Chapter 1
History and Building Blocks of Cloud Computing

Mastering Cloud Computing

Coleman Kane

(based on materials from Paul Talaga)

UNIVERSITY OF Cincinnati

1. Distributed Systems

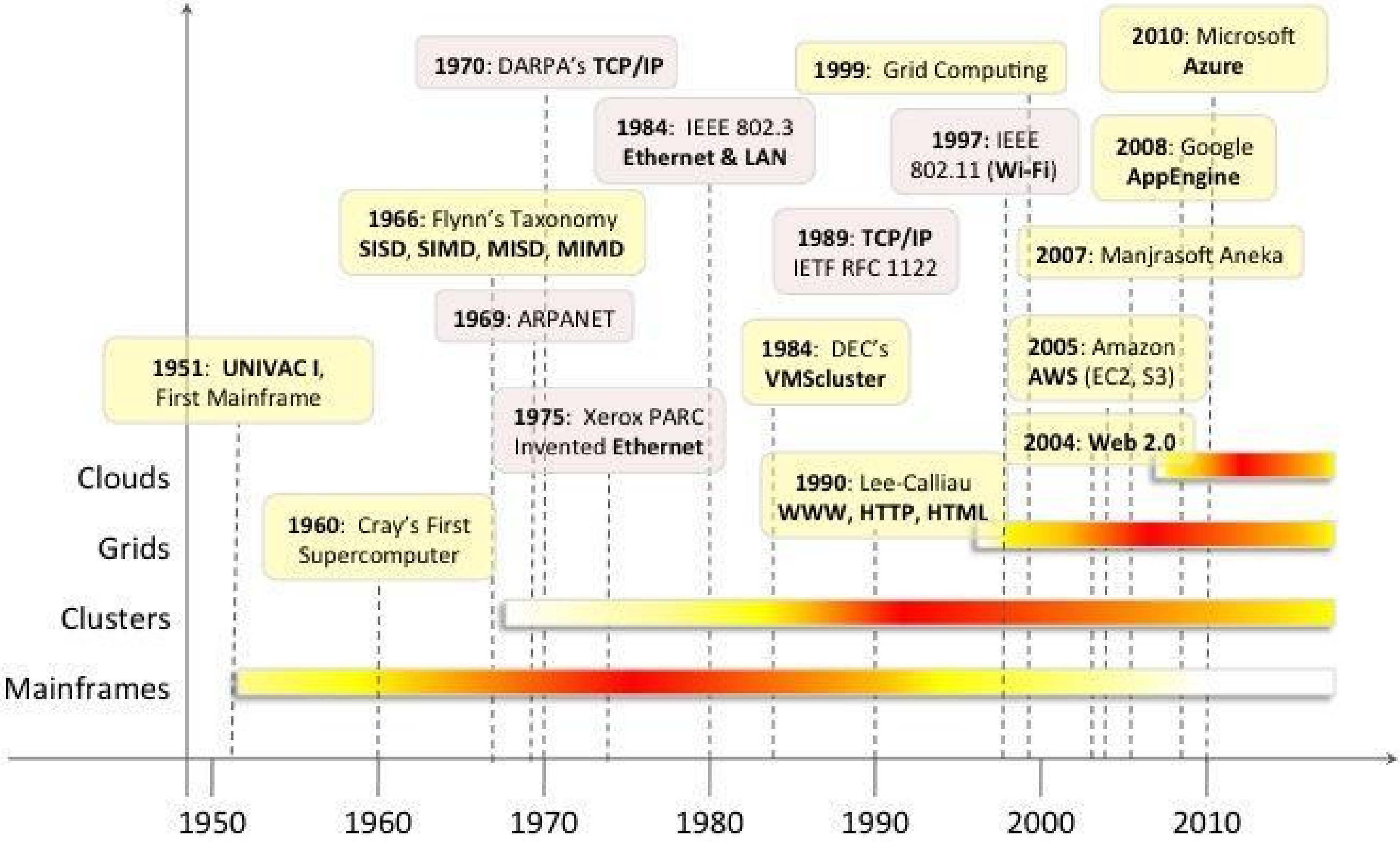
A distributed system is a collection of independent computers that appears to its users as a single coherent system. - Tanenbaum

To share resources for better utilization.

We have a course in Distributed Systems!

UNIVERSITY OF Cincinnati

What is old is new again...



1970: DARPA's **TCP/IP**

1999: Grid Computing

2010: Microsoft **Azure**

1984: IEEE 802.3 **Ethernet & LAN**

1997: IEEE 802.11 **(Wi-Fi)**

2008: Google **AppEngine**

1966: Flynn's Taxonomy **SISD, SIMD, MISD, MIMD**

1989: **TCP/IP** IETF RFC 1122

2007: Manjrasoft Aneka

1969: ARPANET

1984: DEC's **VMScluster**

2005: Amazon **AWS** (EC2, S3)

1951: **UNIVAC I**, First Mainframe

1975: Xerox PARC Invented **Ethernet**

2004: **Web 2.0**

1990: Lee-Calliau **WWW, HTTP, HTML**

1960: Cray's First Supercomputer

Mainframes (1951 onward)

- Multiple processing units
- Powerful, reliable, IO optimized
- Timesharing systems
- Replacement of components while running - always on
- Still used for transaction processing: banking, airline ticketing, registration

IBM z990
2003
256 GB RAM
1k+ VM Linux
11k SSL con/sec

IBM EC12
2012
101 CPUs
5.5Ghz hex core
Integrated SSDs
6k Linux VMs

Clusters (1968 onwards)

- Networked commodity (cheap) machines
- Physically close (room/building/LAN)
- Good for processing work, not IO
- Easy to expand
- Some use spare cpu cycles
- Condor, Beowulf Clusters, Message Passing Interface (MPI)
- Ohio Supercomputer

UNIVERSITY OF Cincinnati

Grid Computing (1990s onwards)

- Like cluster, but heterogeneous nodes
- Large physical distances
- Utility computing idea
- Needed high bandwidth connections (Internet)
- Good for processing work, not IO
- SETI@Home, BOINC

But what about latency!?!

Talaga’s Dissertation/Digression

- A semi full of hard drives has HUGE bandwidth!
- We get sold with bandwidth, but low latency is what we’re after.
- But the speed of light is fixed!
  - Significant drop in sales for > 500ms response
  - Google goal is < 200ms
  - NY to LA is 74ms RTT
- Grids may be good for some workloads, but HORRIBLE for others (IO and communication)
- Mainframe->Cluster->Grid, latency between nodes increases.
- Data location matters, and how your app uses data.

UNIVERSITY OF Cincinnati

Cloud Computing

- Next evolution after grid computing
- Infinite capacity
- Resilient to failures
- Always on
- Built using commodity machines
- Pay-per-use (Utility vision)

True root (my view)

The implementation and wider use of **distributed systems** theory to build distributed databases and file systems allowed cloud computing to take off.

- Huge datasets (multiple petabytes)
- Vector clocks (Lamport timestamps)
- Paxos (Wiki)

2. Virtualization

- Abstract core computing elements away
  - Processor
  - Storage
  - Networking
- Hardware virtualization (VMware, VirtualBox, XEN, EC2, etc….)
  - Most performance issues solved
- Process virtualization (Google AppEngine, Azure, Java)

UNIVERSITY OF Cincinnati

3. Web 2.0

- Web 1.0? - Static pages
- Web 2.0 is:
  - “Web as platform” - *John Battelle and Tim O'Reilly*
  - Interactivity & flexibility - Allow users to change a site’s content!
  - Asynchronous JavaScript and XML (AJAX)
  - Web Services
- Not a ‘next version’ of the web, but a way of using HTTP/HTML
- Examples: Google Docs, Facebook, YouTube, Wikipedia

4. Service-oriented Computing

A component that can perform any function.

- Loosely coupled
- Reusable
- Programming language independent
- Location transparent

By layering services we can build a service-oriented architecture (SOA)

Service-Oriented Computing

Important attributes:

- Quality of Service (QoS)
  - Service attributes, response times, security, uptime, etc…
- New software delivery model
  - Can sell components, not entire programs
  - Access through the internet
  - HTTP
  - Web Service Description Language (WSDL)
  - Simple Object Access Protocol (SOAP)

5. Utility-oriented Computing

Storage, compute, applications, infrastructure, all on a pay-per-use basis.

Old idea.. job queueing systems, OS time-slicing, all developed in mainframes to charge for use.
Buying services/products online common now.

UNIVERSITY OF Cincinnati

Current Platforms

- Amazon Web Services (AWS) - Current leader in IaaS: Elastic Compute Cloud (EC2), Simple Storage Service (S3), many many others…..
- Google AppEngine - PaaS - Python, Java, Go
- MS Azure - IaaS & PaaS
- Hadoop - Open Source framework for MapReduce
- Force.com & Salesforce.com

UNIVERSITY OF Cincinnati

Serverless Architectures

- *Microservice architecture*
- CloudFoundry
- Amazon Lambda
- BlueMix
- Predix
- Microsoft Services Framework

Some Security Implications

- Provisioning in software - automated, online, hands-off
- Full-system imaging & snapshotting, assists forensics investigation
- Partitioning - Better isolation between applications than virtual web hosting servers
- Software networking - pros / cons