjustable if users are added or removed at any point. The traditional model of software distribution, in which software is purchased for and installed on personal computers, is sometimes referred to as software as a product.

Characteristics of SaaS:

Like other forms of Cloud Computing, it is important to ensure that solutions sold as SaaS in fact comply with generally accepted definitions of Cloud Computing. Some defining characteristics of SaaS include

- Web access to commercial software.
- Software is managed from a central location.
- Software delivered in a “one to many” model.
- Users not required handling software upgrades and patches.
- Application Programming Interfaces (APIs) allow for integration between different pieces of software.

1.1.2 Deployment Models – Private, Public, Hybrid

A cloud deployment model represents a specific type of cloud environment, primarily distinguished by ownership, size, and access. There are four types of cloud models available in the market namely

- Public Cloud
- private Cloud
- Hybrid Cloud

Public Cloud

A public cloud is a publicly accessible cloud environment owned by a third-party cloud provider. The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services and the resources are offered as a service, usually over an internet connection, for a pay-per-usage fee. Users can scale their use on demand and do not need to purchase hardware to use the service. Public cloud providers manage the infrastructure and pool resources into the capacity required by its users. Public clouds are available to the general public or large organizations, and are owned by a third party organization that offers the cloud service. A public cloud is hosted on the internet and designed to be used by any user with an internet connection to provide a similar range of capabilities and services. Public cloud users are typically residential users and connect to the public internet through an internet service provider’s network.

The advantages of public cloud include:

- Data availability and continuous uptime
- 24/7 technical expertise
- On demand scalability
- Easy and inexpensive setup
- No wasted resources

Drawbacks of public cloud:

- Data security

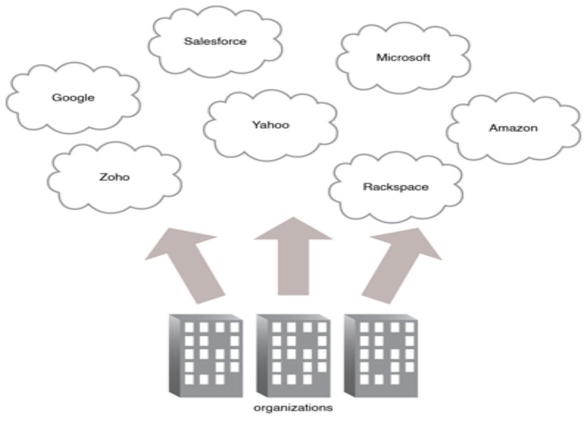


Fig1.2 Public Cloud

Figure 1.2 shows a partial view of the public cloud landscape, highlighting some of the primary vendors in the marketplace.

Private Cloud

A private cloud is owned by a single organization. Private clouds enable an organization to use cloud computing technology as a means of centralizing access to IT resources by different parts, locations, or departments of the organization. When a private cloud exists as a controlled environment, the problems described in the Risks and Challenges section do not tend to apply.
The use of a private cloud can change how organizational and trust boundaries are defined and applied. The actual administration of a private cloud environment may be carried out by internal or outsourced staff.
With a private cloud, the same organization is technically both the cloud consumer and cloud provider (Figure 1). In order to differentiate these roles:

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

- a separate organizational department typically assumes the responsibility for provisioning the cloud (and therefore assumes the cloud provider role)
- departments requiring access to the private cloud assume the cloud consumer role

It is important to use the terms "on-premise" and "cloud-based" correctly within the context of a private cloud. Even though the private cloud may physically reside on the organization's premises, IT resources it hosts are still considered "cloud-based" as long as they are made remotely accessible to cloud consumers. IT resources hosted outside of the private cloud by the departments acting as cloud consumers are therefore considered "on-premise" in relation to the private cloud-based IT resources.

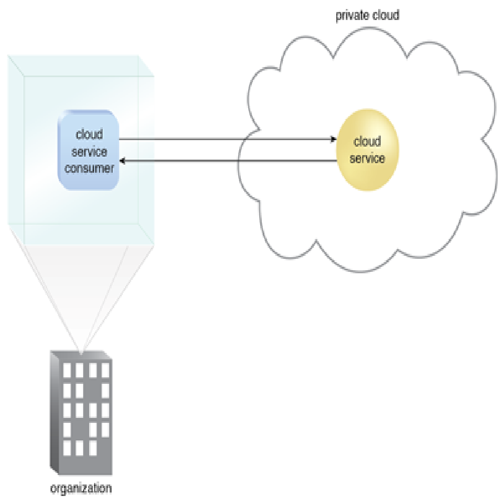


Fig1.3 Private Cloud

Figure 1.3 shows a cloud service consumer in the organization's on-premise environment accesses a cloud service hosted on the same organization's private cloud via a virtual private network. The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise. The cloud infrastructure is accessed only by the members of the organization and/or by granted third parties. The purpose is not to offer cloud services to the general public, but to use it within the organization. For example an enterprise that wants to make consumer data available to their different stores. A private cloud is hosted in the data center of a company and provides its services only to users inside that company or its partners. A private cloud provides more security than public clouds, and cost saving in case it utilizes otherwise unused capacities in an already existing data center. The major drawback of private cloud is its higher cost. When comparisons are made with public cloud; the cost of purchasing equipment, software and staffing often results in higher costs to an organization having their own private.

Hybrid Cloud

Hybrid cloud infrastructure is a composition of two or more clouds that are unique entities, but at the same time are bound together by standardized or proprietary technology that enables data and application portability. Hybrid clouds offer the cost and scale benefits of public clouds, while also offering the security and control of private clouds.

Hybrid deployment architectures can be complex and challenging to create and maintain due to the potential disparity in cloud environments and the fact that management responsibilities are typically split between the private cloud provider organization and the public cloud provider.

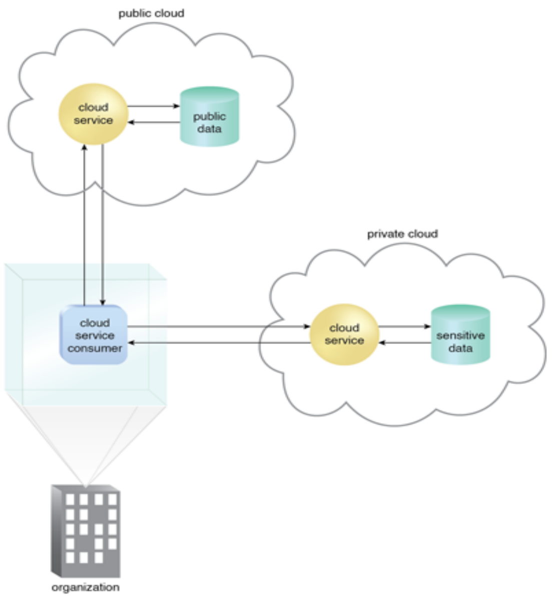


Fig1.4 Hybrid Cloud

Figure1.4 shows an organization using a hybrid cloud architecture that utilizes both a private and public cloud.

1.1.3 Resource Intensive Applications in Clouds

Some resource intensive applications that demand lot of computing, storage, memory and energy power, cannot be run on the mobile devices, since these devices have limitations in the CPU, RAM, storage and batter /power consumptions. To overcome this, we need to design a platform for enriching the user experience while consuming fewer resources on the mobile devices. To address this problem, Aneka offers Mobile Client Library as platform services; for the cloud based mobile applications development.

Aneka Client Library that encapsulates the processes of connecting to cloud, serializing and deserializing messages, sending messages, and collecting their responses. Thus, the effort and complexity of developing a mobile cloud application is decreased. In addition, the library was designed to leverage the Aneka PaaS solution, which provides transparently the resource provisioning and job scheduling services and encapsulates different cloud providers Web APIs. The user has no concern in allocating or deallocating virtual machines or distributing the jobs among the resources.

1.1.4 Dynamic Resource Provisioning for Resource Intensive Applications

The application services hosted under Cloud computing model have complex provisioning, composition, configuration, and deployment requirements. Evaluating the performance of Cloud provisioning policies, application workload models, and resources performance models in a repeatable manner under varying system and user configurations and requirements is difficult to achieve. Due to missing deadlines, the jobs are being rejected. The rejection rate increases. We want to propose on-demand provisioning in order to maintain QoS we need to give extra resource to the job. Mapping performance requirements to the underlying resources in the cloud is challenging. Resource under-provisioning will inevitably hurt performance and resource over-provisioning can result in idle instances, thereby incurring unnecessary costs. There is an increasing demand to efficient management of large-scale application under cost and deadline constrained. To counter such lack of solutions for cost and deadline constrained dynamic resource provisioning and scheduling, to present a coordinated dynamic resource provisioning and scheduling approach that is able to maximize no. of completed execution application within their deadlines and budget.

1.2 Mobile Cloud Applications – Resource Intensive

Mobile Cloud Applications are very similar to Web-based applications. The main similarity is that both mobile cloud apps and Web apps run on servers external to the mobile device and require the use of a browser on the mobile device to display and then use the app user interface (UI). In addition, they both are targeted for multiple mobile devices versus a single mobile device.

Mobile cloud apps do not need to be downloaded and installed on mobile devices. Users view the mobile cloud app UI in a browser window on the remote device. An Internet connection is required to use mobile apps running on a mobile cloud.

Mobile applications may be bonded to cloud resources by following a delegation or offloading criteria. In a delegation model, a mobile device utilizes the cloud to perform resource-intensive operations which are time-consuming, programmable and parallelizable among multiple servers (e.g. based on distributed frameworks like Map Reduce), and computationally unfeasible for offline devices. From a delegation perspective, hybrid cloud and cloud interoperability are essential for mobile scenarios in order to foster the de-coupling of the handset to a specific cloud vendor, to enrich the mobile applications with the variety of cloud services provided on the Web and to create new business opportunities and alliances. However, developing a mobile cloud application in this model involves adapting different Web APIs from different cloud vendors within a native mobile platform. We have studied the delegation of mobile tasks to hybrid clouds in detail and have developed a Mobile Cloud Middleware framework (MCM) that addresses the issues processing, and dynamic allocation of cloud infrastructure of interoperability across multiple clouds, transparent delegation and asynchronous execution of mobile tasks that require resource intensive.

Mobile applications leverage this IT architecture to generate the following advantages:

- Extended battery life
- Improvement in data storage capacity and processing power
- Improved synchronization of data due to “store in one place, access from anywhere” policy
- Improved reliability and scalability
- Ease of integration

1.2.1 Mobile computing in Clouds

Mobile Cloud Computing (MCC) is the combination of cloud computing, mobile computing and wireless networks to bring rich computational resources to mobile users, network operators, as well as cloud computing providers. The ultimate goal of MCC is to enable execution of rich mobile applications on a plethora of mobile devices, with a rich user experience. MCC provides business opportunities for mobile network operators as well as cloud providers.

Mobile Computing is a technology that allows transmission of data, voice and video via a computer or any other wireless enabled device without having to be connected to a fixed physical link**.** Mobile computing involves mobile communication, mobile hardware, and mobile software. Communication issues include adhoc and infrastructure network as well as communication properties, protocols, data formats and concrete technologies. Hardware includes devices or device components. Mobile software deals with the characteristics and requirements of mobile applications. Mobile devices are constrained by their processing power, battery life and storage. However, cloud computing provides an illusion of infinite computing resources. Mobile cloud computing is a new platform combining the mobile devices and cloud computing to create a new infrastructure, whereby cloud performs the heavy lifting of computing-intensive tasks and storing massive amounts of data processing and data storage happen outside of mobile devices.

1.2.2 Mobile Application development

Mobile application development, also known as mobile apps, has become a significant mobile content market. Mobile application development is a term used to denote the act or process by which application software is developed for handheld devices, such as personal digital assistants, enterprise digital assistants or mobile phones. These applications can be pre-installed on phones during manufacturing platforms, or delivered as web applications using server-side or client-side processing (e.g. JavaScript) to provide an "application-like" experience within a Web browser.

Application software developers also have to consider a lengthy array of screen sizes, hardware specifications and configurations because of intense competition in mobile software and changes within each of the platforms. Mobile app development has been steadily growing, both in terms of revenues and jobs created.
As part of the development process, Mobile User Interface (UI) Design is also an essential in the creation of mobile apps. Mobile UI considers constraints & contexts, screen, input and mobility as outlines for design. The user is often the focus of interaction with their device, and the interface entails components of both hardware and software. User input allows for the users to manipulate a system, and device's output allows the system to indicate the effects of the users' manipulation. Mobile UI design constraints include limited attention and form factors, such as a mobile device's screen size for a user's hand(s). Mobile UI contexts signal cues from user activity, such as location and scheduling that can be shown from user interactions within a mobile application. Overall, mobile UI design's goal is primarily for an understandable, user-friendly interface. The UI of mobile apps should: consider users' limited attention, minimize keystrokes, and be task-oriented with a minimum set of functions. This functionality is supported by Mobile enterprise application platforms or integrated development environments (IDEs).

Front-end development tools are focused on the user interface and user experience (UI/UX) and provide the following capabilities

- UI design tools
- SDKs to access device features
- Cross-platform accommodations/support

1. Android

Android is the name of the mobile operating system made by American company; Google. It most commonly comes installed on a variety of smartphones and tablets from a host of manufacturers offering users access to Google's own services like Search, YouTube, Maps, Gmail and more. The OS uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard.

Android is popular with technology companies which require a ready-made, low-cost and customizable operating system for high-tech devices. Android's open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which add new features for advanced users or bring Android to devices which were officially, released running other operating systems. The operating system's success has made it a target for patent litigation as part of the so-called "smart phone wars" between technology companies.

A list of features in the Android operating system
- Messaging.
- Web browser.
- Voice-based features.
- Multi-touch.
- Multitasking.
- Screen capture.
- Video calling.
- Multiple language support.
- Accessibility

Android supports connectivity technologies including GSM/EDGE, WiFi, Bluetooth, LTE, CDMA, E V-DO, UMTS, NFC,IDEN and WiMAX.Android devices can include still/video cameras, touch screens, GPS, accelerometers, gyroscopes, barometers, magnetometers, dedicated gaming controls, proximity and pressure sensors, thermometers, accelerated 2D bit blits (with hardware orientation, scaling, pixel format conversion) and accelerated 3D graphics.

2. IOS

IOS (originally iPhone OS) is a mobile operating system created and developed by Apple Inc. and distributed exclusively for Apple hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod touch. The user interface of iOS is based on the concept of direct manipulation, using multi-touch gestures. Interface control elements consist of sliders, switches, and buttons.

Other iOS features include:

- Integrated search support enables simultaneous search through files, media, applications and email.
- Gesture recognition supports, for example, shaking the device to undo the most recent action.
- Google Maps direction services.
- Push email.
- Safari mobile browser.
- Integrated camera and video.
- Integrated media player.
- Direct access to the Apple Store's catalogue of applications, music, podcasts, television shows and movies.
- Compatibility with Apple's cloud service, iCloud.

3. Windows:

Windows Mobile is a family of mobile operating systems developed by Microsoft for smart phones and Pocket PCs. Most versions of Windows Mobile have a standard set of features, such as multitasking and the ability to navigate a file system similar to that of Windows 9x and Windows NT, including support for many of the same file types. Similarly to its Windows 9x, it comes bundled with a set of applications that perform basic tasks. Windows Mobile is based on the Windows CE kernel and first appeared as the Pocket PC 2000 operating system. It is supplied with a suite of basic applications developed with the Microsoft Windows API, and is designed to have features and appearance somewhat similar to desktop versions of Windows.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Several versions of Windows are-

- Windows CE
- Pocket PC 2000
- Pocket PC 2002
- Windows Mobile 2003
- Windows Mobile 2003 SE
- Windows Mobile 5
- Windows Mobile 6
- Windows Mobile 6.1
- Windows Mobile 6.5
- Windows Embedded Handheld 6.5
- Smartphones

There are three main versions of Windows Mobile for various hardware devices:

- Windows Mobile Professional runs on smartphones with touch screens.
- Windows Mobile Standard runs on mobile phones without touch screen.
- Windows Mobile Classic which runs on personal digital assistant or pocket PCs.

4. Video Streaming

Streaming video is content sent in compressed form over the Internet and displayed by the viewer in real time. With streaming video or streaming media, a Web user does not have to wait to download a file to play it. Instead, the media is sent in a continuous stream of data and is played as it arrives. The user needs a player, which is a special program that uncompress and sends video data to the display and audio data to speakers. A player can be either an integral part of a browser or downloaded from the software maker's Web site. Major streaming video and streaming media technologies include Real System G2 from Real Network, Microsoft Windows Media Technologies (including its NetShow Services and Theater Server), and VDO. Microsoft's approach uses the standard MPEG compression algorithm for video. Streaming video is usually sent from prerecorded video files, but can be distributed as part of a live broadcast "feed." In a live broadcast, the video signal is converted into a compressed digital signal and transmitted from a special Web server that is able to do multicast, sending the same file to multiple users at the same time. Mobile video comes in several forms including 3GPP, MPEG-4, RTSP and Flash Lite.

1.3 Cloud Platform for Mobile App development

1.3.1 Aneka Platform as a Service

Aneka is a Manjra soft product which plays the role of Application Platform as a Service for Cloud Computing. MANJRASOFT Pvt. Ltd. Is one of best companies that works on developing future technologies for saving time and money? ANEKA is one of its first cloud computing technologies that work on developing clouds using .NET framework. MANJRASOFT besides working on future technologies also develops software compatible with distributed networks across multiple servers. It manages resources in cloud without violating service level agreements (SLA's) thus enabling less cost, application scheduling etc.

The word ANEKA means in many ways i.e. it has multiple programming models, multiple scheduling strategies, multiple authentication models and distributive environment for operating system. The main aim of ANEKA is to support open-ended set of abstractions and features for distributed computing and deployment scenarios. Aneka acts as a framework for building customized applications and deploying them on either public or private Clouds. One of the key features of Aneka is its support for provisioning resources on different public Cloud providers such as Amazon EC2, Windows Azure and Go Grid.

Aneka works on RAD (Rapid Application Development) environment to manage interconnected networks of systems. The word market oriented in context of ANEKA specifies that it is possible to build, schedule, monitor results by giving some money for using IT services like Quality of Service (QoS) in both public as well as private clouds. ANEKA is available at PaaS in cloud environment. It means that it provides programming application programming interfaces (API's) for developing distributed applications and virtual execution environment in which the applications developed as per API can be made to run.

Features of ANEKA

There are several features of ANEKA that helps in development of enabling cloud based environment for faster accessing of resources.

- ➢ It consists of RAD tools and framework.
- ➢ It combines with multiple virtual machines or existing machines to provide results of applications
- ➢ It uses provision interface thus following parameters like Quality of Service (QoS) and SLA (service level agreements)
- ➢ It supports multiple programming environments
- ➢ In this multiple applications can be executed simultaneously which increases utilization of resources.
- ➢ ANEKA means many forms. So, it has ability to provide different ways of working in distributed network with the help of programming models like Task Model, Map Reduce model and many more

1.3.2 Mobile App development in Aneka for Resource Intensive Applications

The latest developments in mobile devices technology have made smartphones as the future computing and service access devices. Users expect to run computational intensive applications on Smart Mobile Devices. Mobile device capabilities and increasing battery lifetime through the extension of cloud services and resources, resulting in an enhanced user experience. However, the development of a mobile cloud application is challenging because it involves dealing with different cloud providers and mobile platforms. To tackle the above issues, mobile cloud architecture is proposed to asynchronously delegate resource-intensive mobile tasks in order to handle the mobile device load and, consequently, extend the battery life. We demonstrate this capability by developing an interface that supports the delegation of heavy tasks from mobile apps running under the Android mobile platform to a cloud computing environment managed by the Aneka Cloud Application Platform.

The Aneka Mobile Client Library encapsulates the processes of communicating to cloud is provided, thus, the effort and complexity of developing a mobile cloud application is decreased**.** Aneka Mobile Client Library for Android platform that encapsulates the processes of connecting to cloud, serializing and de serializing messages, sending messages, and collecting their responses. Thus, the effort and complexity of developing a mobile cloud application is decreased. In addition, the library was designed to leverage the Aneka Cloud Application Platform, which provides transparent resource provisioning and job scheduling services and encapsulates different cloud providers Web APIs.

1.3.3 Deadline and budget based provisioning in Clouds for Resource Intensive Applications

Clouds can be used to provide extra resources whenever required in order to achieve it requires both policies defining when and how cloud resources are allocated to applications and a platform implementing not only these policies but also the whole software stack supporting management of applications and resources. Aneka is a cloud application platform capable of provisioning resources obtained from a variety of sources, including private and public clouds, clusters, grids, and desktops grids. Aneka implement deadline driven provisioning mechanism, which is responsible for supporting quality of service (QoS)-aware for execution of scientific applications. This mechanism shows that Aneka is able to efficiently allocate resources from different sources in order to reduce application execution times.

When the application resources may be insufficient in certain periods of time which can lead to long waiting times for utilization of these resources, or the available resources for one application may be insufficient to complete the application before its deadline. In these cases, scientific resources may be complemented by cloud resources. Moreover, by leasing cloud computing services on a pay-per-use basis, even can easily access a large number of resources, which are utilized and paid for only for the time they are actually utilized. To achieve this a middleware supporting provisioning of resources from both local infrastructures and public clouds is required, so that applications can transparently migrate to public virtual infrastructures. Aneka is a software platform for building and managing a wide range of distributed systems, allowing applications to receive resources provisioned from different sources, such as desktop grids, scientific grids, clusters, private clouds, and public clouds managed transparently by Aneka.

Aneka platform, which enables not only utilization of clouds, but also utilization of virtually any kind of computational resource available for applications, including idle desktops from local networks, clusters, and grids which present deadline-driven provisioning of resources for scientific applications.

Deadline-driven resource provisioning algorithm
Algorithm: Deadline-driven provisioning in Aneka.

```
1.  for each request with QoS constraints do
2.  resources available resources for the application;
3.  pendingTasks←number of tasks in the queue;
4.  eft ← pending Tasks  × average Task Runtime;
            Resources
5.  if eft> application Time Remaining then
6.  extra Resources ← pending Tasks × average Task
Runtime application Time Remaining
7.  // invokes Algorithm 2 for resource provisioning
8.  Provisioner.selectResources(applicationId,  extra
Resources);
9.  else
10. to Release ← 0;
11. if pending Tasks< resources then
12. to Release ← pending Tasks − resources;
13. end
14. else
15. pending Tasks ← pending Tasks + runningTasks;
16. left ←pending Tasks    × averageTaskRuntime;
resources
17. if eft<applicationTimeRemaining then
18. to Release ←resources −pending Tasks       ×
averageTaskRuntime applicationTimeRemaining
19. end
20. end
21. Provisioner.releaseResources(applicationId,    to
Release);
22. end
23. end
```

This algorithm related to the management of resources from different sources, other important aspects to be considered that

(i) When such a process of resource allocation takes place and how many resources are requested, and
(ii) Which of the available resource sources are used in a particular provisioning request?

This decision is driven by the application QoS, which is expressed in terms of the deadline for application completion. The deadline driven policy is a best-effort algorithm that considers the time left for the deadline and the average execution time of tasks that compose an application to determine the number of resources required. For each request with QoS constraints, the Service provider considers its deadline, number of tasks, and task runtime estimation to determine if the deadline can be met. If the Service detects that the deadline cannot be met, it determines the number of extra resources required and submits a request, containing the request and the number of resources, to the Service provider.

This process takes place either to scale a single application across multiple sites or to provide dynamic resources to multiple applications in execution in the Aneka cloud, and the process is repeated every time a new request is received by Aneka and every time a task completes or fails. On the other hand, if the Service provider detects that the request does not require all the resources currently allocated to it, it indicates to the Service provider that some of these resources can be released to be used by other requests.

For a decision about the specific source of resources to be used for each resource request By default, resources are allocated first from local static and dynamic resources, then from ‘‘free’’ resource pools. Inside each of these classes of resource sources (e.g., different public Cloud providers in the case of paid external resources), the source of resources and the preferred order is defined by the Aneka administrator in an input configuration file. Notice that this algorithm operates in a best effort manner: if the provider cannot allocate the exact number of resources requested by the user, it returns as much as it can get from all the available sources. Moreover, the time for preparation of the system (e.g., start up of VMs) is not taken into account by the provisioning mechanism, and thus there may be small delays in task processing.

Other policies, which consider data locality and performance costs of file transferring, have currently to be defined by the system administrator, though these policies are planned to be automatically provided by Aneka in the future. Once dynamic resources join the Aneka cloud, they must be properly managed and these resources are typically subject to a usage cost. In particular, current practices for billing by use of cloud resources consider their usage in terms of time units whose granularity varies among providers.

II. LITERATURE SURVEY

Introduction:

Literature survey plays a major role in collecting the literatures based on the objective. Using the literature which was collected I am going to implement my objective. Literature survey is an important step in a development process. Before developing the concept it is necessary to collect the literatures based on the objective.
In this chapter, I try to accomplish the following important objectives in preparing a literature review:

1. The review should provide an overview of previous research on the topic **“Deadline-driven provisioning of resources for scientific applications in hybrid”.**In this some applications require large computing power, traditionally exceeding the amount that is available within the premises. Clouds can be used to provide extra resources whenever required. To achieve this, it requires both policies defining when and how cloud resources are allocated to applications and a platform implementing not only these policies but also the whole software stack supporting management of applications and resources. Aneka is a cloud application platform capable of provisioning resources obtained from a variety of sources, including private and public clouds.

In this paper, we present Aneka’s deadline driven provisioning mechanism, which is responsible for supporting quality of service (QoS)-aware execution of scientific applications in clouds composed of resources obtained from a variety of sources resulting that Aneka is able to efficiently allocate resources from different sources in order to reduce application execution times.

2. Another overview on previous research on the topic **“Cost-based scheduling of Scientific Workflow Applications on Utility Grids”.** Grid technologies have progressed towards a service-oriented paradigm that enables a new way of service provisioning based on utility computing models. Users consume these services based on their QoS (Quality of Service) requirements. In such “pay-per-use” workflow execution cost must be considered during scheduling based on users QoS constraints.

In this paper, we propose a cost-based workflow scheduling algorithm that minimizes execution cost while meeting the deadline for delivering results. It can also adapt to the delays of service executions by rescheduling unexecuted tasks.

3. Another overview on previous research on the topic **“Outsourcing Resource-Intensive Tasks from Mobile Apps to Clouds: Android and Aneka Integration”.** In this Mobile Cloud Computing enables augmenting mobile device capabilities and increasing battery lifetime through the extension of cloud services and resources, resulting in an enhanced user experience. However, the development of a mobile cloud application is challenging because it involves dealing with different cloud providers and mobile platforms. To tackle the above issues, mobile cloud architecture is proposed to asynchronously delegate resource-intensive mobile tasks in order to handle the mobile device load and, consequently, extend the battery life.

In this paper demonstrated that to handle this capability by developing an interface that supports the delegation of heavy tasks from mobile apps running under the Android mobile platform to a cloud computing environment managed by the Aneka Cloud Application Platform. The Aneka Mobile Client Library encapsulates the processes of communicating to cloud is provided and the effort and complexity of developing a mobile cloud application is decreased. A performance evaluation is conducted showing the feasibility of architecture through the reduction of application execution time and extension of mobile device battery life.

3.1 System architecture

The layered architecture is depicted in Figure 3.1. The layers are – Resource layer, Platform layer, SDK and Application developments. The detailed description of each layer is given below.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

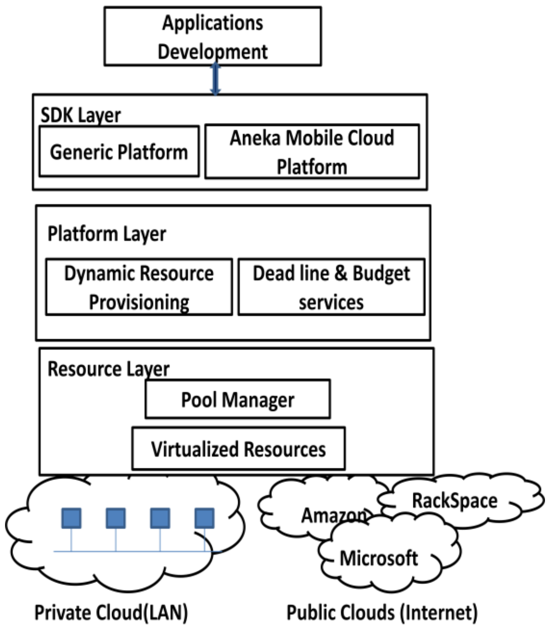


Fig3.1 Layered Architecture.

Figure3.1 provides a layered view of the framework. It is an essentially implementation of the PaaS model, and it provides a runtime environment for executing applications by leveraging the underlying infrastructure of the cloud. Developers can express distributed applications by using the API contained in the Software Development Kit (SDK) or by porting existing legacy applications to the cloud. Such applications are executed on the Aneka mobile cloud platform, represented by a collection of nodes connected through the network hosting the Aneka. System framework is the building block of the middleware, and it represents the runtime environment for executing applications, it contains the core functionalities of the system and is constituted of an extensible collection of services that allow administrators to customize the cloud.

3.2 Elements:

There are three types of elements in layered architecture that provides services to the applications namely
1. Resource Layer
2. Platform Layer
3. SDK Layer

1. Resource Layer:

Resource layer consist of
a. Pool manager
b. Virtualization resources.

Resource layer is a container of virtual resources that mostly come from the same resource provider. A resource pool is in charge of managing the virtual resources it contains and eventually releasing them when they are no longer in use. Since each vendor exposes its own specific interfaces, the resource pools are

- Encapsulates the specific implementation of the communication protocol required to interact with it and
- Provides the pool manager with a unified interface for acquiring, terminating, and monitoring virtual resources.

Pool Manager:

The pool manager performs most of all the management tasks of the pools and the place where dynamic provisioning strategies can be implemented. The main responsibility of the pool manager is controlling the life cycle of pools and manages all the registered resource pools and decides how to allocate resources from those pools and provides a uniform interface for requesting additional resources from any private or public cloud and redirects a provisioning request, release, or query to the appropriate pool which manages multiple pools to the resource provisioning service. The pool manager also notifies the provisioning service when a dynamic resource is activated and terminated.

Virtualization Resource:

Virtualization resource is used to monitor and manage the infrastructure and acquiring resources from different implementations of virtualization technologies such as Xen, KVM and VMware can help in building the foundations of a virtual Infrastructure in order to scale applications on demand. This task can be performed by Virtual machine manager (VMM) technology that is able to provide virtual infrastructure by creating and managing templates, creating and controlling the life cycle of Virtual machines.

Platform Layer:

It provides a runtime environment for executing applications by leveraging the underlying infrastructure of the cloud. This layer consist of
a. Dynamic Resource Provisioning.
b. Deadline and Budget Services

a. Dynamic Resource Provisioning:

This service is responsible for satisfying a provisioning request. It mainly performs the following operations: resource provision, resource release, resource status query, and resource pool status query and these functionalities can be handled by pool manager.
Aneka identifies two types of private resources: static and dynamic resources. Static resources are constituted by existing physical workstations and servers that may be idle for a certain period of time. Their membership to the Aneka cloud is manually configured by provider and does not change over time. Dynamic resources are mostly represented by virtual instances that join and leave the Aneka cloud and are controlled by resource pool managers that provision and release them when needed.

b. Dead line and budget services:

In order to schedule the execution of the tasks within the expected deadline. If the local resources are not enough to execute all the tasks in time, a request for additional resources is issued.

3. SDK Layer:

Developers can express distributed applications by using the API contained in the Software Development Kit (SDK) or by porting existing legacy applications to the cloud. Such applications are executed on the Aneka cloud.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

3.2.1 Virtualization architecture – Xen

Virtualization is a framework or methodology of dividing the resources of a computer into multiple execution environments, by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service etc. Basically one physical machine runs only one OS at any time where as by using virtualizing the machine, we are able to run several operating systems (and all of their applications) at the same time.

E.g. Xen

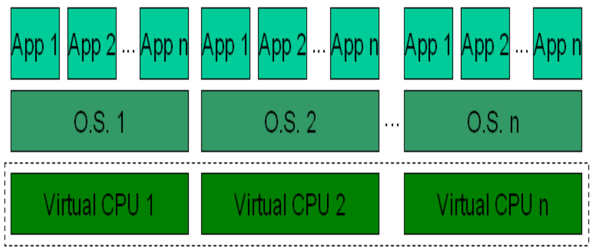


Fig3.2 Virtualization

Xen is virtualization software which providing services that allows multiple computer operating systems to execute on the same computer hardware concurrently in which we are going to create VMs.

Creating VMs:

VMs can be created by
- ➢ Physical to Virtual Conversion (P2V)
- ➢ Cloning an existing VM

Physical to Virtual Conversion (P2V)

P2V is the process by which an existing Windows operating system on a physical server — its file system, configuration, and so on — is converted to a virtualized instance of the operating system. This is then is transferred, instantiated, and started as a VM on the Xen Server host

Cloning an Existing VM.

VMs are prepared from templates. A template is a "image" that contains all the various configuration settings to instantiate a specific VM. Xen Server ships with a base set of templates, which are "raw" VMs, on which you can install an operating system. Different operating systems require different settings in order to run at their best. Xen Server templates are tuned to maximize operating system performance.

There are two basic methods by which you can create VMs from templates:
- Using a complete pre-configured template
- Installing an operating system from a CD, ISO image or network repository onto the appropriate provided template

3.2.2 Aneka – Xen Dynamic Resource provisioning

Aneka provides resource provisioning facilities in dynamic fashion. Applications managed by the Aneka that can be dynamically mapped to heterogeneous resources, which can grow or shrink according to the application's needs. This elasticity is achieved by means of the Aneka resource provisioning framework.

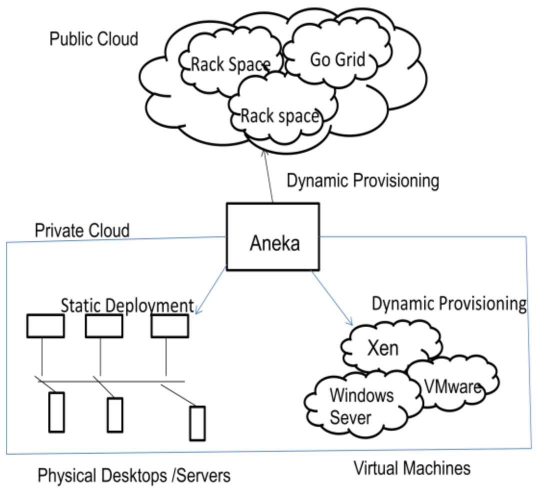


Fig3.3 Aneka Resource Provisioning Framework

Figure 3.3explain an overview of Aneka resource provisioning over private and public clouds, it combines privately owned resources with public rented resources to dynamically increase the resource capacity to a larger scale.

Private resources identify computing and storage elements kept in the premises that share similar internal security and administrative policies. Aneka identifies two types of private resources: static and dynamic resources. Static resources are constituted by existing physical workstations and servers that may be idle for a certain period of time. Their membership to the Aneka cloud is manually configured by administrators and does not change over time. Dynamic resources are mostly represented by virtual instances that join and leave the Aneka cloud and are controlled by resource pool managers that provision and release them when needed.

Public resources reside outside the boundaries are provisioned by establishing a service-level agreement with the external provider. Even in this case we can identify two classes: on-demand and reserved resources. On-demand resources are dynamically provisioned by resource pools for a fixed amount of time (for example, an hour) with no long-term commitments and on a pay-as-you-go basis. Reserved resources are provisioned in advance by paying a low, one-time fee and mostly suited for long-term usage. These resources are actually the same as static resources, and no automation is needed in the resource provisioning service to manage them. The resources are managed uniformly once they have joined the Aneka cloud and all the standard operations that are performed on statically configured nodes can be transparently applied to dynamic virtual instances.

Moreover, specific operations pertaining to dynamic resources, such as join and leave, are seen as connection and disconnection of nodes and transparently handled. This is mostly due to the indirection layer provided by the Aneka that abstracts the specific nature of the hosting machine.

4.2 ALGORITHMIC IMPLEMENTATION PART – DEAD LINE AND BUDGET

The implementation is on the main algorithm used in this project. Here are the main algorithms to implementing in the proposed system

Algorithm4.2.1:

```
Deadline-driven resource provisioning in Aneka algorithm
Input: No. of VM's
Output: No. of resources
1.    for each request with QoS constraints do
2.    resource available resources for the application;
3.    pending Tasks ← number of tasks in the queue;
4.    left ← pending Tasks   × average Task Runtime;
Resources
5.    if eft> application Time Remaining then
6.    extra Resources ← pending Tasks × average Task
Runtime application Time Remaining
// invoking for resource provisioning
1.    Provisioner.selectResources(applicationId, extra
Resources);
2.    else
3.    to Release ← 0;
4.    if pending Tasks< resources then
5.    to Release ← pending Tasks − resources;
6.    end
7.    else
8.    pending Tasks ← pending Tasks + runningTasks;
9.    eft    ←    pending  Tasks    ×
averageTaskRuntime;
i.    resources
10.   if eft<applicationTimeRemaining then
11.   to Release ←
12.   resources −    pending  Tasks    ×
averageTaskRuntime
13.   applicationTimeRemaining
14.   end
15.   end
16.   Provisioner.releaseResources(applicationId,   to
Release);
17.   End
18.   End
```

This algorithm related to the management of resources from different sources, important aspects to be considered
(i) When such a process of resource allocation takes place and how many resources are requested, and
(ii) Which of the available resource sources are used in a particular provisioning request.

This decision is driven by the application QoS, which is expressed in terms of the deadline for application completion. The deadline driven policy is a best-effort algorithm that considers the time left for the deadline and the average execution time of tasks that compose an application to determine the number of resources required.
For each request with QoS constraints, the Service provider considers its deadline, number of tasks, and task runtime estimation to determine if the deadline can be met. If the Service detects that the deadline cannot be met, it determines the number of extra resources required and submits a request, containing the request and the number of resources, to the Service provider. This process takes place either to scale a single application across multiple sites or to provide dynamic resources to multiple applications in execution in the Aneka cloud, and the process is repeated every time a new request is received by Aneka and every time a task completes or fails. On the other hand, if the Service provider detects that the request does not require all the resources currently allocated to it, it indicates to the Service provider that some of these resources can be released to be used by other requests.

For a decision about the specific source of resources to be used for each resource request By default, resources are allocated first from local static and dynamic resources, then from ‘‘free’’ resource pools. Inside each of these classes of resource sources (e.g., different public Cloud providers in the case of paid external resources), the source of resources and the preferred order is defined by the Aneka administrator in an input configuration file. Notice that this algorithm operates in a best effort manner: if the provider cannot allocate the exact number of resources requested by the user, it returns as much as it can get from all the available sources. Moreover, the time for preparation of the system (e.g., start up of VMs) is not taken into account by the provisioning mechanism, and thus there may be small delays in task processing.

Other policies, which consider data locality and performance costs of file transferring, have currently to be defined by the system administrator, though these policies are planned to be automatically provided by Aneka in the future. Once dynamic resources join the Aneka cloud, they must be properly managed and these resources are typically subject to a usage cost. In particular, current practices for billing by use of cloud resources consider their usage in terms of time units whose granularity varies among providers.

4.2 Implementation

In this, we are going to implement Xen Server for creating images and templates, dynamic resource provisioning, Aneka, deadline, budget and mobile context and Mandel Droid Mobile App.

4.2.1 Xen Server

Xen server that allow the services to multiple computer operating systems to execute on the same computer hardware concurrently. It is a virtualization platform that lowers the total cost of ownership for desktop, clouds, server’s virtualization infrastructure.
a. First install Xen in server

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

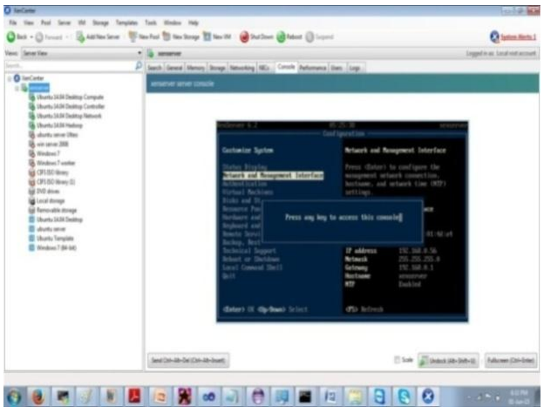
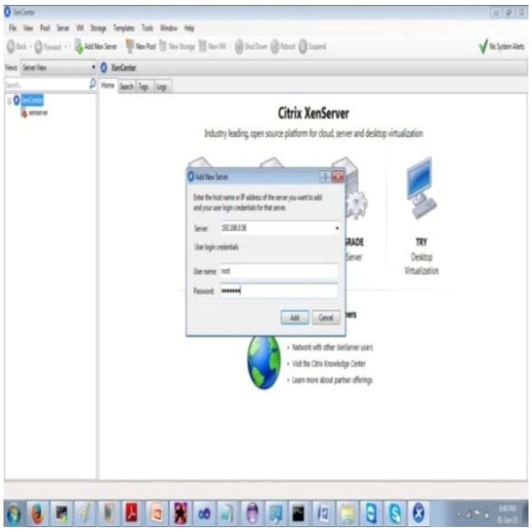
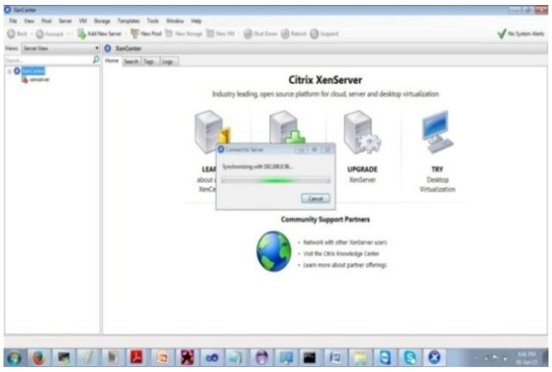
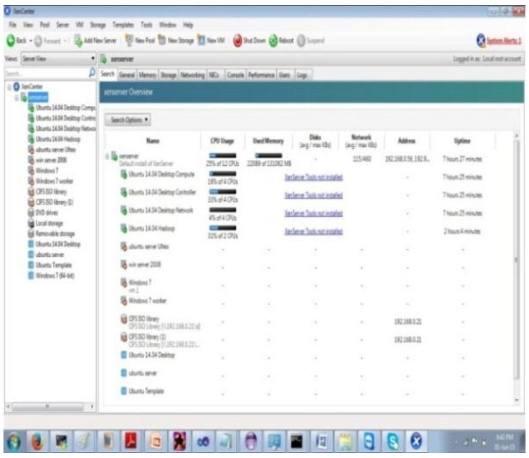

XenCenter
Press any key to access this console
Customize System
Network and Management Interface

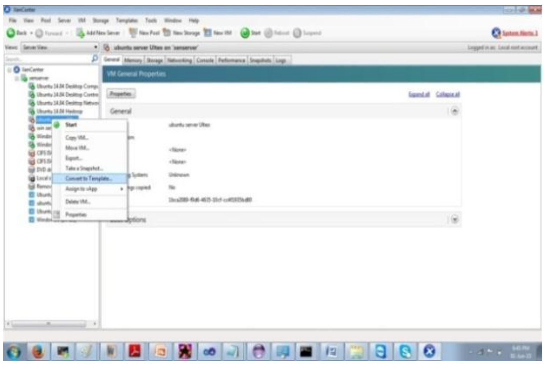
Fig4.1Xen Sever

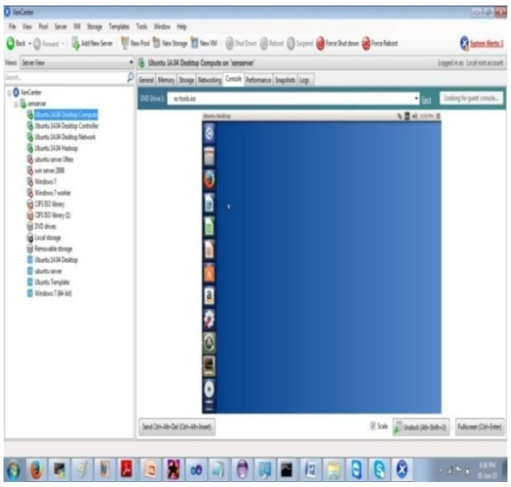

Citrix XenServer
Industry leading, open source platform for cloud, server and desktop virtualization
Add New Server
TRY
Desktop Virtualization
Add
Cancel

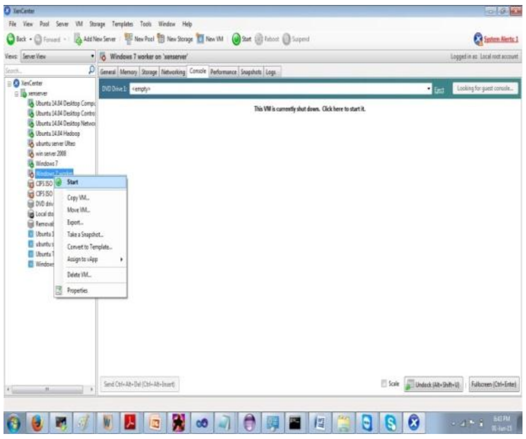

Citrix XenServer
Industry leading, open source platform for cloud, server and desktop virtualization
UPGRADE
XenServer
TRY
Desktop Virtualization
Community Support Partners
Cancel



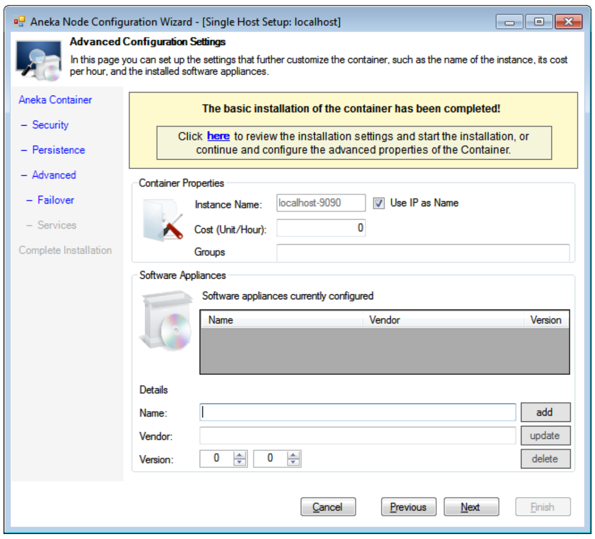
4.2.2 Images and templates







4.2.3 Deadline, Budget and Mobile Context


Aneka Node Configuration Wizard - [Single Host Setup: localhost]

Advanced Configuration Settings

In this page you can set up the settings that further customize the container, such as the name of the instance, its cost per hour, and the installed software appliances.

The basic installation of the container has been completed!

Click here to review the installation settings and start the installation, or continue and configure the advanced properties of the Container.

Container Properties

Instance Name: localhost-9090 [✓ Use IP as Name]

Cost (Unit/Hour): 0

Groups

Software Appliances

Software appliances currently configured

Name	Vendor	Version

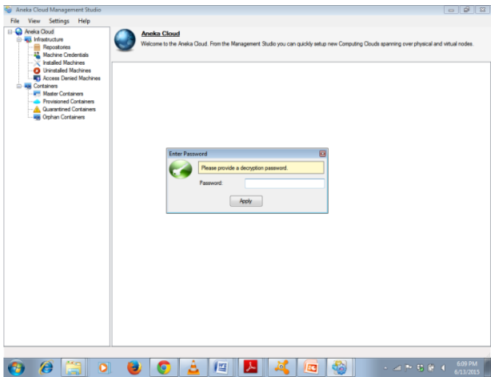
Details

Name:

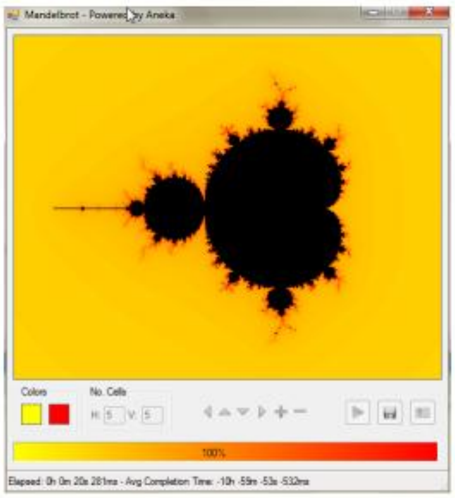
Vendor:

Version: 0 0

4.2.3 Dynamic Resource provisioning Xen and Aneka


Aneka Cloud Management Studio
File View Settings Help
Aneka Cloud
Enter Password
Please provide a decryption password.
Password
Apply

4.2.5 Mandel Droid Mobile App


Mandelbrot - Powered by Aneka
Colors
No. Cells
100%
Elapsed: 0h 0m 20s 281ms - Avg Completion Time: -10h -59m -53s -532ms

III. CONCLUSIONS AND FUTURE DIRECTIONS

Cloud computing quickly became the platform of choice in many practical scenarios in a business context. Nevertheless, its adoption is limited in the context of computational science. Scientific applications requiring larger amounts of computing power than can be delivered by local resources within a given timeframe can utilize clouds, which can deliver this required capacity with minimal effort in terms of the configuration of hardware platforms. An obstacle for the adoption of clouds for scientific applications is taking advantage of such platforms when legacy systems are still used. Different operating systems, programming languages, and software platforms supported by each system can make this integration hard. Aneka addresses these issues by supporting seamless integration of resources from a range of sources that include desktop grids, clusters, grids, public clouds, and private clouds to support QoS-aware execution of applications. Aneka’s features were demonstrated in experiments that showed that it is able to efficiently allocate resources from different sources in order to reduce application execution times. Improvements in Aneka’s dynamic resource provisioning are under development, applications will run more efficiently in hybrid resources.

The experiment presented in this paper addressed the case of applications requiring a small amount of data transfer: the input files, output files, and application together were smaller than 1 MB. Provisioning mechanisms more suitable for data-intensive HPC applications – such as data location-aware provisioning of hybrid resources, which attempts to select providers that contains all or part of the data required by the application – are also the subject of future research. We are developing support for the integration of multiple Clouds in Aneka according to the Intercloud [26] model. In this model, providers interact via a marketplace where they can either negotiate resources for serving their jobs or they can outsource jobs to other Clouds upon compensation to the party receiving the job. This will further expand the range of different sources of resources that can be integrated by Aneka, leading to its ultimate goal of supporting QoS-aware execution of applications using any relevant programming model.

REFERENCES

[1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: vision, hype, and reality for delivering IT services as the 5th utility, Future Generation Computer Systems 25 (6) (2009) 599–616.

[2] I. Foster, K. Kesselman (Eds.), The Grid 2: Blueprint for a New Computing Infrastructure, 2nd ed., Morgan Kaufmann Publishers, 2004.

[3] C. Vecchiola, X. Chu, M. Mattess, R. Buyya, Aneka — integration of private and public clouds, in: R. Buyya, J. Broberg, A. Goscinski (Eds.), Cloud Computing: Principles and Paradigms, Wiley Press, 2011.

[4] C. Vecchiola, X. Chu, R. Buyya, Aneka: a software platform for. NET-based cloud computing, in: W. Gentzsch, L. Grandinetti, G. Joubert (Eds.), High Performance and Large Scale Computing, IOS Press, 2009.

[5] F. Gagliardi, M. Begin, EGEE—providing a production quality grid for e-Science, in: Proceedings of the IEEE International Symposium on Mass Storage Systems and Technology, 2005.

[6] C. Catlett, TeraGrid: a foundation for US cyber infrastructure, in: Proceedings of the International Conference on Network and Parallel Computing, 2005.

[7] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery,K. Blackburn, T. Wenaus, F. Wurthwein, I. Foster, R. Gardner, M. Wilde,A. Blatecky, J. McGee, R. Quick, The open science grid, Journal of Physics: Conference Series 78 (1) (2007) 12–57.

[8] W. Gropp, E. Lusk, A. Skjellum, Using MPI—2nd Edition: Portable Parallel Programming with the Message Passing Interface (Scientific and Engineering Computation), The MIT Press, 1999.
[9] K. Keahey, T. Freeman, Science clouds: early experiences in cloud computing for scientific applications, in: Proceedings of the Cloud Computing and its Applications Conference, 2008.
[10] C. Evangelinos, C. Hill, Cloud computing for parallel scientific HPC applications: feasibility of running coupled atmosphere–ocean climate models on Amazon'sEC2, in: Proceedings of the Cloud Computing and Its Applications Conference, 2008.
[11] E. Deelman, G. Singh, M. Livny, B. Berriman, J. Good, The cost of doing science on the cloud: the Montage example, in: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, 2008.
[12] J. Miller, S. Ragsdale, The Common Language Infrastructure Annotated Standard, Addison Wesley, 2004.
[13] J. Dean, S. Ghemawat, Map Reduce: simplified data processing on large clusters, in: Proceedings of the 6th Symposium on Operating Systems Design and Implementation, 2004.
[14] C. Varela, G. Agha, Programming dynamically reconfigurable open systems with SALSA, ACMSIGPLAN Notices 36 (12) (2001) 2034
[15] M. Kirley, R. Stewart, Multi objective evolutionary algorithms on complex networks, in: Proceedings of 4th International Conference Evolutionary Multi-Criterion Optimization, 2007.
[16] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms, JohnWiley & Sons, 2001.
[17] C. Vecchiola, M. Kirley, R. Buyya, Multi-objective problem solving with offspring on enterprise clouds, in: Proceedings of the 10th International Conference on High-Performance Computing in Asia-Pacific Region, 2009.
[18] M.D. de Assunção, A. di Costanzo, R. Buyya, A cost-benefit analysis of using cloud computing to extend the capacity of clusters, Cluster Computing 13 (3)(2010) 335–347.
[19] D. Kondo, B. Javadi, P. Malecot, F. Cappello, D.P. Anderson, Cost-benefit analysis of cloud computing versus desktop grids, in: Proceedings of the 18thInternational Heterogeneity in Computing Workshop, 2009.
[20] H. Kim, Y. el Khamra, S. Jha, M. Parashar, Exploring application and infrastructure adaptation on hybrid grid–cloud infrastructure, in: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, 2010.
[21] S. Ostermann, R. Prodan, T. Fahringer, Extending grids with cloud resource management for scientific computing, in: Proceedings of the 10th IEEE/ACM International Conference on Grid Computing, 2009.
[22] W. Tan, R. Madduri, A. Nenadic, S. Soiland-Reyes, D. Sulakhe, I. Foster, C. Goble,CaGrid Workflow Toolkit: a taverna based workflow tool for cancer grid, BMC Bioinformatics 11 (1) (2010) 542.
[23] C. Vázquez, E. Huedo, R.S. Montero, I.M. Llorente, Dynamic provision of computing resources from grid infrastructures and cloud providers, in: Proceedings of the Workshops at the Grid and Pervasive Computing Conference, 2009.
[24] P. Marshall, K. Keahey, T. Freeman, Elastic site: using clouds to elastically extend site resources, in: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010.
[25] B. Soto mayor, R.S. Montero, I.M. Llorente, I. Foster, Virtual infrastructure management in private and hybrid clouds, Internet Computing 13 (5) (2009)14–22.
[26] R. Buyya, R. Ranjan, R.N. Calheiros, Inter Cloud: utility-oriented federation of cloud computing environments for scaling of application services, in: Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing, 2010.

BIOGRAPHY

Authors have the option to publish a biography together with the paper, with the academic qualification, past and present positions, research interests, awards, etc. This increases the profile of the authors and is well received by international reader.