# Peer-to-Peer and Cloud Computing

Prof. Dr. Jörg Hähner

## Part 7: Cloud Computing Basics

- Introduction
- Basics
- Cloud architectures
- Cloud offerings
- Conclusion and Literature

# Reminder: What is Cloud Computing?

“One of the appealing aspects of Cloud Computing is that it hides the complexity of IT technology from users and developers. No need to know details of how a service is generated – it is the service provider’s job to provide a corresponding abstraction layer.”

[Source: Christian Baun: Cloud Computing: Web-Based Dynamic IT Services, Springer Verlag 2011]

# Two Technologies for Agility

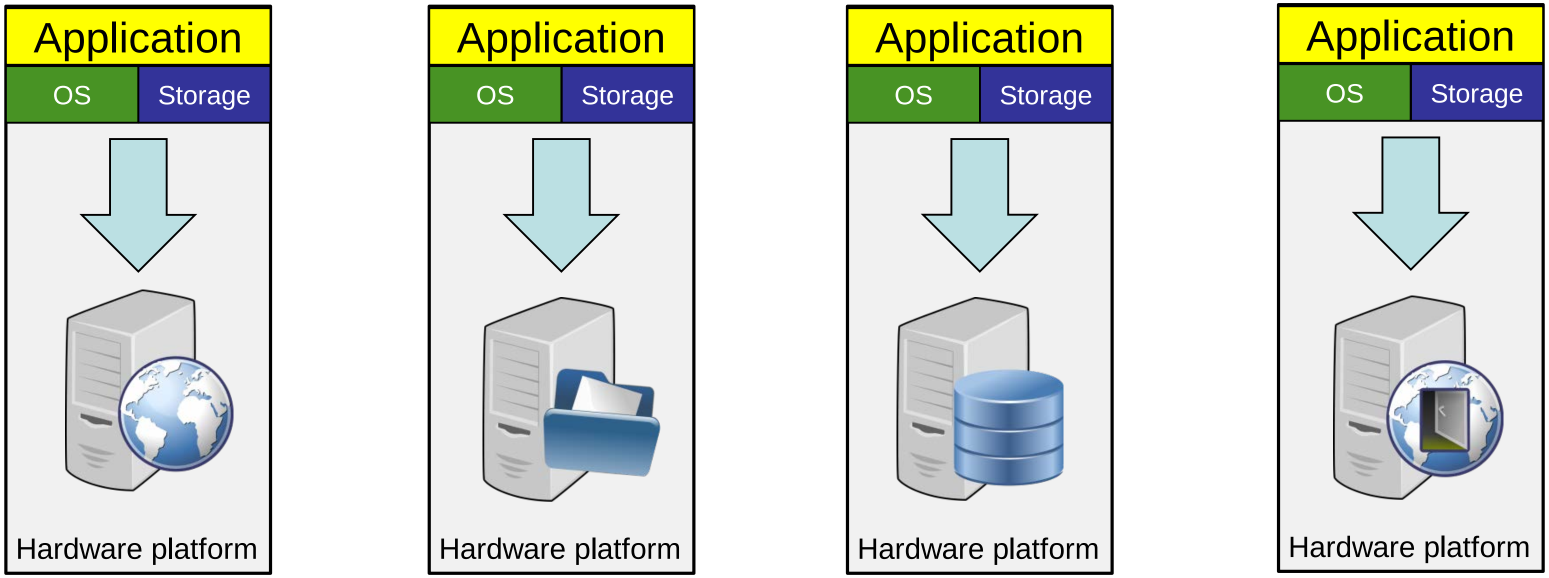
- **Virtualisation:**

  *The ability to run multiple operating systems on a single physical system and share the underlying hardware resources.*
  [VMware white paper, Virtualisation Overview]

- **Cloud Computing:**

  *"The provisioning of services in a timely (near on instant), on-demand manner, to allow the scaling up and down of resources".*
  [Alan Williamson, quoted in Cloud BootCamp March 2009]

# The Traditional Server Concept



Application
OS Storage
Hardware platform

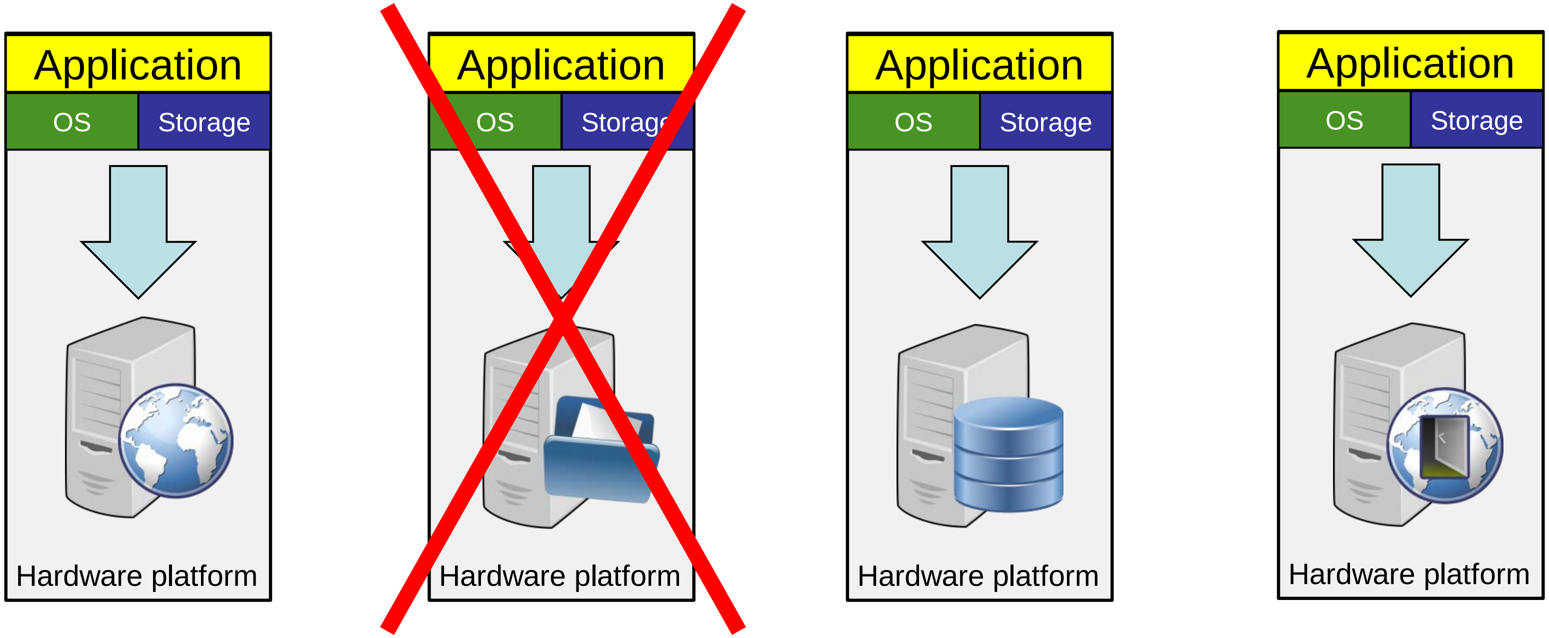
Application
OS Storage
Hardware platform

Application
OS Storage
Hardware platform

Application
OS Storage
Hardware platform

| **Web Server** | **App Server** | **DB Server** | **Email Server** |
|---|---|---|---|
| Windows | Linux | Linux | Windows |
| IIS | Glassfish | MySQL | Exchange |

# And if something goes wrong ...



| **Web Server** | **App Server** | **DB Server** | **Email Server** |
|---|---|---|---|
| Windows | DOWN! | Linux | Windows |
| IIS | | MySQL | Exchange |

# The Traditional Server Concept (2)

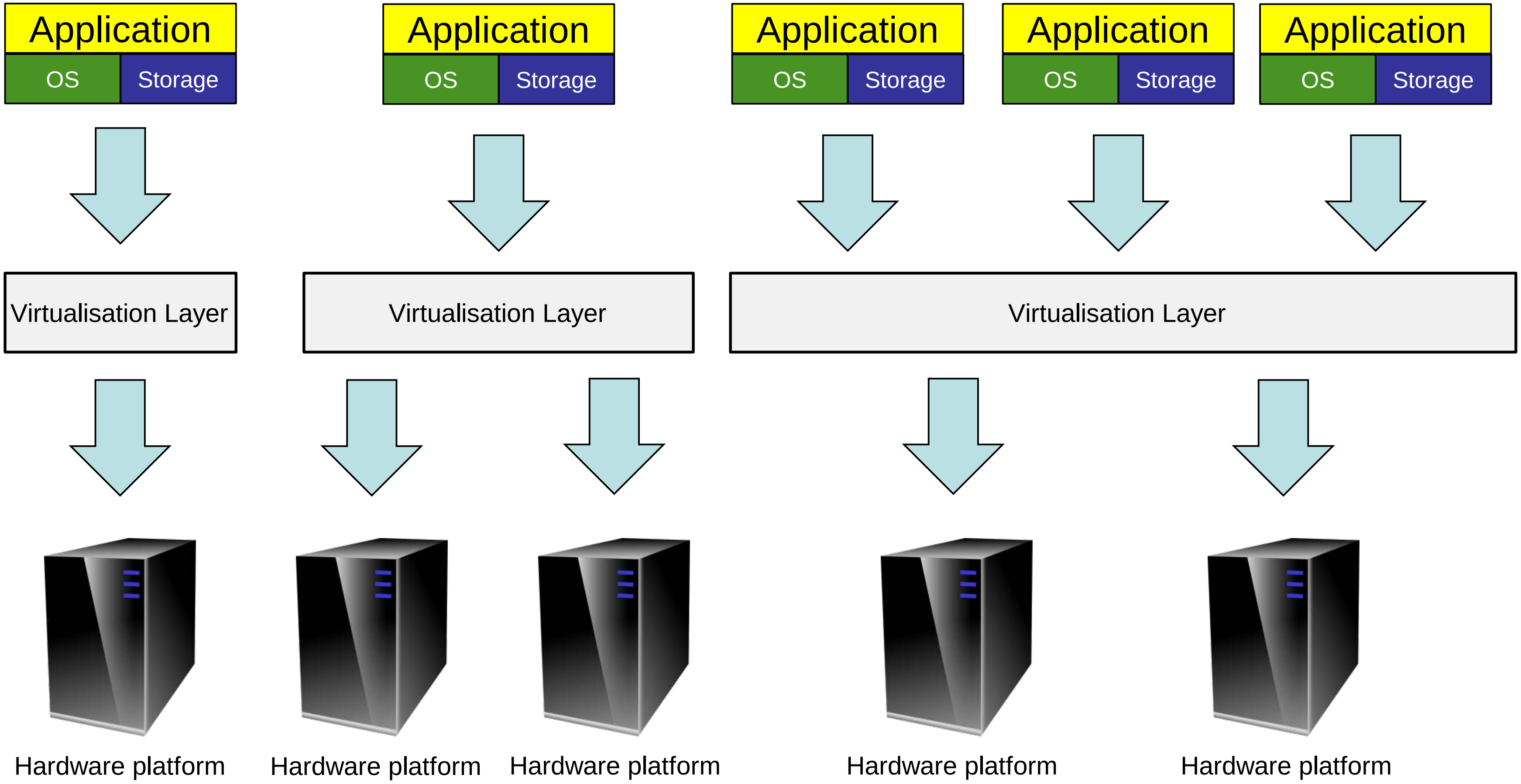
- System Administrators often talk about servers as a whole unit that includes the hardware, the OS, the storage, and the applications.
- Servers:
  - are often referred to by their function i.e. the Exchange server, the SQL server, or the File server.
  - If the File server fills up or the Exchange server becomes overtaxed then the System Administrators must add in a new server.
  - Unless there are multiple servers, if a service experiences a hardware failure, then the service is down.
  - System Admins can implement clusters of servers to make them more fault tolerant.  However, even clusters have limits on their scalability, and not all applications work in a clustered environment.

# The Traditional Server Concept (3)

- Pros
  - Easy to conceptualise.
  - Fairly easy to deploy.
  - Easy to backup.
  - Virtually any application / service can be run from this type of setup.

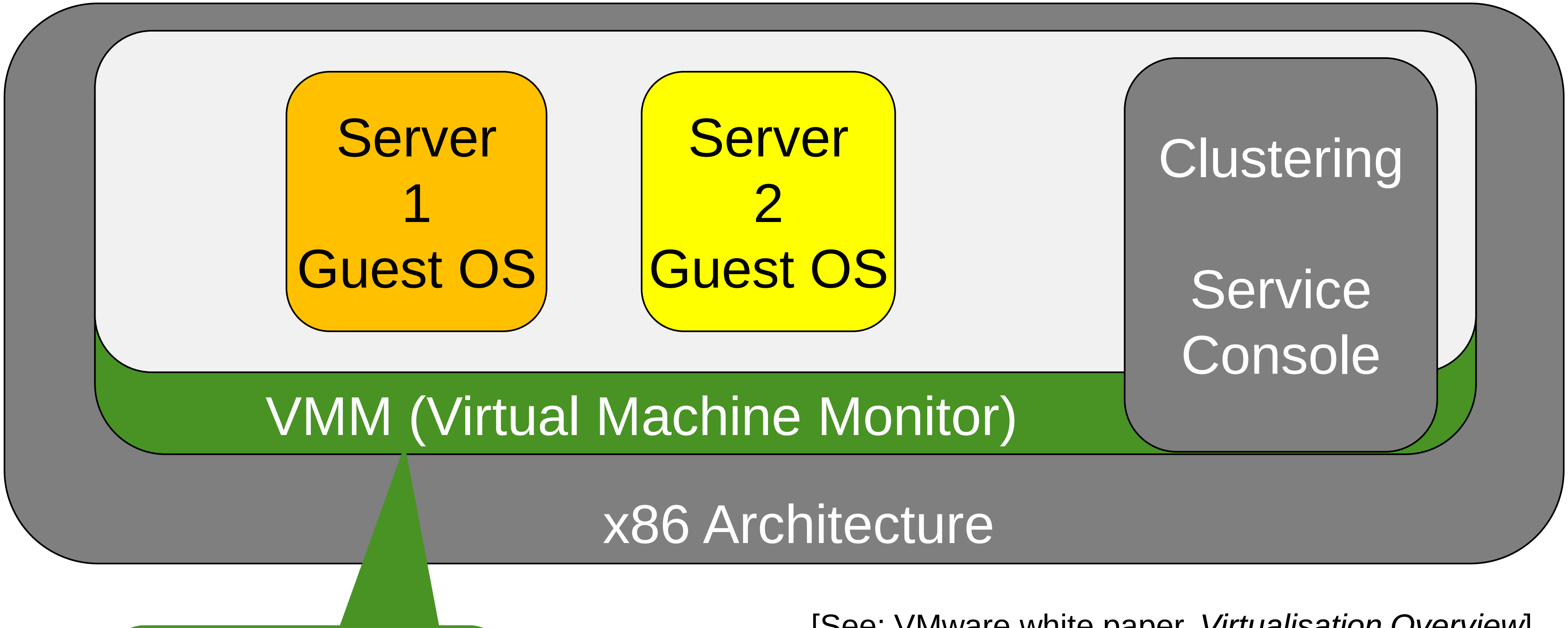
- Cons
  - Expensive to acquire and maintain hardware.
  - Not very scalable.
  - Difficult to replicate.
  - Redundancy is difficult to implement.
  - Vulnerable to hardware outages.
  - In many cases, processor is under-utilised.

# The Virtual Server Concept



Virtual Machine Monitor (VMM) layer between *Guest OS* and hardware.

# Detailed View on a Cloud System



Intercepts hardware requests

[See: VMware white paper, *Virtualisation Overview*]

# The Virtual Server Concept (2)

- Virtual servers (VS)
  - VS seek to encapsulate the server software away from the hardware.
  - This includes the OS, the applications, and the storage for that server.
  - Servers end up as mere files stored on a physical box, or in enterprise storage.
  - A VS can be serviced by one or more hosts, and one host may house more than one virtual server.
  - A VS can still be referred to by its function i.e. email server, database server, etc.
  - If the environment is built correctly, VSs will not be affected by the loss of a host.
  - Hosts may be removed and introduced almost at will to accommodate maintenance.

# The Virtual Server Concept (3)

- Virtual servers can be scaled out easily.
  - If the administrators find that the resources supporting a virtual server are being taxed too much, they can adjust the amount of resources allocated to that virtual server.
- Server templates can be created in a virtual environment to be used to create multiple, identical virtual servers.
- Virtual servers themselves can be migrated from host to host almost at will.

# The Virtual Server Concept (4)

- Pros
  - Resource pooling.
  - Highly redundant.
  - Highly available.
  - Rapidly deploy new servers.
  - Easy to deploy.
  - Reconfigurable while services are running.
  - Optimises physical resources by doing more with less.

- Cons
  - Slightly harder to conceptualise.
  - Slightly more costly (must buy hardware, OS, Apps, and now the abstraction layer).

# Virtualisation Status

- Offerings from many companies
  - e.g. VMware, Microsoft, Oracle (Sun), ...
- Hardware support
  - Fits well with the move to 64 bit (very large memories) multi-core (concurrency) processors.
  - Intel VT (Virtualisation Technology) and AMD-V provides hardware to support the Virtual Machine Monitor layer.
- **Virtualisation is now a well-established technology!**

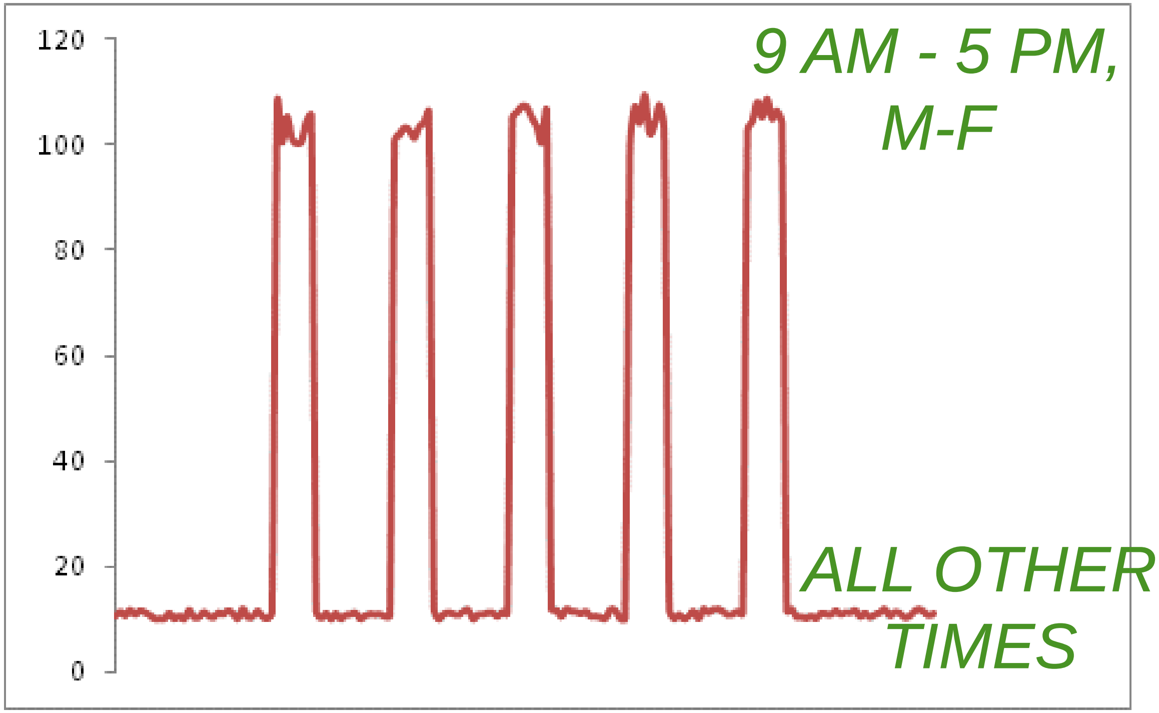
→ More details later on in this lecture…

# Example: Forbes.com

Suppose you are Forbes.com

- Forbes is an American business magazine.
- Forbes offers on-line real time stock market data.
- Why pay for capacity weekends, overnight?

Rate of Server Accesses



9 AM - 5 PM, M-F

ALL OTHER TIMES

# Example: Forbes.com (2)

- Forbes’ solution:
  - Host the web site in Amazon's EC2 *Elastic Compute Cloud.*
  - Provision new servers every day, and deprovision them every night.
  - Pay just $0.10 per server per hour.
    (more for higher capacity servers)
  - Let Amazon worry about the hardware!
- Cloud Computing is the next step in virtualisation!
  - You don’t have to own the hardware!
  - You “rent” it as needed from a cloud.
  - There are public clouds (e.g. Amazon EC2, Microsoft, IBM, Sun, ..).
  - A company can create a private one (with more control over security, etc.).

# Goals of Cloud Computing

- Goal 1: Cost Control
  - Many systems have variable demands:
    - Batch processing (e.g. New York Times)
    - Web sites with peaks (e.g. Forbes)
    - Startups with unknown demand
  - Reduce risk
    - Don't need to buy hardware until you need it.

- Goal 2: Business Agility
  - More than scalability - *elasticity*!
  - Scaling back is as important as scaling up!

# Goals of Cloud Computing (3)

- Goal 3: Stick to your business
  - Most companies don't WANT to do system administration.
  - Forbes says: We are a publishing company, not a software company!
  - But beware: Do you really save much on sys-admins?
  - You don't have the hardware, but you still need to manage the OS!

- Example: Ely Lilly
  - Ely Lilly: company in rapidly changing health care business.
    - Used to take 3 - 4 months to give a department a server cluster, then they would hoard it!
  - Using EC2, about 5 minutes!
    - And they give it back when they are done!

# Cloud Computing: the general concept

- How Cloud Computing works:
  - – Various providers let you create virtual servers.
    - Set up an account, perhaps just with a credit card.
  - – You create virtual servers ("virtualisation").
    - Choose the OS and software each "instance" will have.
    - It will run on a large server farm located somewhere.
    - You can instantiate more on a few minutes' notice.
    - You can shut down instances in a minute or so.
  - – They send you a bill for what you use.

- Vorlesungsumfrage

- Introduction
- Basics
- Cloud architectures
- Cloud offerings
- Conclusion and Literature

# Technology of Cloud Computing

- Cloud Computing is based on several existing techniques, e.g.:
  - Virtualisation
  - Service-oriented architectures (SOA)
  - Web-services
- Virtualisation
  - … allows an abstract, logical view on the physical resources.
  - … includes servers, data stores, networks, and software.
  - Physical resources are pooled and managed as a whole.
  - Individual requests are served as currently required.
  - I.e.: dynamically generate a certain platform for a specific application exactly in that moment when it is needed.
  - Instead of real machines, virtual ones are used.

## Advantages of virtualisation for IT providers

- Resource usage
  - Physical servers seldom work to capacity to cover peak usage.
  - Virtual machines satisfy demands from the resource pool.
  - Hence, purchase of new capacity can be delayed or avoided.
- Management
  - Resource pool management can be automated.
  - Virtual machines are generated and configured automatically.
- Consolidation
  - Different classes of applications can run on a smaller number of physical components.
  - From server or storage consolidation up to entire system landscapes, data and databases, networks, and desktops.
  - Leads to increased efficiency and thus to cost reduction.

# Virtualisation (2)

## Advantages of virtualisation for IT providers (continued)

- Energy consumption
  - The cost of energy required to operate a server is higher than its purchase price.
  - Consolidation reduces the number of physical components – which, in turn, reduces the expenses for energy supply.
- Space requirements
  - Data and server centers are really expensive – less space is cheaper!
  - The same performance can be obtained on a smaller footprint and the costly expansion of an existing data center might possibly be avoided.
- Emergency planning
  - Virtual machines can be moved from one machine to another.
  - Better availability of services
  - Hardware maintenance windows are not longer necessary!

# Virtualisation (3)

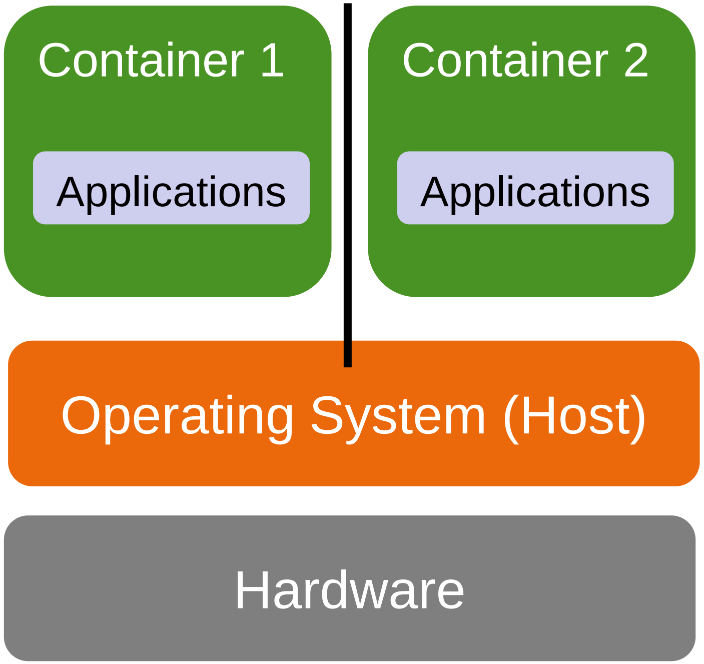
## Advantages of virtualisation for customers

- Availability
  - Services: highly available, can be used 24/7 without stop.
  - Hot-migration of applications in case of technology upgrades: virtual machines can easily be moved to an up-to-date system.
- Dynamic Behaviour
  - Any request can be satisfied just in time and without any delays.
  - In case of bottlenecks, a virtual machine can draw on additional resources (such as storage space, I/O capabilities).
- Access
  - Isolation of each virtual machine from others and the physical infrastructure
  - Virtual systems feature multitenant capabilities
  - Role concept: safely delegate management functionality to the customer.
  - Customer emancipation: purchase IT capabilities from a self-service portal

# Drawbacks of Virtualisation

- Drawbacks of virtualisation:
  - Operation of the abstraction layer itself requires resources
    - Due to the particularly effective interaction of current multicore systems with virtualisation technology, this performance loss plays only a minor role in today’s systems.
    - View point of possible savings: the quality benefits perceived by the customers when using virtualisation pays off in nearly all cases.
  - After the consolidation: more systems need to be operated and managed
    - Besides the virtual machines, there is the physical infrastructure.
    - Current sophisticated management tools achieve a positive balance as much less staff is required in practice.

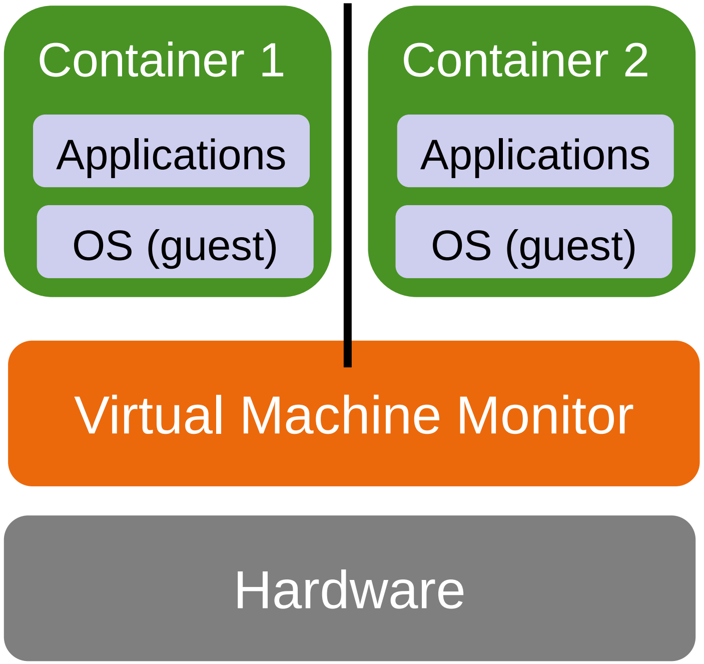
1. Operating system (OS) virtualisation („container“ or „jails“)
   - Multiple identical and isolated system / runtime environments run under one operating system kernel.
   - Outside view: virtual environments appear as autonomous.
   - Use the same kernel, but only the same virtual environment accessible.
   - Pro: security and confidentiality
   - Con: Reduced flexibility as different OSs cannot run simultaneously.
   - Examples:
     - IBM LPARs
     - Used by Internet Service Providers offering virtual root serves.
     - Container technology from Sun Solaris, Linux-VServer, FreeBSD Jails, and Virtuozzo



# Concepts of Virtualisation (2)

## 2. Platform virtualisation

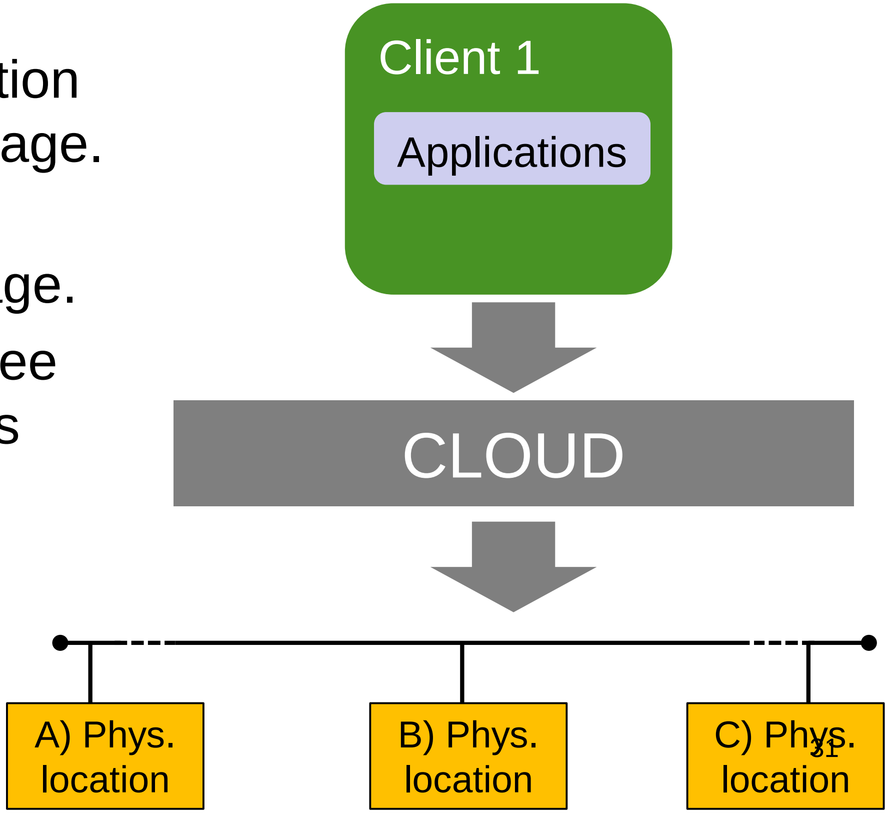
- Goal: run any desired OS and applications in virtual environments
- Distinguish between full virtualisation and paravirtualisation
  - Full virtualisation: simulation of an entire virtual computer with virtual resources (e.g. CPU, RAM)
  - Paravirtualisation: Provides an application interface instead of an emulated hardware layer
- Both solutions are implemented using a virtual machine monitor or hypervisor
- Hypervisor: minimalistic meta-OS used for distributing hardware resources among guest systems and for access coordination
  - Type 1: built directly on top of the hardware
  - Type 2: runs under a traditional basic OS



# Concepts of Virtualisation (3)

## 3. Storage virtualisation

- Goal: dynamically scaling storage space as service.
- Approach: separate data store from classical file servers and pool the physical storage systems.
- Additional abstract administration layer between clients and storage.
- Representation of data is decoupled from physical storage.
- E.g. Amazon creates up to three copies in different data centres when storing data!



# Concepts of Virtualisation (4)

## 4. Network virtualisation

- Techniques such as load balancing are essential in cloud environments as offering dynamically scaling services is needed.
- Most prominent approach: Implement resources as web objects.
- Establish virtual local network (VLAN)
    - Transparency: Distributed devices can be pooled together in a single logical network (i.e. geographically separated elements).
    - Security: Subsystems requiring special protection are hidden in a separate virtual network.
- Disadvantage: VLANs cause overhead for network administration and for programming active network components (i.e. switches).

# Concepts of Virtualisation (5)

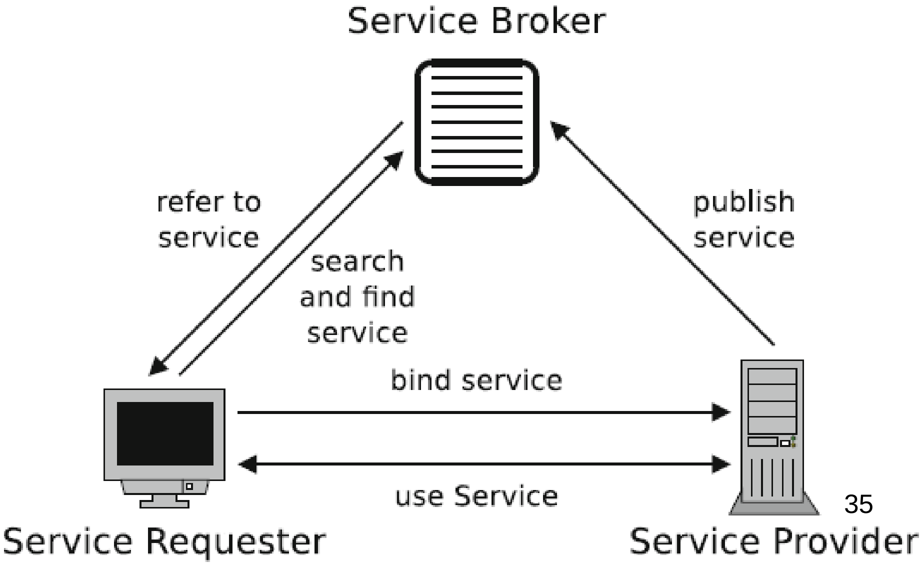
5. Application virtualisation
   - Centrally managed applications are offered over the network.
   - Advantages compared to traditional software:
     - Easier administration
     - Automatic management of updates and patches
     - Compatibility: all users work with the same software portfolio
     - Global availability
   - Distinguish between two alternative ways of deployment:
     - Hosted application: The application is available on the Internet and is transmitted to the client, e.g. using a streaming protocol.
     - Virtual appliance: The application can be downloaded and used on the customer’s own computer.
   - Virtual environment acts as a buffer between application and OS.
   - Result: SaaS (Software as a Service) concept which is used for the dynamic provision of software components.

# Service-oriented Architectures (1)

- Service-oriented architectures (SOA) are architectures whose components are implemented as independent services.
- Typical properties of an SOA are:
  - Consists of distributed components (i.e. the services).
  - Heterogeneous service consumers and service providers are interoperable across platforms; different programming languages and platforms are used to implement individual services.
  - Services are loosely coupled and bound dynamically at runtime.
  - SOAs consequently allow dynamic adjustments, which have a local (but no systemwide) effect.

# Service-oriented Architectures (2)

- Basic, theoretical interaction scheme at runtime:
  1. Service consumer can locate a service in a service directory. Alternative: discover it using a broker.
  2. If a suitable service has been found / brokered, service consumer receives a reference (address, endpoint) for accessing the service (i.e. exchanging messages with it).
  3. Service can be called (i.e. a message can be sent).
  4. The service provider replies by sending a message back.



# Service-oriented Architectures (3)

- Practice: service endpoints are often directly communicated as part of the message.
- Attempts to set standards for service directories:
  - E.g. Universal Description, Discovery and Integration (UDDI)
  - Not successful
- Sending endpoints directly in a message: brokering and binding services at runtime dynamically.

# Web Services

“A Web service is a software application identified by a URI, whose interfaces and bindings are capable of being defined, described and discovered by XML artifacts and supports direct interactions with other software applications using XML-based messages via internet-based protocols.”

[Source: The Web Services Architecture Working Group of the W3C]

→ Web services (WS) are a kind of distributed middleware, which enables machine-to-machine communication on the basis of Web protocols.

# Aspects of Web Services (1)

- Interoperability
  - A Web Service (WS) describes standards required to format and process messages and standards for service interfaces.
  - Two most popular approaches:
    - SOAP / WSDL-based Web services
    - RESTful (REpresentational State Transfer) services
  - Basic form:
    - WSs describe primitives required to exchange documents (data) between service consumers and service providers.
    - Standards for transactional, reliable, and secure services for SOAP/WSDL are available.
    - WS Platform Architecture: modular composition.

# Aspects of Web Services (2)

- SOAP
  - SOAP is a messaging protocol and WSDL (Web Services Description Language) is an interface description language.
  - Defines an XML-based message format, specifies processing rules for messages, describes conventions, and allows mappings to different Internet transport protocols (including HTTP).
  - Consisting of two parts:
    - The SOAP envelope: contains the two elements SOAP header (information such as routing and security details for authentication and authorisation)
    - The SOAP body: accommodates the actual message and may include information on the data exchange or instructions for a remote procedure call.

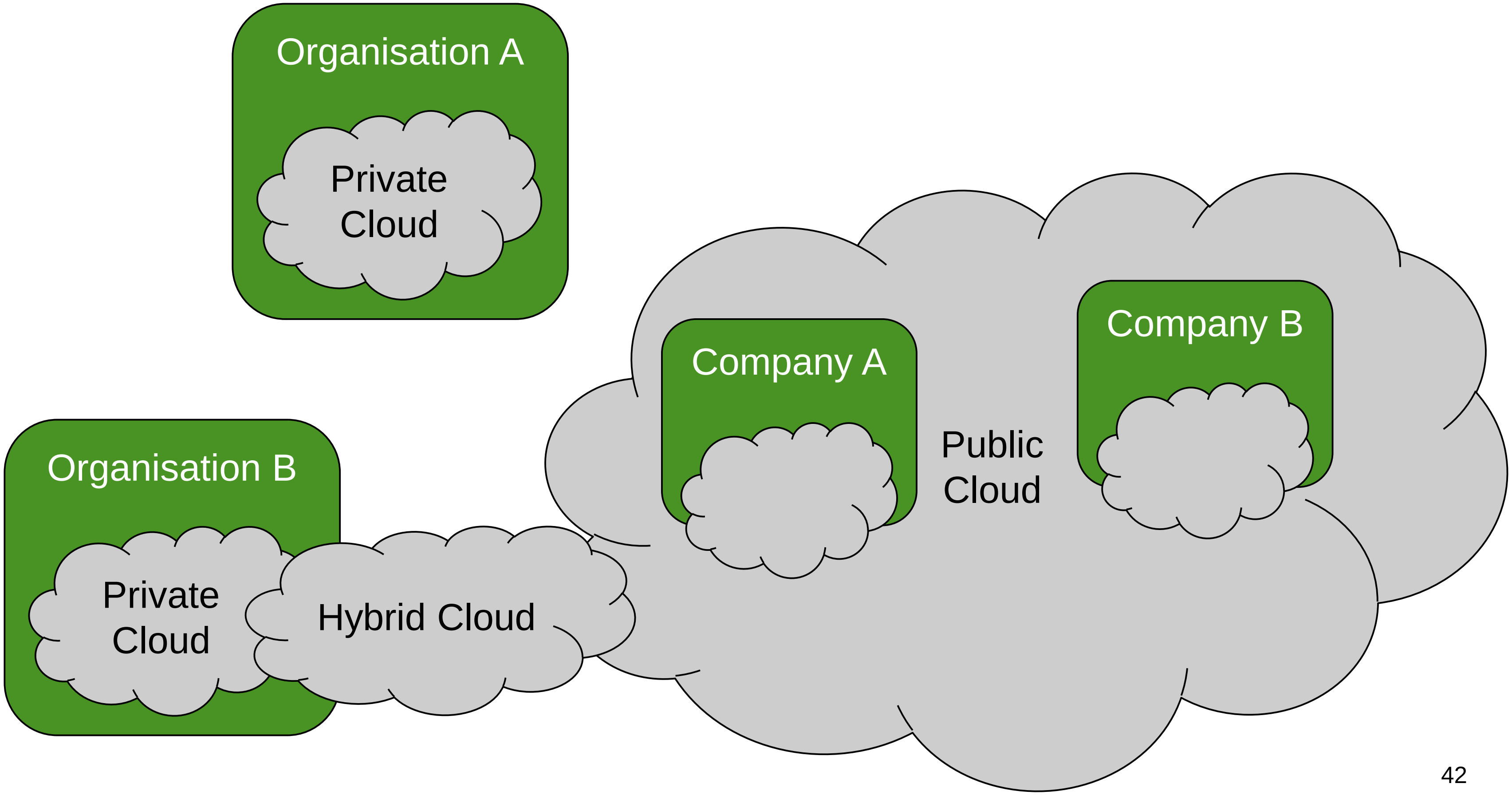
# Aspects of Web Services (3)

- REST
  - REST describes a style of software architecture, which is built on top of HTTP.
  - RESTful services: only invoked from uniform HTTP interfaces.
  - Uses HTTP semantics.
    → Stateless communication (i.e. the server does not maintain any client state information).
  - Features point-to-point connections, any necessary information is encoded in the message itself.
  - This facilitates, e.g. the interposition of caches or the replication of servers.

- Introduction
- Basics
- **Cloud architectures**
- Cloud offerings
- Conclusion and Literature

# Classification of Clouds

- Private, public and hybrid clouds



# Classification of Clouds (2)

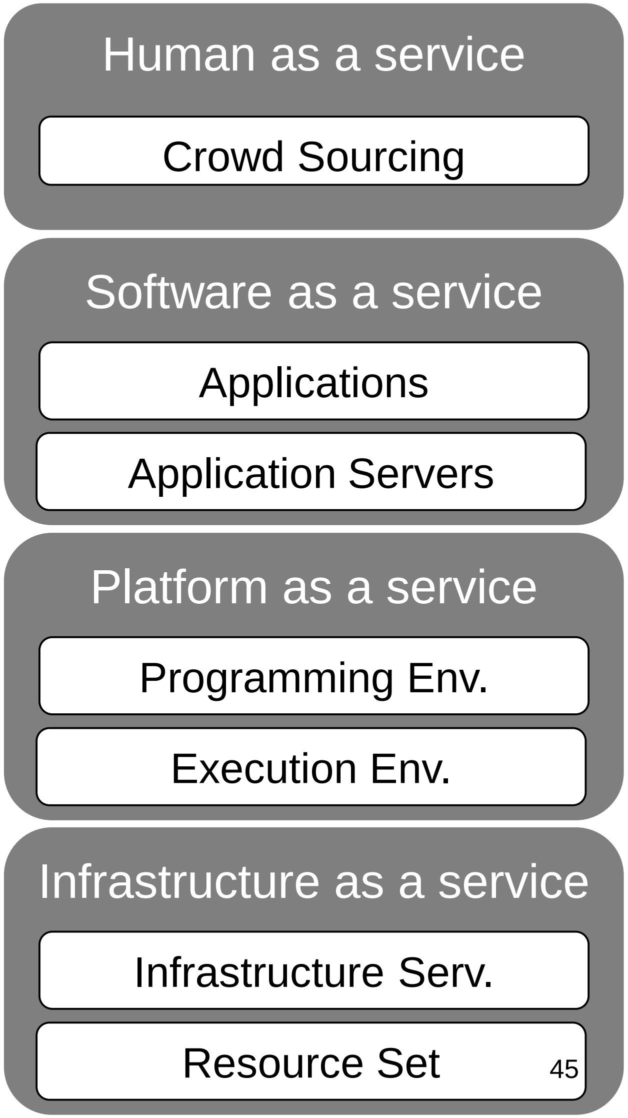
- Public Clouds (External Clouds)
  - – Provider and customer do not belong to the same authority / organisation.
  - – Cloud is public and accessible (usually) via Web-Portals.
  - – Customers specify desired scope of services .
  - – No superior frame agreement.
  - – Contract is specified by the scope agreement.
  - – Billing is done according to actual resource usage and timing.

# Classification of Clouds (3)

- Private Clouds (Internal Clouds / IntraClouds)
  - Advantage: Security issues
  - Although belonging to the same organisation, Private Clouds are addressed similar to Public Clouds
    → Easier to switch between and to combine clouds
- Hybrid Clouds
  - Combination of Private and Public Clouds
  - Typical usage:
    - Outsourcing of certain functionalities
    - Peaks in resource demands are covered externally

# Technical Cloud Architectures

- Cloud Computing allows for providing and using varying
  - IT-infrastructures,
  - platforms and
  - applications as Web-Services.
- Distinguish between four types:
  - Human as a service (HuaaS)
  - Software as a service (SaaS)
  - Platform as a service (PaaS)
  - Infrastructure as a service (IaaS)
- Can be seen as a layered architecture where each layer makes use of the subjacent one (and its interfaces)



# Infrastructure as a Service

- Provides users an abstract view on the hardware, i.e.: Computer, storage, networks
  - Resource Set (RS) is a user interface for the management of resources in order to allocate a subset for the customer‘s demand
  - Physical Resource Set (PRS) are based on a proprietary physical hardware and offer it to the user
  - Virtual Resource Set (VRS) are based on virtualisation technologies (like XEN) and offer virtual instances to the user
  - Infrastructure Service Set (ISS) have an own application focus, e.g.:
    - Hadoop Map Reduce (computation tasks)
    - Amazon S3, Zumodrive, Dropbox (storage)
    - OpenFlow (networks)

# Platform as a Service

- Addresses mainly developers, two parts:
  - Programming Environments (PE):
    Extend existing programming languages (e.g. class libraries).
  - Execution Environments (EE):
    Application (from PE) is processed in decoupled runtime environments.
- An EE is typically combined with at least one PE
- Examples:
  - PE: Django Framework (Python-based framework for w.a.)
  - PE: SUN Caroline (Development and execution of w.a.)
  - EE: Google App Engine (scalable runtime environment w.a.)
  - EE: Microsoft Azure (development and execution environment for Microsoft applications)

w.a.: web applications

# Software as a Service

- Directly addressing customers and their demands
  - Users do not have to install software
- Distinguish between two types:
  - Application Services: A specific service can be directly used
    → e.g. Google‘s Maps Service
  - Applications: Complex and integrated applications, where a combination of individual application services is needed
    → e.g. integration of social networks using OpenSocial
- Further examples:
  - Google Docs: Online Office Suite
  - Microsoft Office Live: Online Office Suite
  - Salesforce.com: Platform where application providers can develop and offer chargeable services

# Human as a Service

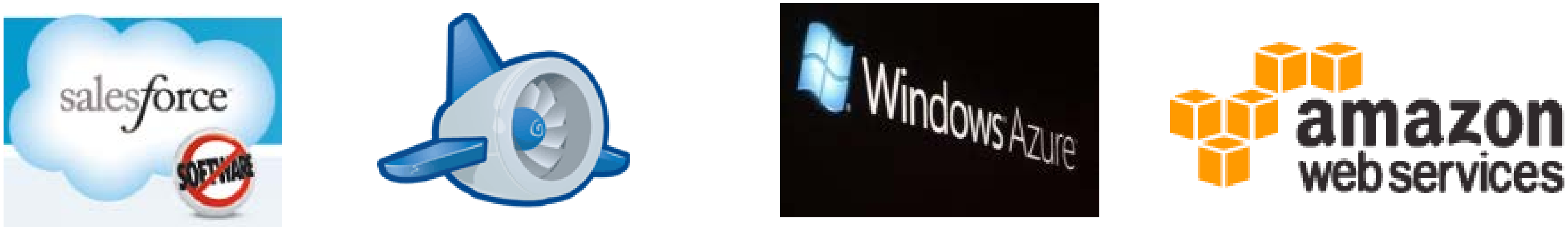
- Demonstrates the generic character of the Cloud Computing approach: it is not restricted to IT-services!
- From a technical perspective, humans are mainly needed as experts, where technical systems are characterised by missing knowledge.
  - Or: solving a task needs creativity.
- Most prominent application: Crowd-Sourcing
  - A group of humans processes tasks of varying complexity and size over the Internet for a certain client.
  - E.g.: Amazon‘s Mechanical Turk provides an interface to assign (mostly fine-grained) tasks to humans, which in turn receive a corresponding reward.
  - E.g.: Prediction markets (wheather, voting results)

- Introduction
- Basics
- Cloud architectures
- Cloud offerings
- Conclusion and Literature

# The Cloud Computing Market

- Confusing and unclear market
  - Lots of small vendors:

  - And the big four:



Google AppEngine

See: http://cloud-computing.findthebest.com/

# The Cloud Computing Market (2)

- Let‘s consider three (most popular) examples:
  - Amazon WebServices (AWS)
    - Amazon Elastic Computing Cloud (EC2)
    - Amazon Simple Storage Service (S3)
    - Amazon Simple Queue Service (SQS)
    - Amazon SimpleDB
  - Google AppEngine (GAE)
  - Salesforce.com (SFC)

- Amazon Elastic Computing Cloud (EC2)
  - Core element of Amazon‘s cloud offerings
  - Process:
    1. Create Virtual Server
       (a) from predefined Amazon Machine Images (AMI) - including OS and software or
       (b) self-defined.
    2. Choose resource configuration (e.g. processor power, RAM, disk space) and availability zone (which services / processes have access?).
    3. Generate keys: identification is done using public-key approach – public key is assigned to user account, while private key remains in user’s working environment.

# Amazon WebServices (2)

– Process (ctd.):

4. Activate instance: VM is initialised based on previous configuration. It has a public (accessible from the Internet) and a private (accessible from inside the Amazon Cloud) IP address – both addresses are dynamic.
   → But: you can book static Elastic IP addresses.
5. Define security group: Due to security reasons, new instances are not accessible from outside at the beginning. They have to be added to a security group, for which access policies are configured.
6. Persistent state storage: All information gets lost after finishing the instance. Hence, data has to be stored persistently using external sources like Amazon S3.

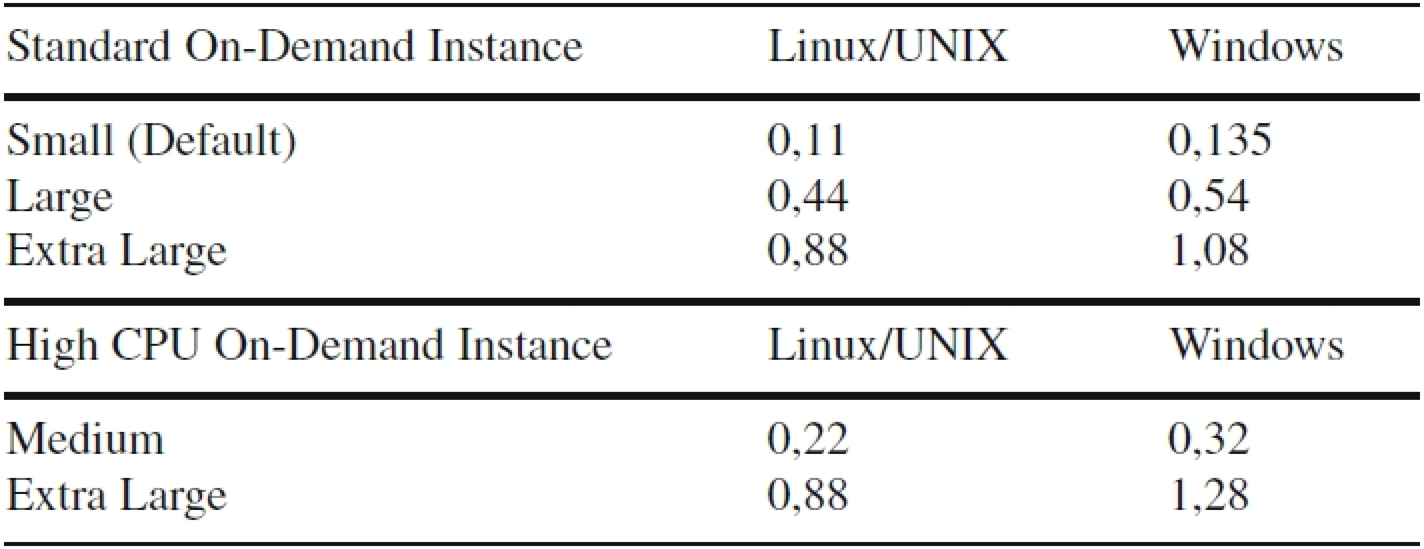
# Amazon WebServices (3)

- Process (ctd.):

7. Create own AMIs: Start with a predefined image (e.g. OS) and add needed applications and services. Afterwards, this customised image is registered with Amazon and accessible for further usage.
8. Tool support: All previous steps can be performed using the command prompt (console). For those preferring more comfortable user interfaces, tools are available:
   - AWS Management Console
   - AWS Toolkit for Eclipse
   - ElasticFox (Plugin for Firefox)

# Amazon WebServices (4)

- Prices for Amazon EC2 on-Demand instances (in $/h) – as provided in 12/2012:



| Standard On-Demand Instance | Linux/UNIX | Windows |
|---|---|---|
| Small (Default) | 0,11 | 0,135 |
| Large | 0,44 | 0,54 |
| Extra Large | 0,88 | 1,08 |
| **High CPU On-Demand Instance** | **Linux/UNIX** | **Windows** |
| Medium | 0,22 | 0,32 |
| Extra Large | 0,88 | 1,28 |

- Example: *High CPU On-Demand Instance Extra Large* is a 64-bit platform with 20 ECUs (2,5 ECUs for each 8 virtual cores), 7 GB RAM and 1690 GB storage.

# Amazon WebServices (5)

- Amazon Simple Storage Service (S3)
  - Cloud-based mass-storage.
  - Web objects of up to 5GB, directly accessible using a unique ID.
  - Access using SOAP or REST API.
    Example: `s3cmd mb s3://Bucket` (creates new bucket)
- Amazon Simple Queue Service (SQS)
  - Message queue
  - Sender (publisher) adds message to SQS queue, registered recipient reads and processes this message
  - De-coupling of several processes and parallelisation become possible
- Amazon SimpleDB
  - Not a complex data base schema or characterised by transactual properties
  - Simple structured, highly reliable data storage

# Google AppEngine

- Google AppEngine
  - PaaS product including programming environment, tool support, and runtime environment
  - Goal: develop web applications for the scalable Google infrastructure with focus on functionality
    → Developer does not have to care about administration (servers).
  - Initially, Java and Python were supported. Now many more.
  - Web application can use standard services:
    - Storage of data
    - Email integration
    - User account services
    - Temporary storage of data (caching)
    - Shortcuts to other applications (e.g. using http)
    - Google services (e.g. calendar)

# Salesforce

- Salesforce.com
  - Market leader for Customer Relationship Management (CRM) Software
  - Serves as SaaS and PaaS provider
  - Portfolio contains four main parts:
    1. Salesforce: SaaS for CRM. This is a web-based solution for marketing, customer service, partner management and distribution. Does not need a local installation.
    2. Force.com: PaaS allowing customers and others (i.e. independent software vendors) to develop own web-based applications and to run them using the Salesforce infrastructure.
    3. AppExchange: Distribute Force.com applications.
    4. DeveloperForce: Organised and moderated developer community for Force.com.

- Introduction
- Basics
- Cloud architectures
- Cloud offerings
- **Management**
- Open Source Cloud Stack
- Conclusion and Literature

# Cloud Management

- Management procedures for Cloud Computing
  - Vital for both, operators and users
  - Services must be described, provisioned, and billed.
  - Automated processes are used to achieve the required service scalability and reliability
  - Security issues are important when transferring services from the local context into the public cloud
    → Following aspects have to be covered:
    1. Service Level Agreements
    2. Management services and tools
    3. Security management
    4. Risk management
    5. Legal compliance

# Service Level Agreements

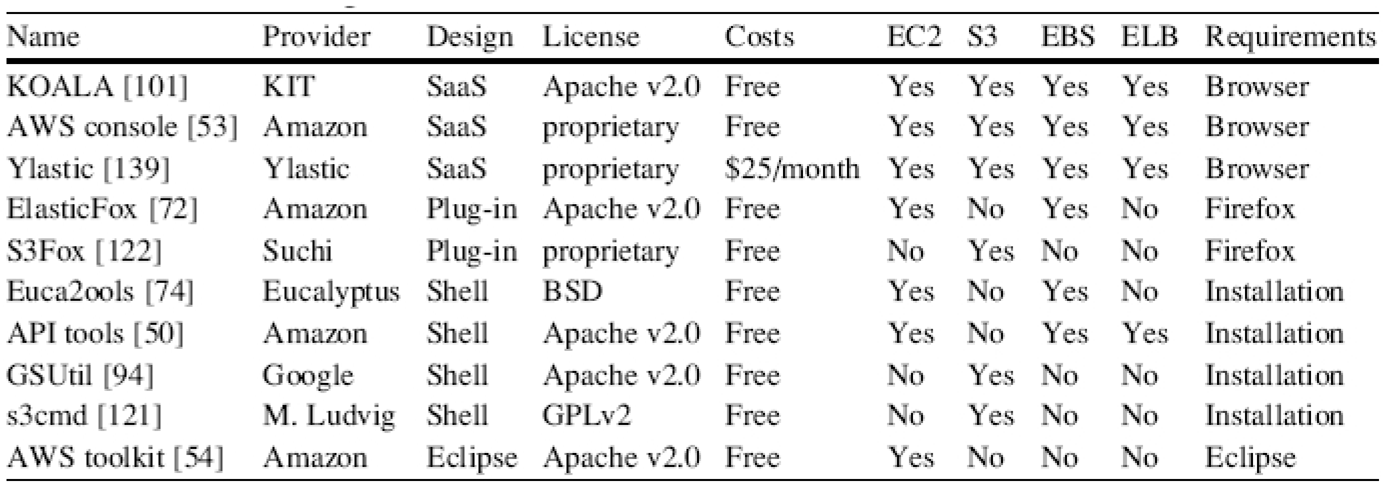
- Service Level Agreements (SLAs)
  - Agreement between service provider and consumer.
  - Related to the service level (quality of service).
  - Not always a formal contract, can be informal as well.
  - SLA implies a mutual agreement with respect to security, priorities, responsibilities, guarantees, and billing modalities.
  - SLA also specifies metrics such as availability, throughput, response times, and others.
  - SLA always consider the output side, i.e. they are drafted from the service consumer’s perspective.
  - Possibility to agree on different quality levels (Basic, Silver, Gold).

- SLAs for Cloud Computing
  - Current cloud offerings:
    - Agreeing and monitoring specific SLAs is only possible in a very basic way.
    - These offerings are usually made on a best effort basis.
    - Failures or service disruptions: provider issues a corresponding credit note.
  - Cloud architecture:
    - Developers should insert a layer into the cloud stack on which both aspects (i.e. service agreement and service monitoring) will be dealt with.
    - SLA@SOI is a project supported by the European Commission in the EU Seventh Framework Programme. It examines the aspects of multi-level SLAs in a market with competing offerings.

- Cloud service‘s life cycle
  1. Service provider defines the scope and quality of the services and describes their properties in a service catalogue.
  2. Consumers select the desired services from the catalogue and instantiate them as required.
  3. SLAs need to be taken into account and monitored.
  4. End of the utilisation period: service orders are closed, service modules are dissociated and resources are reset.
  5. Accounting procedure adds up the usage of all resources.<br>→ Track current status in a fine-grained and time-resolved manner.
  6. Consumers are billed either periodically or each time the costs incurred exceed a certain threshold (i.e. by credit card).

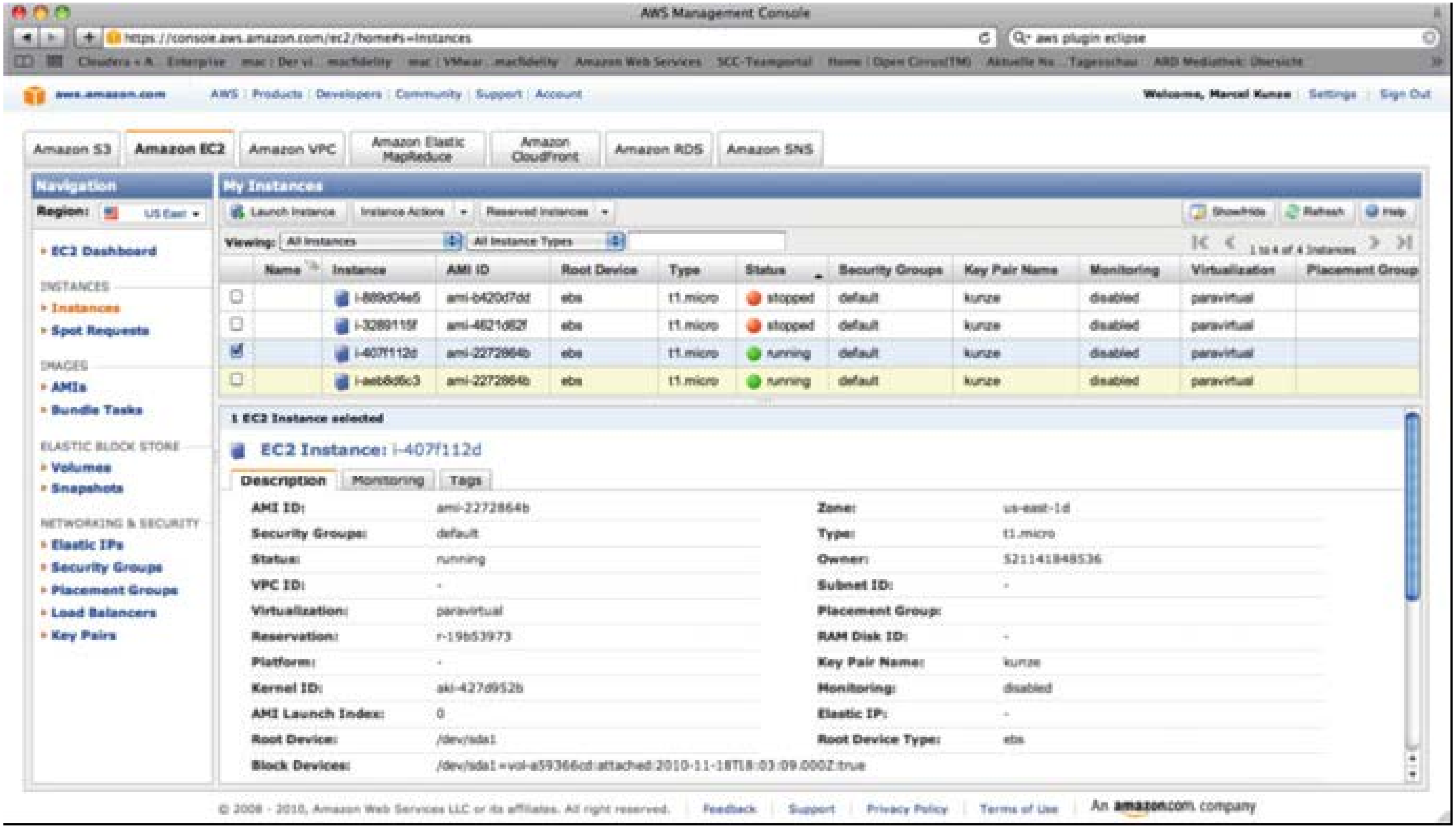
# Management Services and Tools

- Management Services and Tools
  - Cloud providers offer a broad range of tools.



| Name | Provider | Design | License | Costs | EC2 | S3 | EBS | ELB | Requirements |
|---|---|---|---|---|---|---|---|---|---|
| KOALA [101] | KIT | SaaS | Apache v2.0 | Free | Yes | Yes | Yes | Yes | Browser |
| AWS console [53] | Amazon | SaaS | proprietary | Free | Yes | Yes | Yes | Yes | Browser |
| Ylastic [139] | Ylastic | SaaS | proprietary | $25/month | Yes | Yes | Yes | Yes | Browser |
| ElasticFox [72] | Amazon | Plug-in | Apache v2.0 | Free | Yes | No | Yes | No | Firefox |
| S3Fox [122] | Suchi | Plug-in | proprietary | Free | No | Yes | No | No | Firefox |
| Euca2ools [74] | Eucalyptus | Shell | BSD | Free | Yes | No | Yes | No | Installation |
| API tools [50] | Amazon | Shell | Apache v2.0 | Free | Yes | No | Yes | Yes | Installation |
| GSUtil [94] | Google | Shell | Apache v2.0 | Free | No | Yes | No | No | Installation |
| s3cmd [121] | M. Ludvig | Shell | GPLv2 | Free | No | Yes | No | No | Installation |
| AWS toolkit [54] | Amazon | Eclipse | Apache v2.0 | Free | Yes | No | No | No | Eclipse |

# Example: AWS Management Console



AWS Management Console

https://console.aws.amazon.com/ec2/home#s=Instances

aws.amazon.com | AWS | Products | Developers | Community | Support | Account | Welcome, Marcel Kunze | Settings | Sign Out

Amazon S3 | Amazon EC2 | Amazon VPC | Amazon Elastic MapReduce | Amazon CloudFront | Amazon RDS | Amazon SNS

**Navigation**

Region: US East

- EC2 Dashboard

INSTANCES

- Instances
- Spot Requests

IMAGES

- AMIs
- Bundle Tasks

ELASTIC BLOCK STORE

- Volumes
- Snapshots

NETWORKING & SECURITY

- Elastic IPs
- Security Groups
- Placement Groups
- Load Balancers
- Key Pairs

**My Instances**

Launch Instance | Instance Actions | Reserved Instances | Show/Hide | Refresh | Help

Viewing: All Instances | All Instance Types | 1 to 4 of 4 Instances

| | Name | Instance | AMI ID | Root Device | Type | Status | Security Groups | Key Pair Name | Monitoring | Virtualization | Placement Group |
|---|---|---|---|---|---|---|---|---|---|---|---|
| □ | | i-889d04e5 | ami-b420d7dd | ebs | t1.micro | ● stopped | default | kunze | disabled | paravirtual | |
| □ | | i-3289115f | ami-4621d62f | ebs | t1.micro | ● stopped | default | kunze | disabled | paravirtual | |
| ☑ | | i-407f112d | ami-2272864b | ebs | t1.micro | ● running | default | kunze | disabled | paravirtual | |
| □ | | i-aeb8d6c3 | ami-2272864b | ebs | t1.micro | ● running | default | kunze | disabled | paravirtual | |

1 EC2 Instance selected

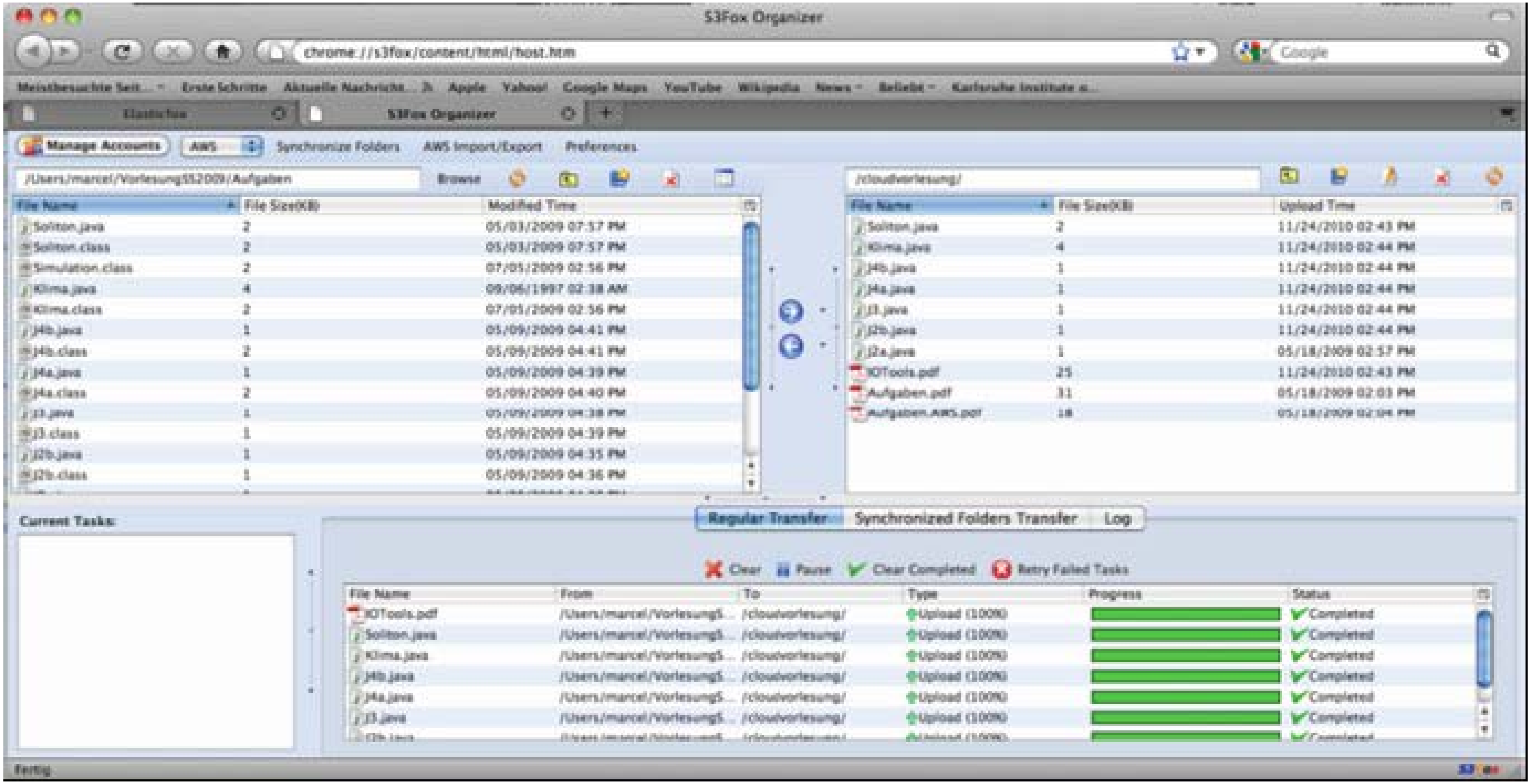
**EC2 Instance:** i-407f112d

Description | Monitoring | Tags

| | | | |
|---|---|---|---|
| **AMI ID:** | ami-2272864b | **Zone:** | us-east-1d |
| **Security Groups:** | default | **Type:** | t1.micro |
| **Status:** | running | **Owner:** | 521141848536 |
| **VPC ID:** | - | **Subnet ID:** | - |
| **Virtualization:** | paravirtual | **Placement Group:** | |
| **Reservation:** | r-19b53973 | **RAM Disk ID:** | - |
| **Platform:** | - | **Key Pair Name:** | kunze |
| **Kernel ID:** | aki-427d952b | **Monitoring:** | disabled |
| **AMI Launch Index:** | 0 | **Elastic IP:** | - |
| **Root Device:** | /dev/sda1 | **Root Device Type:** | ebs |
| **Block Devices:** | /dev/sda1=vol-a59366cd:attached:2010-11-18T18:03:09.000Z:true | | |

© 2008 - 2010, Amazon Web Services LLC or its affiliates. All right reserved. | Feedback | Support | Privacy Policy | Terms of Use | An amazon.com. company

# Example: S3Fox



/Users/marcel/VorlesungSS2009/Aufgaben

| File Name | File Size(KB) | Modified Time |
|---|---|---|
| Soliton.java | 2 | 05/03/2009 07:57 PM |
| Soliton.class | 2 | 05/03/2009 07:57 PM |
| Simulation.class | 2 | 07/05/2009 02:56 PM |
| Klima.java | 4 | 09/06/1997 02:38 AM |
| Klima.class | 2 | 07/05/2009 02:56 PM |
| J4b.java | 1 | 05/09/2009 04:41 PM |
| J4b.class | 2 | 05/09/2009 04:41 PM |
| J4a.java | 1 | 05/09/2009 04:39 PM |
| J4a.class | 2 | 05/09/2009 04:40 PM |
| J3.java | 1 | 05/09/2009 04:38 PM |
| J3.class | 1 | 05/09/2009 04:39 PM |
| J2b.java | 1 | 05/09/2009 04:35 PM |
| J2b.class | 1 | 05/09/2009 04:36 PM |

/cloudvorlesung/

| File Name | File Size(KB) | Upload Time |
|---|---|---|
| Soliton.java | 2 | 11/24/2010 02:43 PM |
| Klima.java | 4 | 11/24/2010 02:44 PM |
| J4b.java | 1 | 11/24/2010 02:44 PM |
| J4a.java | 1 | 11/24/2010 02:44 PM |
| J3.java | 1 | 11/24/2010 02:44 PM |
| J2b.java | 1 | 11/24/2010 02:44 PM |
| J2a.java | 1 | 05/18/2009 02:57 PM |
| IOTools.pdf | 25 | 11/24/2010 02:43 PM |
| Aufgaben.pdf | 31 | 05/18/2009 02:03 PM |
| Aufgaben.AWS.pdf | 18 | 05/18/2009 02:04 PM |

| File Name | From | To | Type | Progress | Status |
|---|---|---|---|---|---|
| IOTools.pdf | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| Soliton.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| Klima.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| J4b.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| J4a.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| J3.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |
| J2b.java | /Users/marcel/VorlesungS... | /cloudvorlesung/ | Upload (100%) | 🟩 | ✓ Completed |

- Security Management
  - Security is not only related to safely accessing resources, but also covers data privacy issues.
  - A study by Berkeley University researchers shows that, for the so-called cloud sourcing, no specific challenges or problems exist that would require measures other than those currently implemented.
  - With respect to security, the same safety objectives apply as those common to the operation of services in a local data centre:
    - Confidentiality
    - Integrity
    - Availability
    - Authenticity
    - Accountability
    - Pseudonymity

# Security and Risk Management

- Security Management
  - IaaS: Highest flexibility, the customer is responsible for security.
  - PaaS: Medium flexibility, both the customer and the provider are responsible for security.
  - SaaS: Lowest flexibility, the provider is responsible for security.
- Risk Management
  - When data has been transferred to the cloud, there is always a risk that access might no longer be available in case of a service disruption or bankruptcy of the provider.
  - These issues are the same as with any outsourcing process and can be handled by concluding suitable contracts or SLAs.
  - Further risk: dependency on the proprietary technology and interfaces of a certain provider (vendor lock-in).
    → Minor problem for IaaS and a major one with PaaS and SaaS.

# Legal Compliance

- Legal Compliance
  - Neither cloud service offerings nor their use may infringe laws, social values, morals, or ethics.
  - This is ensured by the concept of compliance.
  - The same laws are applicable to Cloud Computing as to renting (in case of paid services) or lending (with respect to unpaid services).
  - Important: The geographical location of the cloud provider is decisive in the determination of the laws that will apply to the data stored.
    → For this reason, e.g. Amazon offers cloud services for different regions, such as the U.S., Europe, or Asia.

- Introduction
- Basics
- Cloud architectures
- Cloud offerings
- Conclusion and Literature

# Conclusion

- Cloud Computing (CC)
  - Still a very young and dynamic field with active industry.
  - Lots of industry organisations (even players of the public sector) are taking on CC today, either as a provider or as a consumer.
  - Primarily US-based companies (e.g. Amazon, Google, or Microsoft) are currently shaping the cloud services market.
    → But many smaller companies also show a lot of commitment in this field.
  - CC is a disruptive technology which has the potential to change our understanding of how to provision and leverage IT services in a fundamental and sustainable way.
  - The effects might even be comparable to the introduction of the personal computer some 25 years ago.

- Baun C., Kunze M., Nimis J., Tai S., ”*Cloud Computing”, Springer 2011, ISBN-13: 978-3642209161.*
- Barrie Sosinsky: “*Cloud Computing Bible*”, John Wiley & Sons Publishers 2011, ISBN-13: 978-0470903568.