CLOUD COMPUTING MATERIAL (18PCS8)

II M.Sc Computer Science

(Semester - III)

Cloud:

* A visible mass of condensed water vapour floating in the atmosphere.

Cloud Computing:

It is the use of h/w and s/w to deliver a service over a n/w. (Internet).

* With CC, users, can access files, use applications from any device that can access the internet.

(eg) cloud computing -> Googles GMail.

—.

Roots of cloud computing: Mainframe to cloud

Roots are: Advancement in several Technologies

(ie) (*) hardware (virtualization, multi-core chip)

(*) Internet Technologies [Web service (SOA (service oriented architectutre) etc]

(*) Distributed Computing (clusters, grid)

(*) Systems Management (automated computing, data center automation).

Virtualisation -> creates a virtual version of device or resource (ie) server, storage or ~~n/w~~ n/w or OS.

Grid Computing -> In GC in addition to the OS there is a co-ordinator programme that manages the resources and split the tasks.

Distributed Computing ->

eg. on line games - where multiple users can interact. also known as MMOG's. (Massively Multiplayer Online Games)

Here, the virtual world maintained by the central system. But the processing power of individual computer or game consoles is used at the same time.

Systems Management -> enterprise-wide

- administration of a distributed systems (includes computers).
- n/w management
- Applications performance management.

Layers and Types of clouds.

- CC services are divided into 3 classes.

1. Infrastructure as a Service [IaaS]
2. Platform as a Service [PaaS]
3. Software as a Service [SaaS]

- cloud computing stack -

Service class	Main access & Management Tool	Service content
SaaS	Web Browser	cloud Application (social n/w, office suites CRM, video processing)
PaaS	Cloud Development Environment	Cloud Plat form (Programming languages, Frameworks, structural data) Mashub editor
IaaS	Virtual Infrastructure Manager	Cloud Infrastructure (Computer server, Data storage firewall, Load Balancer

* Mashub — In web development it's a webpage or application that integrates complementary elements from two or more sources.

1. Infrastructure asa Service (IaaS)

* It offers virtualised resources (computation, storage and communication) on demand.

* IaaS is the bottom layer in CC systems.

(eg) Amazon Web Services mainly offers IaaS.

(ie) offers EC2 service — Means offering VM's with a software stack.

users can perform activities in the server like starting & stoping it, customizing by installing s/w package, etc....

2. Platform asa Service (PaaS)

* It offers an environment, on which developers create and deploy applications. [No need to know how many processors, or how much memory that application is using]

* Multiple programming models and specialised services (data access, authentication, payments) are added as building blocks to new applications.

(eg) PaaS — Google AppEngine.
— offers scalable environment for developing and hosting web applications. (uses Python & Java).

3. Software asa Service (SaaS)

- Applications reside on the top of cloud stack.
- Services provided by SaaS accessed by end-user through web portal.

* Traditional word processing & spreadsheets, now can be accessed as a service in the web.

(eg) salesforce.com – which relies on the SaaS model

* offers business productivity applications (CRM) which resides on servers, allows consumers to customize & access applications on demand.

~~Classification~~ Types of cloud:

deployment models

Types of clouds based on deployment models

1) Public (Internet) clouds
2) Private (Enterprise) clouds
3) Hybrid (Mixed) clouds

* 3rd party, multi-tenant cloud infrastructure & services
* Available of subscription basis.

Def: Armbrust et. al. "cloud made available in a pay-as-you-go manner to the gen. public"

Runs within a company's own Data center / infrastructure for internal and/ partner use.

Mixed usage of public & private clouds. Leasing public cloud services when private cloud capacity is insufficient.

Features of a cloud:

1. Self Service:
   - services available on demand
   - instant access to the resources
   - customers can request, customise, pay and use services. without human operators

2. Per-Usage metering & Billing

* Services by short term basis (eg. by hour etc.) allowing users to release resources when they not needed. So,
metering should be done for different types of services (eg. Storage, processing, bandwidth)

3. Elasticity:

(ie) If needed additional resources can be (a) provisioned and if not (b) released when load decreases.

4. Customization:

Resource rented from the cloud is highly customizable.

~~Infraf~~ Infrastructure Management:

key challenges of IaaS providers when building a cloud infrastructure:

* Managing physical & virtual resources
(ie) namely Servers, storage & n/w.

The s/w toolkit is responsible for this, called VIM. (Virtual Infrastructure Manager)

(ie) It's like traditional OS. - Instead of dealing with single computer, aggregate resources from multiple computer, presenting a uniform view to user & applications.

* VIM can also called as "Cloud operating System"

* Also called as "Infrastructure sharing software"
(or) "Virtual ~~Infrast~~ Infrastructure Engine"

Features of VIM :

Basic & Advanced features available in VIM :

* Virtualization support : multiple customers with disparate requirements to be served by a single h/w infrastructure

* Self Service, on demand resource provisioning
— enable users to directly obtain services from clouds
(no human intervention) * highly desirable feature.

* Multiple Backend Hypervisors :
Its a s/w, h/w or firmware. ~~is~~ that creates and runs virtual machines.

* Storage virtualization :
— Means abstracting logical storage from physical storage devices & location.
— creates virtual disks, independent from devices in a data center, (From the available storage devices

* Interface to public clouds : Extra loads can be offloaded to rented resources.

* Virtual N/wing : ~~allo~~creates combining h/w & s/w into a single
n/w resources and n/w functionality
s/w based administrative entity.

* Dynamic Resource Allocation :
Allows business customers to scale up and down their resources based on the needs.

* Virtual clusters : purpose of using VM is to consolidate multiple functionalities on the same server.
(virtual clusters are built with VM)

* Reservation and negotiation Mechanism :

* High availability and data recovery :

— a —.

Infrastructure as a Service providers

* Public IaaS providers offers:
virtual servers containing one or more CPU,
running several OS &
a customized S/W stack.

Features

* Availability of special features that influence the cost benefit ratio. (to be experienced by user)

Most (important) relevant features are:

- * Geographical distribution of data centers
- * Variety of user interface and API to access the system.
- * Specialised components and services that aid particular application (eg. load balancer, firewall)
- * Choice of virtualisation platform and OS
- * Different billing methods & periods. (prepaid, post paid, hourly, monthly etc.)

(eg) Amazon Web Services (AWS)

* Introduced in 2006.

offers:
S3 (storage) service
EC2 (virtual servers)

EC2 - Elastic compute cloud

Cloudfront (content delivery)
cloud streaming (video streaming)
SimpleDB (structured Data store)
RDS (Relational Database)
SQS (reliable messaging)
Elastic MapReduce (Data processing)

Platform as a Service providers

* public Paas offers deployment + environments which allows users to create & run their applicatio

Features:

* Programming Models, Languages + Frameworks:

Paas providers support multiple programming Languages. most commonly used language Python and Java & .Net., Ruby.

Force.com deviced Apex (Programming Language) + Excell like query language.

variety of s/w framework is made available to Paas Developers.

popular web application frame work offers on Ruby on Rails, Spring, Java EE & .Net.

* Persistence option:

persistence layer is very essential. It allows applications to record their state and recovers from crashes, as well as to store user data.

Traditionaly Relational Databases, used by application developers, as persistence method.

(eg) Amazon Simple DB
Google AppEngine

— x —

Challenges and Risks

Issues to be focused includs: privacy, data security, data lock-in, availability of service, disaster recovery, performance scalability, energy- efficiency, and programmability.

(1) Security, privacy and Trust:

* Security and privacy affect the entire cloud stack.
* Trust towards providers is fundamental, to ensure

desired level of privacy for the applications hosted on the cloud.

* Legal and regulatory issues needs attention.

[can upload data to the cloud, anywhere on the planet physical location of data centres determines the set of laws that can be applied to the management of data]

(2) Data Lock-in and Standardization:

* It's a major concern. (data locked-in by certain providers)

* Users may move data & app from a provider, which does not meet their requirements.

Because, there is no standard methods for storing user data & applications in the cloud comp infrastructure So, standardization is needed.

* CCIF - cloud Computing Interoperability Forum. (was formed by Intel, Sun, cisco)

* Development of UCI (Unified cloud Interface) by CCIF - aims at creating a standard programmatic point of access to an entire cloud infrastructure.

* OVF (Open Virtual Format) aims at facilitating Packing & distribution of S/w to be run on VM's.

(3) Availability, fault Tole~~rance, & Disaster recovery~~

* Once user applications are moved to cloud, it should be available when any one needed.

* So SLA (Service Level Agreement) is needed between customer and provider.

SLA contains -> details of service to be provided, availability & performance guarantees.

(4) Resource Management and Energy Efficiency:

* Should take care of the physical resources such as CPU, disk space, n/w bandwidth, because they are in heterogeneous workload.

* Thats why dynamic VM is used.

* Its important to minimize the energy consumption in data centers. This can be done by consolidating the work load, and turning off the idle resources.

Broad approaches to Migrating into the cloud

Cloud computing is a "Techno-business distributive model".

why migration?

* Because of economic and business reason.

Migration of an application into the cloud can happen because of the following ways:

* either the application is clean & independent
* Some degrees of code needs to be modified & adapted
* migration results in the core architechture being migrated for a cloud computing service setting

Migration can happens at one of the following five levels of: applications, code, design, architechture & usage.

migration can be captured as follows:

$$P \rightarrow P'_c + P'_l \rightarrow P'_{OFc} + P'_l$$

"p" – application before migration

P'_c – application part after migration into the cloud.

P'_l – part of application being run . on capture local data caitenters

P'_{OFc} – application part optimized for cloud.

* "If an enterprise application cannot be migrated fully, parts being runs on local data centre".

* when it is fully migrated P'_l is null.

So migration can happen at 5 levels, application, code, design, architechture & usage.

Cloudonomics: meaning the economics of cloud computing

- Core of cloudonomics, is the expression of when a migration can be economically feasible.

- post migration, the ROI on the migration should be positive.

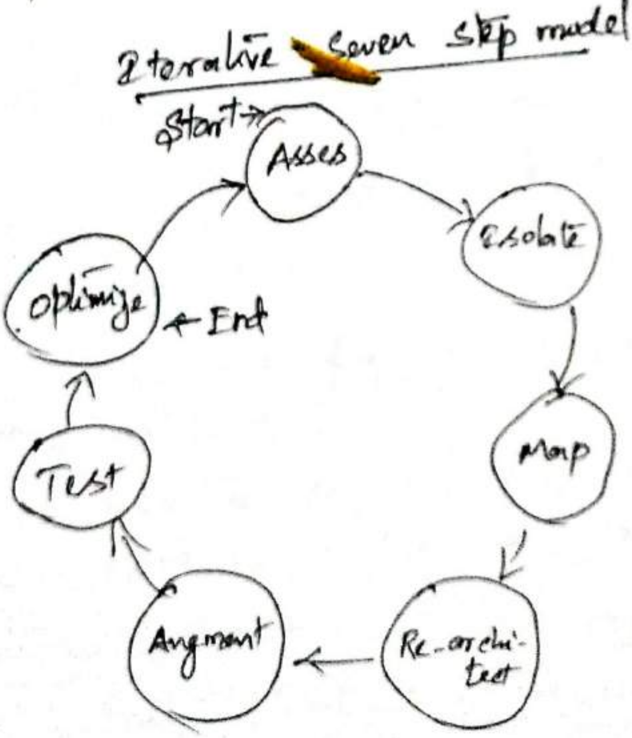
The seven step model of Migration into a cloud

- Migration initiatives into the cloud can be done in phases or stages.

seven step model (Infosys Research)

1. Conduct cloud migration assessments
2. Isolate the dependencies
3. Map the messaging & Environment
4. Re-architect & Implement the local functionalities
5. Leverage cloud functionalities & features
6. Test the migration
7. Iterate and optimize.

Iterative seven step model (Infosys)



1. Assesments Comprise :

* understand the issues at application level or code or design, the architechture or usage leve
* Tools being used, test cases, functionalities and Non Functional Requirement's of the company application.

-> The above gives a meaningful comprehensive migration strategy.

-> prototype for various approaches with pricing parameters.

* In production version "assessment is the cost of migration and also ROI to be achieved".

2. Isolate the dependencies :
systemic & environmental dependencies within the captive data center.
(eg) Run time environment, Licencing, libraries, Applications. Architechture, Latencies + performance bottlenecks.

3. Map:
It comes after step 2. (ie) What shall be remains in the local captive data center and what goes onto the cloud.

4. Re-architect:
Due to mapping substantial part of enterprise application to be re-architechted, redesigned and reimplemented on the cloud.

5. Augment -> *(original functionality of an enterprise)
* Exploiting additional features

6. Test - Validating the new form of enterprise.
* Results should be positive or mixed

7. optimization : * Meet the users requirement or not. If not iterate the process.
Later migration is successful.

* Amazon AWS uses the following steps for Migration:

First phase: Assessment

2nd phase: build reference migration architecture

3rd phase: Data migration

4th phase: application migration

5th phase: leveraging amazon AWS features (eg. elasticity, autoscaling, cloud storage etc)

6th phase: optimization.

Migration Risks and mitigation:

* In the seven step model, Testing includes efforts to identify the key migration risks.

* In optimization step, steps to midigate the identified risks.

Two types of risks here are:

a) migration risks

* performance monitoring & tuning
* business continuity & disaster recovery
* Compliance with standards
* IP & licensing issues
* QoS & SLA's committed
* ownership, transfer, storage of data
* portability & interoperability issues

b) Security related migration risks includes

* issue of trust, issues of privacy inaddition, security at various levels of enterprise application

Unit II

The evolution of Saas:

* IT as a Service (ITaaS) - is recent and most efficient delivery method.
Every single IT resource, activity & infrastructure is being viewed & visualized as a service.

* Integration as a service (IaaS) - is budding
- Fullfills business integration requirements.
- Can get business, technical benefits

Question here is: How to get seamless connectivity between those hosted and on-premise applications

- Here, for the above IaaS uses B_2B integration technology between SaaS solutions & in-house business applications.
- B_2B has provisions to encrypt files for safe ~~package~~ passage accross public n/w,
  - manage large data volumes
  - Transfer batch files
  - convert disparate file formats
  - guranted data delivery accross multiple enterprises.

# Hub(H) is installed at SaaS providers cloud center (for lifting)

#A spocke(S) unit at each user site, acts as a data transfer utility.

# With H&S SaaS providers offers integration services "clouds, being the web based infrastructure"

∴ Web is:

* largest digital information super highway
* largest repository of all kinds of resource.

* open, cost-effective & generic business execution platform.
* global-scale communication infrastructure
* next generation discovery, connectivity and integration middleware.

—*—

The challenges of SaaS Paradigm

The following ~~(deters (growing))~~ are necessary for massive adoption of cloud:

* Controlability
* Visibility & Flexibility
* Security and privacy
* High performance & availability
* Integration and Composition
* Standards.

Challenges:

Integration Conundrum (4Q's):
majority of SaaS applications are point solutions and service one line of business. So, company may not synchronise multiple line of businesses. Therefore can't get accurate data, forecast etc..

API's are insufficient:
accessing via API, requires, a significant amount of Coding. [For modification & updates]. So a unique method is needed for API of each SaaS applications.

Data Transmission Security
Security needed while transferring data from on-premise systems to client data centres.

—*—.

Approaching The SaaS Integration Enigma

* Integration as a Service (IaaS) – means migration of functions of Enterprise Application Integration (EAI) hub / Enterprise Service Bus (ESB) into cloud. for smooth data transport between enterprise and SaaS application.

∴ cloud middleware is needed. (It should be made available as a service) eg. EAI, ESB etc..

For service integration – ESB (Enterprise Service Bus)

" data " – EDB (Enterprise Data Bus)

Message oriented Middleware - MOM

* Events are comming fast, there are Complex Event Processing engine (CEP) that ~~received~~ receives stream of events from diverse sources, process them, accordingly select and activate one or more target applications. Therefore a lighter connectivity occurs between initiating and destination applications.

* Service integration through ESB is loosely coupled. systems, whereas, CEP connects decoupled systems [changes made in one system without having an effect on any other system

(eg) Amazon's Simple Queue Service (SQS).

Why integration is hard?

The probable reasons are:

* New Integration Scenario
* Access to the cloud may be limited
* Dynamic resources
* Performance.

New Integration Scenarios

Three major integration scenarios are:

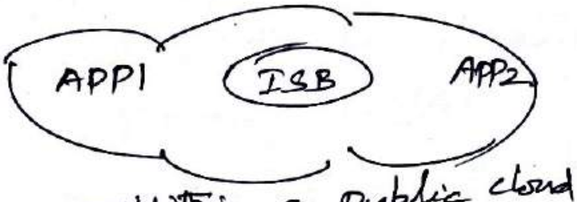
1) Within a public cloud
2) Homogeneous clouds
3) Hetrogeneous clouds.

1. Within a public cloud:

* Two different applications are hosted in cloud. The role of cloud integration middleware (ESB Enterprise Service Bus or ISB Internet Service Bus) is to seamlessly enable these applications to talk to each other.

* The two applications may be owned by two different Companies.

* They may be within a single physical server but run on different Virtual Machines (VM)

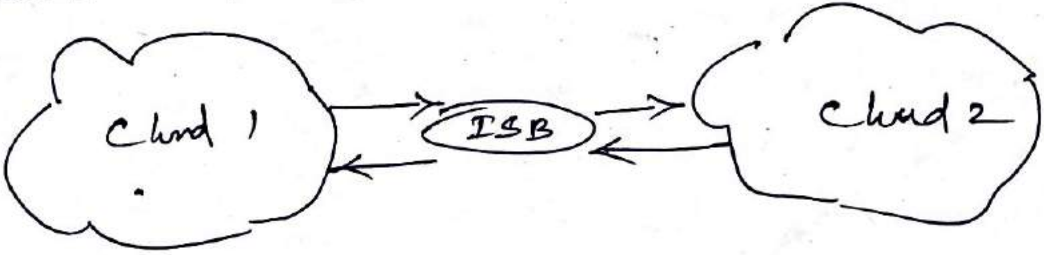


— Within a public cloud —

2. Homogeneous clouds

* The applications to be integrated are posted in two different cloud infrastructure.

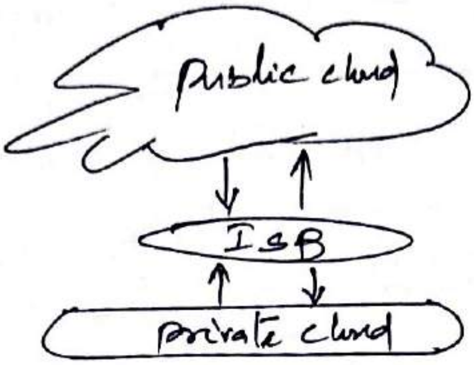
Therefore the intergration middleware can be in cloud 1 or cloud 2 or in a separate cloud.



3. Hetrogeneous clouds

* one application is in public cloud and another

in a private cloud.
(i.e) business enterprises are subscribing to popular on-demand enterprising package. (From salesforce.com, Ramco systems, Netsuites etc).



— x —

The Integration Methodologies

* There are 3 types of cloud integration [excludes custom integration through hand-coding]

① Traditional Enterprise Integration Tools can be empowered with special connectors to access cloud-located applications

* mostly used by IT organisations, (because they have invested a lot for integration suites for their applications integration needs).
* Based on necessity, special drivers, connectors and adapters are being built and incorporated.
* We have integration platform/methods like

EAI/ESB

② Traditional Enterprise Integration Tools are hosted in the cloud:

- Similar to the above one. except the integration s/w is now hosted in any 3rd ~~party~~ party cloud, so enterprise may not worry about procuring and managing h/w or ~~s/w~~ installing integration software.

* This is good fit for IT organisations.

③ Integration-as-a Service (IaaS) or

on demand integration offerings

* These are SaaS applications, designed to deliver the integration service securely over Internet

* Able to integrate cloud applications with on-premise systems, cloud-to-cloud applications.

* On-premise systems can be integrated with other on-premise systems or applications via this int service.

This is good for the companies ~~who~~ who insist:

- # ease of use
- # ease of maintenance
- # time to deployment
- # on a tight budget.

SaaS Integration product & platform

Jitterbit

* Fully graphical integration solution, provides users a versatile platform to reduce the integration efforts.

* Can be used standalone or with existing EAI (users can create a new project or modify the existing)

* gives coll integration among confidential and corporate data.

* Gives powerful graphical environment to design, implement, test, deploy and manage the integration projects quickly.

Two major components:

1. JitterbitIntegration Environment.

using point-&-click GUI can configure, Test, deploy, manage integration projects on jitter server.

2. Jitterbit Integration Server

Power ful & scalable runtime engine, that processes all integration operations.

* Jitterbit is making integration easier, faster & more affordable than ever before.

Boomi S/w:

Fullfils the vision "Integration on Demand"

∴ Its an on-demand-service.

It can connect any combination of SaaS, PaaS, cloud, & on-premise applications.

* It offer solution that enables to quickly develop and deploy connections between applications, regardless of delivery models.

Bungee Connect

* For professional developers
* used to integrate applications from multiple data sources & facilitating instant deployment.

OpSource Connect.

* Expands on OpSource Service Bus (OSB)
* provides infrastructure for 2-way web service interaction
* Allows consumers to consume & publish applications accross a common web service infrastructure.
* It unifies different SaaS applications & also legacy applications running behind firewall.
* provides platform for web services adoption & integration.

SnapLogic

* Free community edition is used for integration task
* Professional edition – upgraded version with price.

The Pervasive Data cloud:

* Its a unique multi-tenant platform.

It delivers:

1. IaaS for both hosted and on-premise applications & data source
2. Packaged Turnkey integration
3. Integration that supports every integration scenario
4. Connectivity to hundreds of different apps. and data sources.

Pervasive Datasynch : - family of packaged integratoo.
- subscription based.. uses turn key approach.
- used in popular applications like saleferce, Quickbooks + Microsoft dynamics.

Pervasive data integrater :
- runs on cloud or on-poremise
- It's a design once and deploy anywhere solution to any integration ~~solution~~ scenario.

* provides multi-tenant, multi-application and multi-customer deployment.

Blue wolf :
- It's used to integrate diverse saas solutions.
- remote monitoring of integration jobs.
- proactively alerts customer, if any issues with integration, and solves quickly.

* For administrators: Enable administrator to manage data flow between front & back office systems with little or no It support.

On-line MQ :
* It's a internet based queuing system.
* It's a complete secure on-line messaging solution.

Advantages
# ease of use # No maintenance # easy integration
# load balancing and high availability.

Cloud MQ : like online MQ.

Linxter:
* It's a cloud messaging framework for connecting all kinds of applications, devices and systems.

SaaS Integration Services

1) Informatica on - Demand

- provides informatica On-Demand services.

* Multi-tenant architechture

- used to integrate data in SaaS applications accross internet with data in on-premise applications

* Its subscription based.

* If any features developed, it is made available to its users.

2) Microsoft Internet Service Bus (ISB)

Azure - cloud OS from microsoft.

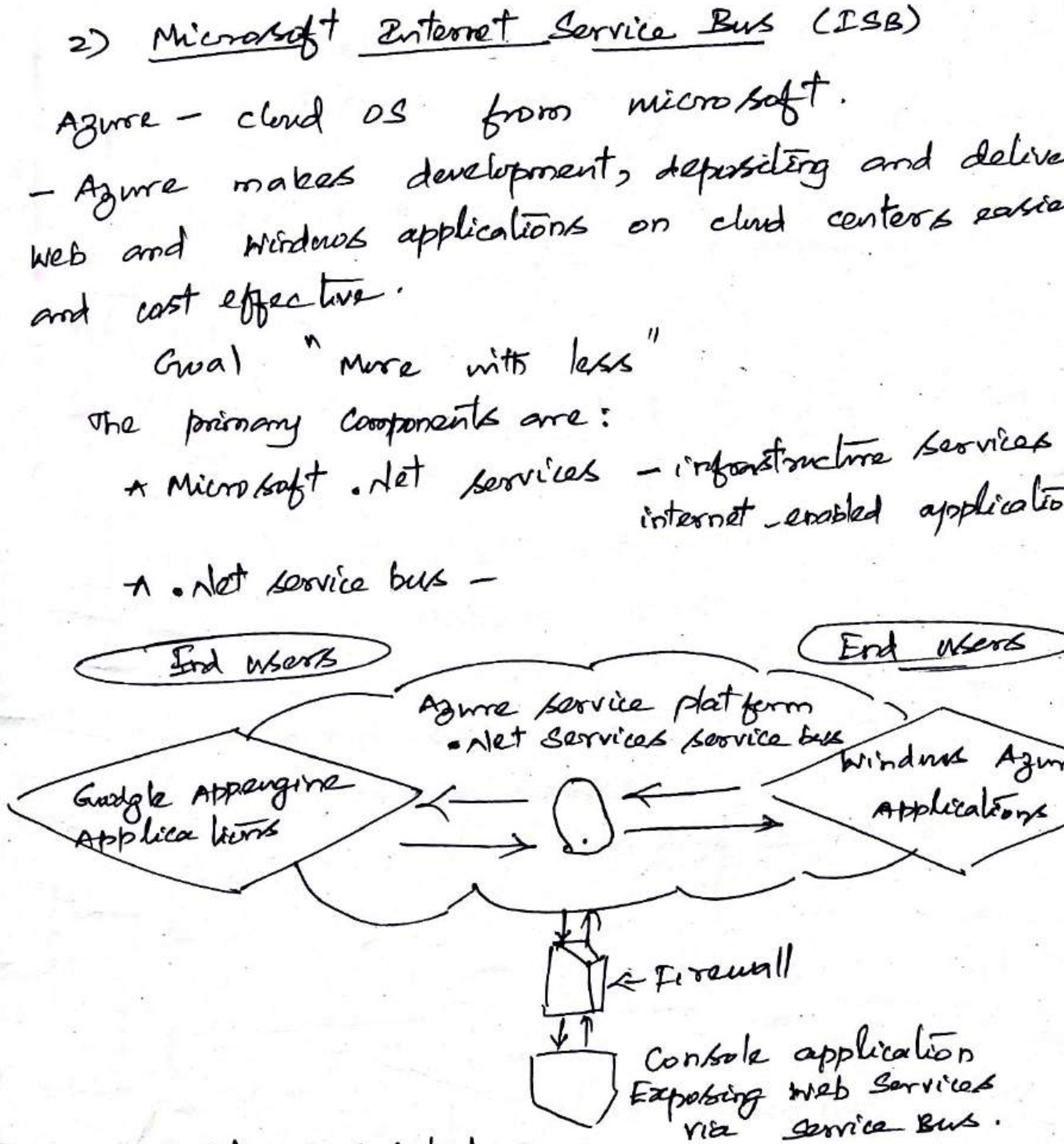
- Azure makes development, depositing and delivering web and windows applications on cloud centers easier and cost effective.

Goal "More with less"

The primary components are:

* Microsoft .Net services - infrastructure services for internet enabled applications.

* .Net service bus -


End users
End users
Azure service platform
.Net Services service bus
Google Appengine Applications
Windows Azure Applications
Firewall
Console application Exposing web Services via service Bus.

* .Net access Control service

* .Net workflow service

Business-To-Business Integration (B2Bi) services.

* used to connect geographically distributed businesses, product vendors must have a competent B2B hubs.

* It is good for IaaS.

* provide A2A Connectivity. (Application to Application) (10) secure data exchange accross the corporate firewall

EAI – for internal data sharing.

* Ability to encrypt file for ~~safe~~ safe passage in public n/w, manage large data, transfer batch files, Convert disparate file formats and guarantee data accuracy, integrity, Confidentiality, &delivery.

* provides adapter libraries for rapid integration with various business systems.

— * —

The Enterprise cloud Computing Paradigm

Background :

According to NIST, cloud computing is composed of 5 essential characteristics:

* on-demand self service
* broad network access
* resource pooling
* rapid elasticity
* Measured service.

* The above may vary according to the deployment model used.

Relevant deployment models for Enterprise cloud Computing:

Models accepted by cloud stake holders are:

i. public cloud – for general public, under utility based pay-per-use model.

ii. private cloud – built & operated by organizations for its internal use.
virtual private cloud.

iii Community clouds - Shared by several organisations and support by a specific community
eg. OpenCirrus - formed by HP, Intel, Yahoo & others

iv. Managed clouds - physical infrastructure ~~and~~ is owned by the company with management and security plane controlled by managed service provider.

v. Hybrid clouds - Combination of 2 or more clouds (private, community, public)

* Selection of a Deployment model depends on:
  * opportunity to increase earning & reduce cost. (ie) capital expenses (CAPEX) & operating expenses (OPEX)

Adoption ~~&~~ and Consumption Strategies

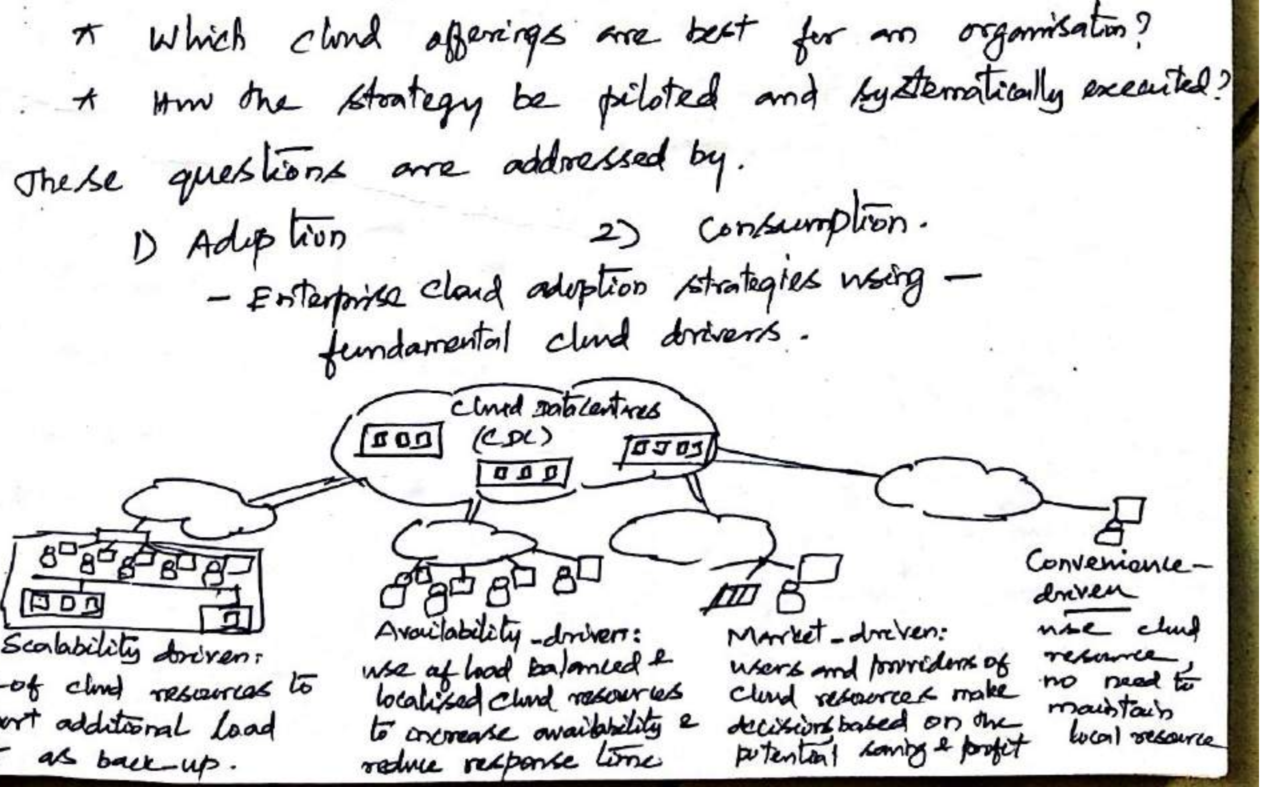
Before adoption we need answers for the following question:

* Will enterprise cloud strategy ~~will~~ increase overall business value?
* efforts & risks while transitioning is worth?
* which area of business be considered for enterprise cloud?
* Which cloud offerings are best for an organisation?
* How the strategy be piloted and systematically executed?

These questions are addressed by.

1) Adoption 2) Consumption.

- Enterprise cloud adoption strategies using - fundamental cloud drivers.


cloud datacentres
(CDC)
Scalability driven:
use of cloud resources to support additional load or as back-up.
Availability-drivers:
use of load balanced & localised cloud resources to increase availability & reduce response time
Market-driven:
users and providers of cloud resources make decisions based on the potential saving & profit
Convenience-driven
use cloud resource, no need to maintain local resource

1) Scalability driven strategy: objective is to support the increasing work loads of the organisations without investments and expenses exceeding returns.

2) Availability driven strategy: have has close relation with the above with the assurance that IT capabilities and functions are accessible, usable and acceptable by the standards of user.

3) Market driven strategy. useful for small, agile organisations that do not have massive investment on their IT infrastructure. Objective is identify and acquire IT capabilites as demand & supply change.

4) Convenience-Driven strategy: objective is to reduce the load and need for dedicated system admin and to make access on IT Capabilities by user easier, regardless of their location and Connectivity.

Cloud Consumption strategies:

4 strategies are:

1) Software provision: Its relevant when:

* the elasticity requirement is high for S/w + low for data
* Controlability concerns are low for S/w & high for data
* Cost reduction concerns are high for S/w

Implementing to the organisation, requesting, software to be delivered as a service (SaaS) by CDC or to access some portion of CDC's infrastructure as a service (IaaS).

2) Storage provision: Its relevant when:

* the elasticity requirement is high for data + low for S/w
* the controlability concerns is low for data + high for S/w

Adv: ease of sharing data between organisations, availability management of storage utilisation

3) Solution provision: It's relevant when:

* elasticity and cost reduction requirement are high for S/W and data

* but controlability requirements can be entrusted to the CDC organisation trust the CDC, sufficiently, to manage access + usage control of its s/w & data.

4) Redundancy services:

* This is hybrid enterprise cloud strategy, where the organisation switches between traditional, s/w, storage or solution management based on changes in its operational conditions and business demands.

It has the name "Redundancy strategy" because the CDC is used for situations such as disaster recovery, fail-over and load balancing.

objective -> business continuity.

Issues for Enterprise Applications on the cloud

* most comprehensive enterprise application today is ERP.

ERP -> Is to equip enterprise with a tool to optimize their business process with a seamless, integrated information flow from suppliers through to manufacturing and distribution, and, ability to plan effectively and control all resources. ∴ ERP is backbone of any organisation.

* Due to market changes, organisation needs new solutions to compete ∴ Enterprise cloud computing platform is becoming attractive as a potential ERP environment.

* Such transition will require a balance of strategic and operational steps guided by socio-technical considerations, continuous evaluation and tracking mechanisms.

* First one of the issue is that of infrastructural availability.

OLTP – On Line Transaction Processing

OLTP type ERP components are:

* Sales & Distribution (SD)
* banking and Financials
* Customer Relationship Management (CRM)
* & * Supply chain Management (SCM)

* These applications face major technical and Non-Technical challenges to deploy in cloud environments

OLAP -> On Line Analytical processing

OLAP applications are:

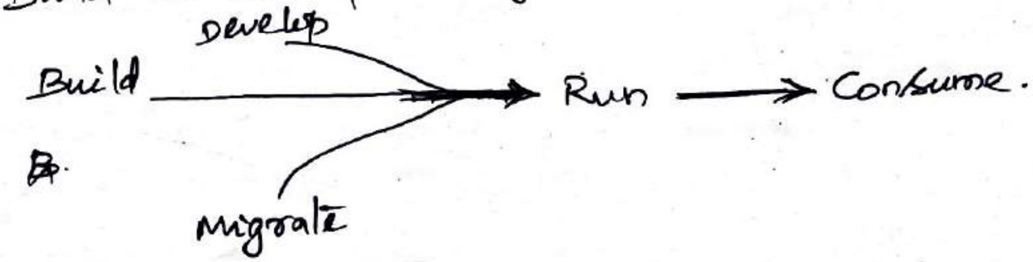
* Business reporting * Marketing } called Business Intelligence (BI)
* Budgetting & * forecasting. }

* The above are read-only or read-most.
* This type of applications benefit greatly in cloud.

— ∝ —

Transition challenges:

5 catigories are:
1) Build 2) Develop 3) migrate 4) Run 5) Consume.



* A the moment private and hybrid models are relevant for ERP transitioning.

Immediate challenge: What is the already available IT assets of org? and can or can't be sublimed. ([illegible])

* Unplanned cloud spreads throughout the organisation, coherency becomes a challenge. (sticking together)

* Second challenge is migration of existing to the cloud.

~~When~~ ~~*~~ During migration to cloud, the processes to be decoupled.

challenges for cloud operation:

(1) running enterprise cloud

* difficulties there in terms of IT operations on their day-to-day operations.
* needs upgrading & updating IT depts components

Very hard to upgrade are: human,

2) running application on enterprise cloud.

— ∝ —

Unit – III

The Anatomy of cloud Infrastructure

IaaS providers ~~have~~ share 5 characteristics:

1. on demand provisioning of Computational resource
2. use virtualization technologies to lease these resources
3. provides public and simple remote interfaces to lease these resources
4. use pay-as-you-go cost model.
5. operates data centres to provide unlimited resource to their clients.

VI requires fair configuration, including s/w & h/w. (n/w envis)

VIM - used to allocate the resources.

(eg) VI Management solutions

ISF, VMware ~~Vs~~ vSphere.

open source :-> Enomoly Computing platform, OVirt.

Distributed Management of VM

* Setup a custom s/w environment for VMs:
  * setting up & managing n/wing for interrelated VM's
  # reducing overheads involved in using VM's
  - the above tasks are handled by VI Manager.

* VI Manager has to support flexible & Complex scheduling policies and also ability of VMs to suspend, resume & migrate.

Reservation-based provisioning of virtualised resources

* Requesting the resources exactly when needed.

(ie) A particular persons need it at 2 to 4 pm it is reserved and provisioned on that time. (ie) resource available at 2pm.

Provisioning to meet SLA commitments

IaaS clouds can be used to deploy services that will be consumed by users other than the one that deployed the services.

(eg) a company depends on IaaS provider to deploy 3 tier applications :. web front-end, application server, database server., for its customers.

Here, there is a difference between

cloud consumers & End users of resource
((ie) service owner) (the users that access the application)

∴ Service owner will put SLA with end users.
(which cover timelines with which these services will respond)

—1—.

Distributed Management of Virtual Infrastructures

Open Nebula – opensource VI engine. or VI Manager.

* Able to manage group of interconnected VM's
* supports Xen, KVM & VMW platforms.
* used to built hybrid clouds

It's manages VM's in VI.

VM Model and life cycle:

primary target of openNebula is to manage VM's.

* VM modeled in openNebula have the attributes:
  - # capacity. (Memory & cpu)
  - # set of NICs attached to one or more VI
  - # A set of disk images
  - # A state file or recovery file

Life cycle of VM

the stages are:

* Resource selection
* Resource preparation
* vm creation
* vm migration
* vm Termination

VM management.

Three different Management areas are,

1) Virtualisation by interfacing with physical resource's hypervisors, such as Xen, KVM or VMWare to control VM

2) Image management by transfering the VM images from an image repository to the selected resource and by creating on-the-fly temporary images.

3) Networking by using LAN.

Scheduling Techniques for Advance reservation of capacity.

VI managers like openNebula can hadle in all the time, of managing VM's in a pool of physical resources.

Existing approaches to capacity reservation

In job scheduling, Efficient reservation of resources in resource mgmt system, is important.

* PBS-pro creates a new queue that will be band to the reserved resources, (i.e) job submitted in the queue will be executed on them.

* Maui and Moab - allows users to specify that a submitted job should ~~be~~ use the reserved resource

There are no mechanism to directly login to reserved resources. [other than through an interactive job]

* Advance reservation leads to utilisation problem.

* Traditional schedulers : unable to efficiently schedule workloads

* preemption is used for large parallel jobs. (shares)

Chek ~~poit~~ pointing is used in preemption, preemted jobs entire state is saved in disk, allowing to resume its work from the last checkpoint.

Reservations with VMs

overheads of using VM:

- * preparation overhead - ~~when~~ using VMs to implement reservation
- * Run time overhead - VM's entire memory space is saved to disk, and then read from disk

Leasing model

Lease means - " a negotiated and renegotiable agreement between a resource provider and a resource consumer, based on a set of lease terms presented by the resource consumer".

Terms encompasses:

H/w resources - required by consumer such as CPU, Memory, N/w bandwidth.

S/w environment - required on leased resource

Availability period - during which user requests that the H/w + S/w resources be ~~availab~~ available

Availability in terms:

* Start time - specified or unspecified
* Maximum duration - maximum time availability

* Leases can be preemtable -

—α—

Lease Scheduling:

Best-effort leases - are scheduled using queues.

Advance reservation - No queue is used. Start at the request time

— α — α —

RVWS Design: (Resources Via Web Services)

dynamic attribute exposure:

Two attributes of RVWS are State & characteristic.

* State - Covers current activity of service & its resources thus indicating readiness.

(eg) web. service that exposes a cluster, would have dynamic state attribute, that indicates hwo many nodes in cluster are busy and idle.

* Characteristic - Covers operational features of the service, resources behind it, QoS, price & provider information.

(eg) Web service - characteristic is Array of support s/w within the cluster.

Stateful WSDL Document creation:

* When exposing dynamic attributes of resources, the RVWS framework allows web services to expose through the WSDL. document of web service.

* WSDL governs a schema that describes a web service & document written in the schema.

* stateful WSDL document - refers to WSDL document created by RVWS web services. called Resources.

* For each Resource behind web service, a ResourceInfo section exists

ResourceInfo has:

* Resource-id attribute &

* Two child sections (state & characteristics)

Publication in RVWS:

* A dynamic broker was proposed to ease publication

Goal of Dynamic broker - provide an effective publication and discovery service based on service, resource and provider dynamic attributes.

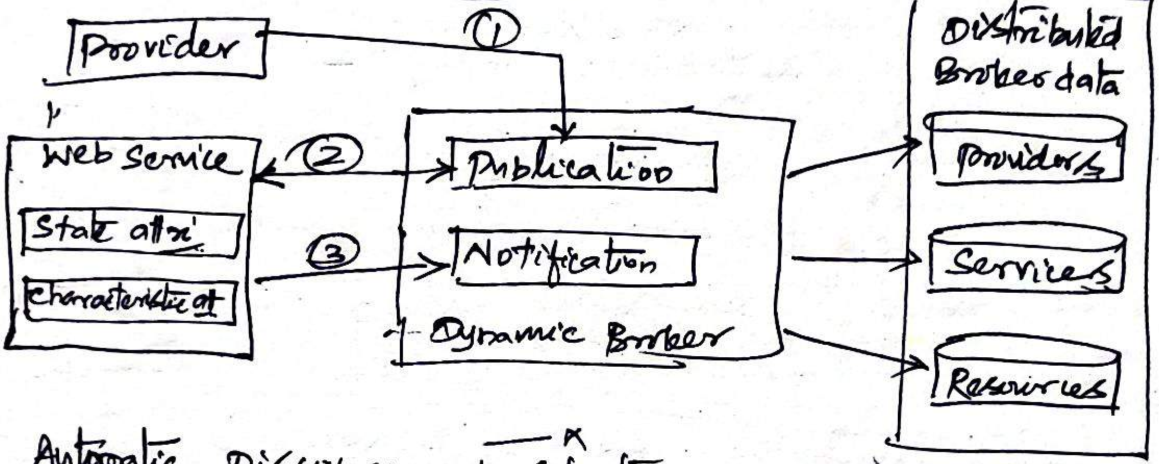
*① When publishing to the broker, provider sends attributes of the web service.

* After publication ①, broker gets the stateful WSDL document from the web services ②

* The Dynamic broker then stores the (stateless) WSDL document and service attributes from ① in service store

* As the webservice changes, notification is send to the Broker ③, which then updates the relevant attribute in the relevant store.

Publication



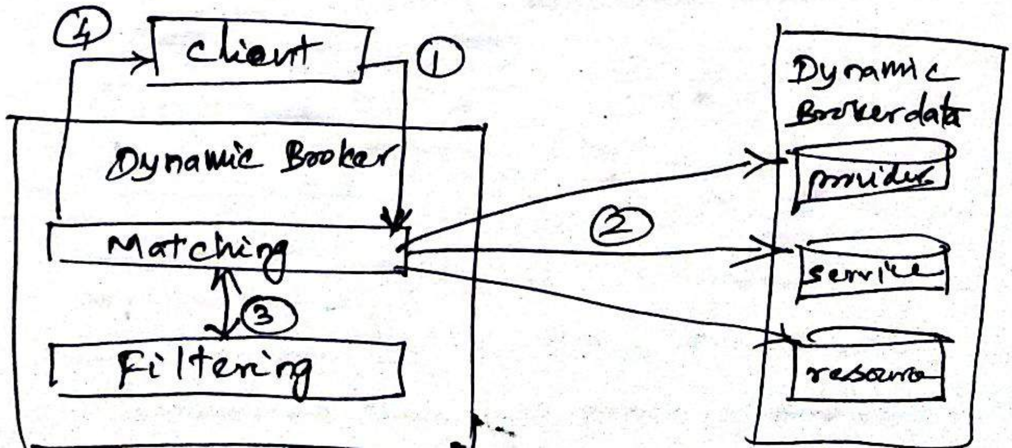
Automatic Discovery and selection :

* When discovering service, client submits 3 groups of requirements to dynamic broker ①. [Services resource and provider)

* Dynamic broker compares each requirement group on the related data store ②

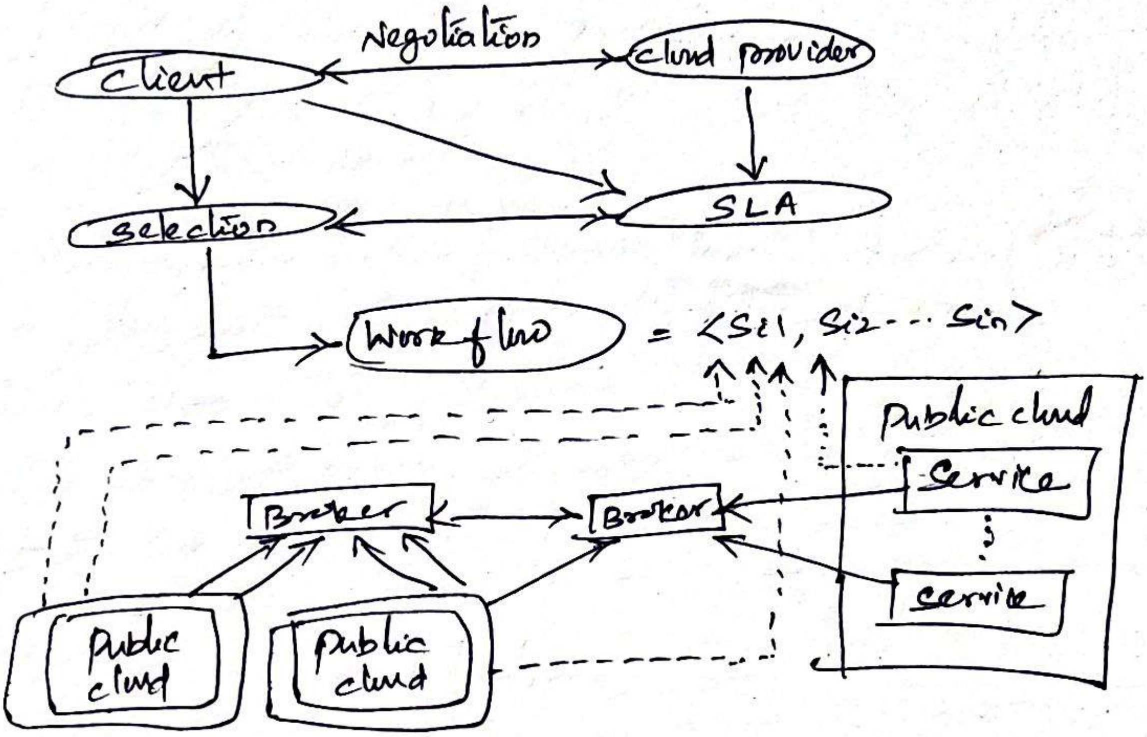
* After getting matches, Broker applies filtering ③

* Finally, filtered results returned to the clients



— Matching Parameters — to attributes
i) →
ii) →

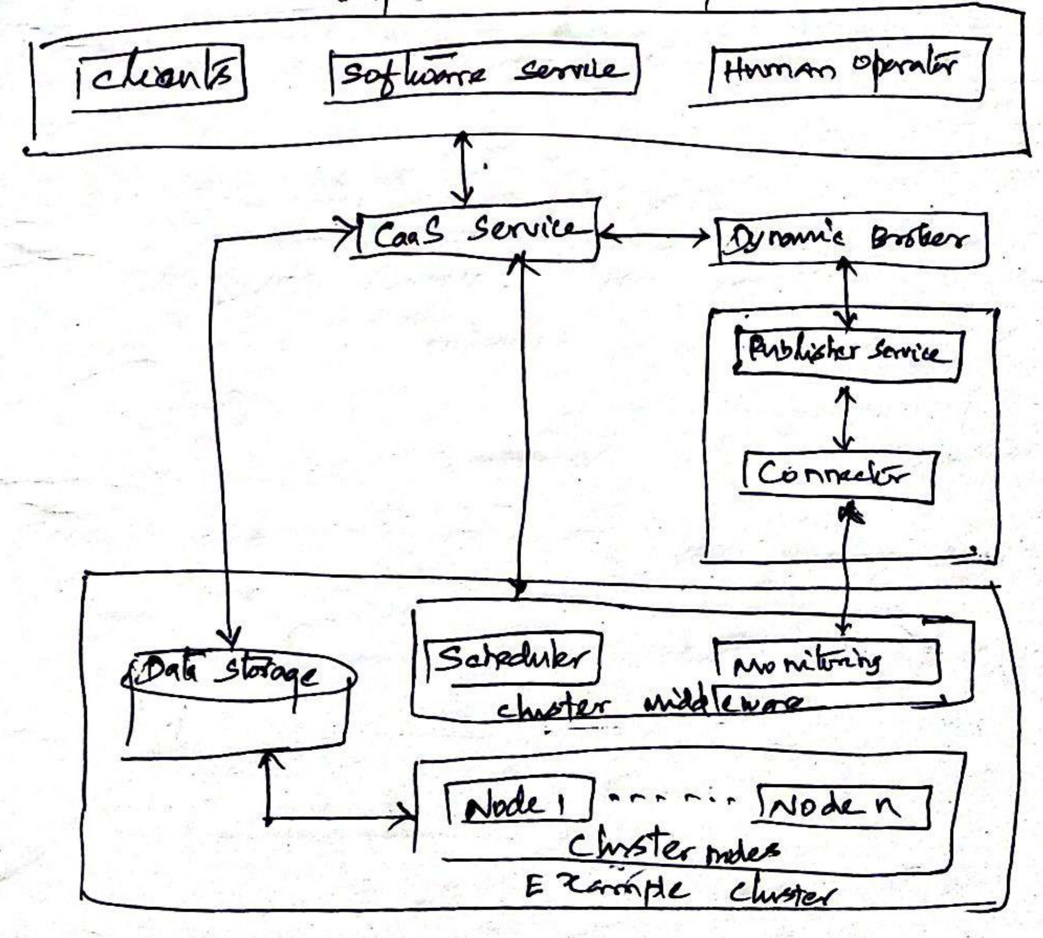
— Dynamic discovery and selection —



Cluster as a Service: The logical design

CaaS overview

— Complete CaaS system —



* A typical cluster consists of 3 elements:

1) Nodes 2) Data storage 3) Middleware

* Middleware virtualizes the clusters into a single system image.

As the time progresses, the amount of free space (ie) memory, disk space and CPU usage of each cluster node changes. (it) depends on the Scheduler.

* RVWS framework is used publishes the cluster information. then, it is published to ~~client~~ dynamic broker, so clients can easily discover clusters.

* To find cluster CaaS Service makes use of dynamic broker.

7.4.2 cluster stateful WSDL Document

* Inside the state and characteristic elements, an XML element for each cluster node was created.

* XML is used to Compare client requirements to resource attributes.

state

Free disk - amount of free disk space

Free memory - " " " memory

OS name - Name of installed OS

OS version - Version " " "

proless-count - Number of processes

proless-running - " " " running

cpu usage-percent - overall % cpu used

Memory free-percent - Amount of free memory on the cluster node.

—

Core-count - No of cores on the cluster Node

Core-speed - speed of each core

Core-speed-unit - Unit of core speed (GB or TB)

h/w architechture - h/w architechture of cluster node (processor)

total disk - total amount of physical storage space

total disk-unit - (eg) GB to TB

Total memory - Total amount of physical memory

Total memory-unit - (eg) GB to TB

s/w name - Name of installed s/w

s/w version - Version " " "

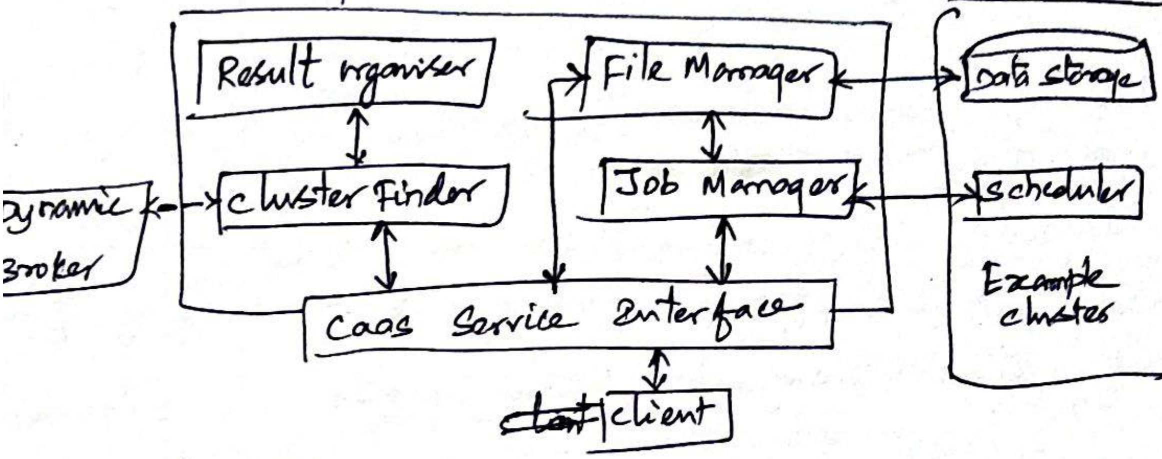
s/w architechture - architechture of " "

Node count - Total no of nodes in cluster (differs from core count)

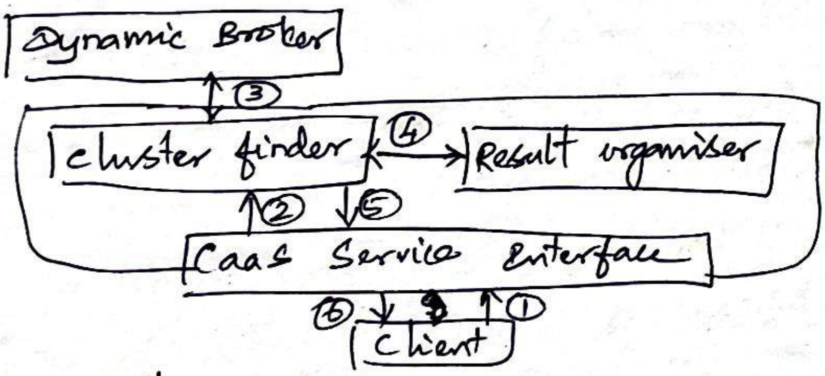
CaaS Service design:

It has 4 main tasks:

1) cluster discovery and selection
2) result organisation
3) Job management
4) File Management. — CaaS. Service Design —



1. cluster discovery

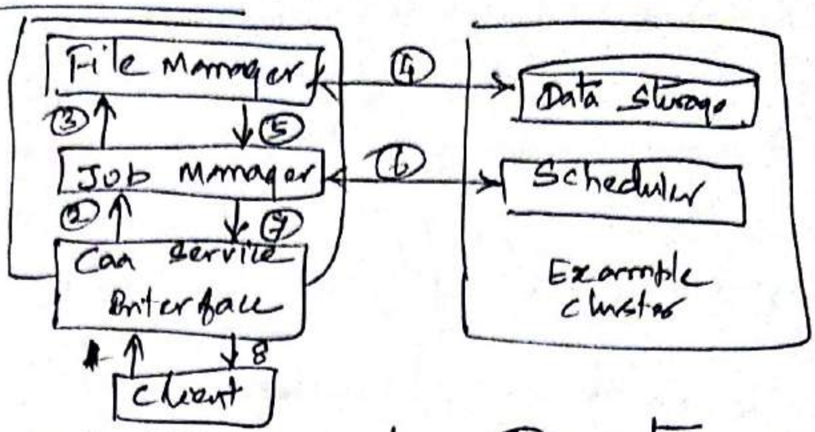


* cluster must be discovered before its usage.

* First, client make request to CaaS service interface in the form of attributes ①
* 2nd CaaS Service interface invoke cluster finder module ②, that communicates with dyamic broker ③ and returns service matches.
* 3rd - to address detailed results from broker, cluster finder invokes result organiser ④ and returns organised version, that is returned to the client ⑤ + ⑥.

The results shows or instruct the clients which cluster satisfies the requirements.

2. Job Submission.



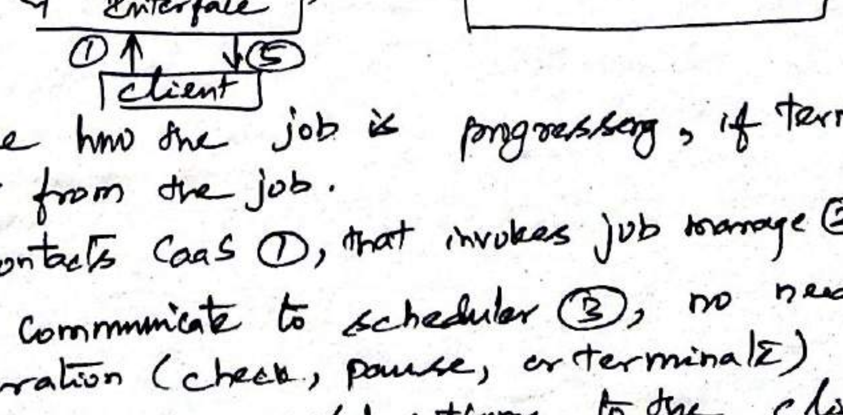
* Required data, paramaters ⓔⓖ run time, are uploaded to CaaS service ①. Upload complete, then job manage involved ②. It resolves transfer of all files to the cluster by invoking file manager ③, that makes connection with storage and commences the transfer of all file ④.

upon completion of transfer ④, outcome reported back to job manage ⑤. ~~I~~ on failure, report sent to client. If transfer successful, the job manager invokes scheduler on the cluster ⑥. The same parameters supplied to CaaS service, is submitted to scheduler If outcome of scheduler ⑥ is successful the client is informed. ⑦ & ⑧.

* Outcome :-> Response from scheduler, the job identifie (scheduler given to the job), any other info scheduler provi

———.

Job Monitoring



* client will see how the job is progressing, if terminate then remove it from the job.

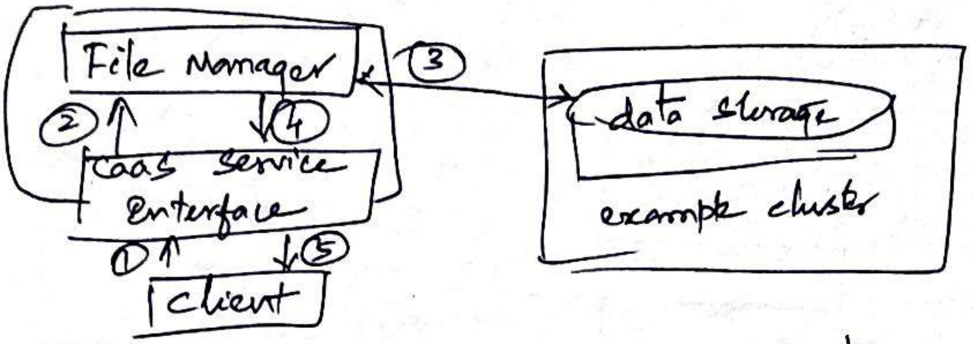
* First client contacts CaaS ①, that invokes job manage ②, m

* Job manager communicate to scheduler ③, no need to see the operation (check, pause, or terminate)

* Reports back, a successful outcome to the client ④ + ⑤

Result collection:

(e) Jobs that have terminated or Completed their execution successfully. In both errors or data files to be transfered to the client.

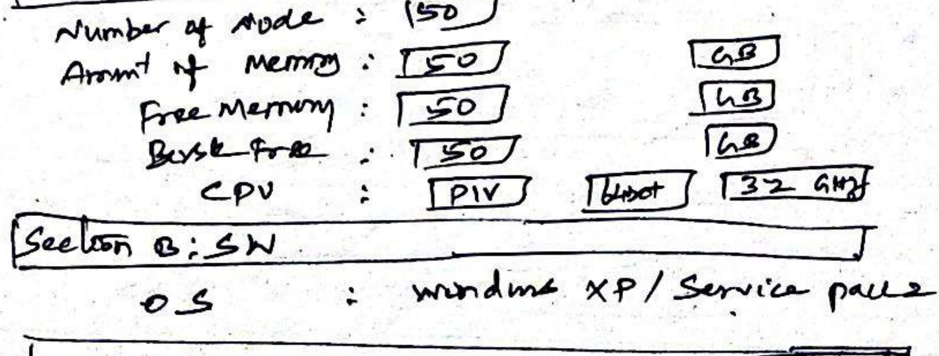


* client start the error or result file transfer by contacting CaaS ①, then it invokes file manager ② to retrieve files from storage ③.
* If there is a transfer error, file Manager attempts to resolve the issue before informing to client.
* If the transfer of file successful ③ files returned to CaaS ④ and then to client ⑤.

User Interface : CaaS web pages:

* CaaS service has to support S/W clients and human operator clients
* S/W clients have other S/W applications to communicate to CaaS.
* For human operator clients, a series of web pages has been designed.

web page for cluster specification is as follows:



Section A : H/W

Number of node : 150

Amount of Memory : 50 GB

Free Memory : 50 GB

Disk free : 50 GB

CPU : PIV 64bit 3.2 GHz

Section B : SW

OS : windows XP / Service pack 2

Disconnect

* From the above the client can discover the required clusters.

* client specify attributed about required cluster in section A

section B allows the required size the cluster job needs.

— web page for showing matching clusters —

	cluster A Select	cluster B Select
HW		
No of Node :	☑	
Amt of Mem :	☑	
Fre Mem :	☑	
DISK ~~Free~~:	☑	
CPU :	☑	☑
Archit. :	☑	
Sped :		☑
S/W		
OS	☑	
Archit	☑	
Version	☑	

— web page for job specification —

Section A : Identification

Job Name : Travelling sales man

Job owner: Joe Bloggs

Section B; Job File specification

Executable [My_exec-exe] [Browse..]

Script [My_script.pl] [Browse]

Data files [Custom-set.dat] [Browse]

[Add] [Remove] [Clear]

output filename [out.dat]

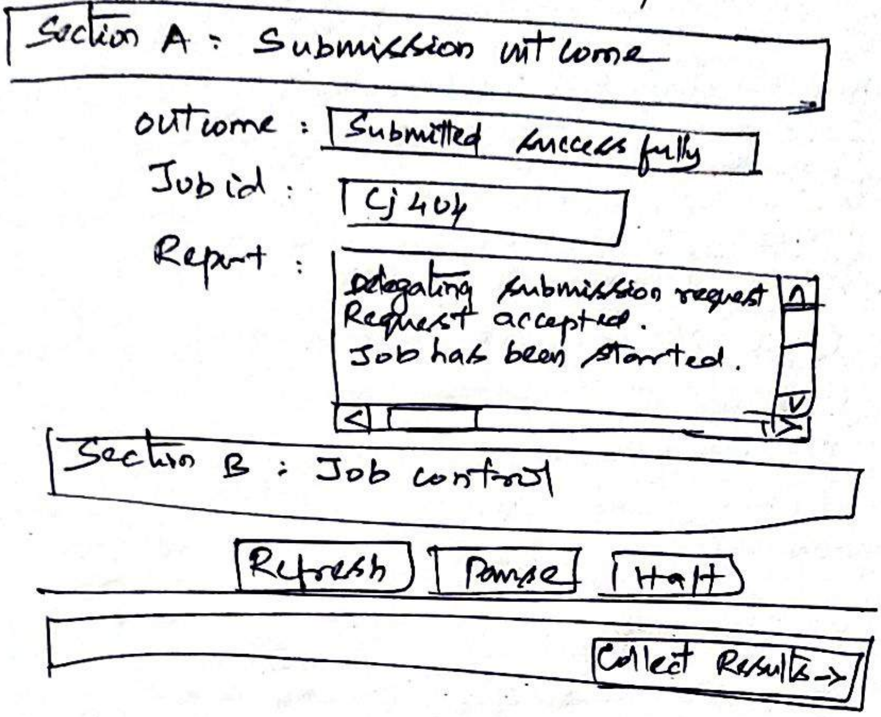
Section C Execution specification

Estimated Time [3 14 h]

[← change ~~customer~~ clusters]

[Submit →]

—Web Page for Monitoring Job execution—



Section A : Submission outcome

Outcome : Submitted successfully

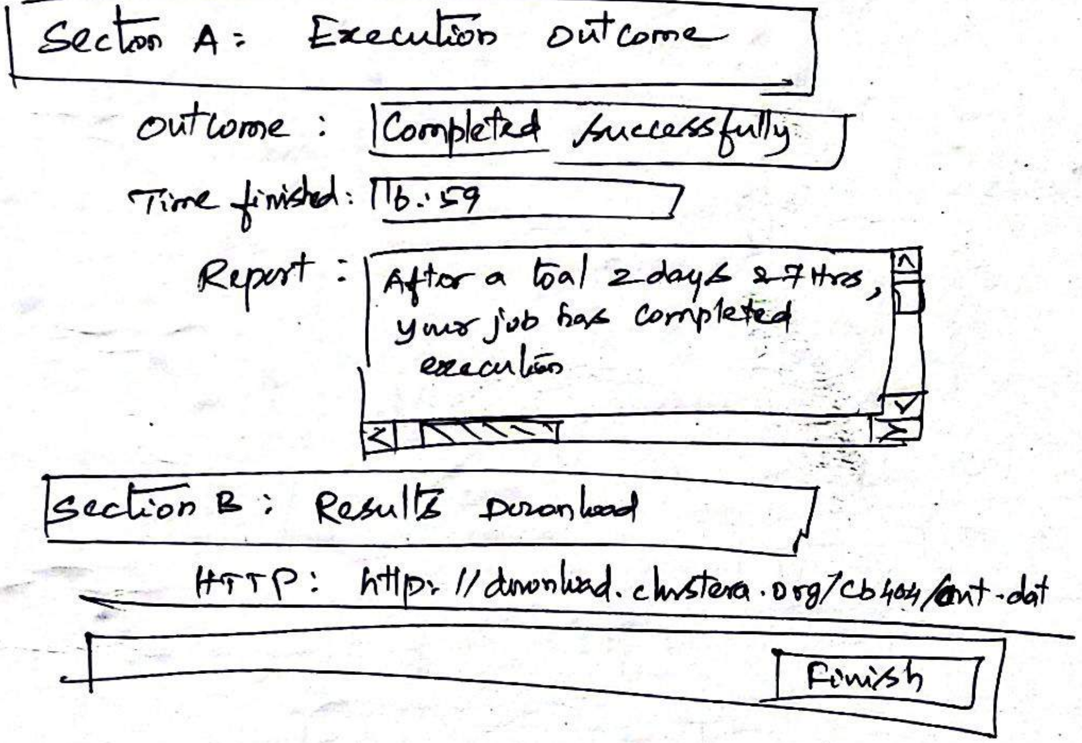
Job id : Cj 404

Report : Delegating submission request
Request accepted.
Job has been started.

Section B : Job control

— a —

—Web Page for collecting Result files—



Section A : Execution outcome

Outcome : Completed successfully

Time finished : 16:59

Report : After a toal 2 days 27 Hrs,
your job has completed
execution

Section B : Results Download

HTTP : http://download.clustera.org/cb404/out.dat

— a —

Unit IV & V

Workflow Management systems and clouds

Benefit of cloud: Application scalability.
(when needed scaling up additional resources and scaling down when demand is low)

* Scalability enables workflow mgmt systems to readily meet QoS requirement of applications.

* SLA is important.

SLA must offer ⓐ better QoS guarantees to consumer

ⓑ clear terms for compensation in the event of violation.

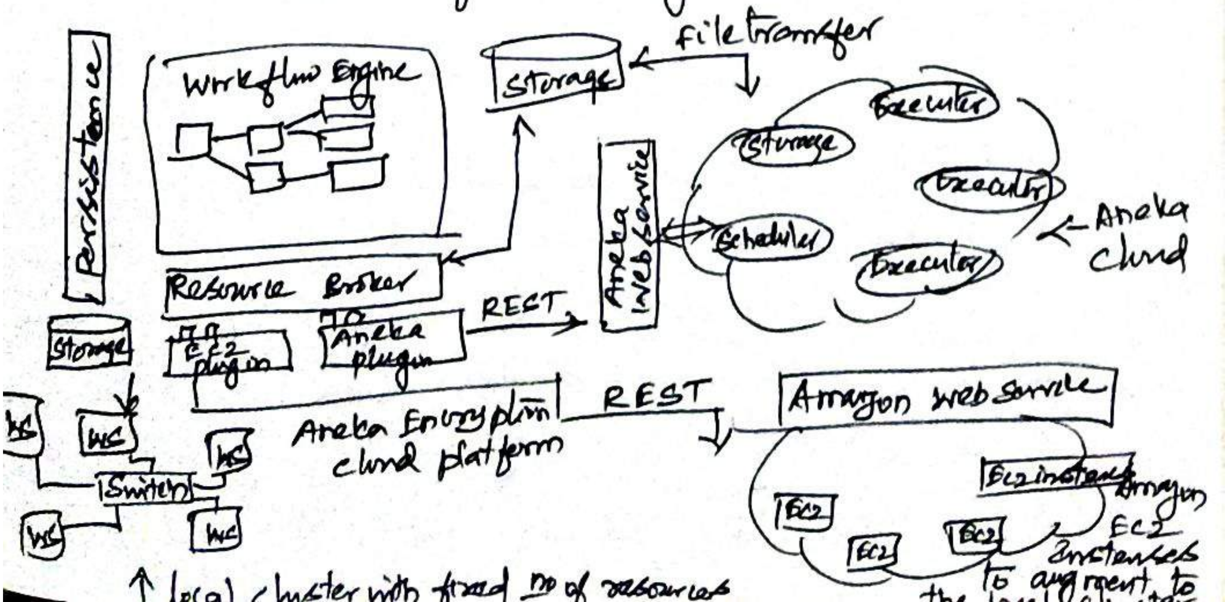
* This allows workflow mgmt syss. to provide better end-to-end guarantees.

* providers has to give storage, computation and bandwidth in lower costs. utilising public cloud service is economical. & cheaper.

* Benefits of using virtualised resources for workflow execution is using the technique sand boxing. [Reduced need from malicious code for physical resources].

WS → Work station

— Workflow Engine in the cloud —



12.3.1 Architectural Overview

Figure 12.1 presents a high-level architectural view of a Workflow Management System (WfMS) utilizing cloud resources to drive the execution of a scientific

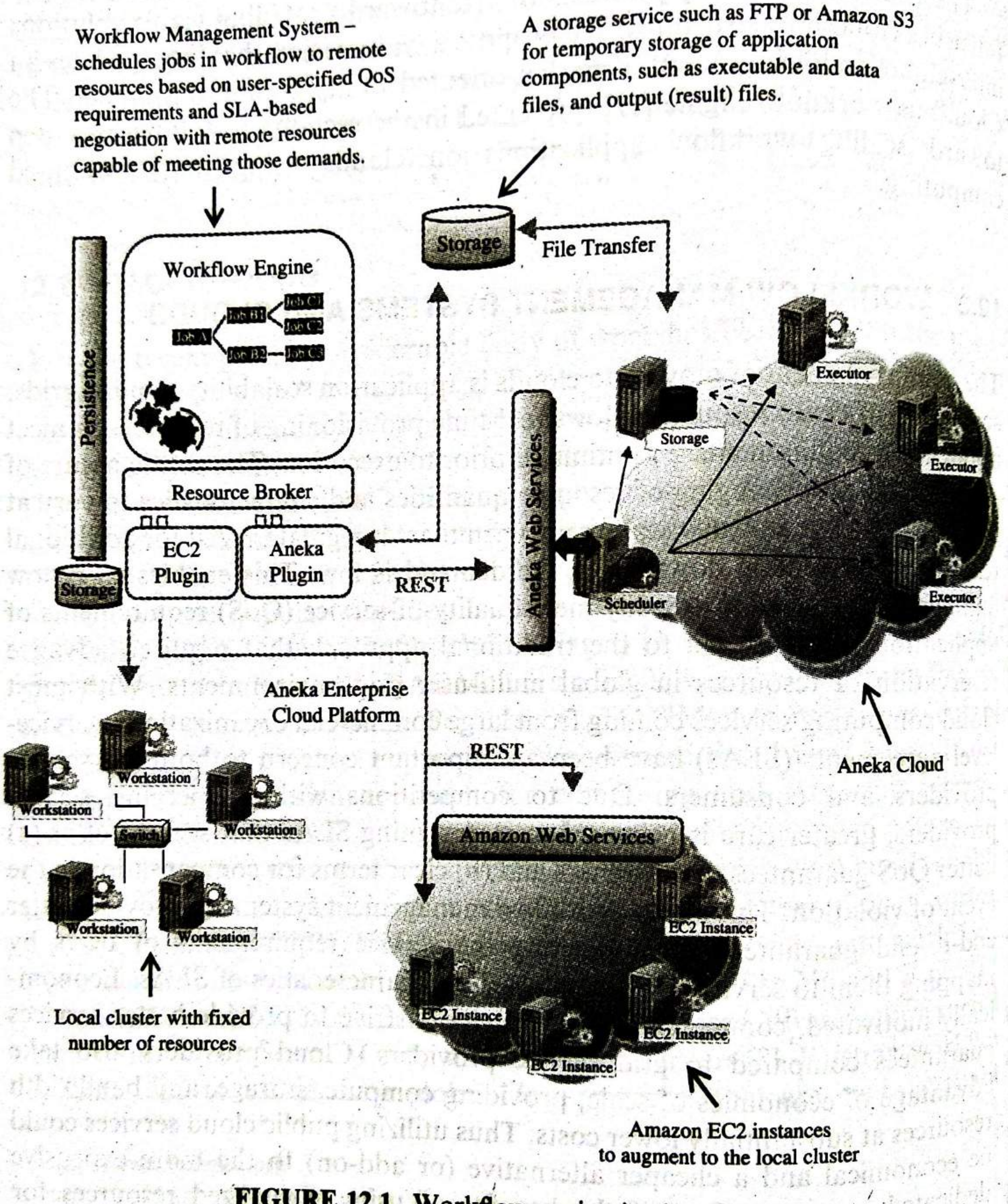

Workflow Management System – schedules jobs in workflow to remote resources based on user-specified QoS requirements and SLA-based negotiation with remote resources capable of meeting those demands.
A storage service such as FTP or Amazon S3 for temporary storage of application components, such as executable and data files, and output (result) files.
Storage
File Transfer
Workflow Engine
Job A
Job B1
Job B2
Job C1
Job C2
Job C3
Persistence
Resource Broker
EC2 Plugin
Aneka Plugin
Storage
REST
Aneka Web Services
Storage
Scheduler
Executor
Executor
Executor
Aneka Cloud
Aneka Enterprise Cloud Platform
REST
Workstation
Workstation
Workstation
Switch
Workstation
Workstation
Local cluster with fixed number of resources
Amazon Web Services
EC2 Instance
EC2 Instance
EC2 Instance
EC2 Instance
Amazon EC2 instances to augment to the local cluster

FIGURE 12.1. Workflow engine in the cloud.

workflow management system (WfMS) ~~is~~ as shown in the above diagram.

It Comprises of:

* Workflow engine, a resource broker, and plug-ins, such as Amazon & Aneka EC2.

In above diagram, where Aneka platform is used to complete the workflow. and Amazon EC2 is used to supplement a local cluster when there are insufficient resources to meet the QoS requirements of the application Aneka.

(i.e) Given limited resources in local networks, Aneka is capable of provisioning additional resources by acquiring new resources in 3rd party cloud service such as Amazon EC2.

* Aneka also provides a set of web services for service negotiation, job submission & job monitoring.

* Based on user-specified QoS requirements, the WfMS schedules workflow tasks to resources that are located at the local cluster and in the cloud.

Parameters for scheduling: (1) deadline (Time)
(2) budget (cost)

—— α ——.

Architechture of WfMS:

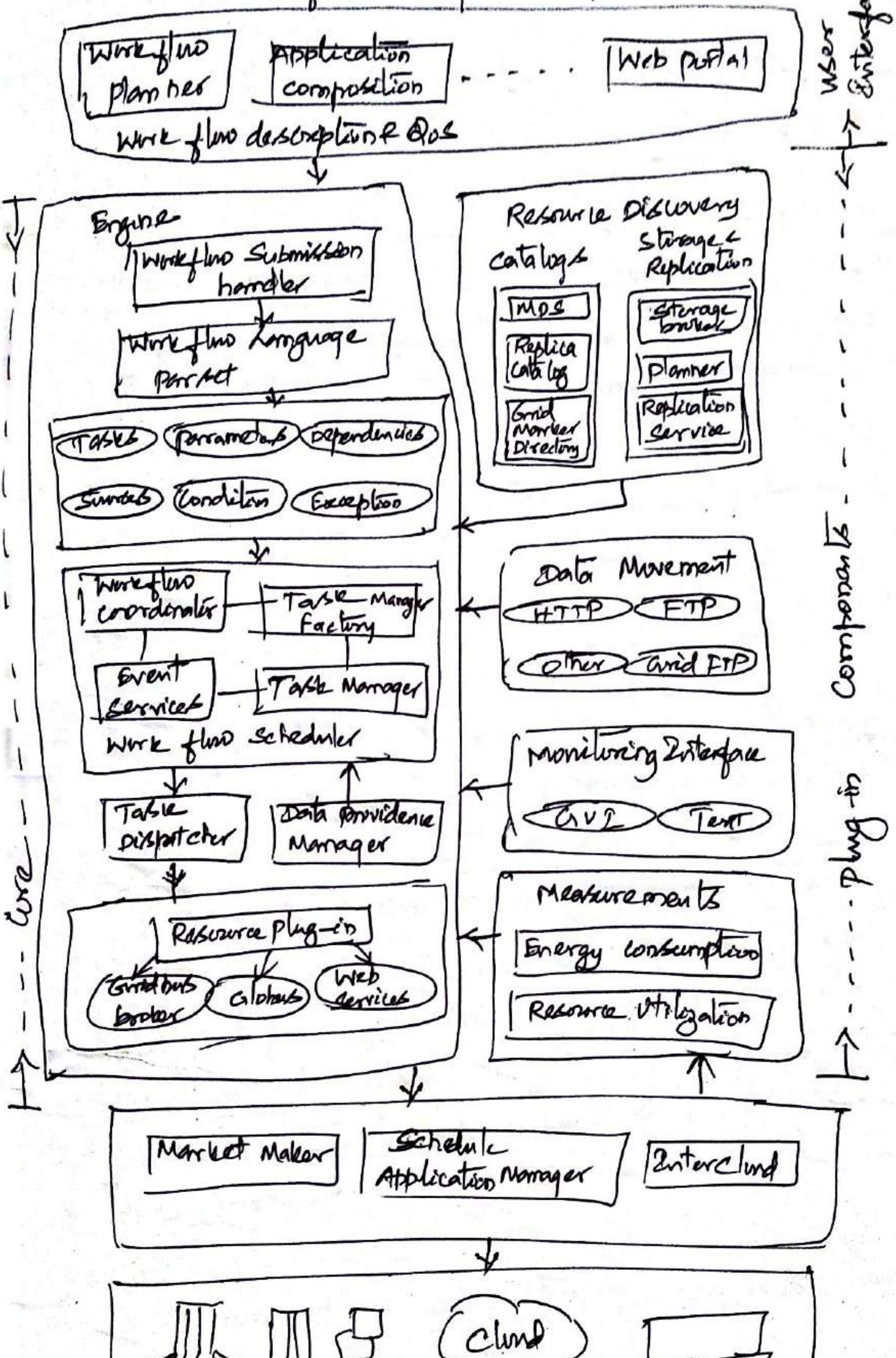
* Scientific applications are typically modeled as workflows. (Consists of tasks, data elements, control sequences. & data dependencies).

* WfMS are responsible for managing and executing these workflows.

The architechture is as follows.

— Architechture of WfMS —

Workflow Management System



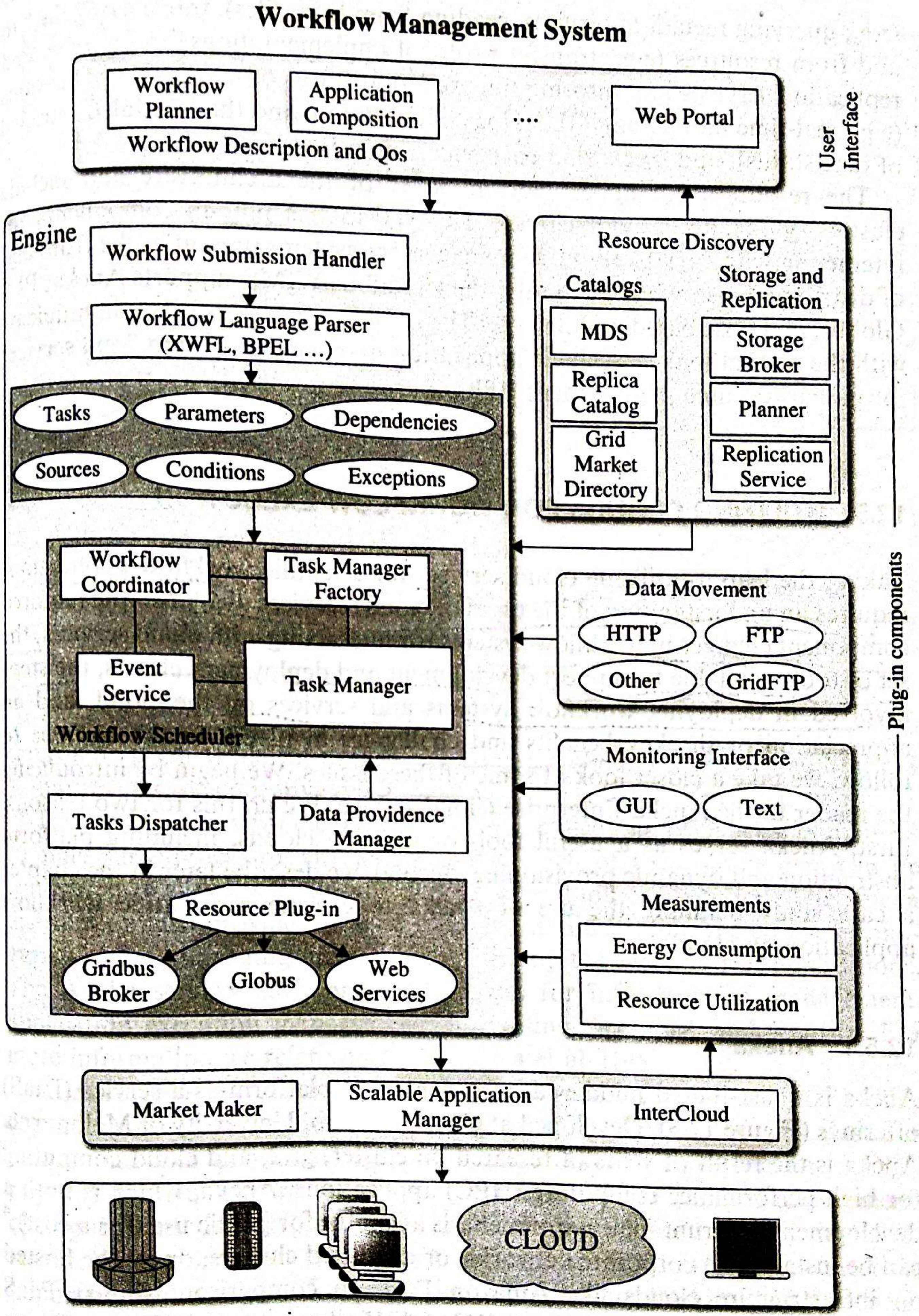


FIGURE 12.2. Architecture of Workflow Management System.

Three parts of the architechture:

(1) The user-interface

(2) The core

(3) Plug-ins.

<u>user interface</u>

- allows end users to work with workflow composition, uf execution planning, submissio and monitoring. These features are delivered through a web portal or through a stand alone application that is installed at the user's end.
- <u>Workflow</u> composition done through xWFL [XML Bas work flow language]

<u>Core</u>:

* Manages execution of workflow (Components inside core
* Facilitate in translation of high-level work flow descriptions (XML) to <u>task</u> & data objects.
  * These <u>tasks</u> can be used by execution sub-sys
* <u>scheduling</u> component applies users selected policies and plan at various stages in execution.
* <u>Tasks</u> & <u>data dispatchers</u> interact with resource interfa plug-ins, to continuously submit & monitor tasks in the workflow.

<u>Plug-ins</u>:

* supports work flow execution on different enviro and platform.
* plug-ins are avaible for quering tasks & data characteristics, transferring data to and from resource monitoring the execution status of tasks & applications and measuring energy consumption.

* Resources are at the bottom layer.

It includes:

*clusters * global grids and * clouds

* WfMS plugins are present in the front-end of distributed resources.

* currently WfMS supports -> Aneka, Pbs, Globus and fork-based middleware

* For global resource Management, resource managers may communicate with Market maker, Scalable application manager and Inter cloud services.

—^—

Utilizing clouds for Workflow Execution

It requires the understanding of: the types of cloud services available.

* required component changes in the workflow
* set of tools available to support developmente deployment efforts
* key benefits and challenges.

Aneka:

- is a distributed middleware for PaaS.
- Developed @ CLOUD's Lab, University of Melbourne
- available for public use (run time envi & Development)
- Developed on MS.NET framework.
- Run on Windows, Linux & MAC
- Runtime environment -> collection of Aneka containers running on physical or virtual nodes
- Development enviroment -> provides rich set of API's for developing application. (used as free resource)
- Storage service provides a temporary repository for application files.

* Aneka exposes 3 SOAP web services for negotiation, reservation & task submission

[SOAP — Simple Object Access Protocol]
its a messaging protocol specification for exchanging structured info in the implementation of webservice in computer N/W

* Moving workflow engine to cloud requires:
  - ⓐ Architectural changes and
  - ⓑ Integration of cloud management tools.

ⓐ Architectural changes

* Most components of WfMS can be separated from core engine
* Separated components can communicate to centralised or workflow engine using events
* Manager — responsible for co-ordination and distribution of load.

ⓑ Integration of cloud management Tools:

As one WfMS is breaken into components, we need a mechanism to ⓐ access, transfer and store data ⓑ enable + monitor execution.

* cloud service providers, providing API's and tools for discovering the VM instances, associated to a users account.
* For data storage & access, cloud has to provide data sharing, data movement & access rights mgmt capabilities to users application
* cloud measurement tools to calculate the used amount of data and power. (Based on this users are charged).

Tools for utilising clouds in WfMS:

Technologies such as Amazon S3, Googles BigTable and Windows Azure storage services can support most storage requirements for work flow systems, while also being scalable, reliable and secure.

* Amazon's AWS Import / Export - service that speed up data movement by transferring large amounts of data in portable storage devices.
  * Data shipped to/ from Amazon and offloaded into/from S3 buckets using Amazon's high speed internal n/w.
* Most cloud providers offers service and API's for tracking resource usage and the costs incurred.
* Google App Engine & Windows Azure provides, platform for building scalable interactive web applications.

SAGA-Based Scientific Applications that utilise clouds:

SAGA -> Simple API for Grid Applications

* SAGA standard, SAGA implementations provide higher level programming application to developers.
* SAGA is used to develop scientific application, can utilize set of infrastructure ranging from vannila clouds to open source.

* [SAGA Standard and SAGA implementations provides higher level programming abstraction to developers.] *

* SAGA has also been used with openNebula.

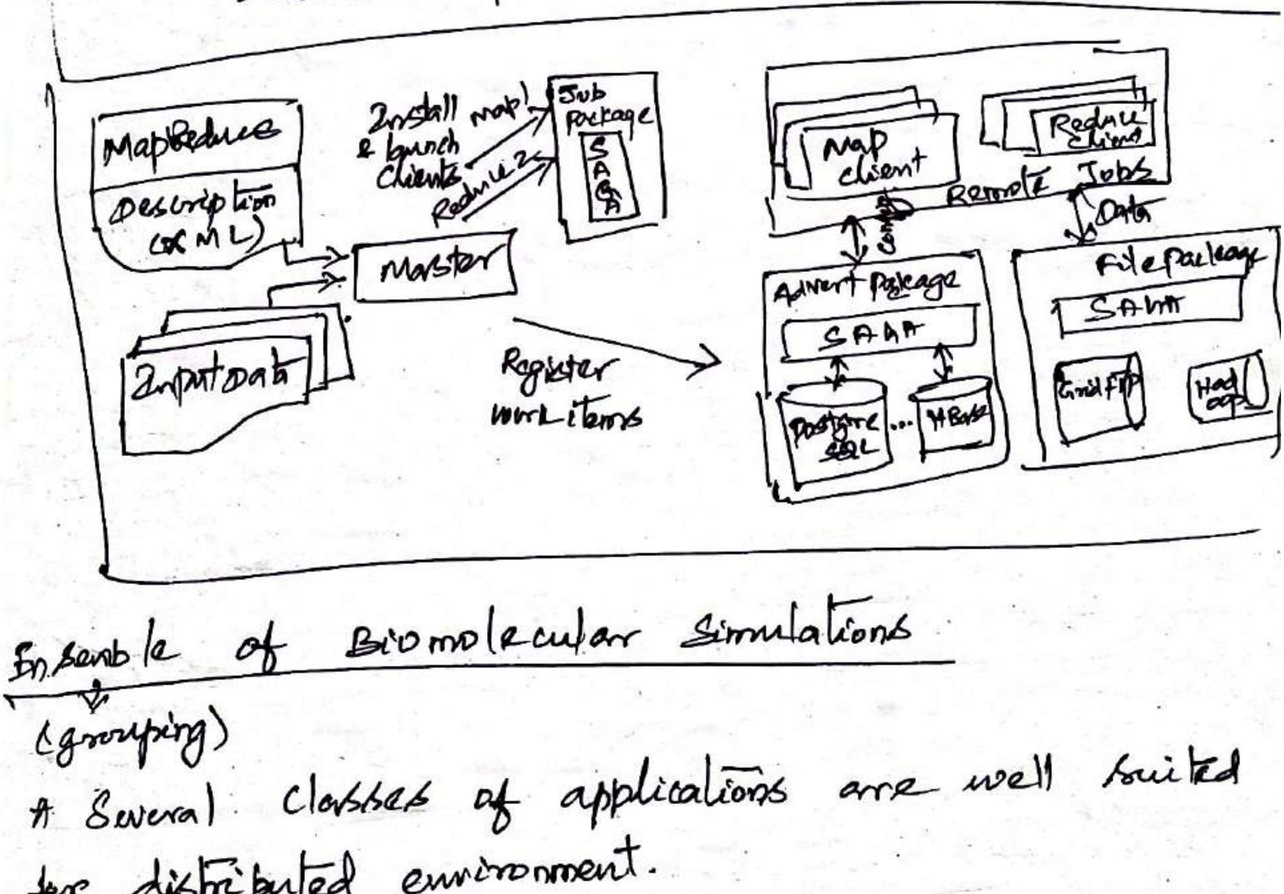
MapReduce (MR)

* It is an example for PaaS.

* SAGA MapReduce provides a MR development and runtime environment.

Advantage of SAGA based approach -> is infrastructure independent.

— SAGA MapReduce Framework —



Ensemble of Biomolecular Simulations
(grouping)

* Several classes of applications are well suited for distributed environment.

(eg) ensemble of decoupled jobs.

* Ensemble used to overcome limitations of insufficient timescales

* To efficiently execute ensemble of batch jobs without necessity to queue each individual job. (These applications uses SAGA Bigjob framework

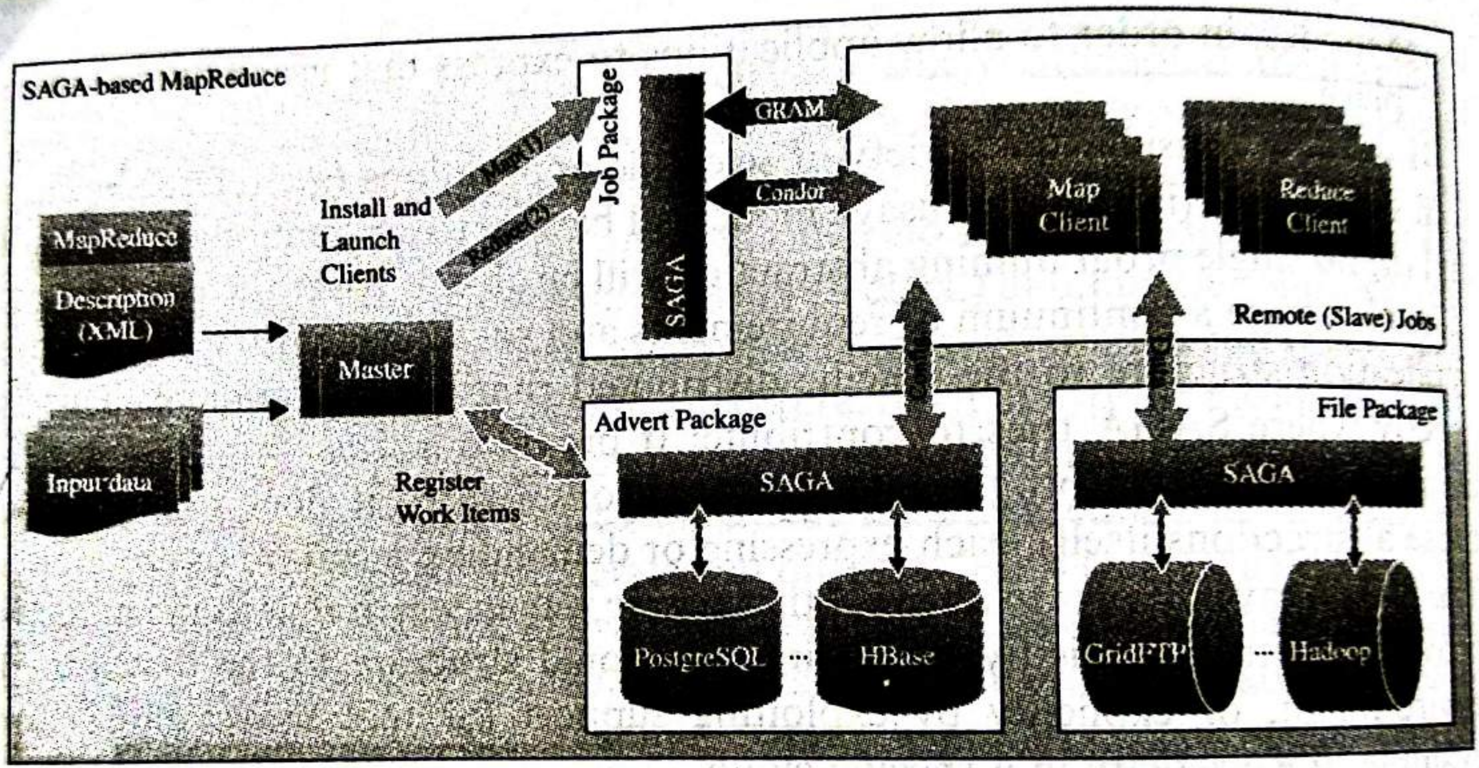


FIGURE 13.2. SAGA MapReduce framework. A master-worker paradigm is used to implement the MapReduce pattern. The diagram shows several different infrastructure options that can be utilized by the application.

BigJob — is a Pilot Job framework.

Pilot Job — Execution abstraction

↳ used to :- * improve utilization of resources

* reduce net wait time of collection of tasks

* facilitate bulk throughput simulations

* implement application-specific scheduling decisions and policy decisions.

— α — α —

MapReduce programming model

* Its a Software framework for solving many large-scale computing problems.

* It allows users to easily express their computation as Map & Reduce function.

* both Map & Reduce function written by the user.

(eg) for MR → word count - counts no of occurences of each word in large collection of document.

Map function - process a key/value pair to generate a set of intermediate key/value pair.

Reduce function - Merges all intermediate values associated with the same intermediate key.

Main features of MR:

* Data-Aware

* Simplicity

* Manageability (To manage I/O data)

* Scalability (increasing the no of nodes)

* Fault tolerance and reliability

Reliability: (*) Running all the tasks when a host goes to off-line

(*) Rerunning failed task on another node

(*) launching backup tasks.

— a —

MR execution overview:

When user program calls MR function, first splits input file into 'M' pieces [(ie) 16 to 64 MB.] and starts many copies of program. One is 'Master' ↓ Responsible for scheduling & monitoring and remaining are "Worker'. ↓

Reads the content, and emits a key/value pairs to the user defined Map function. Each workers result is buffered and periodically written in local disk.

—

Spotlight on Google MapReduce Implementation

(eg) Google's MR implementation targets large clusters of Linux PC's.

* Tasks are forked using remote procedure calls

* Runtime library is written in C++ with the interfaces in Python and Java.

— a —