Recap on Cluster and Cloud Computing – Lecture 12.2

Professor Richard O. Sinnott
Director, eResearch
University of Melbourne
rsinnott@unimelb.edu.au

Course Contents

- Lectures 1 & 2 – 2nd March
  - Information Session & How we got here (Distributed Systems, Grid…)
    - Richard Sinnott
- Lectures 3 & 4 – 9th March
  - Domain Drivers – tour of some big data projects
    - Richard Sinnott
- Lectures 5 & 6 – 16th March
  - Parallel Systems, Distributed Computing and HPC/HTC
    - Richard Sinnott
- Lectures 7 & 8 – 23rd March
  - HPC @ UniMelb and Practicalities of HPC/HTC
    - Richard Sinnott, Lev Lafayette & Farzad Khodadadi
    - Linux and HPC practicalities
    - Using mpi4py workshop

Programming Assignment handed out (24th Mar – due 6th April)

Course Contents

- Lectures 9 & 10 – 30th March
  - Cloud Computing – Programming Clouds: Getting to grips with NeCTAR
    - Richard Sinnott & Farzad Khodadadi
    - Introduction to Cloud Computing
    - Getting to grips with OpenStack/NeCTAR
    - Scripting for the Cloud (Introduction to Boto & Ansible demonstration) workshop

Second Programming Assignment handed out (7th April – 10th May)

- Lectures 11 & 12 – 6th April
  - Big Data and Related Technologies
    - Luca Morandini (Data Architect, AURIN)
    - Big Data V-challenges, Cap Theorem and noSQL technologies
    - CouchDB workshop
- Lectures 13 & 14 – 13th April
  - Service-oriented architectures & Other Things Needed for Assignment II
    - Farzad Khodadadi & Luca Morandini
    - SOA & SOAP vs ReST
    - Hands-on examples of coding/demonstrating SOAP, ReST
    - Code versioning systems and GitHub (Git) workshop

Easter Break 17th-21st April

Course Contents… ctd

- Lectures 15 & 16 – 27th April
  - Big Data Analytics
    - Luca Morandini
    - Big Data Technologies – Hadoop, HDFS, Spark, …
    - Hadoop cluster on Cloud and practical application workshop
- Lectures 15 & 16 – 4th May
  - Cloud Underpinnings and Other Things
    - Richard Sinnott & Farzad Khodadadi & Luca Morandini
    - Virtualisation background
    - Compare and Contrast EC2 with NeCTAR Research Cloud
    - Microhosting Environments (Docker)
    - Discussion of 2nd Programming Assignment
- Lecture 17 & 18 – 11th May
  - Over to you
    - “Some” teams randomly chosen to present their assignment II
    - 15minutes each

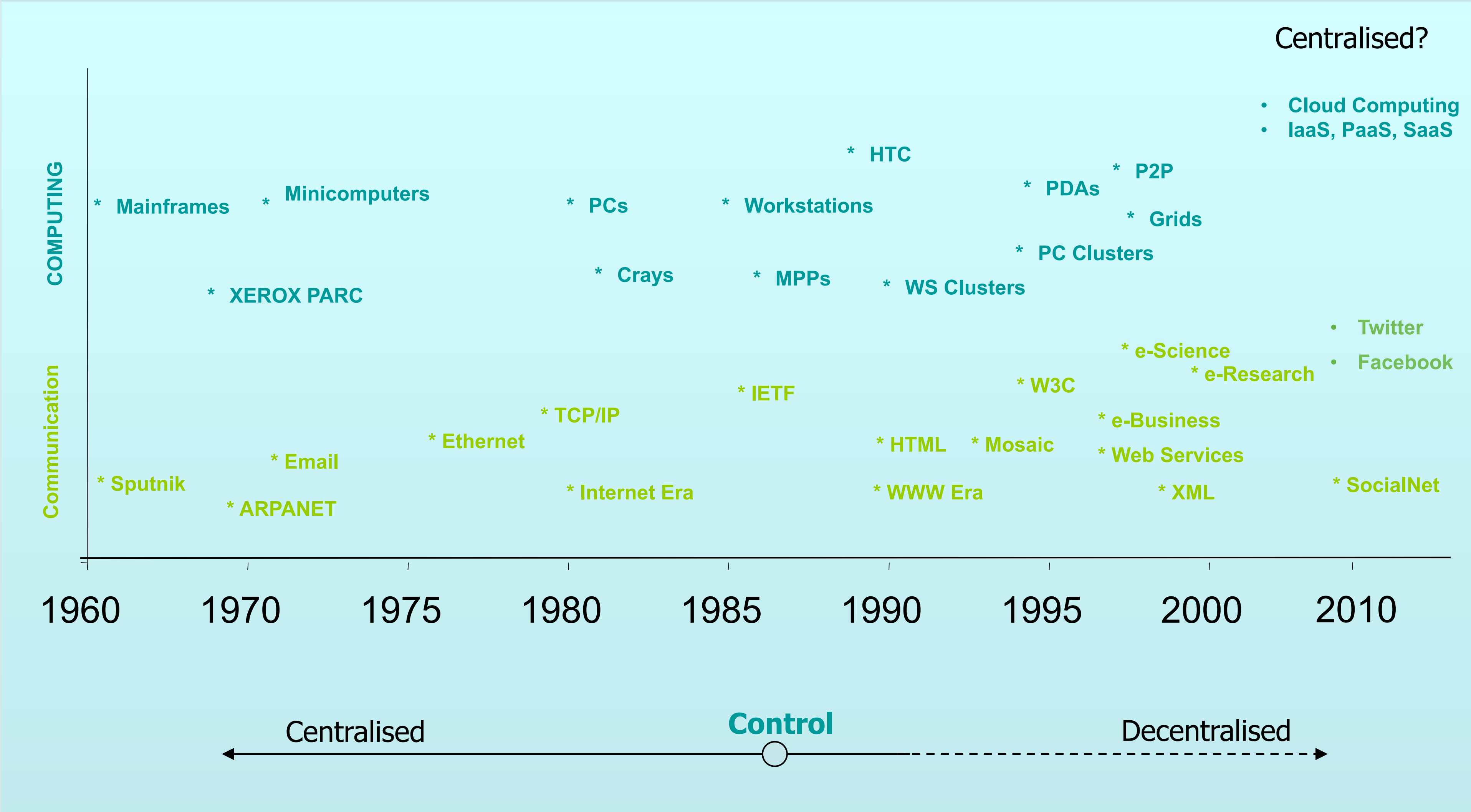
Course Contents… ctd

- Lecture 21 & 22 – 18th May
  - Over to you
    - "Some" more teams randomly chosen to present their assignment II
    - 15minutes each

- Lecture 23 & 24 – 25th May
  - Security and Clouds
  - Subject Review
  - Working Through Past Papers
  - Feedback and SES
    - Richard Sinnott

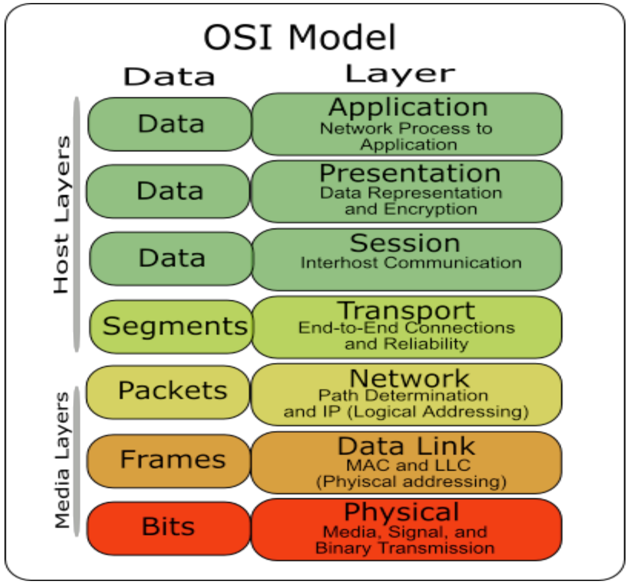
Week 1 – How we got here

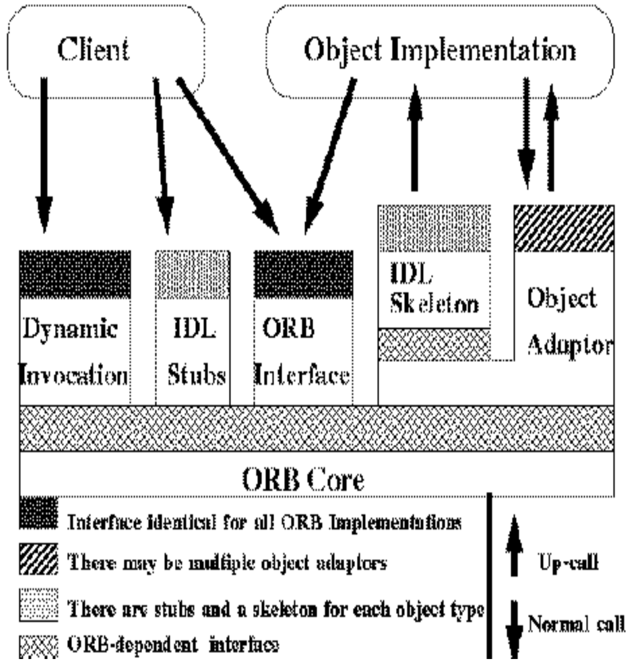
Computing and Communication Technologies (r)evolution: 1960-…!



Distributed Systems - A Very Brief History

- Once upon a time we had standards
  - With very detailed conformance, consistency and compliance demands
    - Services, protocols, inter-operability, …
- Then we had more standards
  - Open distributed processing
  - With slightly less rigorous compliance demands
    - OMG Common Object Request Broker Architecture (CORBA)
    - Distributed Computing Environment
    - Multiple technologies
      - Client server, remote procedure call, …



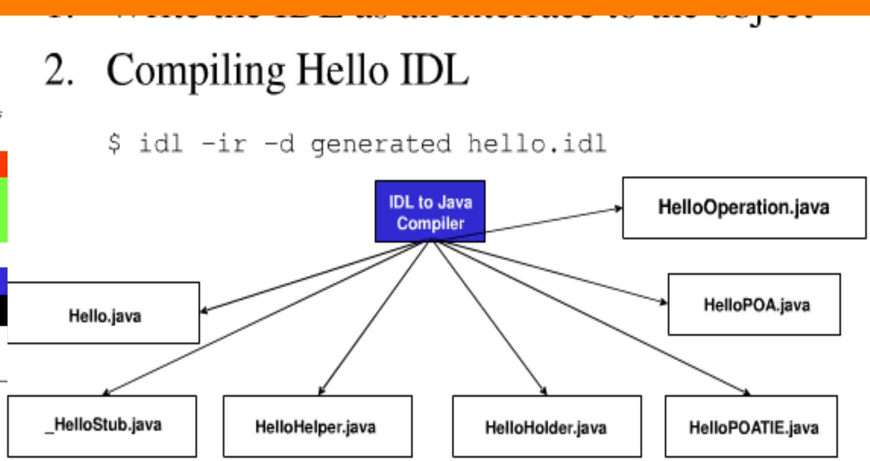


Key distributed systems focus mid-90s

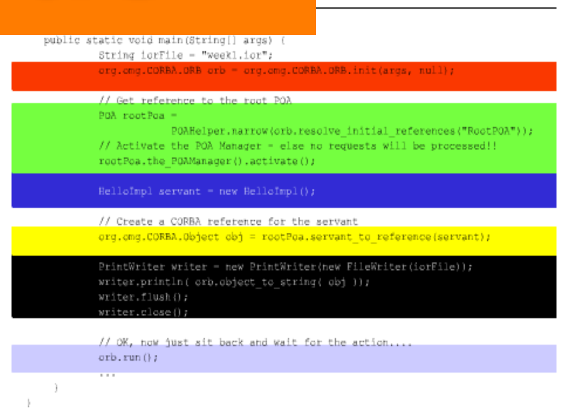
- Transparency and heterogeneity of computer-computer interactions
  - finding/discovering resources (trader!),
  - binding to resources in real time,
  - run time type checking,
  - invoking resources
  - c

Computer-computer interaction focus

Client.java

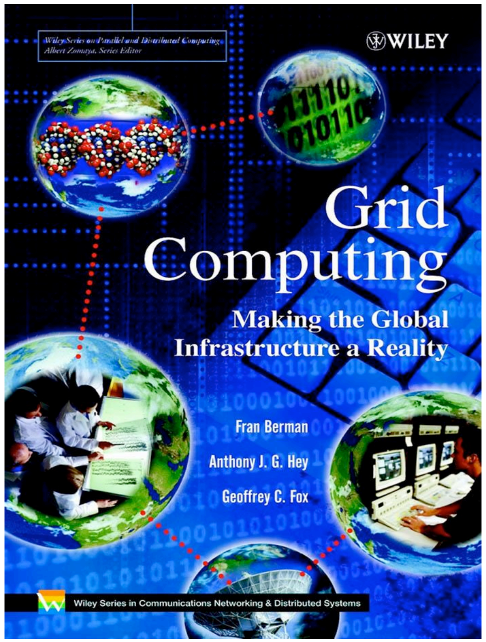


(Simplified)

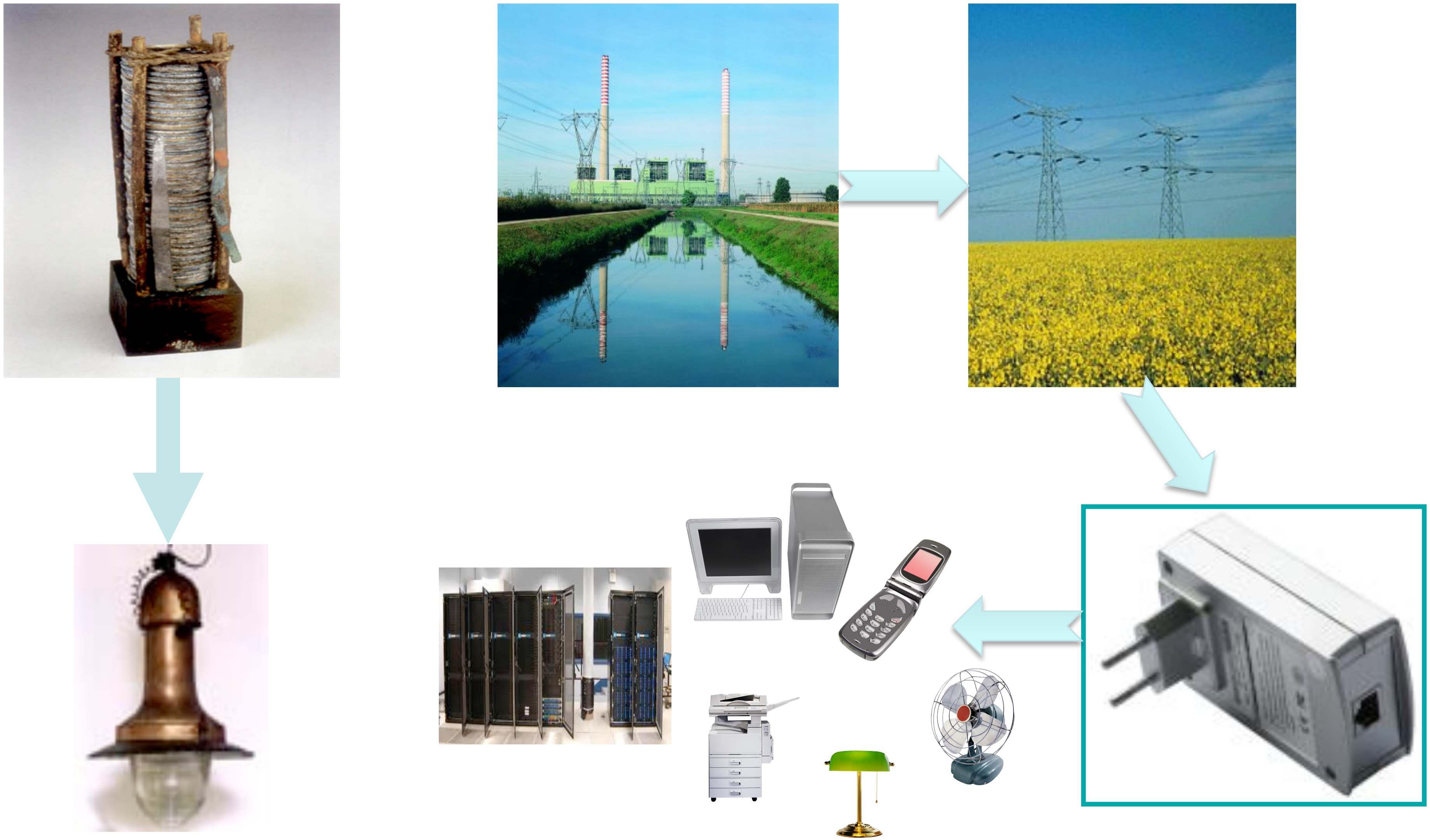


Distributed Systems History…ctd

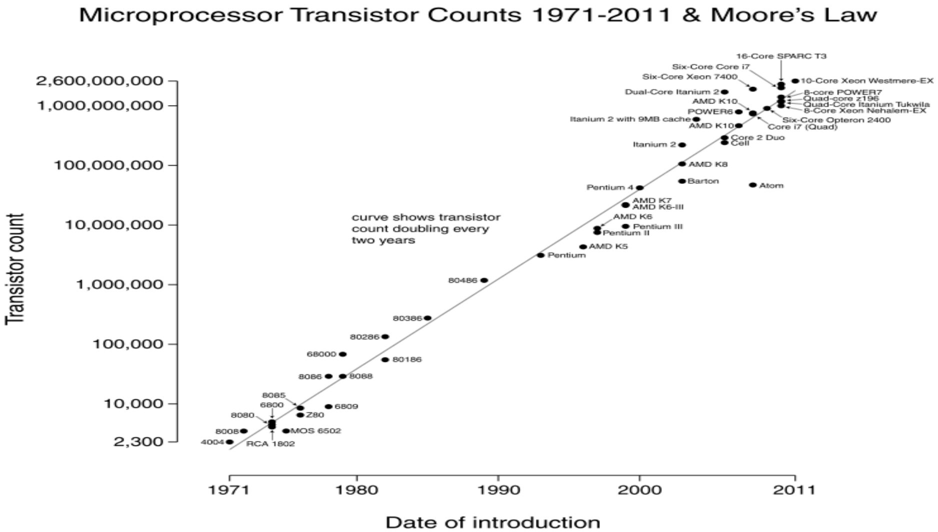
- Enter the web era
  - My first ftp 1993 put/get files to/from Australia
  - Then the web pretty much exploded
- Peer-peer processing
  - File sharing …
- Scaling of…
  - machines,
  - people,
  - domains of application
- Grid computing
  - From computer-computer focus
  - To organisation-organisation focus

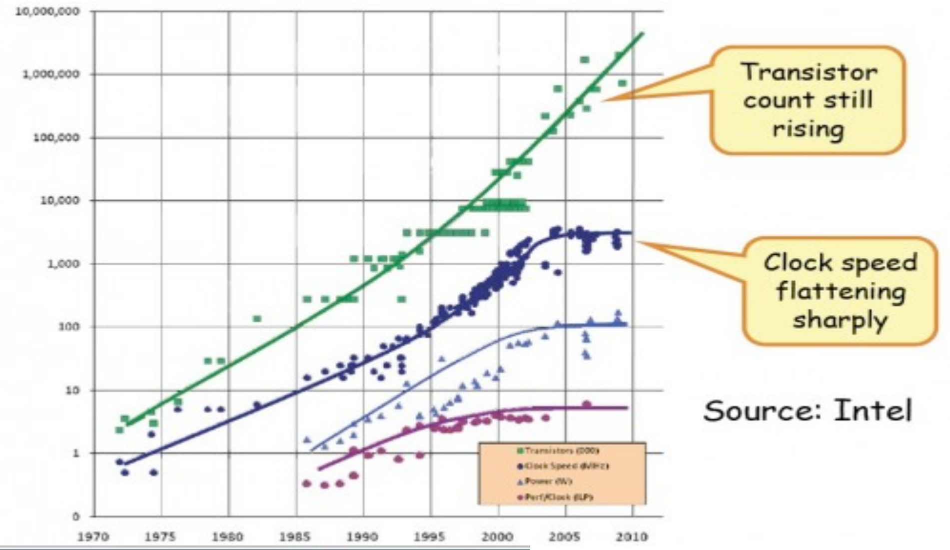


The Grid Metaphor

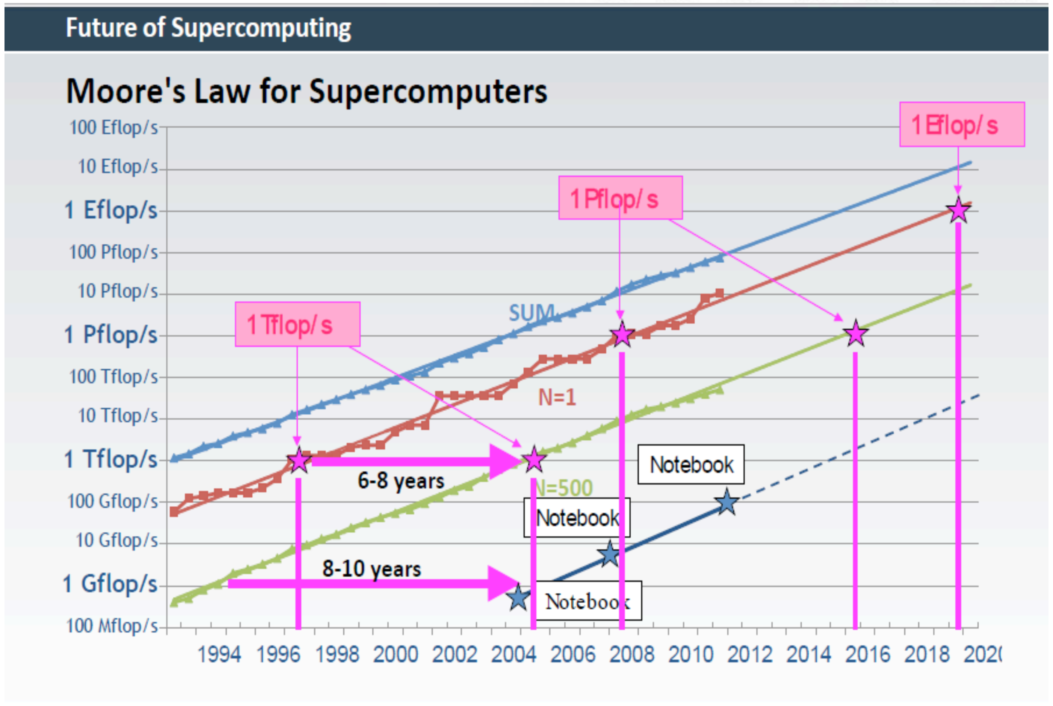


Week 2 – How we got here and domain drivers
Compute Scaling

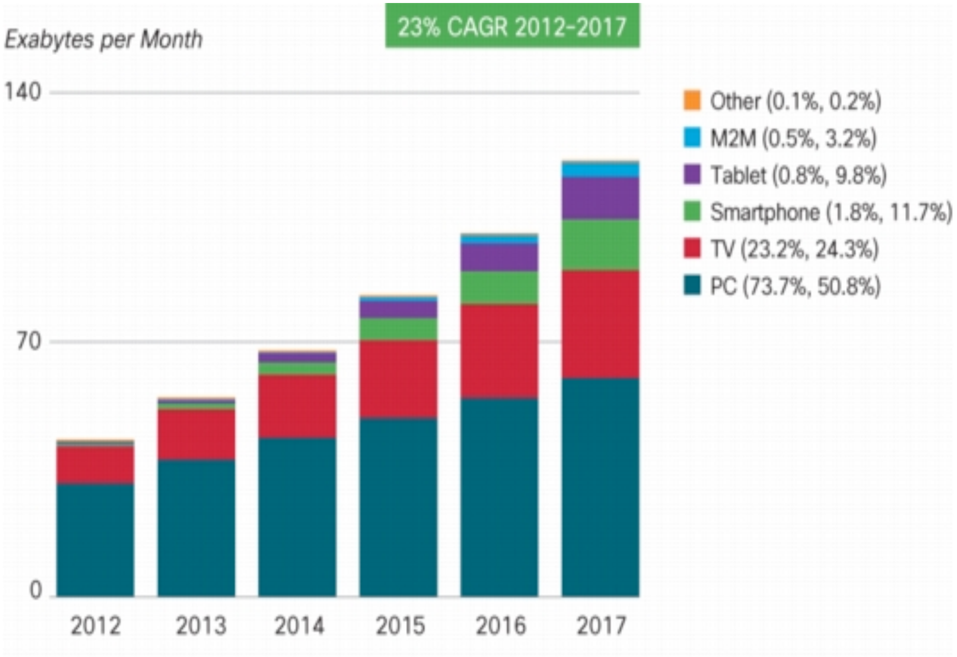




Source: Intel



Network Scaling



Source: Cisco VNI, 2013
The percentages within parenthesis next to the legend denote the relative traffic shares in 2012 and 2017.

Table 6. Table A-1 Global IP Traffic, 2012-2017

IP Traffic, 2011-2016	2012	2013	2014	2015	2016	2017	CAGR 2012-2017
By Type (PB per Month)							
Fixed Internet	31,339	39,295	47,987	57,609	68,878	81,818	21%
Managed IP	11,346	14,679	18,107	21,523	24,740	27,668	20%
Mobile data	885	1,578	2,798	4,704	7,437	11,157	66%
By Segment (PB per Month)							
Consumer	35,047	45,023	56,070	68,418	82,683	98,919	23%
Business	8,522	10,530	12,822	15,417	18,372	21,724	21%
By Geography (PB per Month)							
Asia Pacific	13,906	18,121	22,953	28,667	35,417	43,445	26%
North America	14,439	18,788	23,520	28,667	34,457	40,672	23%
Western Europe	7,722	9,072	10,568	12,241	14,323	16,802	17%
Central and Eastern Europe	3,405	4,202	5,167	6,274	7,517	8,844	21%
Latin America	3,397	4,321	5,201	5,975	6,682	7,415	17%
Middle East and Africa	701	1,049	1,483	2,013	2,659	3,465	38%
Total (PB per Month)							
Total IP traffic	43,570	55,553	68,892	83,835	101,055	120,643	23%

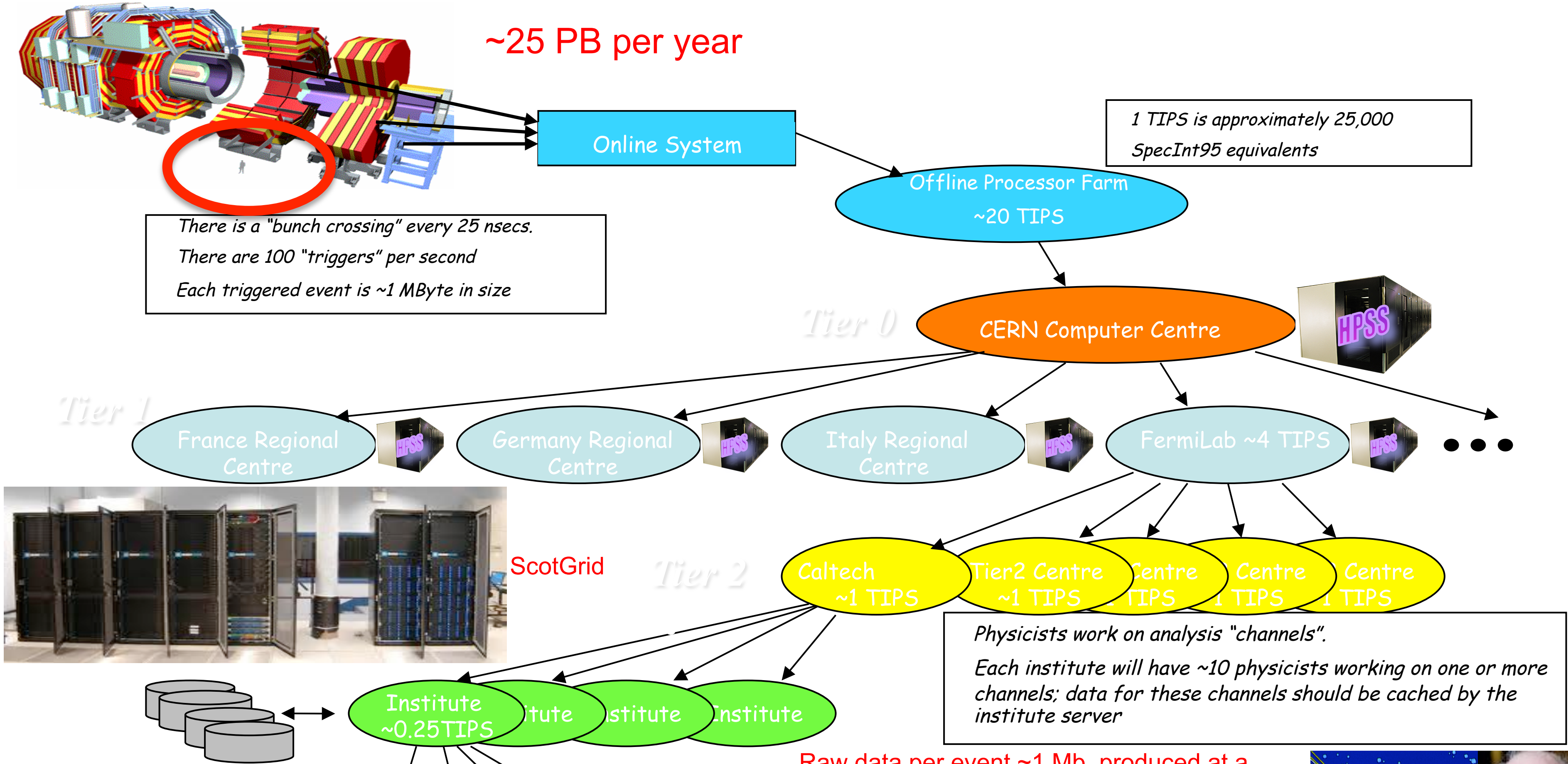
Source: Cisco VNI, 2013

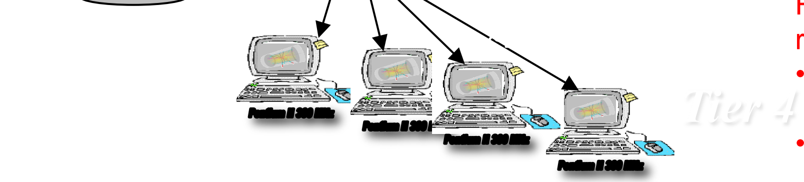
Table 1. The VNI Forecast Within Historical Context

Year	Global Internet Traffic
1992	100 Gigabytes per Day
1997	100 Gigabytes per Hour
2002	100 Gigabytes per Second
2007	2,000 Gigabytes per Second
2012	12,000 Gigabytes per Second
2017	35,000 Gigabytes per Second

Source: Cisco VNI, 2013

Compute Infrastructure for High Energy Physics

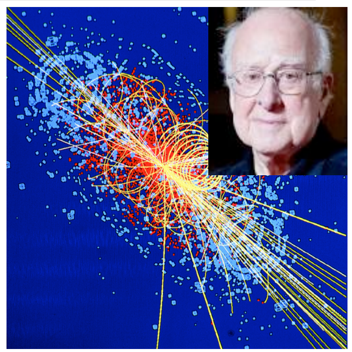




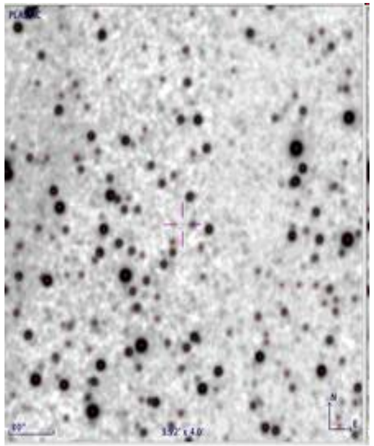
Raw data per event ~1 Mb, produced at a rate of about 40 million events per second.

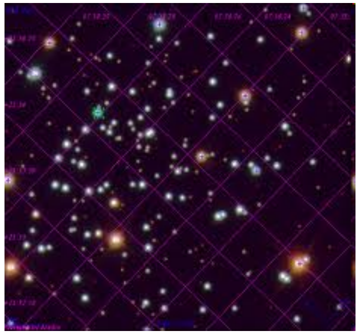
- Basic filtering reduces to 100,000 events per second
- Advanced filtering to 100 or 200 events per second

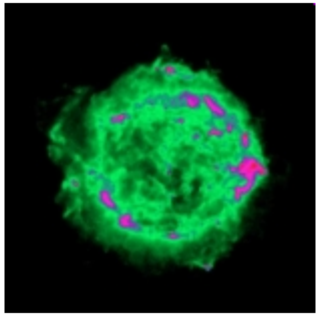
This two-stage data re-processing is performed several times a year on all data acquired since the LHC start-up.

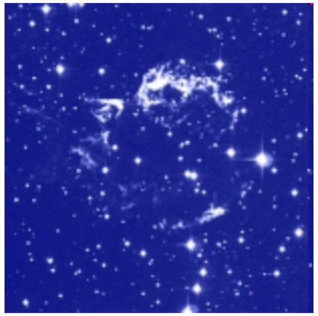


Mapping the Skies



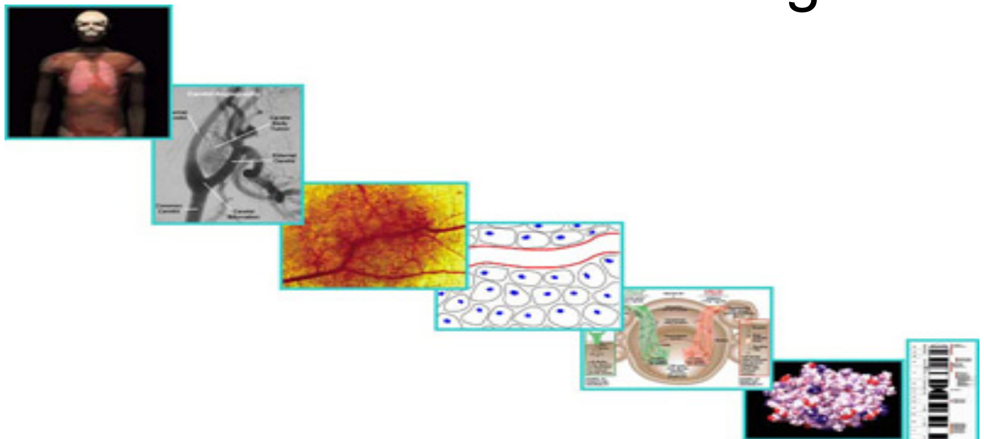




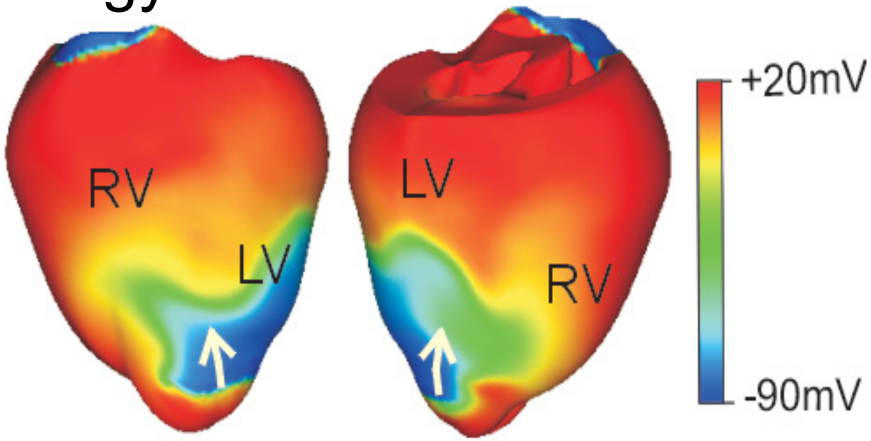


Macro-micro Simulations

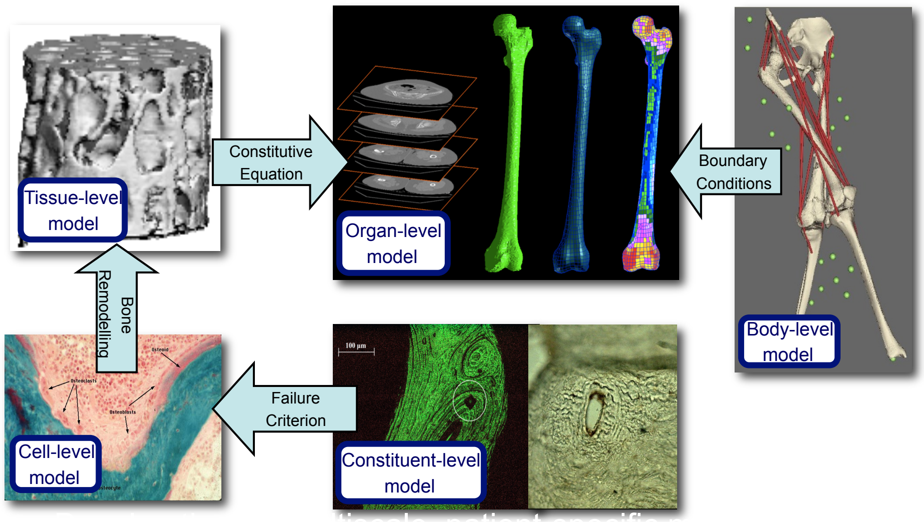
Integrative Biology

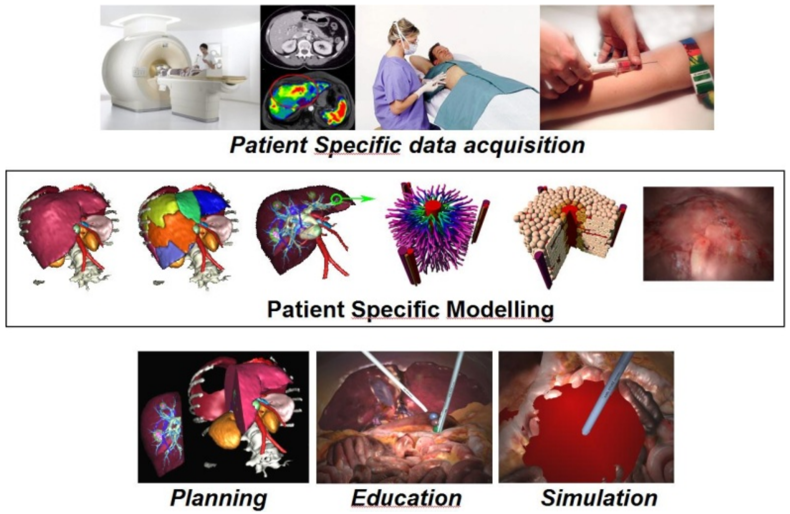


DNA to whole organism modelling

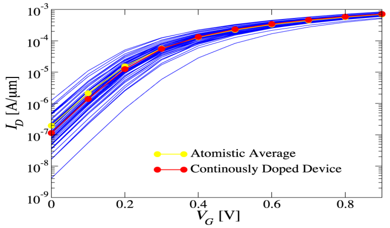


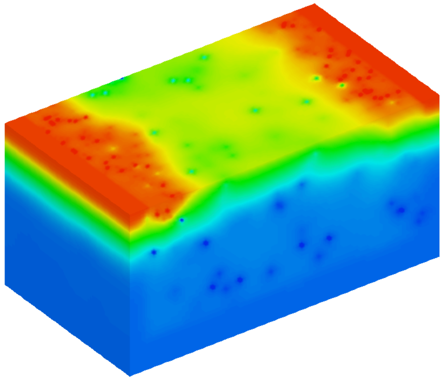
Direction of wave propagation

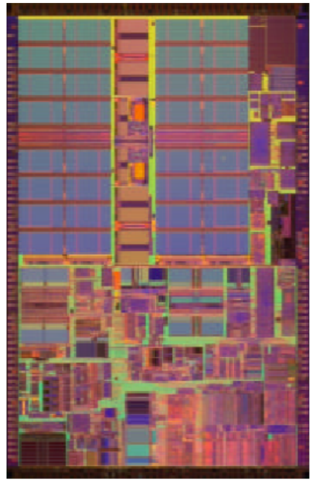


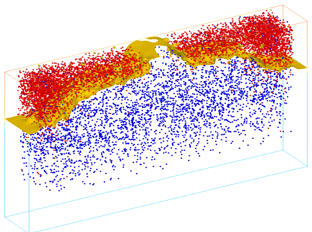


Challenges of NanoCMOS Design





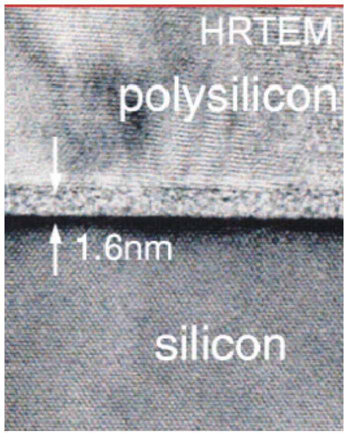


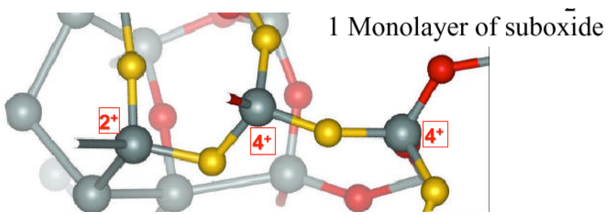


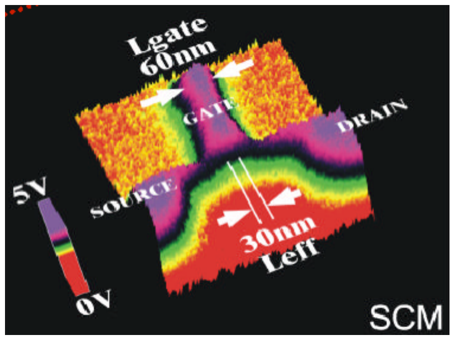
3D

+

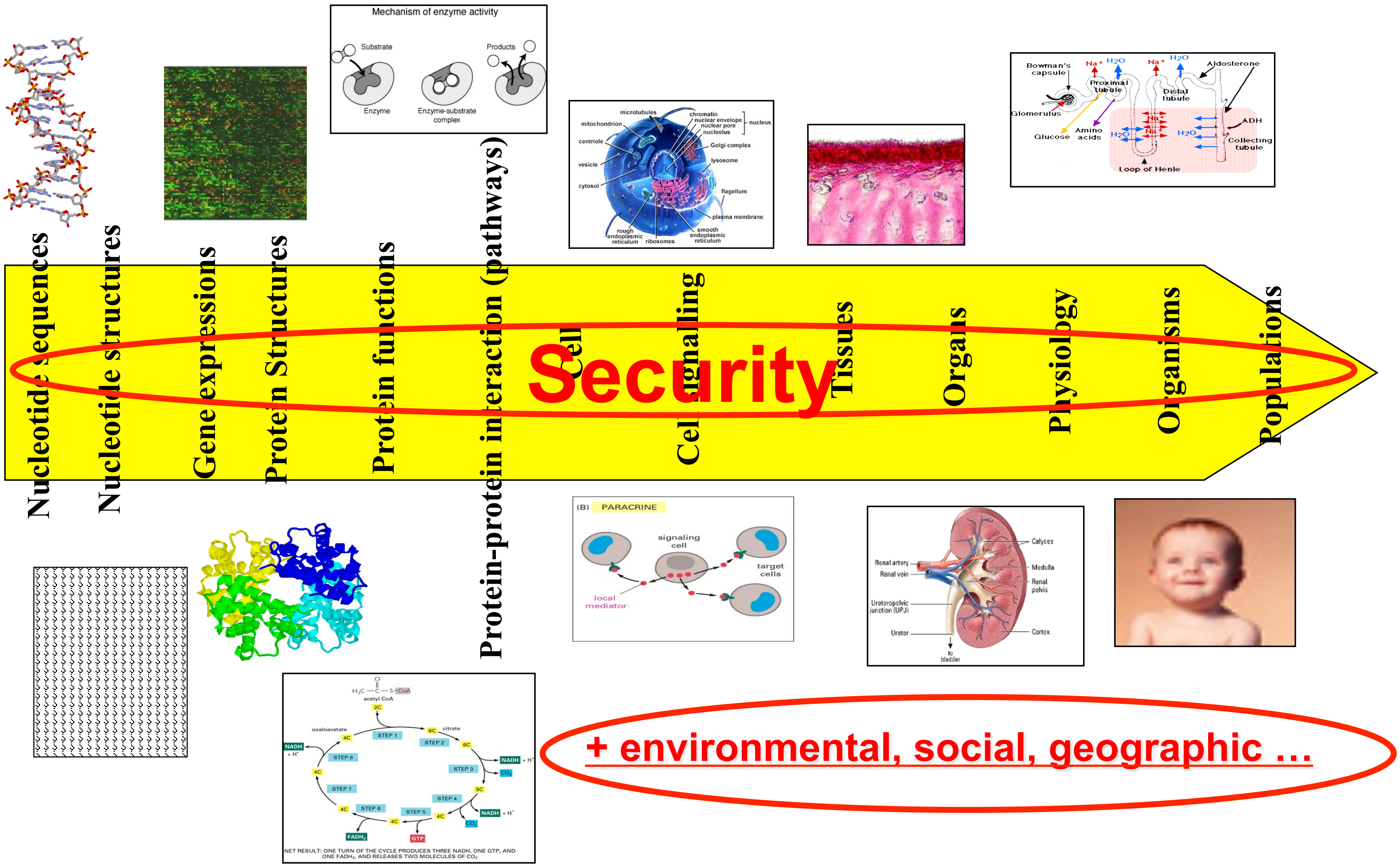
Statistical





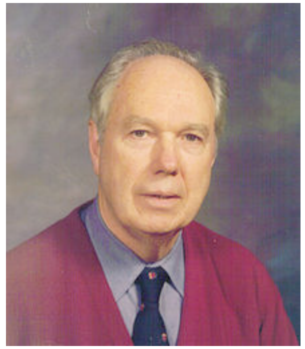


The e-Health Future…



More (and more and more) genomes…

Week 3 - Computer Architectures



- Flynn's Taxonomy
  - Single Instruction, Single Data stream (SISD)
  - Single Instruction, Multiple Data streams (SIMD)
  - Multiple Instruction, Single Data stream (MISD)
  - Multiple Instruction, Multiple Data streams (MIMD)

Flynn's taxonomy

	Single instruction	Multiple instruction
Single data	SISD	MISD
Multiple data	SIMD	MIMD

Week 3 - Amdahl's Law

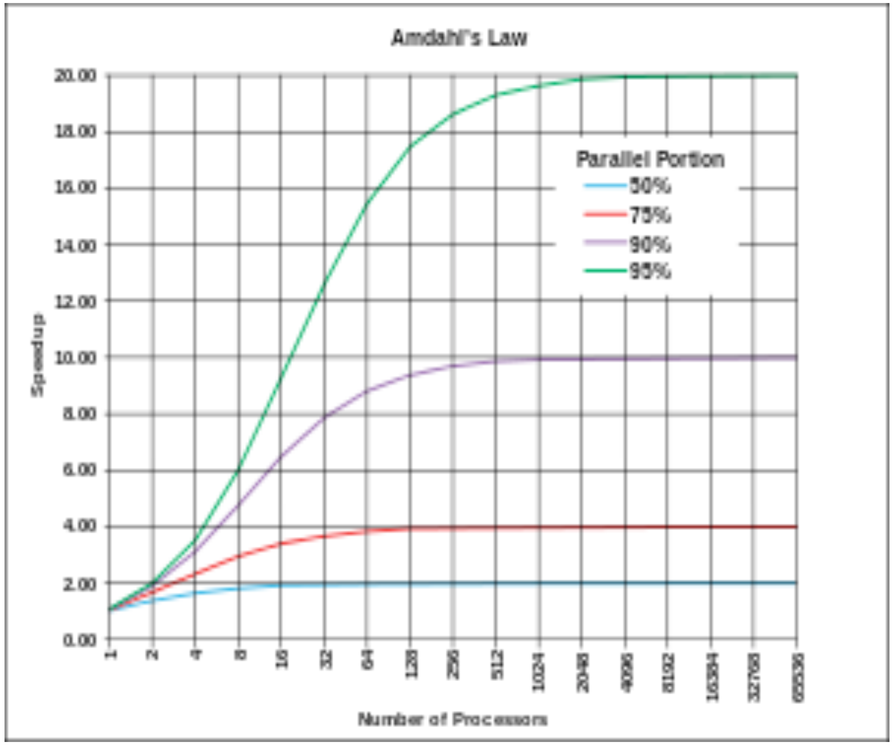
$$T(1) = \sigma + \pi \qquad T(N) = \sigma + \frac{\pi}{N}$$

$$S = \frac{T(1)}{T(N)} = \frac{\sigma + \pi}{\sigma + \pi/N} = \frac{1 + \pi/\sigma}{1 + (\pi/\sigma) \times (1/N)}$$

$$\pi/\sigma = \frac{1-\alpha}{\alpha}$$

α Fraction of running time sequential program spends on parallel parts

$$S = \frac{1 + (1-\alpha)/\alpha}{1 + (1-\alpha)/(N\alpha)} = \frac{1}{\alpha + (1-\alpha)/N} \approx \frac{1}{\alpha}$$



If 95% of the program can be parallelized, the theoretical maximum speedup using parallel computing would be 20×, no matter how many processors are used, i.e. if the non-parallelisable part takes 1 hour, then no matter how many cores you throw at it it won't complete in <1 hour. However this assumes a fixed problem size – sometimes can't predict length of time required for jobs, e.g. state space exploration or differential equations that don't solve…

Week 3 - Gustafson-Barsis's Law

- Gives the “scaled speed-up”

$$T(1) = \sigma + N\pi \quad \text{and} \quad T(N) = \sigma + \pi$$

$$S(N) = \frac{T(1)}{T(N)} = \frac{\sigma + N\pi}{\sigma + \pi} = \frac{\sigma}{\sigma + \pi} + \frac{N\pi}{\sigma + \pi}$$

π Fixed parallel time per process

α Fraction of running time sequential program spends on parallel parts

$$\pi/\sigma = \frac{1-\alpha}{\alpha}$$

$$S(N) = \alpha + N(1-\alpha) = N - \alpha(N-1)$$

Speed up S using N processes is given as a linear formula dependent on the number of processes and the fraction of time to run sequential parts. Gustafson's Law proposes that programmers tend to set the size of problems to use the available equipment to solve problems within a practical fixed time. Faster (more parallel) equipment available, larger problems can be solved in the same time.

Week 3 - Parallelisation Paradigms

- Task-Farming/Master-Worker
- Single-Program Multiple-Data (SPMD)
- Pipelining
- Divide and Conquer
- Speculation
- Parametric Computation

Week 3 - Erroneous Assumptions of Distributed Systems

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is zero
8. The network is homogeneous
9. Time is ubiquitous

Week 4 – HPC & SPARTAN & MPI

SLURM and PBS/Torque
- sbatch vs qsub
- squeue vs showq
- squeue -j vs qstat
- scancel vs qdel

…

SLURM -101

```
#!/bin/bash
#SBATCH p cloud
#SBATCH time= 01:00:00
#SBATCH nodes= 1
#SBATCH ntasks= 1
module load myappcompiler/version
myapp data
```

```
MPI_INIT
MPI_FINALIZE
MPI_COMM_SIZE
MPI_COMM_RANK
MPI_SEND
MPI_RECV
```

+ examples

Week 5 - The Most Common Cloud Models

Deployment Models

- Private
- Community
- Public
- Hybrid

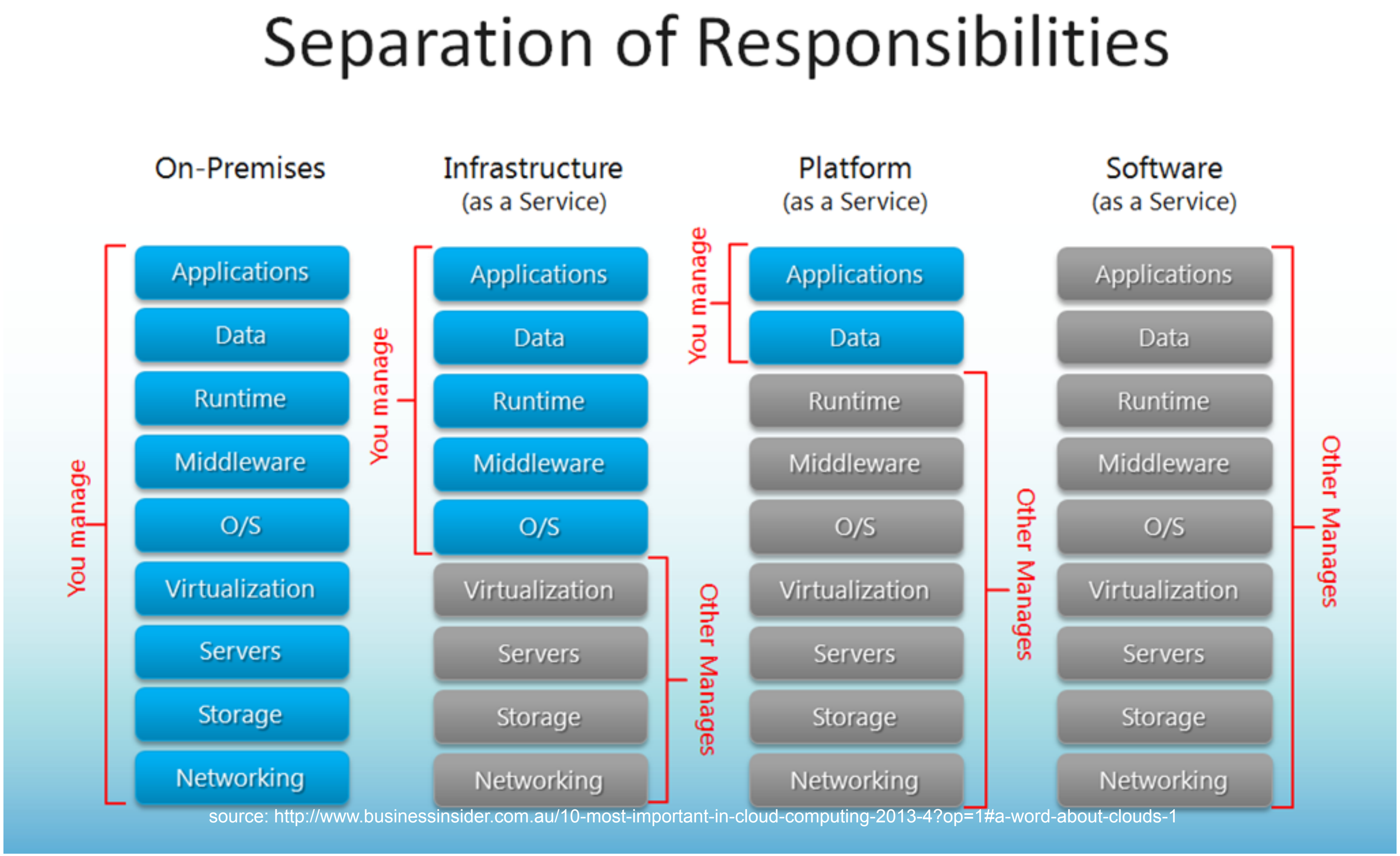
Delivery Models

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Week 5 - Delivery Models



Week 5 - Automation

- Deploying complex cloud systems require a lot of moving parts
  - Easy to forget what software you installed, and what steps you took to configure system
  - Might be non-repeatable
  - Snapshots are monolithic – provides no record of what has changed
- Automation:
  - Provides a record of what you did
  - Codifies knowledge about system
  - Makes process repeatable
  - Makes it programmable – "Infrastructure as code"

HANDS ON BOTO & ANSIBLE

Weelk 6 - “Big data” Is Not Just About “Bigness”

The “Vs” :

- **Volume**: yes, volume (Giga, Tera, Peta, Exa, …) is a criteria, but not the only one
- **Velocity**: the frequency of new data being brought in to the system and analysis performed
- **Variety**: the variability and complexity of data schema. The more complex the data schema(s) you have, the higher the probability of them changing along the way, adding more complexity.
- **Veracity**: the level of trust in the data accuracy; the more diverse sources you have, the more unstructured they are, the less veracity you have.

Week 6 - Consistency, Availability, Partition-Tolerance

- Consistency

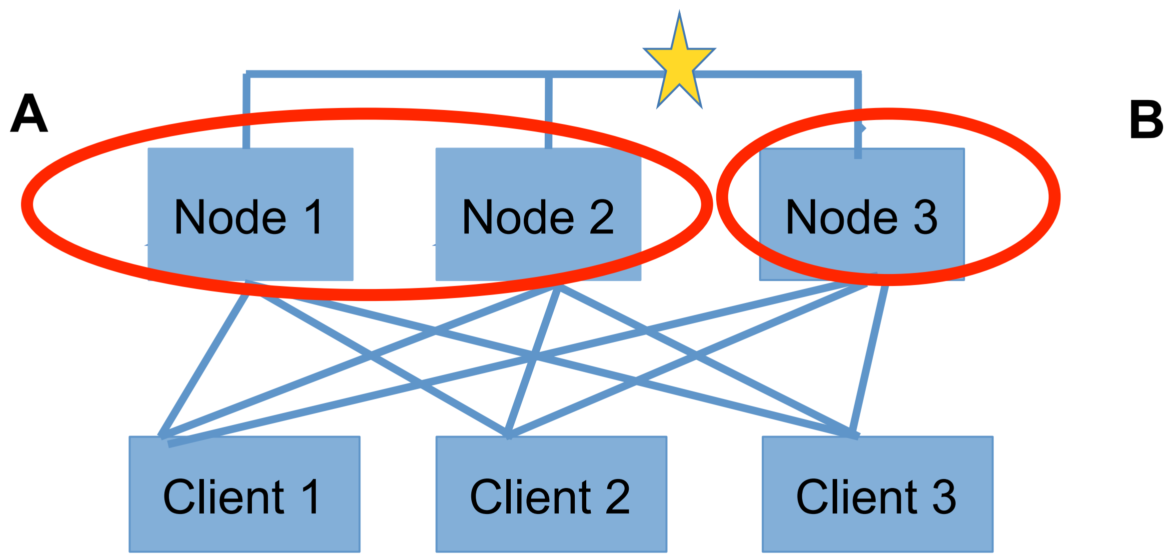
  Every client receiving an answer receives the same answer from all nodes in the cluster

- Availability

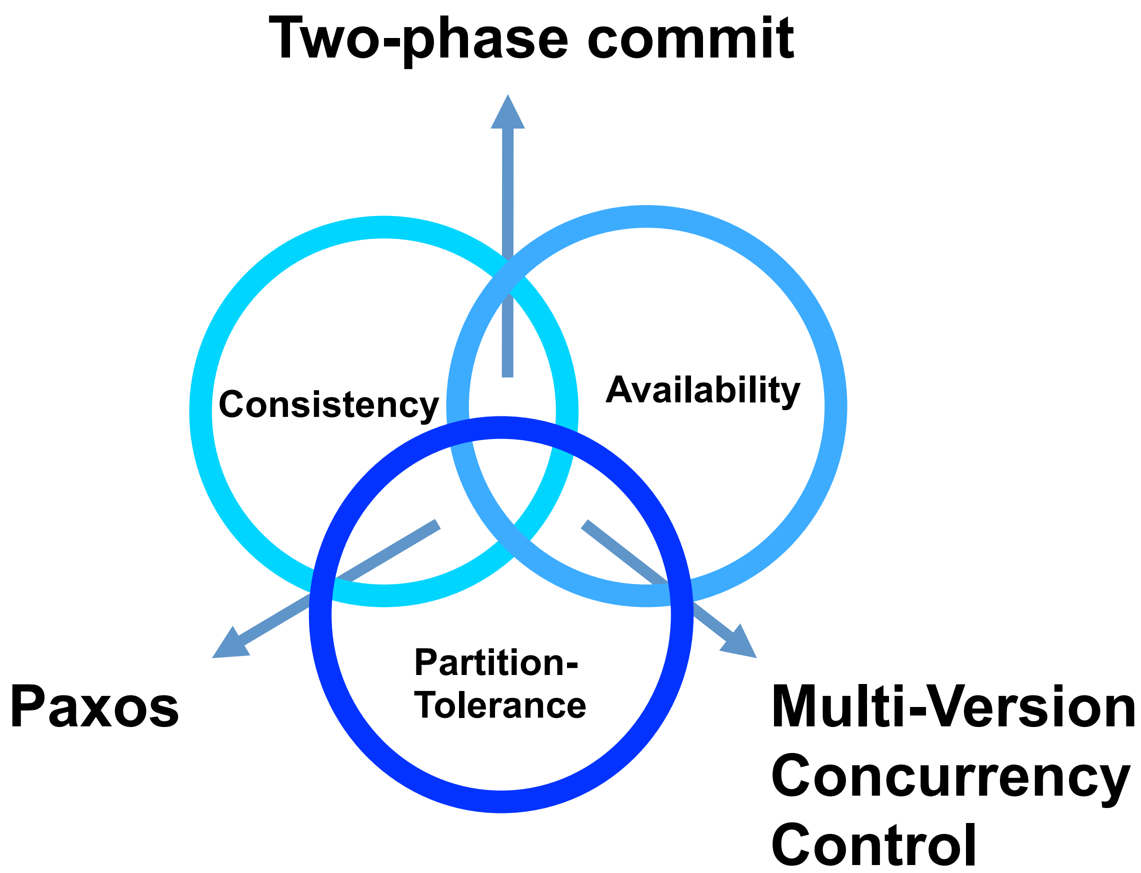
  Every client receives an answer from any nodes in the cluster

- Partition-tolerance

  The cluster keeps on operating when one or more nodes cannot communicate with the rest of the cluster



Week 6 - CAP Theorem and the Classification of Distributed Processing Algorithms



Week 6 – CouchDB Hands-on

Week 7 - SOA Core Ideas

A Service Oriented Architecture (SOA) has several core ideas (IBM):

- A set of services that a business wants to provide to their customers, partners, or other areas of an organization
- An architectural pattern that requires a service provider, mediation, and service requestor with a service description
- A set of architectural principles, patterns and criteria that address characteristics such as modularity, encapsulation, loose coupling, separation of concerns, reuse and composability
- A programming model complete with standards, tools and technologies that supports web services, ReST services or other kinds of services
- A middleware solution optimized for service assembly, orchestration, monitoring, and management

Week 7 - SOA Principles #1

- **Standardized service contract:** Services adhere to a communications agreement, as defined collectively by one or more service-description documents.
- **Service loose coupling:** Services maintain a relationship that minimizes dependencies and only requires that they maintain an awareness of each other.
- **Service abstraction:** Beyond descriptions in the service contract, services hide logic from the outside world.
- **Service reusability:** Logic is divided into services with the intention of promoting reuse.
- **Service autonomy:** Services have control over the logic they encapsulate.
- **Service statelessness:** Services minimize resource consumption by deferring the management of state information when necessary
- **Service discoverability:** Services are supplemented with communicative meta data by which they can be effectively discovered and interpreted.
- **Service composability:** Services are effective composition participants, regardless of the size and complexity of the composition.

Week 7 - SOA Principles #2

- **Service granularity**: A design consideration to provide optimal scope at the right granular level of the business functionality in a service operation.
- **Service normalization**: Services are decomposed and/or consolidated to a level of normal form to minimize redundancy. In some cases, services are denormalized for specific purposes, such as performance optimization, access, and aggregation.
- **Service optimization**: All else equal, high-quality services are generally preferable to low-quality ones.
- **Service relevance**: Functionality is presented at a granularity recognized by the user as a meaningful service.
- **Service encapsulation**: Many services are consolidated for use under a SOA. Often such services are not planned to be under a SOA.
- **Service location transparency**: The ability of a service consumer to invoke a service regardless of its actual location in the network.

SOAP vs REST and other WS flavours (geospatial)

- Resource-oriented architectures and do’s don’ts

Code versioning and Git demo

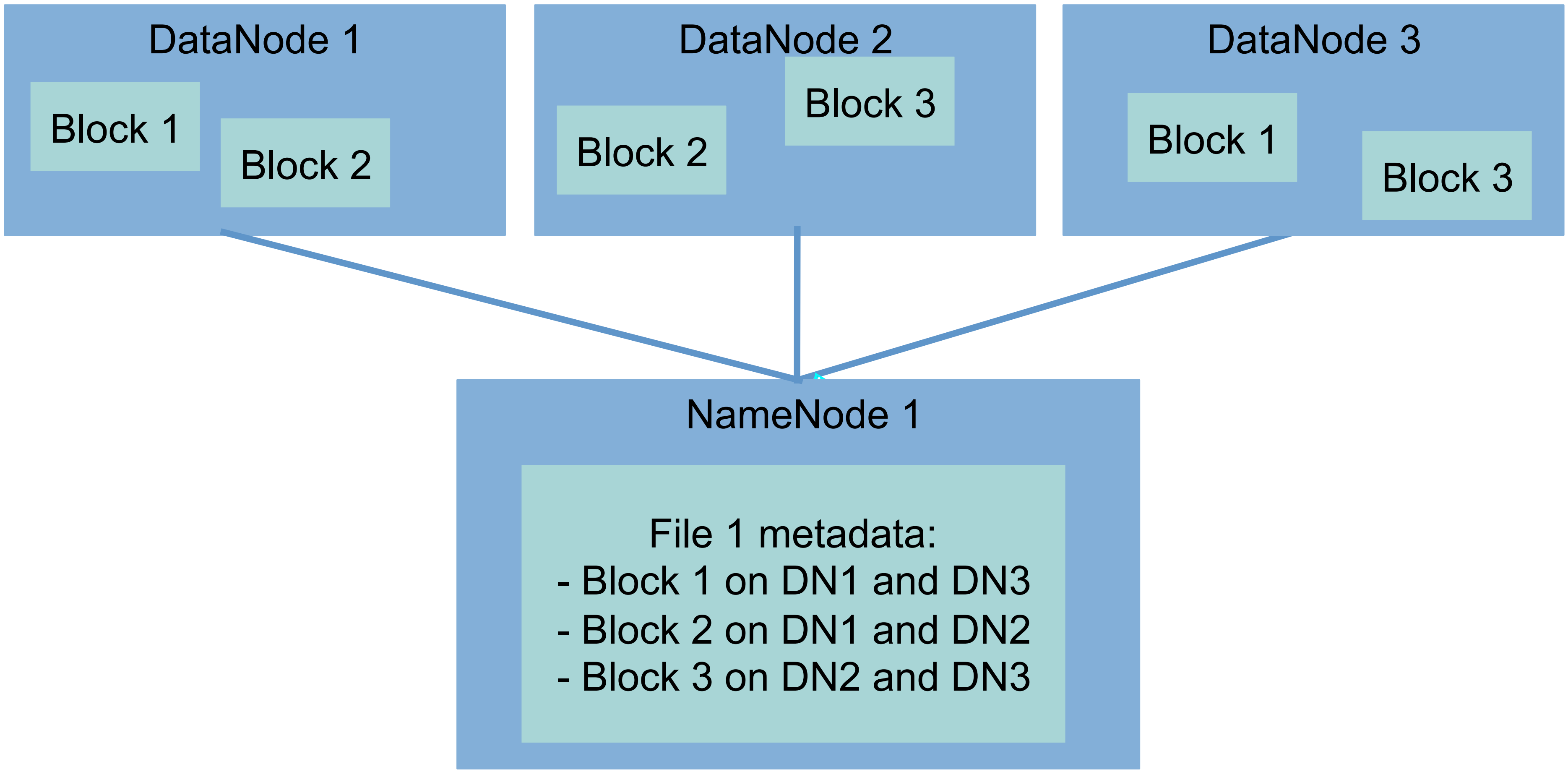
Week 8 - Challenges of Big Data Analytics

A framework for analysing big data has to distribute both data and processing over many nodes, which implies:

- Reading and writing distributed datasets
- Preserving data in the presence of failing data nodes
- Supporting the execution of MapReduce tasks
- Being fault-tolerant (a few failing compute nodes may slow down the processing, but not actually stop it)
- Coordinating the execution of tasks across a cluster

Week 8 – big data and HDFS Architecture

An HDFS file is a collection of blocks stored in *datanodes*, with metadata (such as the position of those blocks) that is stored in *namenodes*



Week 8 - Why Spark?

While Hadoop MapReduce works well, it is geared towards performing relatively simple jobs on large datasets.

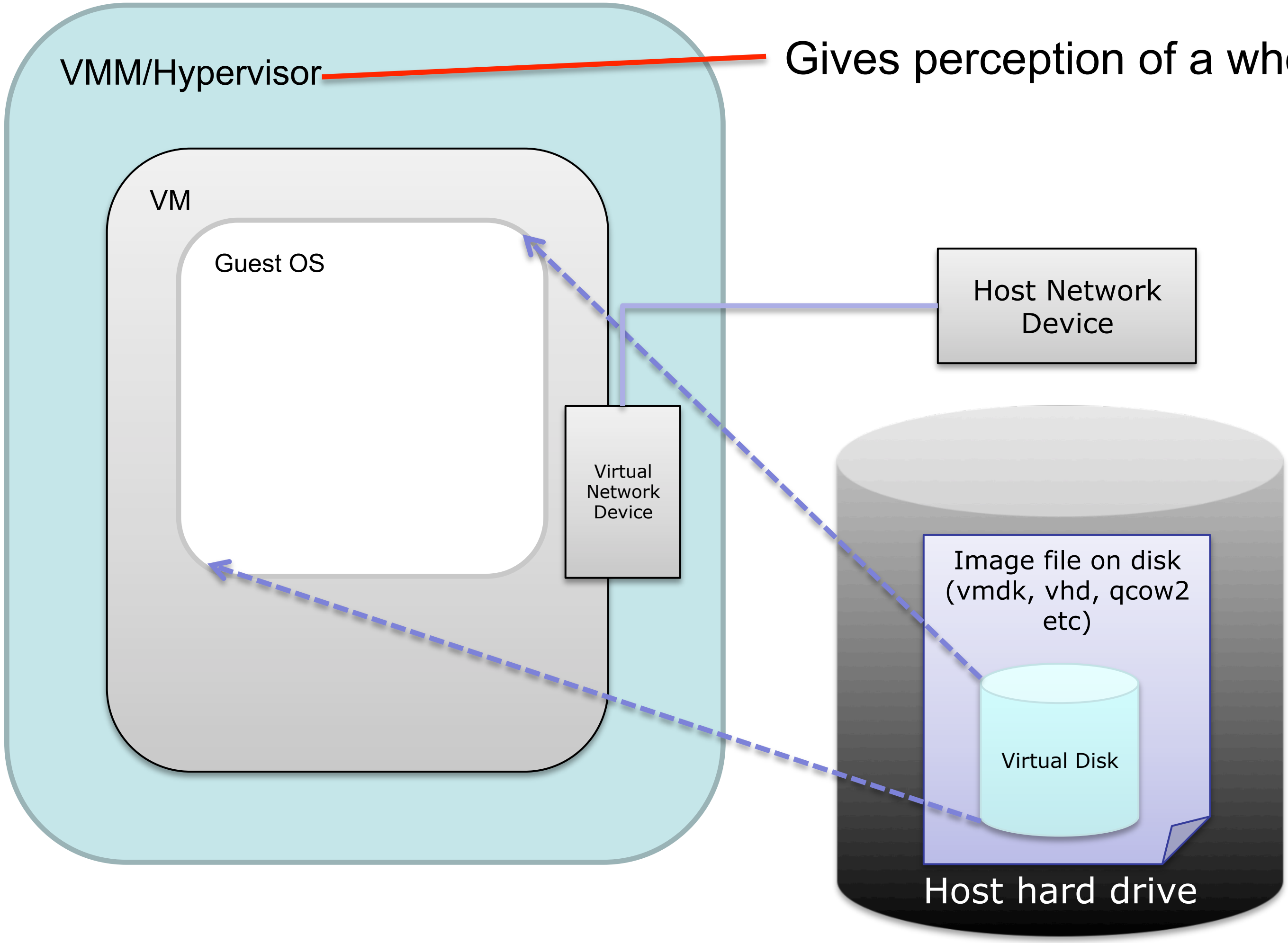
However, when complex jobs are performed (say, machine learning or graph-based algorithms), there is a strong incentive for caching data in memory and in having finer-grained control on the execution of jobs.

Apache Spark was designed to reduce the latency inherent in the Hadoop approach for the execution of MapReduce jobs.

Spark can operate within the Hadoop architecture, using YARN and Zookeeper, and storing data on HDFS.

Examples of using SPARK

Week 9 - What Happens in a VM



VHD (Virtual Hard Disk represents a virtual hard disk drive (HDD). May contain what is found on a physical hard disk, such as disk partitions and a file system, which in turn can contain files and folders.

VMDK (Virtual Machine Disk) describes containers for virtual hard disk drives to be used in virtual machines like VMware.

qcow2 (QEMU Copy On Write) file format for disk image files used by QEMU. It uses a disk storage optimization strategy that delays allocation of storage until it is actually needed.

- Guest OS apps “think” they write to hard disk but translated to virtualised host hard drive by VMM
- Which one is determined by image that is launched

Week 9 - Classification of Instructions

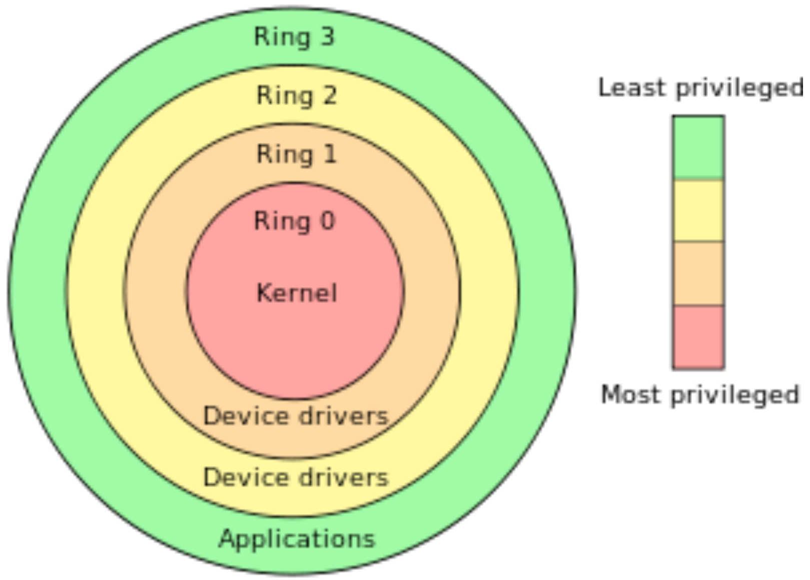
- Privileged Instructions: instructions that trap if the processor is in user mode and do not trap in kernel mode
- Sensitive Instructions: instructions whose behaviour depends on the mode or configuration of the hardware
  - Different behaviours depending on whether in user or kernel mode
    - e.g. POPF interrupt (for interrupt flag handling)
- Innocuous Instructions: instructions that are neither privileged nor sensitive
  - Read data, add numbers etc

Week 9 - Origins - Principles

- Theorem (Popek and Goldberg)
  - For any conventional third generation computer, a virtual machine monitor may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions
    - i.e. have to be trappable

Example of Privilege Rings

- Ring 0: Typically hardware interactions
- Ring 1: Typically device drivers
- Specific gates between Rings (not ad hoc)
- Allows to ensure for example that spyware can't turn on web cam or recording device etc



- Significance
  - The IA-32/x86 architecture was not originally virtualisable

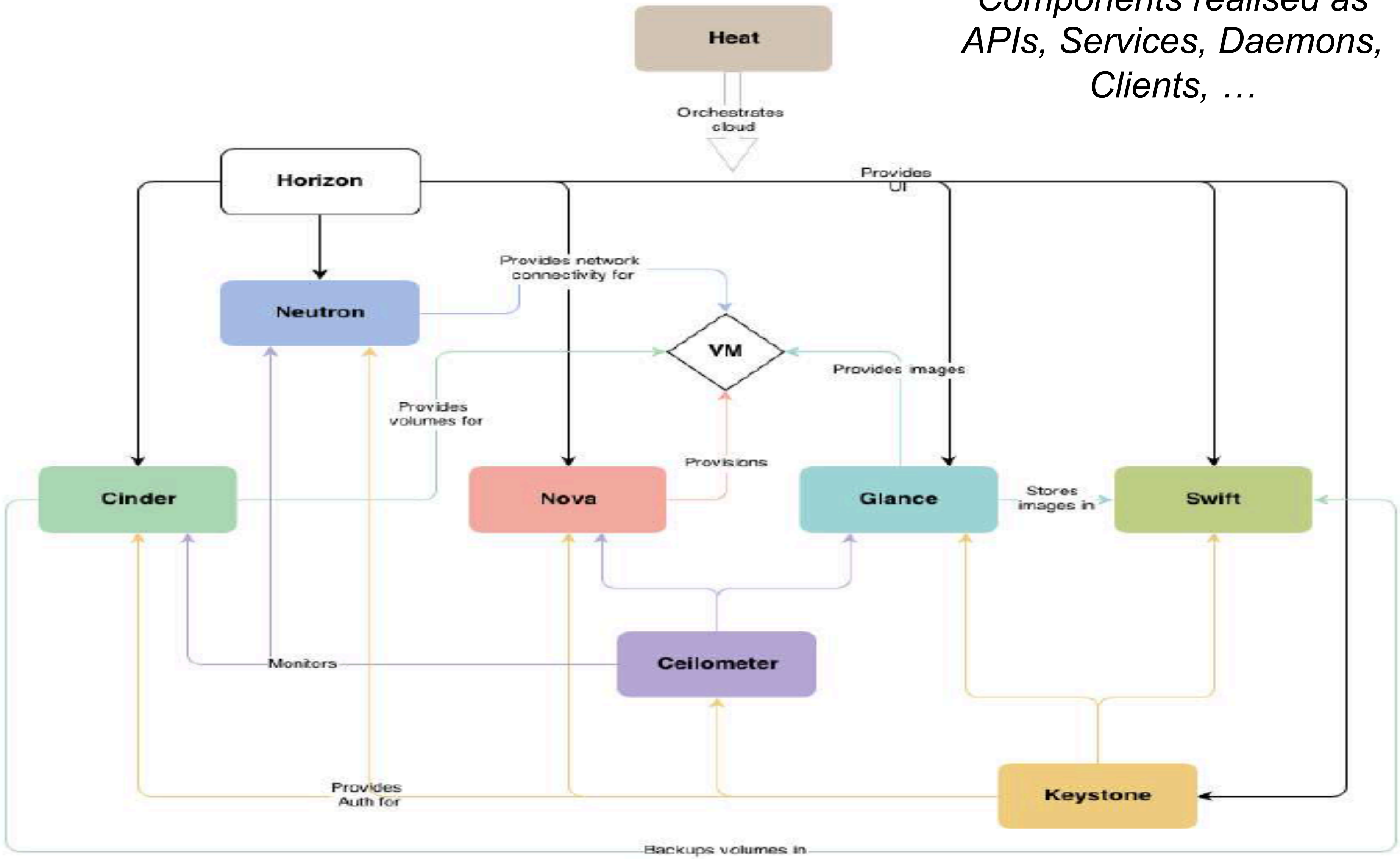
Week 9 - OpenStack Components

- Many associated/underpinning services
  - Compute Service (code-named Nova)
  - Image Service (code-named Glance)
  - Block Storage Service (code named Cinder)
  - Object Storage Service (code-named Swift)
  - Security Management (code-named Keystone)
  - Orchestration Service (code-named Heat)
  - Network Service (code-named Neutron)
  - Metering Service (code-named Ceilometer)
  - Database service (code-named Trove)
  - Dashboard service (code-named Horizon)
  - Search service (code-named Searchlight)
  - Security API (code named Barbican)…

http://docs.openstack.org/

Week 9 – (v. simplified) OpenStack Architecture

Components realised as APIs, Services, Daemons, Clients, …



Week 9 - VM and Containers (Docker etc)

Parameter	Virtual Machines	Containers
Guest OS	Run on virtual HW, have their own OS kernels	Share same OS kernel
Communication	Through Ethernet devices	IPC mechanisms (pipes, sockets)
Security	Depends on the Hypervisor	Requires close scrutiny
Performance	Small overhead incurs when instructions are translated from guest to host OS	Near native performance
Isolation	File systems and libraries are not shared between guest and host OS	File systems can be shared, and libraries are
Startup time	Slow (minutes)	Fast (a few seconds)
Storage	Large size	Small size (most is re-use)

Weeks 10 & 11 - You

Week 12 - Technical Challenges of Security

- Several key terms that associated with security
  - Authentication
  - Authorisation
  - Audit/accounting
  - Confidentiality
  - Privacy
  - Fabric management
  - Trust

Generally speaking

AAAA

Domain specific
(name -> DOB -> DNA)

Inter-organisational and
Technological challenges

All are important but some applications/domains have more emphasis on concepts than others

Key is to make all of this simple/transparent to users!

Week 12 - Public Key Cryptography

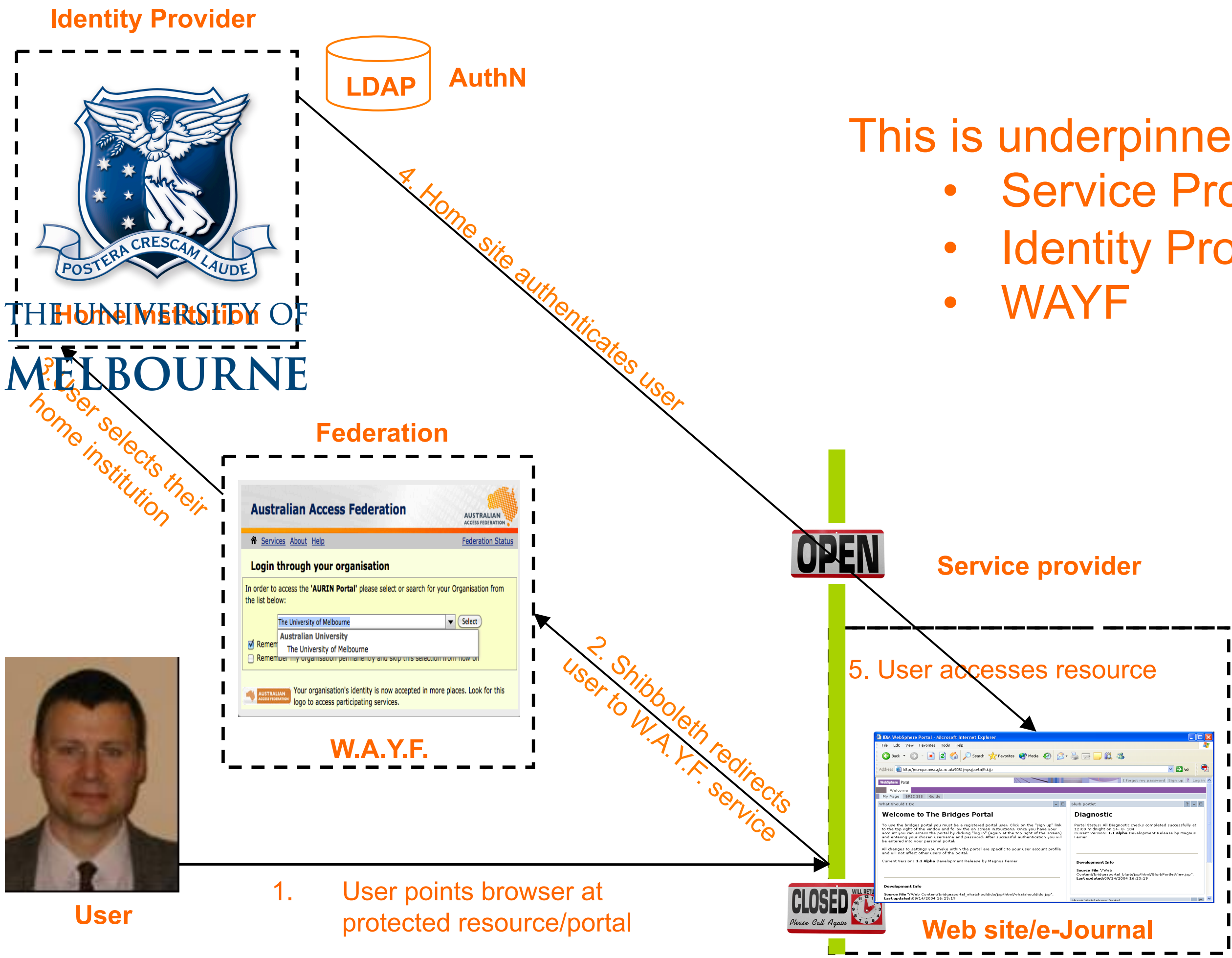
- Also called Asymmetric Cryptography
  - Two distinct keys
    - One that must be kept private
      - Private Key … Duh! ;o)
    - One that can be made public
      - Public Key … Double duh!
  - Two keys complementary, but essential that cannot find out value of private key from public key
    - With private keys can digitally sign messages, documents, … and validate them with associated public keys
      - Check whether changed, useful for non-repudiation, …
- Public Key Cryptography simplifies key management
  - Don’t need to have many keys for long time
    - The longer keys are left in storage, more likelihood of their being compromised
      - Instead use Public Keys for short time and then discard
      - Public Keys can be freely distributed
    - Only Private Key needs to be kept long term and kept securely

Week 12 - PKI and Cloud

- So what has this got to do with Cloud…?
  - IaaS – key pair!
- Cloud inter-operability begins with security!
  - There is no single, ubiquitous CA, there are many
- Your access to:
  - NeCTAR VMs was achieved through proving your identity as a member of the University of Melbourne
  - SPARTAN cluster was through proving your identity as a student enrolled in COMP90024 at the University of Melbourne
- There are many ways to prove your identity
  - OpenId, FacebookId, Visa credit card for Amazon, …
    - Degrees of trust
  - But remember need for single sign-on

Prove identity once and access distributed, autonomous resources!

Week 12 - Decentralised Authentication thru Shibboleth



This is underpinned by a PKI

- Service Providers
- Identity Providers
- WAYF

(identity proven!?)

Supports Single-Sign On (in case you were unaware)

Break

Example Questions

Question 1:

A) Discuss the major trends in research and research computing over the last 20 years that have led to the emergence of Cloud computing. [6]

B) How has the evolution of service-oriented architectures supported Cloud computing? [2]

C) A HTTP method can be *idempotent*.

- What is meant by this italicized term? [1]
- Give an example of an idempotent ReST method. [1]

Example Questions

Question 2:

A) Define Amdahl's law and discuss the challenges of its practical implementation. [2]

B) The actual performance as experienced by users of shared-access HPC facilities such as the Edward cluster at the University of Melbourne can vary – where here performance can be considered as the throughput of jobs, i.e. from the time of first job submission to the time of last job completion.

- Explain why this can happen. [2]
- Explain how the Edward cluster has been set up to minimize this. [2]
- Explain what users can do to optimize their throughput (use) of the Edward cluster. [2]
- Describe some of the challenges with application benchmarking on HPC facilities. [2]

Example Questions

Question 3:

A) Explain the consequences of Brewer's CAP theorem on distributed databases. [4]

B) Describe which aspects of the CAP theorem are supported by the following database technologies:

- non-SQL (unstructured) databases such as CouchDB. [2]
- relational databases such as PostGreSQL. [2]

C) Describe the advantages of MapReduce compared to other more traditional data processing approaches. [2]

Example Questions

Question 4:

A) Compare and contrast Representational State Transfer (ReST) based web services and Simple Object Access Protocol (SOAP)-based web services for implementing service-oriented architectures. [8]

B) Explain the differences between ReST-based PUT and POST methods and explain when one should be used over another. [2]

Example Questions

Question 5:

A) Explain what is meant by the following terms:

- Virtual Machine Monitor/Hypervisor [1]
- Full virtualization [1]
- Para-virtualization [1]
- Shadow page tables [1]
- Explain how hardware virtualization and software virtualization can differ in their treatment of shadow page tables. [2]
- Explain the advantages and disadvantages of virtual machines. [2]
- Describe the typical steps that are required to support live migration of virtual machine instances using a Cloud facility such as the NeCTAR Research Cloud. [2]

Example Questions

Question 6:

A) The Internet2 Shibboleth technology as currently supported by the Australia Access Federation provides *federated authentication* and *single sign-on*.

   a. Explain what is meant by the italicized terms [2].
   b. Explain the role of *trust* and *public key infrastructures* in supporting the Internet2 Shibboleth model. [2]
   c. What are the advantages and disadvantages of the Shibboleth approach for security? [4]
   d. Why isn't Shibboleth used to access Cloud-based systems more generally? [2]

Example Questions

Question 7:

A) Many research domains are facing "big data" challenges. Big data is not just related to the size of the data sets. Explain. [5]

B) What capabilities are currently offered or will be required for Cloud Computing infrastructures such as the NeCTAR Research Cloud to tackle these "big data" challenges. [5]
You may refer to specific research disciplines, e.g. life sciences, astrophysics, urban research (or others!) in your answer to part A) and B) of this question.

Feedback

- All in one 3 hour slot vs separated?
- Too much tech?
- Not enough tech?
- More HPC?
- More hands-on Cloud?
- Individual assignments vs teams?
- Software dev vs sys-admin skills?
- SES feedback completion