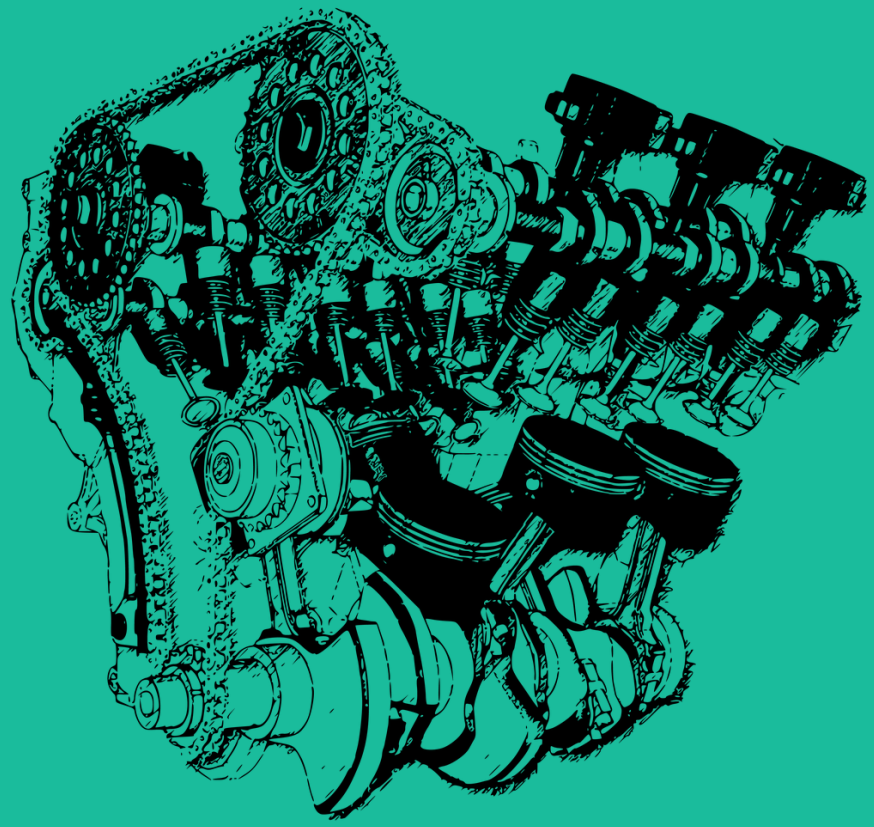
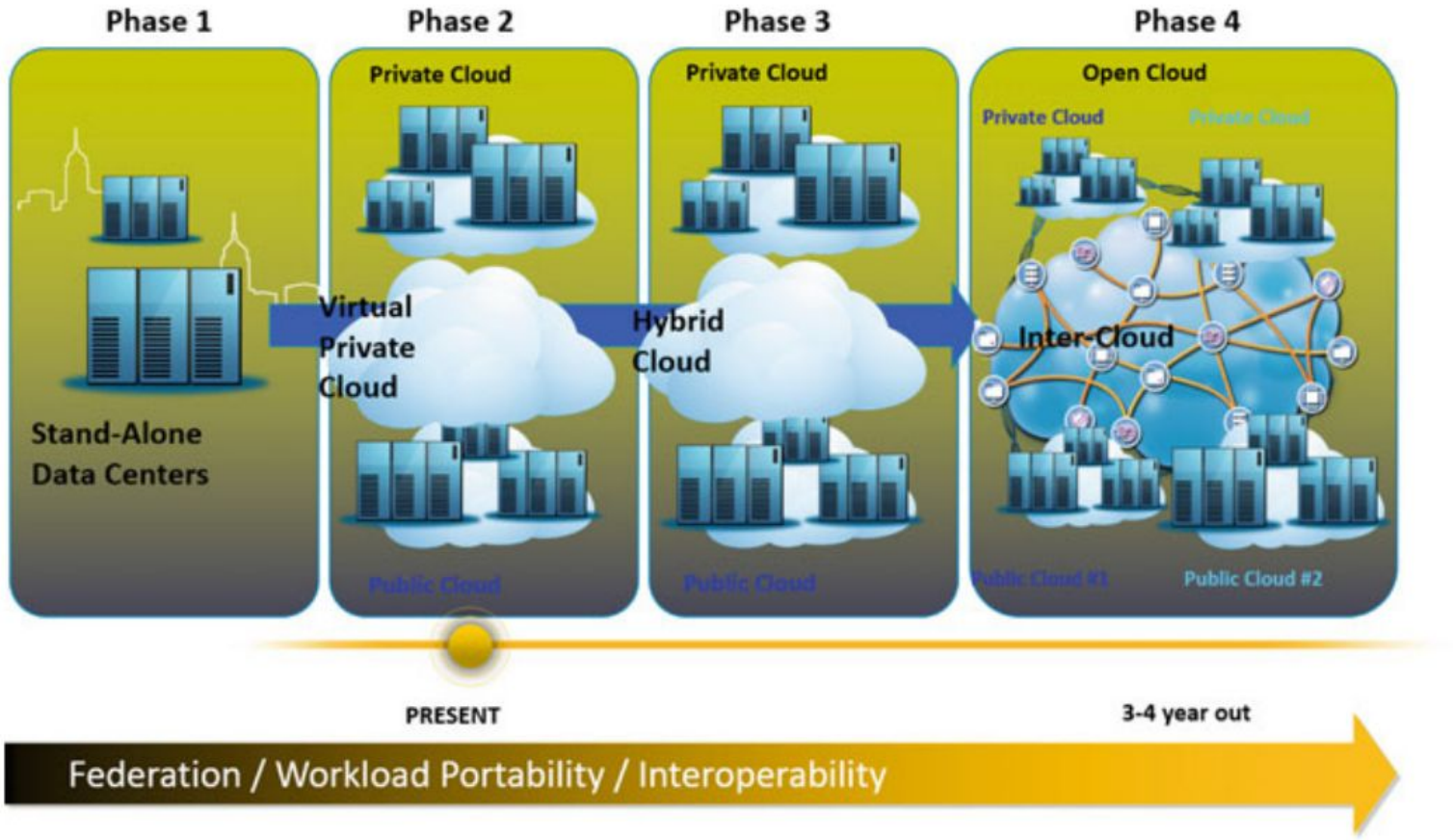


Chapter 2. Features of Cloud

Bilkent University | CS443 | 2021, Spring | Dr. Orçun Dayıbaş

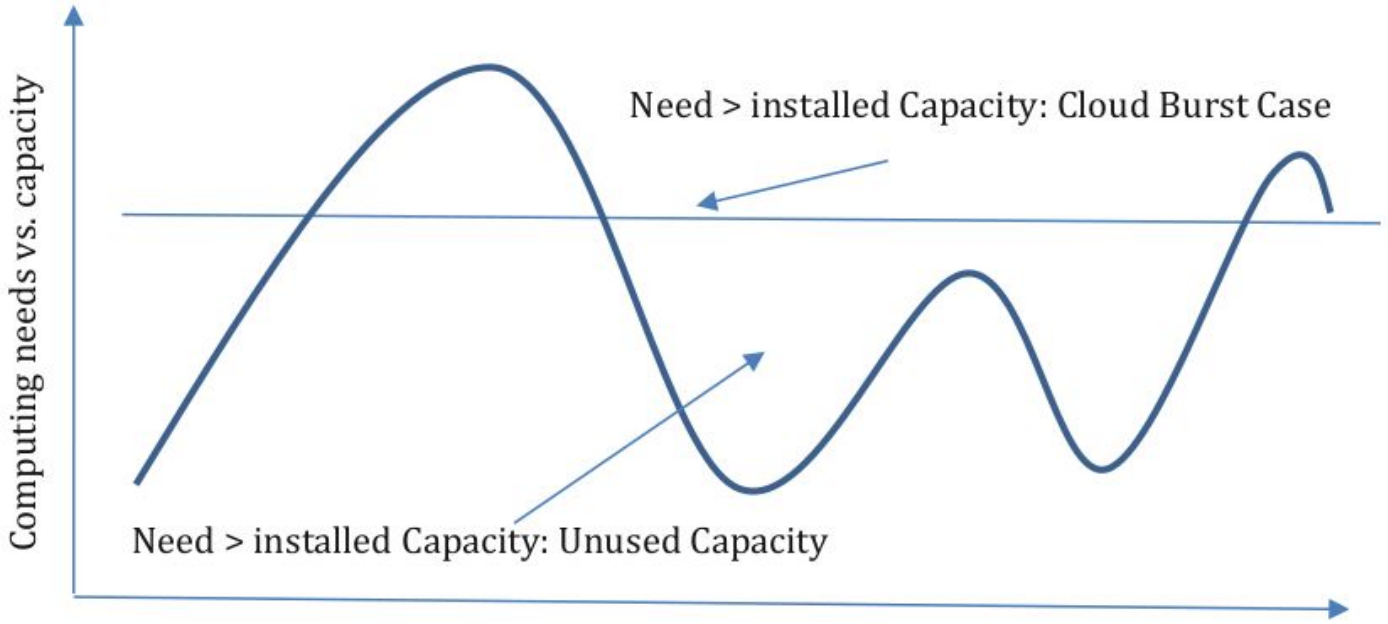
Adoption of Cloud Computing

- **Phased Approach**



Adoption of Cloud Computing

- **Needs vs. Capacity**



"Opportunity cost" represents the investor or business misses out on when choosing one alternative over another.

Reliability of Cloud Computing

- **Availability of various computing systems**

9's	Availability (%)	Downtime per year	Examples
1	90.0	36 days 12 h	Personal computers
2	99.0	87 h 36 min	Entry-level business
3	99.9	8 h 45.6 min	ISPs, mainstream business
4	99.99	52 min 33.6 s	Data centers
5	99.999	5 min 15.4 s	Banking, medical
6	99.9999	31.5 s	Military defense

"Availability", in the context of a computer system, refers to the ability of a user to access information or resources in a specified location and in the correct format.

Reliability of Cloud Computing

- **Availability of a e-commerce site**
  - Multiplying each row gives overall sys. avail.
  - What is the best way to promote?

Component	Availability (%)
Web server	85
Application	90
Database	99.9
DNS	98
Firewall	85
Switch	99
Data center	99.99
ISP	95

Reliability of Cloud Computing

- **Availability of a e-commerce site**
  - 60% is not acceptable, why?
  - Possible solutions
    - Higher reliability for each component (improving the overall reliability)
    - Introducing redundancy (parallelism)
      - The availability of a parallel system: A + ((1-(A/100))*A))
      - Two web server(85%) → 97.75%

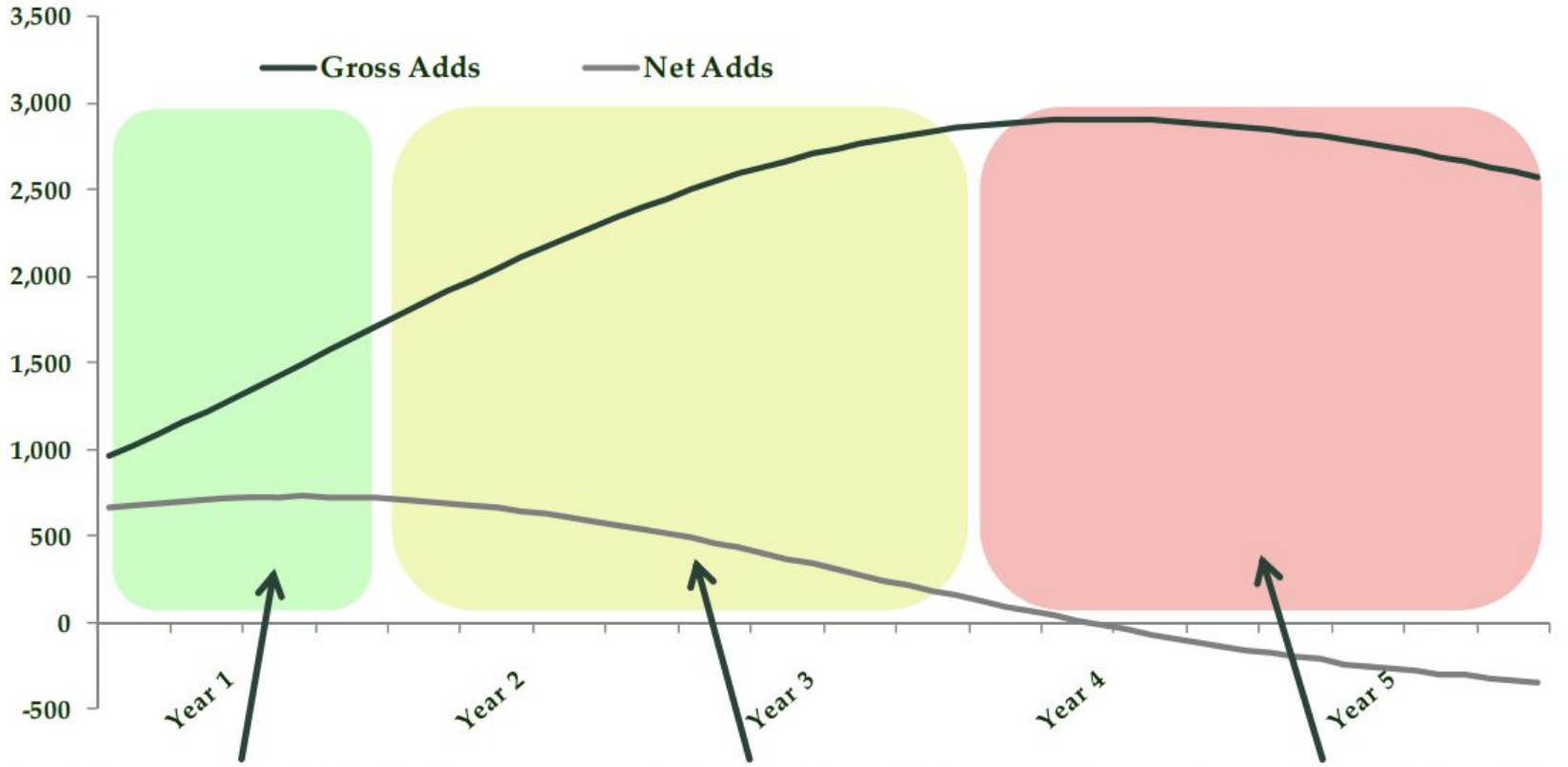
Performance of Cloud Computing

- **Cloud computing performance has inherent variability**
  - What to measure? How to compare?
  - Business effects & trade-offs
- **Application/Service monitoring**
  - Business KPI
    - Avg. order value, customer churn rate, products per order, etc...

"Customer Churn Rate" is the percentage of your customers or subscribers who cancel or don't renew their subscriptions during a given time period.
Customer Churn Rate = Lost Customers ÷ Acquired Customers

Performance of Cloud Computing

- Ex: ACME SaaS performance



Performance of Cloud Computing

- Ex: Cohort table for churn analysis

Column 0 shows changes that happen in the same month the customer joins.

Each row contains one group (cohort) of customers who started paying in a particular month. Rows show retention of that cohort in the lifespan. The columns (1,2,3,etc.) represent the number of months since they joined.

Customer churn cohort (% of customers churned relative to previous month)

		Lifetime month											
	Cohort value	0	1	2	3	4	5	6	7	8	9	10	11
Feb 2014	$999	2.50%	0.80%	5.93%	2.12%	1.35%	0.40%	1.04%	0.90%	0.90%	0.90%	0.90%	0.90%
Mar 2014	$293	0.00%	1.50%	4.09%	3.65%	1.04%	1.43%	1.04%	1.04%	1.04%	1.04%	1.04%	
Apr 2014	$89	1.22%	4.69%	5.80%	4.23%	2.15%	2.46%	2.46%	1.18%	1.18%	1.18%		
May 2014	$999	2.40%	5.66%	5.82%	3.54%	1.35%	3.49%	1.04%	1.32%	1.32%			
Jun 2014	$293	3.50%	2.67%	7.23%	2.12%	1.04%	4.52%	0.90%	1.46%				
Jul 2014	$89	1.55%	2.56%	5.00%	3.65%	2.15%	5.55%	1.04%					
Aug 2014	$999	1.34%	0.80%	4.09%	4.23%	1.35%	6.58%						
Sep 2014	$293	2.50%	1.50%	4.12%	3.54%	1.04%							
Oct 2014	$89	0.00%	4.69%	3.80%	2.12%								
Nov 2014	$999	1.22%	5.66%	3.93%									
Dec 2014	$293	2.40%	2.67%										
Jan 2015	$89	3.50%											
Average	$89	1.80%	2.98%	5.04%	3.39%	1.51%	3.49%	1.04%	1.18%	1.11%	1.04%	0.97%	0.90%

The first two columns show the month and the value of this specific cohort (the total MRR or customer count)

The reason these cells are empty is because this is the future for this example

"Cohort analysis" is a subset of behavioral analytics that takes the data from a given data set and rather than looking at all users as one unit, it breaks them into related groups for analysis. These related groups, or cohorts, usually share common characteristics or experiences within a defined time-span.

"Monthly Recurring Revenue (MRR)" is all of your recurring revenue normalized into a monthly amount. It's a metric usually used among subscription and SaaS companies.

Performance of Cloud Computing

- **Application/Service monitoring**
  - Service Level Agreement (SLA)
    - Monitor failure ratio: failure reqs. / total reqs. (Availability)
    - Measure both client-side and server-side latencies per API method (Latency)
    - “Synthetic Transaction Monitoring” (Consistency)
      - CRUD operations on production sys.

Performance of Cloud Computing

- **Application/Service monitoring**
  - Compute infrastructure
    - CPU utilization
    - Workload vs. CPU utilization (cost-effectiveness)
    - System memory (total and %)
    - Garbage collection count & time spent
    - Disk space (total and %)
    - etc.

Performance of Cloud Computing

- **Application/Service monitoring**
  - Network infrastructure
    - Bandwidth limit and maximum number of open connections
  - Dependencies
    - Health of external services (SSO, payment, advertisement, etc.)
    - Availability, latency (mean, p99), etc.
      - How can you measure latency for a app./service?

Performance of Cloud Computing

- **Latency metrics**
  - **Mean:** average latency
    - Can be misleading (hides outliers)
  - **50th percentile (Median):** The max. latency for the fastest half of all requests
  - **99th percentile (p99):** The max. latency for 99% of all requests
    - Way better than “maximum” (can be distorted by outliers)
    - Ex: says “1% of all your customers are experiencing 800+ ms latencies”

Performance of Cloud Computing

- **Recap**
  - If you can’t measure it, you can’t fix/improve it
  - Measuring everything creates noise (trade-off)
  - DevOps/SRE practices kicks in here
    - Ex: 12-factor App principles, Canary release, A/B Testing, etc...

Q/A

Cloud Workload Characterization

- **Different players in the cloud have different business and technical needs**
  - “A problem well stated is a problem half solved”
- **Workload = Nature of business + solution design**
  - Spikes (e.g. black friday → AWS)
    - Not normal but must be handled
  - Application category
    - Normal behavior

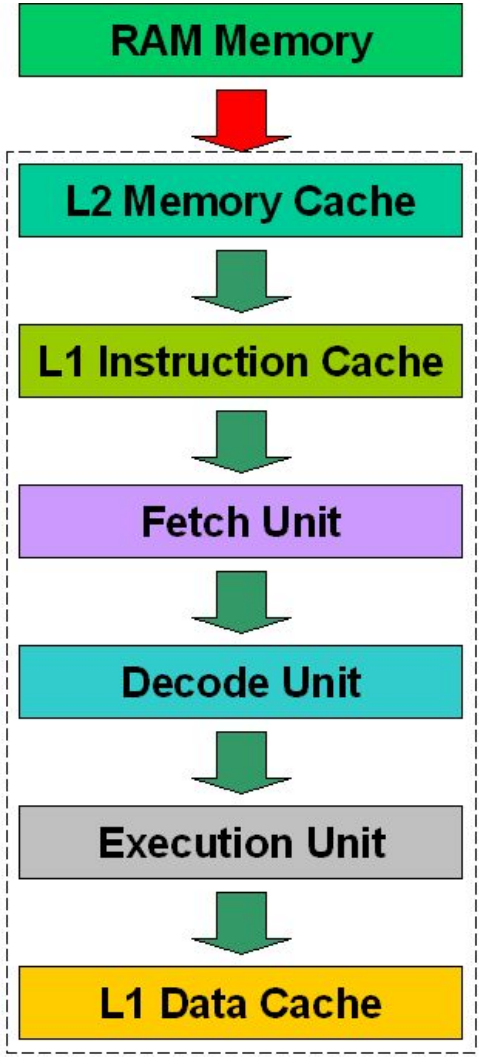
Cloud Workload Characterization

- **Typical Applications**

Workload Category	Example	Limiting Resources
Big streaming data	Netflix	Network bandwidth
Big database calculation	Google	Persistent storage, computational capability, caching
Many tiny tasks (ants)	Simple games, translator	Network, many processors
Single computer intensive jobs	EDA tools (simulation, etc.)	Computational capability
Highly interactive multi-person jobs	Google Docs, Facebook	Network, Processor assignment (VMs)

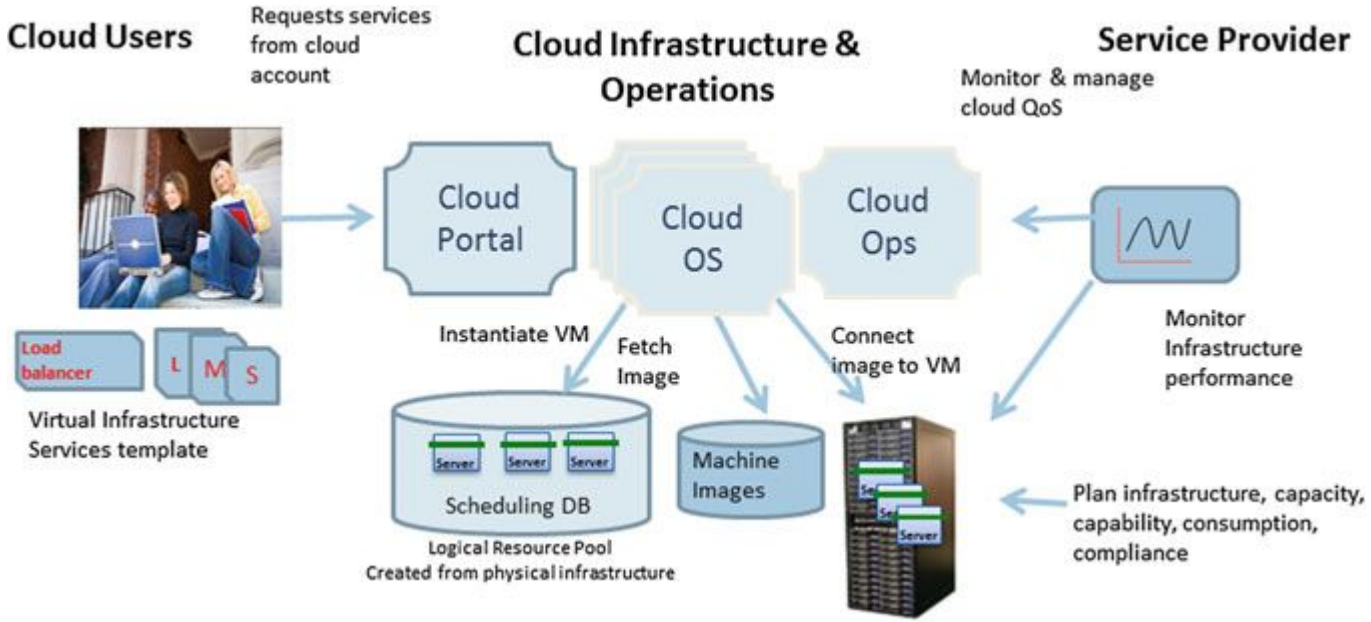
Cloud Workload Characterization

- **Low-level or Hardware Metrics of Utilization**
  - Instruction per Cycle(IPC)
    - Clock speed vs. IPC
  - LLC (Last-level cache) misses
    - Longest-latency before memory
  - L1 data cache misses
    - The fastest data provider
  - # of lines fetched from memory
    - The pressure on the memory



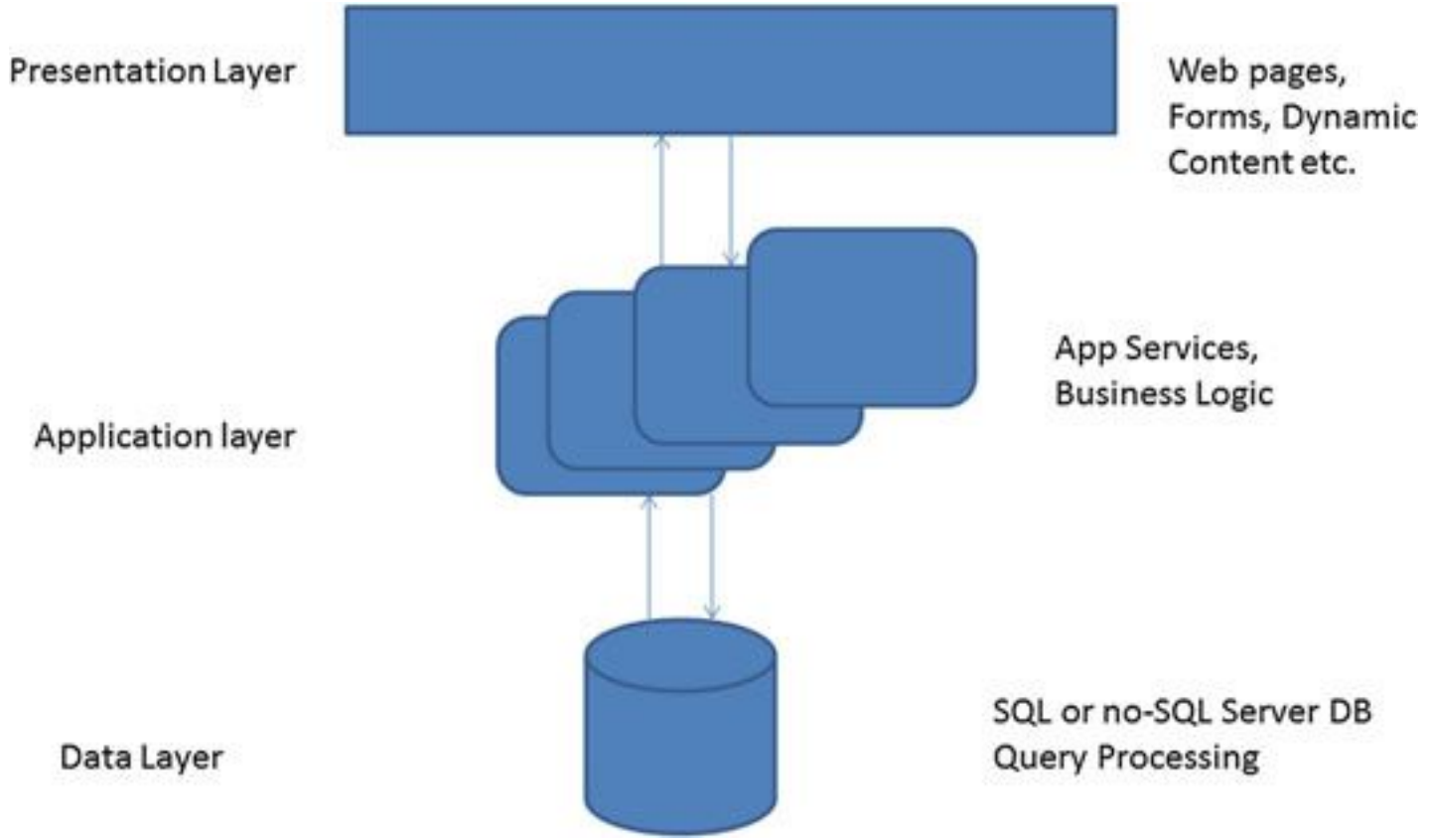
Cloud Workload Characterization

- **Dynamic monitoring**
  - As the VM density increases, co-scheduling VMs that are least destructive to each other on the same physical cores is crucial.



Cloud Workload Characterization

- **Example:** An airline's website
  - 3-tier architecture



Cloud Workload Characterization

- **Recap**
  - Many applications combine different workloads. For instance;
    - online maps system
      - read large file first (I/O), build a graph and then computation heavy
    - airline ticketing system
      - I/O heavy, transactional
  - Therefore, we need to understand common cloud application architectures…

Distributed Systems

- **Definition**
  - “A collection of independent computers that appear to its users as one computer” A.T.
  - Three characteristics
    - The computers run concurrently
    - The computers fail independently
    - The computers don’t share a global clock
  - Three topics to discuss
    - Storage, Computation, Messaging

Distributed Systems

- **Implementation complexity**

	Runs on Single Machine	Runs on Multiple Machines
Runs for Single User	X	10X
Runs for Multiple User	10X	100X

- **Patterns help us to mitigate the risk**

> **"A reference architecture"** provides a template solution for an architecture for a particular domain. It also provides a common vocabulary with which to discuss implementations, often with the aim to stress commonality.

Q/A