

IT ARCHITECTURE

CLOUD COMPUTING

Willy Sudiarto Raharjo

Teknik Informatika
Universitas Kristen Duta Wacana

Grid Computing

What is all about Grid ?

- 1) Sharing, Selecting, aggregation of resources with a policy universally agreed.
- 2) Distributive supercomputing for a better future



Data Movement on New Installment

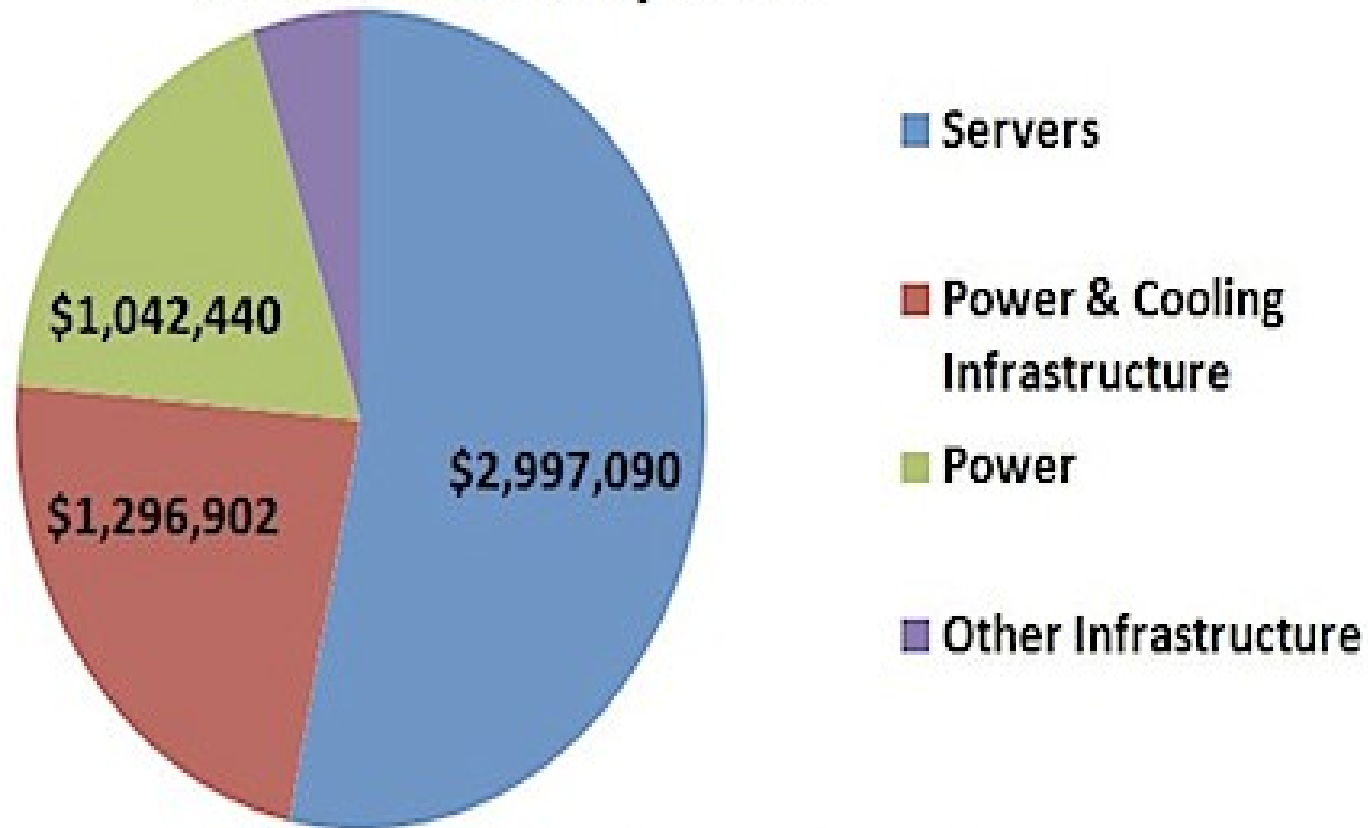


Server Maintenance



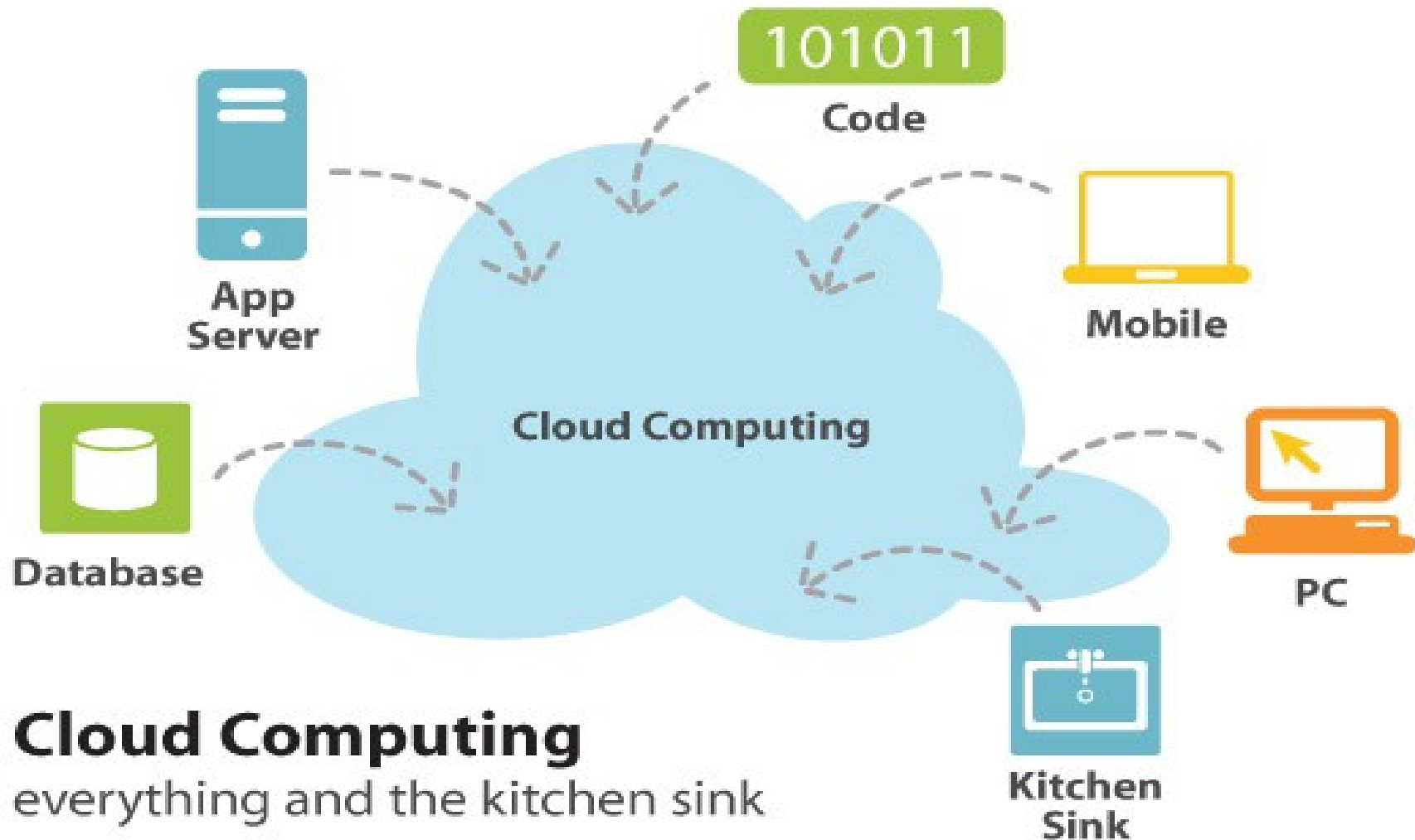
Maintenance Cost

\$284,686 Monthly Costs

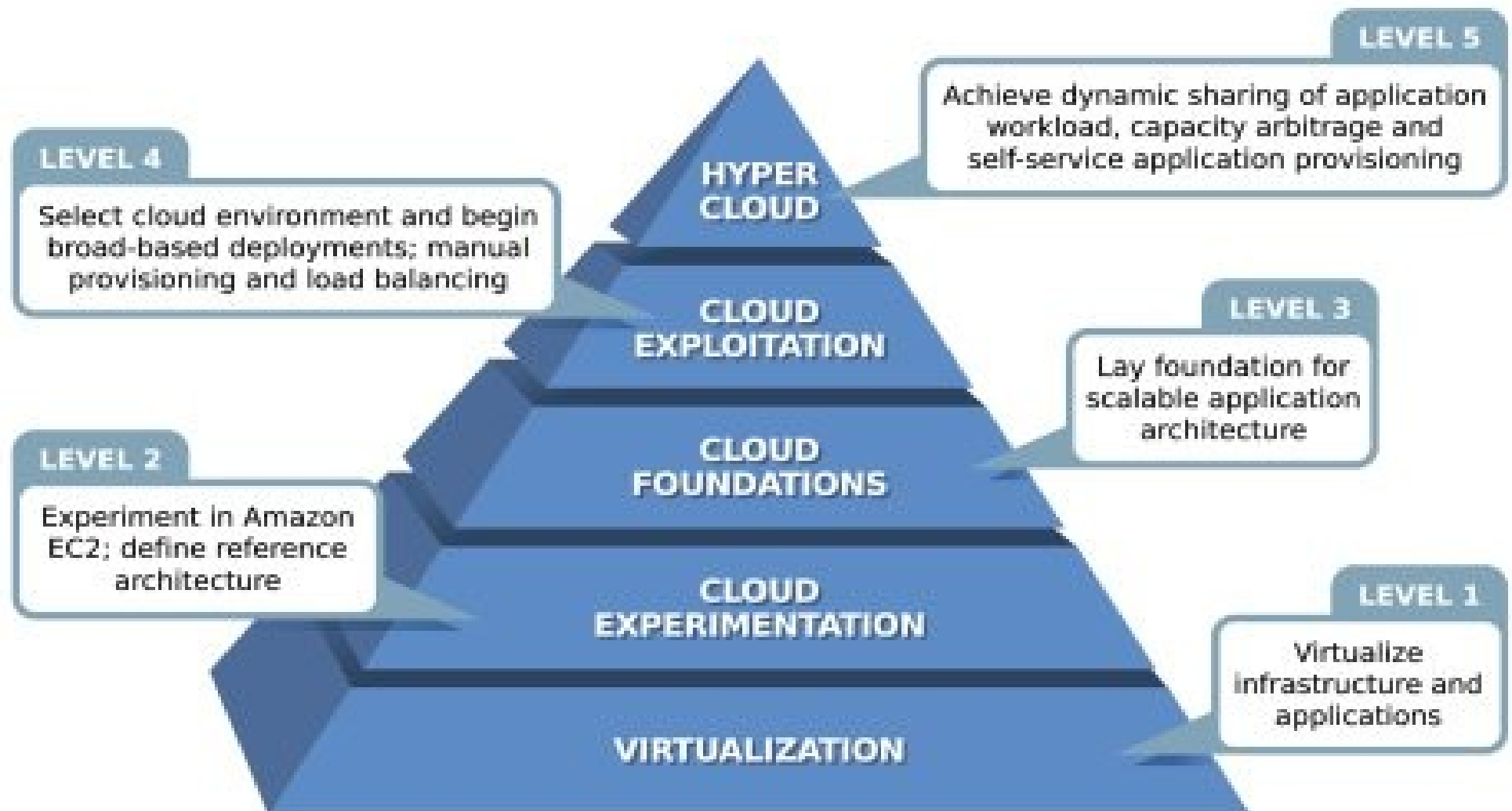


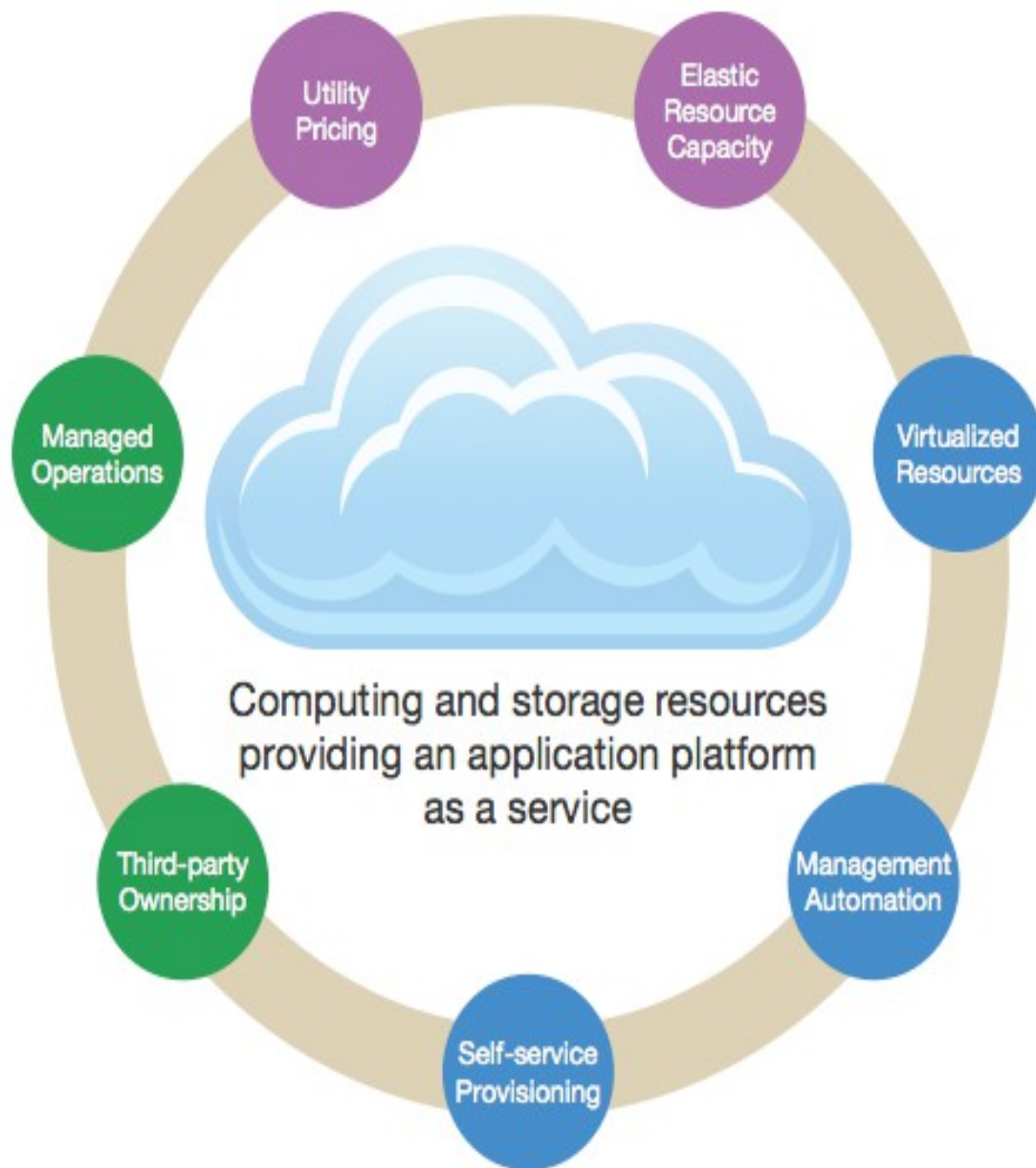
3yr server & 15 yr infrastructure amortization

Welcome to Cloud World



THE CLOUD COMPUTING ADOPTION MODEL





Economic Elements:

Pay-as-you-go,
pay-as-you-grow,
no CAPEX.

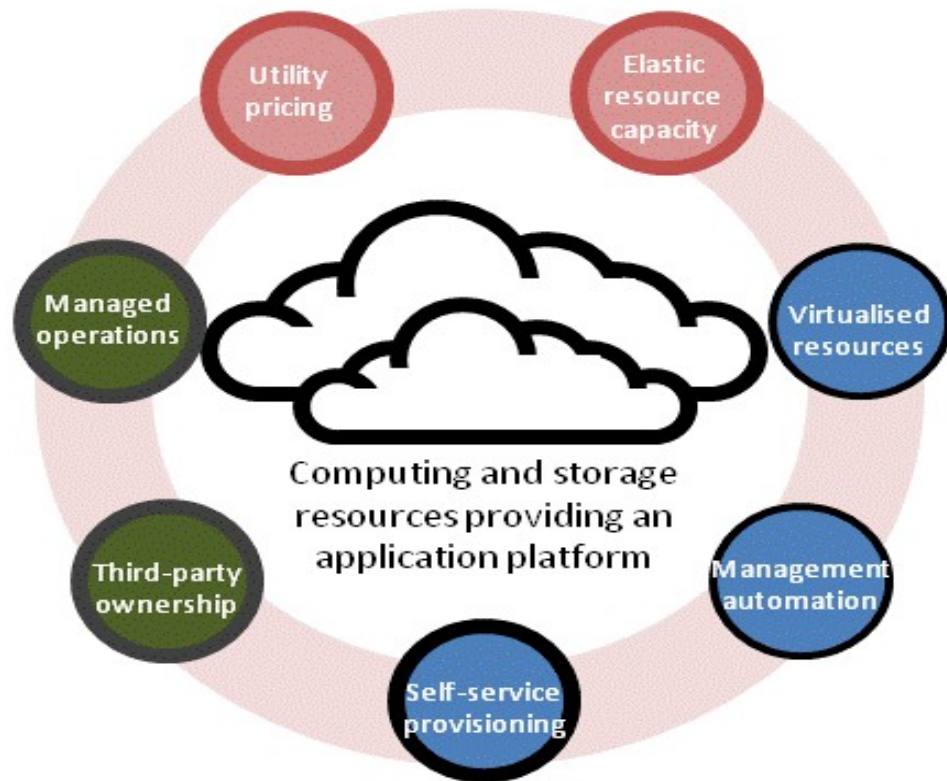
Architectural Elements:

Simple, abstract
environment for
development.

Strategic Elements:

Focus on your core business,
leave the rest to
someone else.

Public clouds



Private clouds

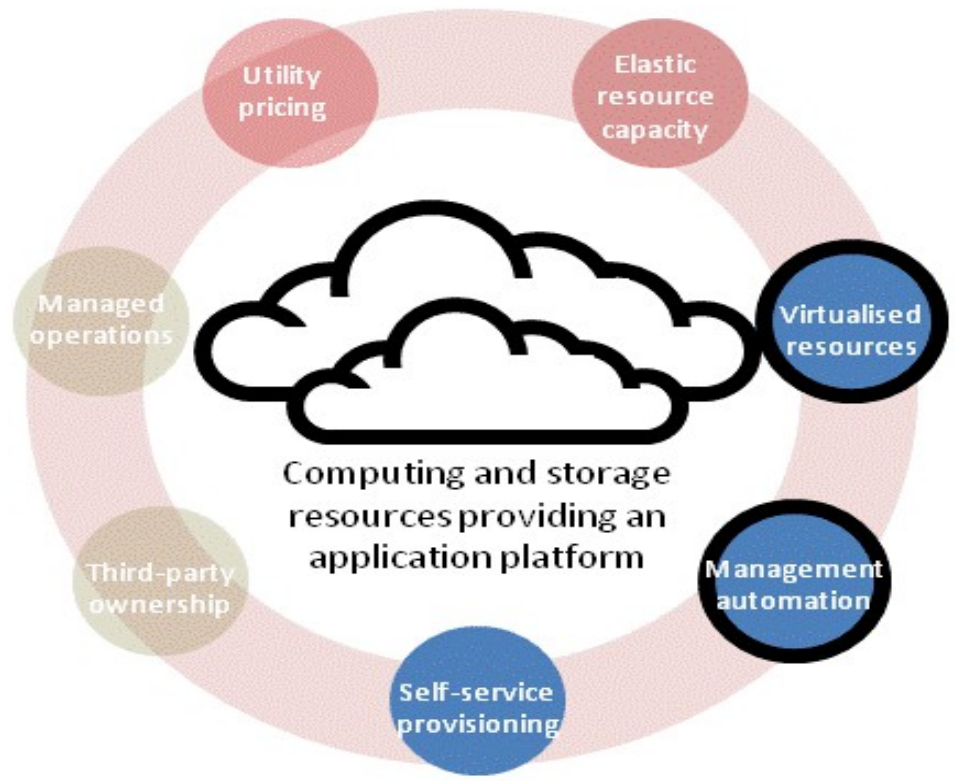
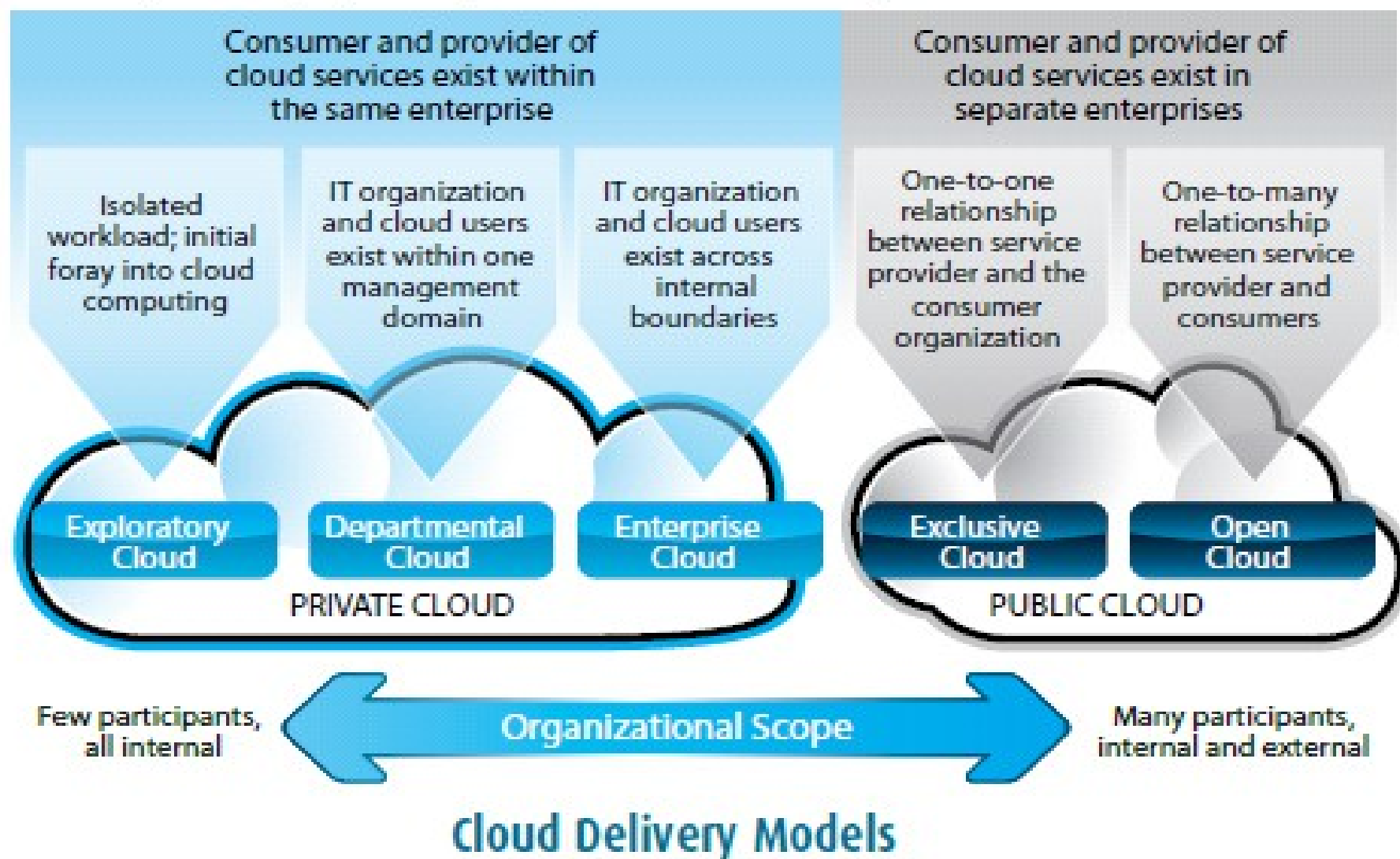


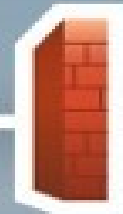
Figure 1.4 - Cloud subtypes.

Source IBM, White Paper, Defining a Framework for Cloud Adoption



HYBRID

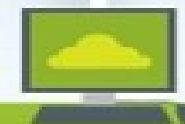
private / internal



public / external

PUBLIC CLOUD

AMAZON S3



USERS

Cloud Ecosystem

■ Classic Computing

- Repeated every 18 months
- Buy and Own
- Install, Configure, Test, Manage
- Use
- Pay \$\$\$\$\$\$

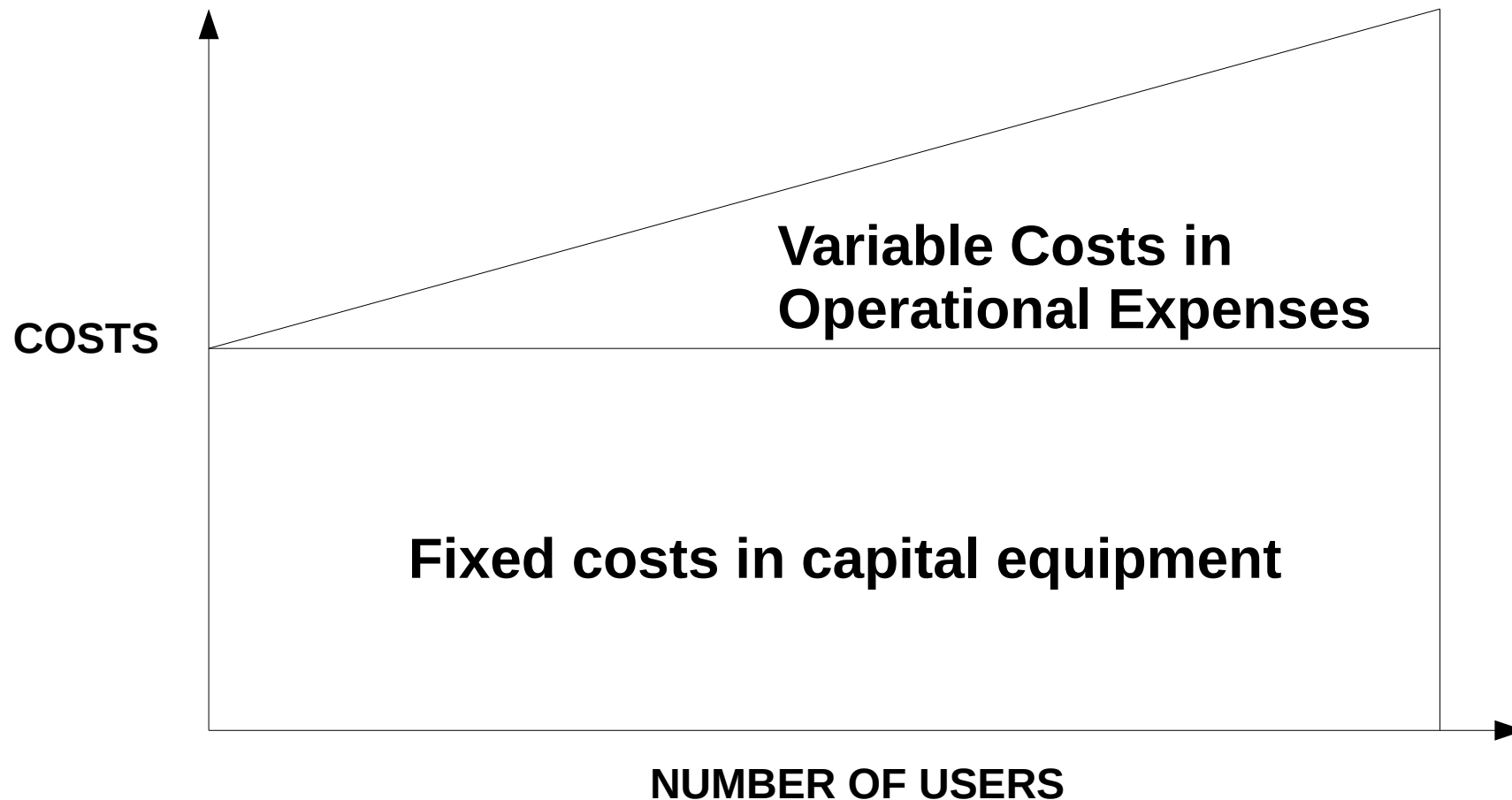
■ Cloud Computing

- Pay as you go per service provided
- Subscribe
- Use
- Pay for what you use (based on QoS)

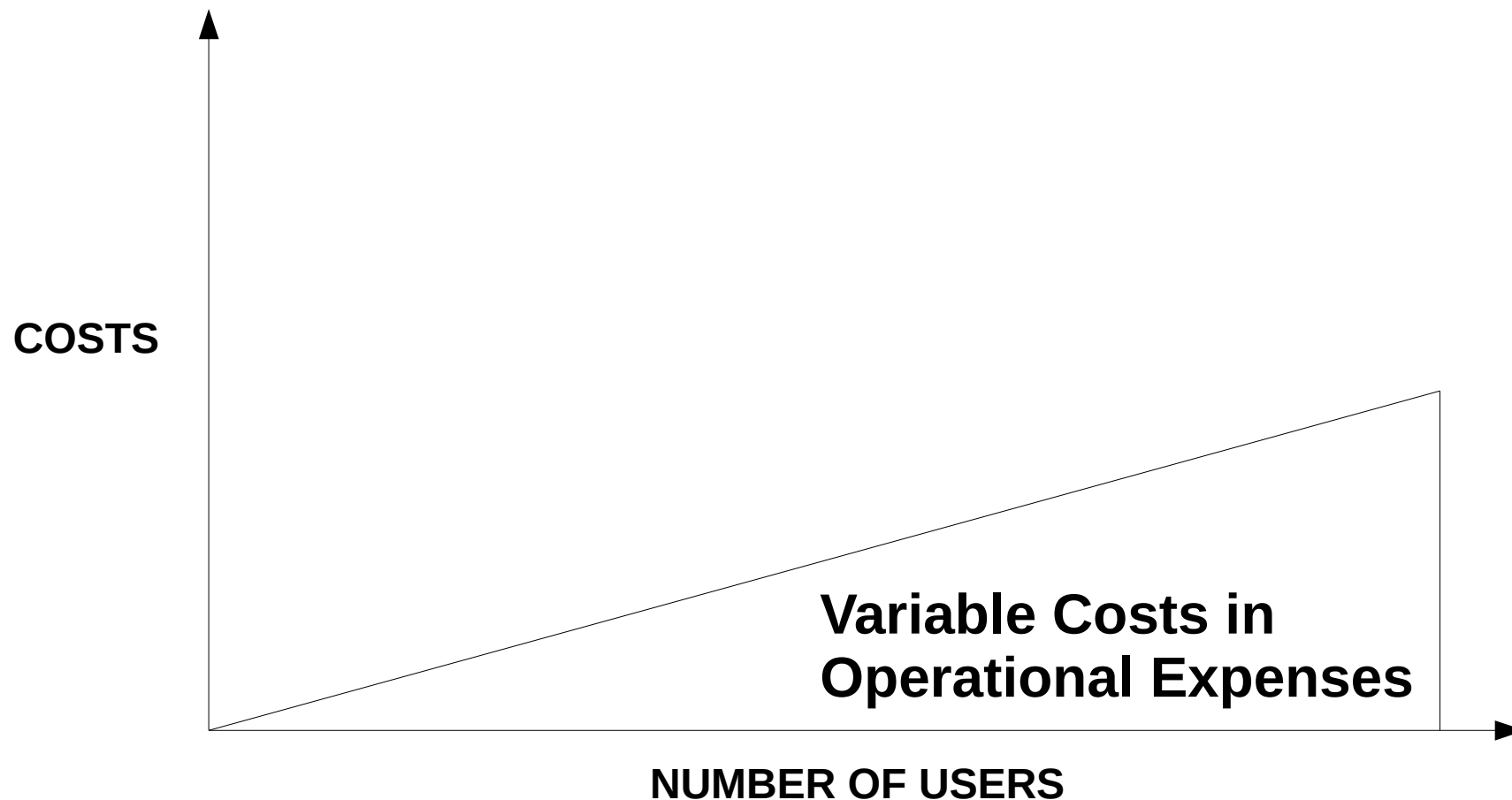
Cloud Design Objectives

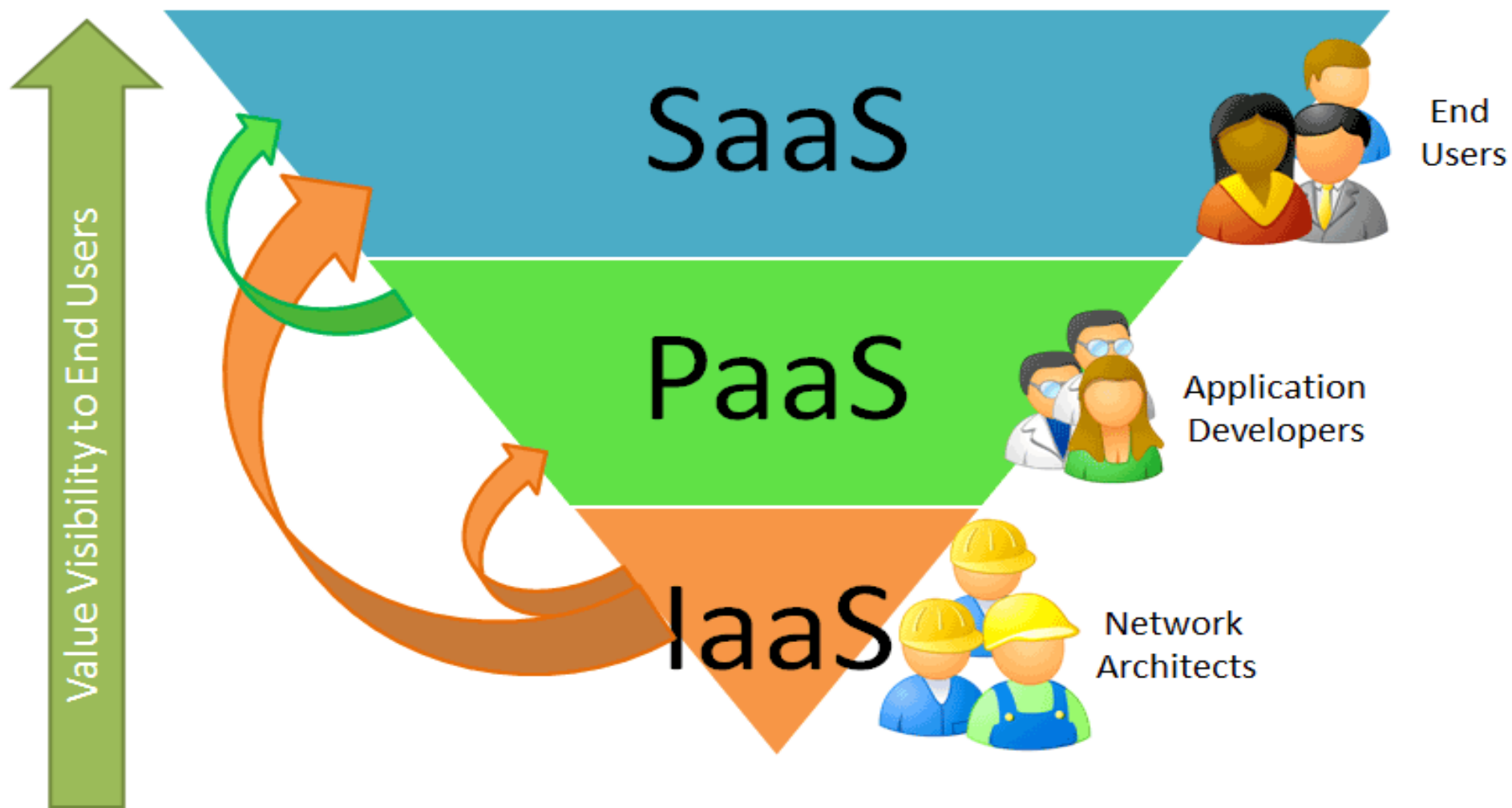
- Shifting computing (desktop → data centers)
- Service provisioning and cloud economics
 - SLA
 - Pay-as-you-go policy
- Scalability in performance
- Data privacy protection
- High quality of cloud services : Standardized QoS
- New standards and interfaces

Traditional IT Cost Model



Cloud Computing Cost Model





SaaS

- Software as a service
- Operating environment largely irrelevant, fully functional applications provided, e.g. CRM, ERP, email

PaaS

- Platform as a service
- Operating environment included, e.g. Windows/.NET, Linux/J2EE, applications of choice deployed

IaaS

- Infrastructure as a service
- Virtual platform on which required operating environment and application are deployed
- Includes storage as a service offerings

Application

OS + App Server Stack



Infrastructure

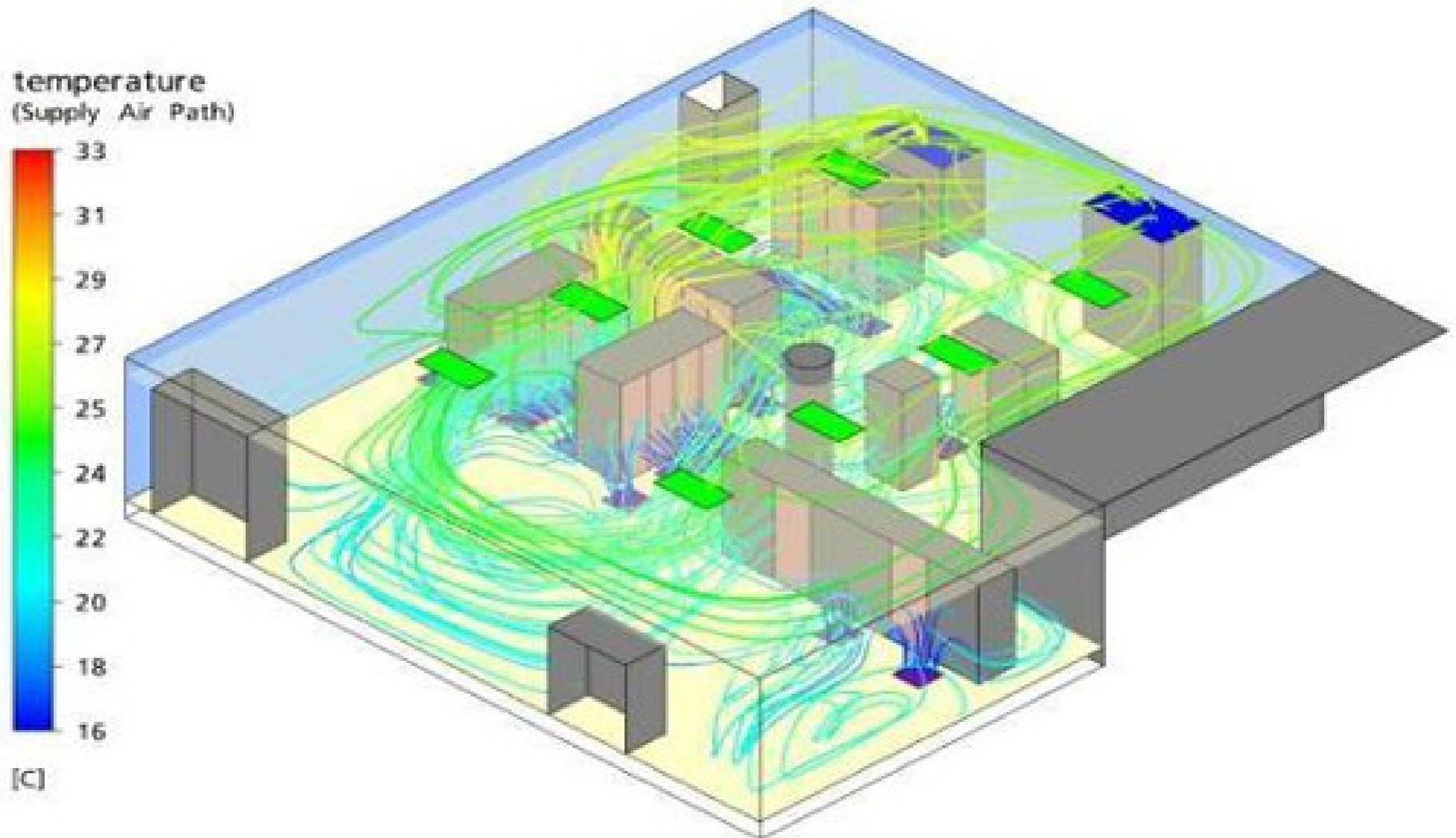
Platform

Software Application

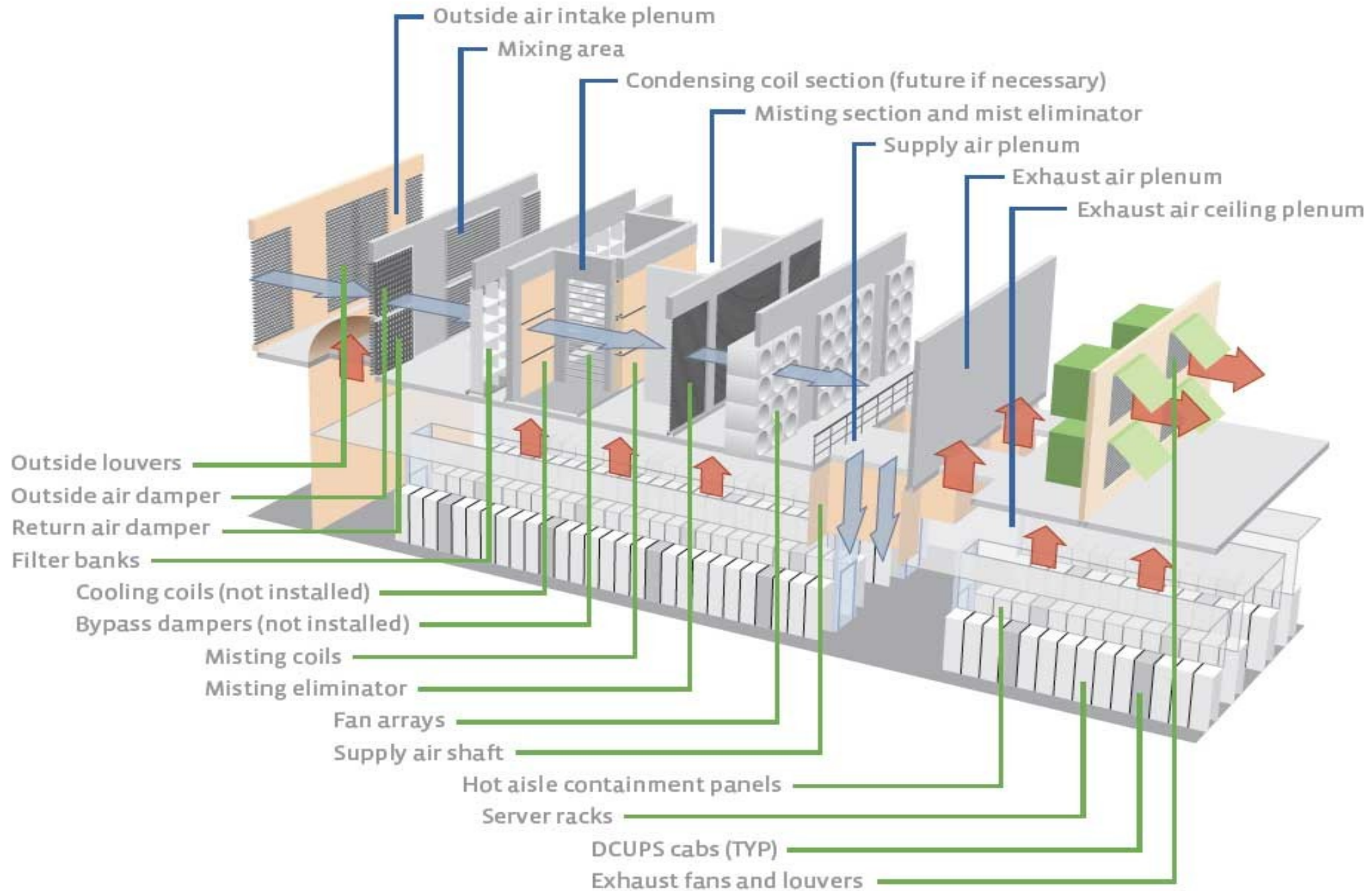
Model	IBM	Amazon	Google	Microsoft	Salesforce.com
Platform as a service	BlueCloud, Websphere CloudBurst Appliance, Research Compute Cloud (RC2)		Google App Engine	Windows Azure	Force.com
Infrastructure as a service	Ensembles	Elastic Compute Cloud, Simple Storage Service, Simple Queue Service, SimpleDB			
Software as a service	Lotus Live		Gmail, Docs	.NET service, dynamic customer relationship management (CRM)	Online CRM, Giftag
Reported services	Service-oriented architecture, B2, Tivoli Service Automation Manager, Rational Application Developer, Web 2.0	Amazon Web Services, Hadoop	GFS, BigTable, MapReduce	Live, Structured Query Language, Azure, Hotmail	Apex, Visualforce, record security
Security features	WebSphere2 and PowerVM tuned for protection	Public-key infrastructure and VPN for security, Elastic Block Store to recover from failure	Some HW security in data centers	Replicated data, rule-based access control	Administrative record security, metadata API

*Blank table entries refer to unknown services or cloud applications that are still under development.

Data Center Design



Open Compute Project Data Center



Data-Center Interconnection Networks

- Low latency
- High bandwidth
- Low cost
- Message-passing Interface (MPI) communication
- Fault tolerance

```

/*
"Hello World" MPI Test Program
*/
#include <mpi.h>
#include <stdio.h>
#include <string.h>

#define BUFSIZE 128
#define TAG 0

int main(int argc, char *argv[])
{
    char idstr[32];
    char buff[BUFSIZE];
    int numprocs;
    int myid;
    int i;
    MPI_Status stat;

    MPI_Init(&argc,&argv); /* all MPI programs start with MPI_Init; all 'N' processes exist thereafter */
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs); /* find out how big the SPMD world is */
    MPI_Comm_rank(MPI_COMM_WORLD,&myid); /* and this processes' rank is */

    /* At this point, all programs are running equivalently, the rank distinguishes
       the roles of the programs in the SPMD model, with rank 0 often used specially... */
    if(myid == 0)
    {
        printf("%d: We have %d processors\n", myid, numprocs);
        for(i=1;i<numprocs;i++)
        {
            sprintf(buff, "Hello %d! ", i);
            MPI_Send(buff, BUFSIZE, MPI_CHAR, i, TAG, MPI_COMM_WORLD);
        }
        for(i=1;i<numprocs;i++)
        {
            MPI_Recv(buff, BUFSIZE, MPI_CHAR, i, TAG, MPI_COMM_WORLD, &stat);
            printf("%d: %s\n", myid, buff);
        }
    }
    else
    {
        /* receive from rank 0: */
        MPI_Recv(buff, BUFSIZE, MPI_CHAR, 0, TAG, MPI_COMM_WORLD, &stat);
        sprintf(idstr, "Processor %d ", myid);
        strncat(buff, idstr, BUFSIZE-1);
        strncat(buff, "reporting for duty\n", BUFSIZE-1);
        /* send to rank 0: */
        MPI_Send(buff, BUFSIZE, MPI_CHAR, 0, TAG, MPI_COMM_WORLD);
    }

    MPI_Finalize(); /* MPI Programs end with MPI_Finalize; this is a weak synchronization point */
    return 0;
}

```

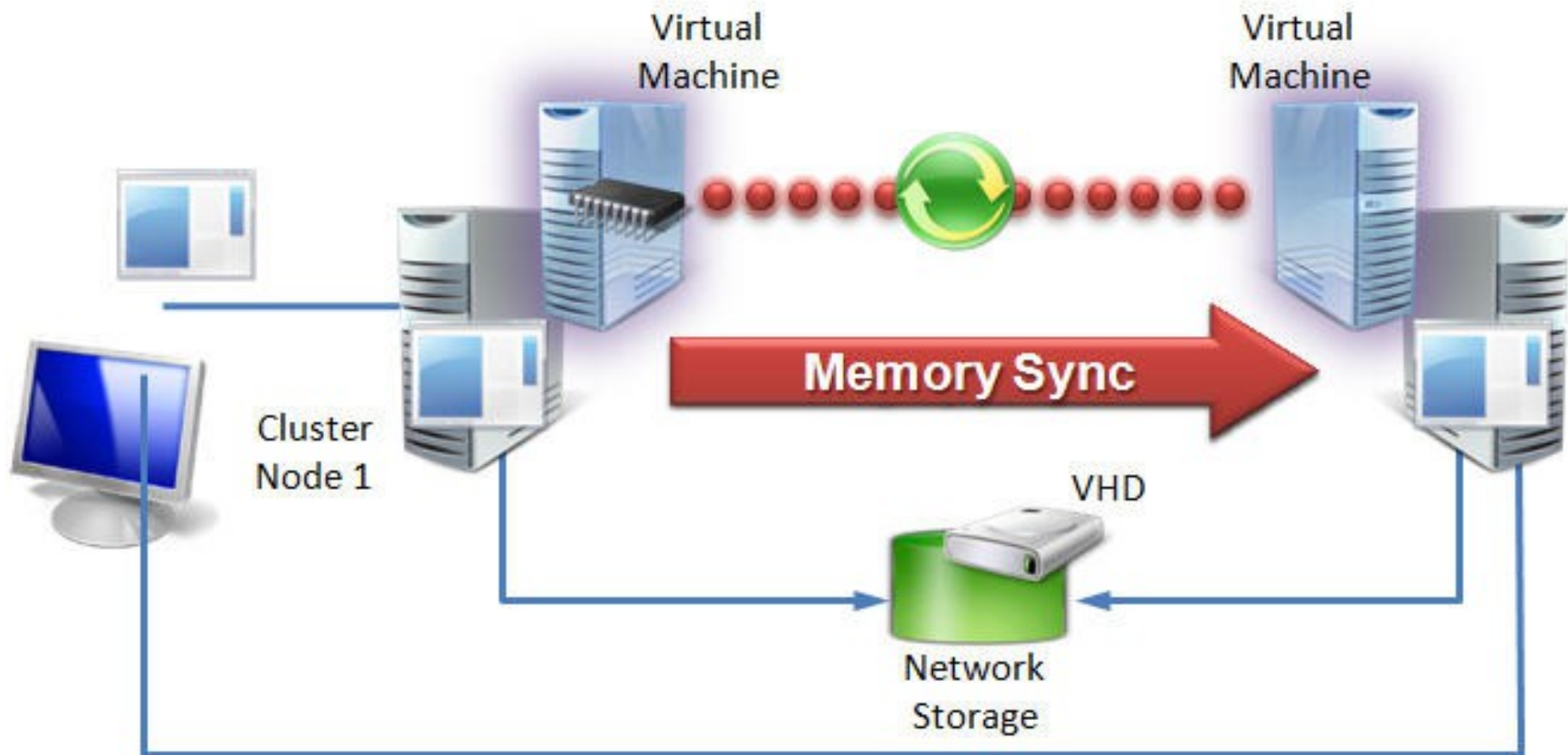
Management Issues

- Making common users happy:
 - long quality service (30 years)
- Controlled information flow:
 - Sustained service and high availability
- Multiuser manageability
- Scalability
- Reliability in virtualized infrastructure
 - Failover
 - Fault tolerance
 - Live migration

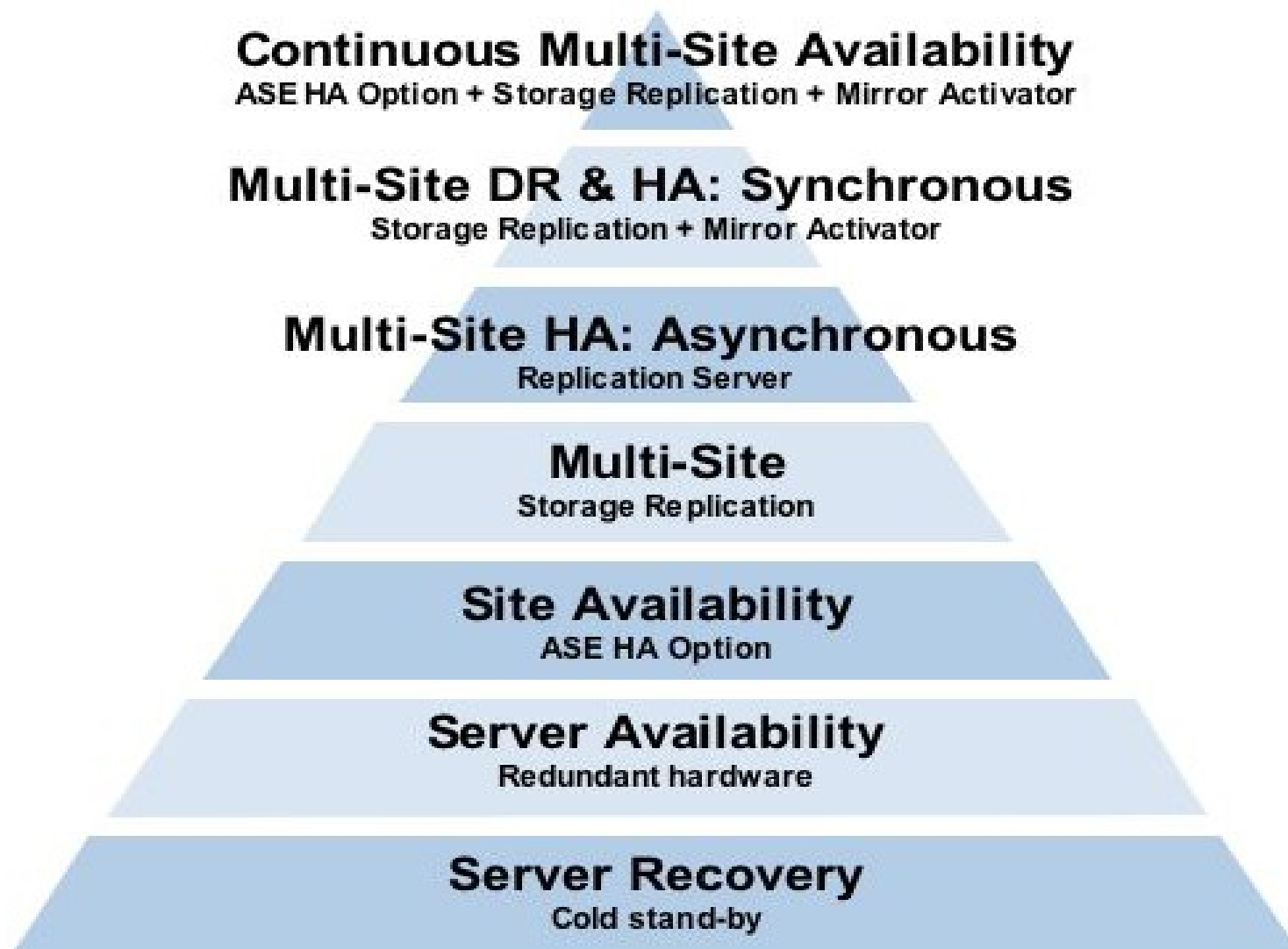
Management Issues

- Low Cost to both user and providers
- Security enforcement and data protection
- Green information technology

Live Migration



Disaster Recovery Hierarchy



Architectural Design Challenges

- Service Availability and Data Lock-in Problem
- Data Privacy and Security Concerns
- Unpredictable Performance and Bottlenecks
- Distributed Storage and Widespread Software Bugs
- Cloud Scalability, Interoperability, and Standardization
- Software Licensing and Reputation Sharing