

TCCS 562:
SOFTWARE ENGINEERING
FOR CLOUD COMPUTING

Intro to Cloud Computing &
Term Project

Wes J. Lloyd
School of Engineering and Technology
University of Washington - Tacoma



1

FEEDBACK FROM 10/7

- Perspective on material: 6.17 (→ mostly new to me)
- Pace: 5.04 (~ just right)
- 23 respondents
- Story of cloud computing from bare metal to container

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.2

2

FEEDBACK - 2

- What about research on papers for term project?
 - Finding an idea from a research paper, and extending into a project could work. Please see instructor.
 - Alternate project ideas are for those wanting MORE challenge than the standard project
- If you're not in a group yet, can you join a group of three?
 - Groups of 2 or 4 are permitted if needed
 - We are using CANVAS to collect names into groups
 - If having already identified group members, PLEASE form a group via the Canvas signup

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.3

3

FEEDBACK - 3

- Is it possible to conduct the feedback on some online platform?
- EMAIL: Common for students to send email to ask questions on topics, assignments, or provide general feedback.
- Email feedback is worked in, in addition to the surveys
- Daily survey results are generally fast to compile

October 9, 2019



TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.4

4

INTRODUCTION TO CLOUD
COMPUTING

CHAPTER 3:
UNDERSTANDING CLOUD
COMPUTING



October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.5

5

OBJECTIVES - 2

- Introduction to Cloud Computing
 - Why study cloud computing?
 - History of cloud computing
 - Business drivers
 - Cloud enabling technologies
 - Terminology
 - Benefits of cloud adoption
 - Risks of cloud adoption

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.6

6

TECHNOLOGY INNOVATIONS
LEADING TO CLOUD

- What is cluster computing?
 - For scalability, what are cluster limitations?
 - For availability, what are cluster limitations?
- What is grid computing?
 - For scalability, what are grid limitations?
 - For availability, what are grid limitations?
- What does the cloud offer beyond a cluster?
- What does the cloud offer beyond the grid?
- What is virtualization?
 - Besides splitting many CPUs of dense servers, what other important features do VMs provide?

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.7

7

KEY TERMINOLOGY

- On-Premise Infrastructure
 - Local server infrastructure not configured as a cloud
- Cloud Provider
 - Corporation or private organization responsible for maintaining cloud
- Cloud Consumer
 - User of cloud services
- Scaling
 - Vertical scaling
 - Scale up: increase resources of a single virtual server
 - Scale down: decrease resources of a single virtual server
 - Horizontal scaling
 - Scale out: increase number of virtual servers
 - Scale in: decrease number of virtual servers

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.8

8

VERTICAL SCALING

- Reconfigure virtual machine to have different resources:
 - CPU cores
 - RAM
 - HDD/SDD capacity
- May require VM migration if physical host machine resources are exceeded

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.9

9

HORIZONTAL SCALING

- Increase (scale-out) or decrease (scale-in) number of virtual servers based on demand

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.10

10

HORIZONTAL VS VERTICAL SCALING

- What are the tradeoffs for ... ?
 - Cost
 - Latency time before resources are available
 - Configuration effort (for VMs, for disks...)
 - Hardware requirements to implement
 - Limited by... ?

Horizontal Scaling	Vertical Scaling

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.11

11

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.12

12

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
More latency before resources are available	IT resources typically instantly available

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.13

13

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
More latency before resources are available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.14

14

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
More latency before resources are available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed
Additional servers required	No additional servers required

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.15

15

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
More latency before resources are available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed
Additional servers required	No additional servers required
Not limited by individual server capacity, limited by the number of servers	Limited by individual server capacity

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.16

16

KEY TERMINOLOGY - 2

- Cloud services
 - Broad array of resources accessible "as-a-service"
 - Categorized as Infrastructure (IaaS), Platform (PaaS), Software (SaaS)
- Service-level-agreements (SLAs):
 - Establish expectations for: uptime, security, availability, reliability, and performance

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.17

17

GOALS AND BENEFITS

- Cloud providers
 - Leverage economies of scale through mass-acquisition and management of large-scale IT resources
 - Locate datacenters to optimize costs where electricity is low
- Cloud consumers
 - Key business/accounting difference:
 - Cloud computing enables anticipated capital expenditures to be replaced with operational expenditures
 - Operational expenditures always scale with the business
 - Eliminates need to invest in server infrastructure based on anticipated business needs
 - Businesses become more agile and lower their financial risks by eliminating large capital investments in physical infrastructure

October 9, 2019


TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.18

18

CLOUD BENEFITS - 2

- On demand access to pay-as-you-go resources on a short-term basis (less commitment)
- Ability to acquire “unlimited” computing resources on demand when required for business needs
- Ability to add/remove IT resources at a fine-grained level
- Abstraction of server infrastructure so applications deployments are not dependent on specific locations, hardware, etc.
 - The cloud has made our software deployments more agile...



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.19

19

CLOUD BENEFITS - 3

- Example: Using 100 servers for 1 hour costs the same as using 1 server for 100 hours
- Rosetta Protein Folding: Working with a UW-Tacoma graduate student, we recently deployed this science model across 5,900 compute cores on Amazon for 2-days...
- What is the cost to purchase 5,900 compute cores?**
- UW Tacoma Dell bioinformatics server purchase example: 20 cores on 2 servers for \$4,478...
- Using this ratio 5,900 cores costs \$1.3 million (purchase only)

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.20

20

OH YOU NEED MORE SERVERS?



INTERESTING... I HAVE SOMETHING TO SHOW YOU...

Gene Wilder, Charlie and the Chocolate Factory

October 9, 2019

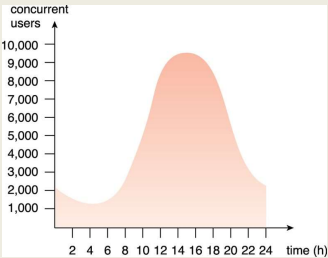
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.21

21

CLOUD BENEFITS

- Increased scalability
 - Example demand over a 24-hour day →
- Increased availability
- Increased reliability



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.22

22

CLOUD ADOPTION RISKS

- Increased security vulnerabilities**
 - Expansion of trust boundaries now include the external cloud
 - Security responsibility shared with cloud provider
- Reduced operational governance / control**
 - Users have less control of physical hardware
 - Cloud user does not directly control resources to ensure quality-of-service
 - Infrastructure management is abstracted
 - Quality and stability of resources can vary
 - Network latency costs and variability

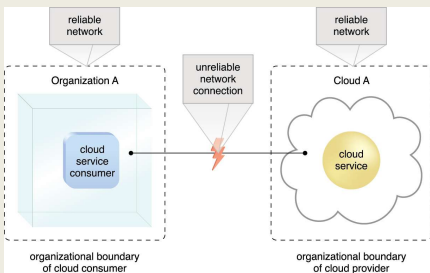
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.23

23

NETWORK LATENCY COSTS



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.24

24

CLOUD RISKS - 2

- **Performance monitoring of cloud applications**
 - Cloud metrics (AWS cloudwatch) support monitoring cloud infrastructure (network load, CPU utilization, I/O)
 - Performance of cloud applications depends on the health of aggregated cloud resources working together
 - User must monitor this aggregate performance
- **Limited portability among clouds**
 - Early cloud systems have significant “vendor” lock-in
 - Common APIs and deployment models are slow to evolve
 - Application containers help make applications more portable, but containers still must be deployed
- **Geographical issues**
 - Abstraction of cloud location leads to legal challenges with respect to laws for data privacy and storage

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.25

25

CLOUD: VENDOR LOCK-IN

The diagram shows a cloud consumer connected to two cloud providers, Cloud A (Cloud Provider X) and Cloud B (Cloud Provider Y). Cloud A supports message encryption and digital signatures. Cloud B supports message encryption only. The cloud consumer requires encryption and digital signing of messages. This creates a dependency on Cloud A's capabilities, leading to vendor lock-in.

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.26

26

CHAPTER 4: FUNDAMENTAL CONCEPTS AND MODELS

The image shows the cover of the book 'Cloud Computing: Concepts, Technology & Architecture' by Thomas H. Davenport and D.J. Patil. The cover features a network diagram and images of various devices connected to the cloud.

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.27

27

OBJECTIVES

- **From: Cloud Computing Concepts, Technology & Architecture:**
- Cloud Computing Concepts and Models
 - **Roles and boundaries**
 - Cloud characteristics
 - Cloud delivery models
 - Cloud deployment models

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.28

28

ROLES

- **Cloud provider**
 - Organization that provides cloud-based resources
 - Responsible for fulfilling SLAs for cloud services
 - Some cloud providers “resell” IT resources from other cloud providers
 - Example: Heroku sells PaaS services running atop of Amazon EC2
- **Cloud consumers**
 - Cloud users that consume cloud services
- **Cloud service owner**
 - Both cloud providers and cloud consumers can own cloud services
 - A cloud service owner may use a cloud provider to provide a cloud service (e.g. Heroku)

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.29

29

ROLES - 2

- **Cloud resource administrator**
 - Administrators provide and maintain cloud services
 - Both cloud providers and cloud consumers have administrators
- **Cloud auditor**
 - Third-party which conducts independent assessments of cloud environments to ensure security, privacy, and performance.
 - Provides unbiased assessments
- **Cloud brokers**
 - An intermediary between cloud consumers and cloud providers
 - Provides service aggregation
- **Cloud carriers**
 - Network and telecommunication providers which provide network connectivity between cloud consumers and providers

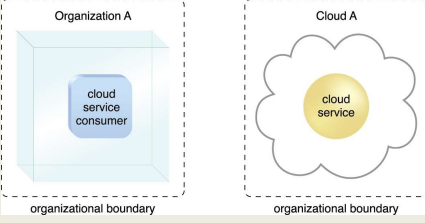
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.30

30

ORGANIZATION BOUNDARY



Organization A

Cloud A

cloud service consumer

cloud service

organizational boundary

organizational boundary

October 9, 2019

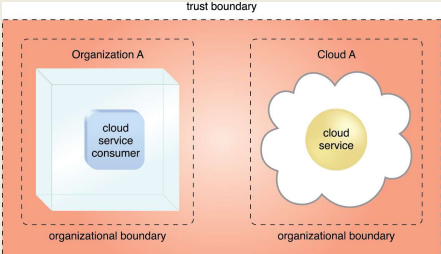
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.31

31

TRUST BOUNDARY

■ With cloud computing, trust boundary expands to encompass the organization and cloud provider(s).



trust boundary

Organization A

Cloud A

cloud service consumer

cloud service

organizational boundary

organizational boundary

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.32

32

OBJECTIVES

- **From: Cloud Computing Concepts, Technology & Architecture:**
- Cloud Computing Concepts and Models
 - Roles and boundaries
 - **Cloud characteristics**
 - Cloud delivery models
 - Cloud deployment models

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.33

33

CLOUD CHARACTERISTICS

■ Outline:

- On-demand usage
- Ubiquitous access
- Multitenancy (resource pooling)
- Elasticity
- Measured usage
- Resiliency

■ Assessing these features helps measure the value offered by a given cloud service or platform

October 9, 2019

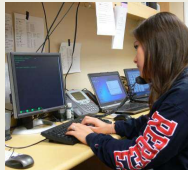
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.34

34

ON-DEMAND USAGE

- The freedom to self-provision IT resources
- Generally with automated support
- Automated support requires no human involvement
- Automation through software services interface



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.35

35

UBIQUITOUS ACCESS

- Cloud services are widely accessible
- Public cloud: internet accessible
- Private cloud: throughout segments of a company's intranet
- 24/7 availability

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.36

36

MULTITENANCY

- Cloud providers pool resources together to share them with many users
- Serve multiple cloud service consumers
- IT resources can be dynamically assigned, reassigned based on demand
- Multitenancy can lead to performance variation

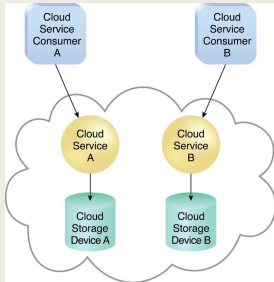
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.37

37

SINGLE TENANT MODEL



October 9, 2019

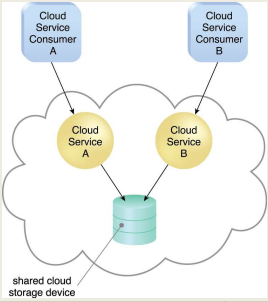
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.38

38

MULTITENANT MODEL

- Resource is "multiplexed" and share amongst multiple users
- Goal is to increase utilization
- Often server resources are underutilized
- There are many "sunk costs" whether usage is 0% or 100%
- Cloud computing tries to maximize "sunk cost" investments



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.39

39

MEASUREMENT STUDY OF SERVER UTILIZATION IN PUBLIC CLOUDS

H. Liu, A Measurement Study of Server Utilization In Public Clouds, Proc. 9th IEEE International Conference on Cloud and Green Computing (CAG'11), Sydney, Australia, Dec 2011, pp.435-442.

- H. Liu characterized CPU utilization across a public cloud by analyzing CPU temperature
- Liu's approach averages thermal measurements of the CPU from small VMs which are context switched across the physical host's CPU cores for extended periods to approximate CPU die temperature
- Local tests on private cluster established correlation between CPU die temperature and CPU utilization
- Using this approach Liu observed CPU utilization using 20 m1.small VMs on Amazon EC2 in 2011 for 1 week and estimated average CPU utilization of the physical hosts to be around 7.3%

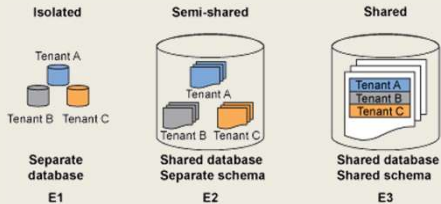
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.40

40

MULTITENANT DATABASE



October 9, 2019

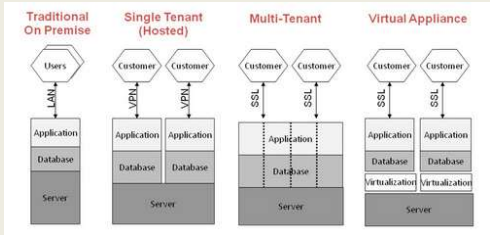
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.41

41

MULTITENANCY OF RESOURCES

Where is the multitenancy?



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.42

42

ELASTICITY

- Automated ability of cloud to transparently scale resources
- Scaling based on runtime conditions or pre-determined by cloud consumer or cloud provider
- Threshold based scaling
 - CPU-utilization > threshold_A, Response_time > 100ms
 - Application agnostic vs. application specific thresholds
 - Why might an application agnostic threshold be non-ideal?
- Load prediction
 - Historical models
 - Real-time trends

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

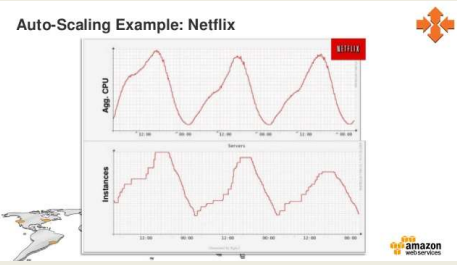
L5.43

43

PREDICTABLE DEMAND

- Example:

Auto-Scaling Example: Netflix



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.44

44

MEASURED USAGE

- Cloud platform tracks usage of IT resources
- For billing purposes
- Enables charging only for IT resources actually used
- Can be time-based (minute, hour, day)
- Can be throughput-based (MB, GB)

- Not all measurements are for billing
- Some measurements can support auto-scaling
- For example CPU utilization

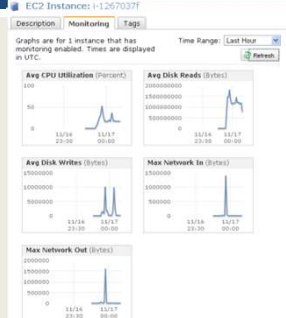
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.45

45

EC2 CLOUDWATCH METRICS



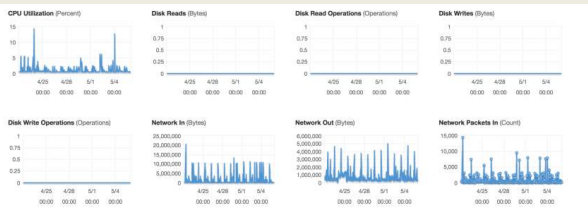
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.46

46

EC2 CLOUDWATCH METRICS



October 9, 2019

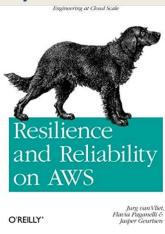
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.47

47

RESILIENCY

- Distributed redundancy across physical locations
- Used to improve reliability and availability of cloud-hosted applications
- Very much an engineering problem
- No “resiliency-as-a-service” for user deployed apps
- Unique characteristics of user applications make a one-size fits all service solution challenging



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.48

48

OBJECTIVES

- From: Cloud Computing Concepts, Technology & Architecture:
- Cloud Computing Concepts and Models
 - Roles and boundaries
 - Cloud characteristics
 - Cloud delivery models
 - Cloud deployment models

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.49

49

CLOUD DELIVERY MODELS

- What is the appropriate level of abstraction?
- How should applications be deployed?
 - IaaS, PaaS, SaaS, DbaaS, FaaS
- How do we ensure Quality-of-Service?
 - Performance, Availability, Responsiveness, Fault Tolerance
- How is scalability provided?
- How do we minimize hosting costs?
 - How do we estimate hosting costs?

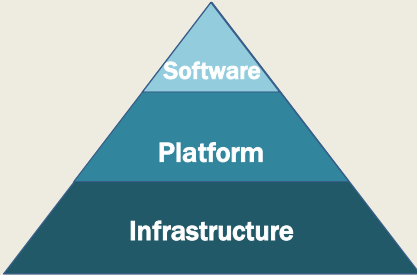
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.50

50

CLASSIC CLOUD DELIVERY MODELS



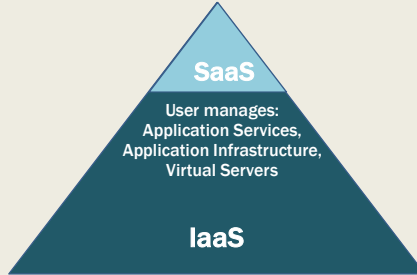
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.51

51

CLASSIC CLOUD DELIVERY MODELS



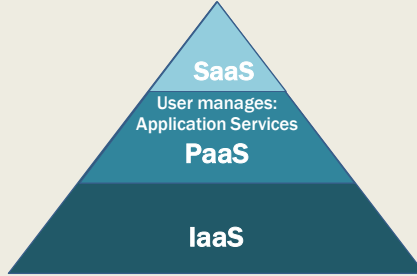
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.52

52

CLASSIC CLOUD DELIVERY MODELS



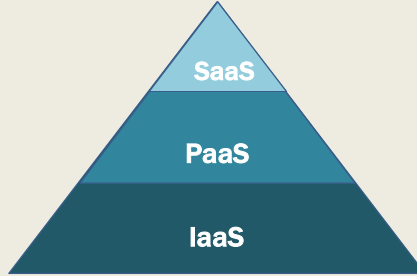
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.53

53

CLASSIC CLOUD DELIVERY MODELS



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.54

54

EXAMPLE CLOUD SERVICES

SAAS

Software as a Service

Email CRM Collaborative ERP

CONSUME

PAAS

Platform as a Service

Application Development Decision Support Legacy Web Streaming

BUILD ON IT

IAAS

Infrastructure as a Service

Caching File Networking Technical Security System Mgmt

MIGRATE TO IT

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.55

55

END USER APPLICATIONS

Many different "cloud" providers

Many cloud providers are also cloud consumers

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.56

56

INFRASTRUCTURE-AS-A-SERVICE

- Compute resources, on demand, as-a-service
 - Generally raw "IT" resources
 - Hardware, network, containers, operating systems
- Typically provided through virtualization
- Generally not-preconfigured
- Administrative burden is owned by cloud consumer
- Best when high-level control over environment is needed
- Scaling is generally **not** automatic...
- Resources can be managed in bundles
- AWS CloudFormation: Allows specification in JSON/YAML of cloud infrastructures

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.57

57

SC1 SC2 SC3 SC4

SC5 SC6 SC7

SC8 SC9 SC10

SC11 SC12 SC13

SC14 SC15

M: Tomcat ApplicationServer
D: Postgresql DB
F: nginx file server
L: Log server (Codebeamer)

58

58

SC1 SC2 SC3 SC4

SC5 SC6 SC7

SC14 SC15

M: Tomcat ApplicationServer
D: Postgresql DB
F: nginx file server
L: Log server (Codebeamer)

Bell's Number:

k: number of ways
n components can be
distributed across containers

n	k
4	15
5	52
6	203
7	877
8	4,140
9	21,147
n	...

59

59

SC1 SC2 SC3 SC4

SC5 SC6 SC7

SC14 SC15

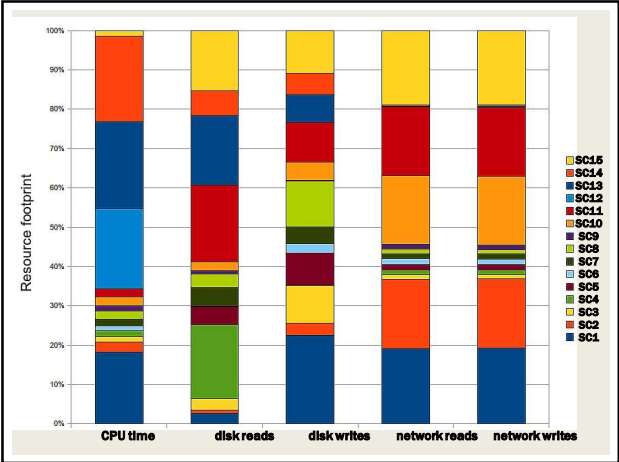
M: Tomcat ApplicationServer
D: Postgresql DB
F: nginx file server
L: Log server (Codebeamer)

Component Composition Example

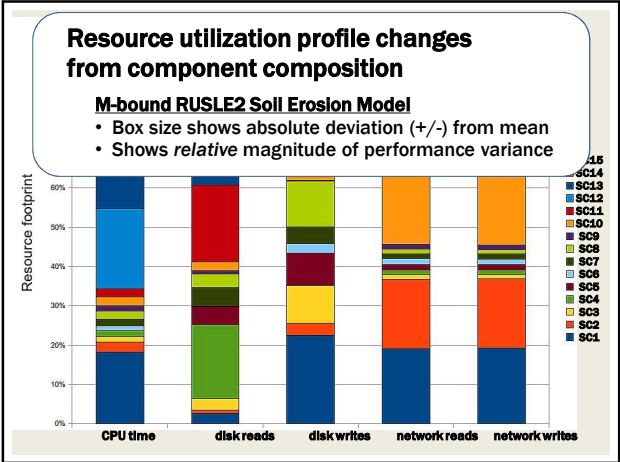
- An application with 4 components has 15 compositions
- One or more component(s) deployed to each VM
- Each VM launched to separate physical machine

60

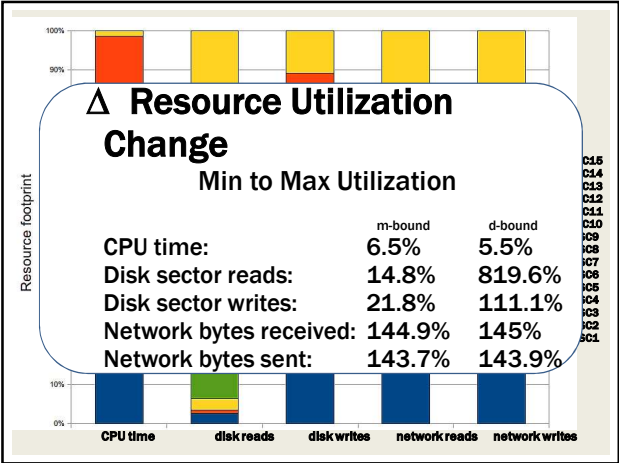
60



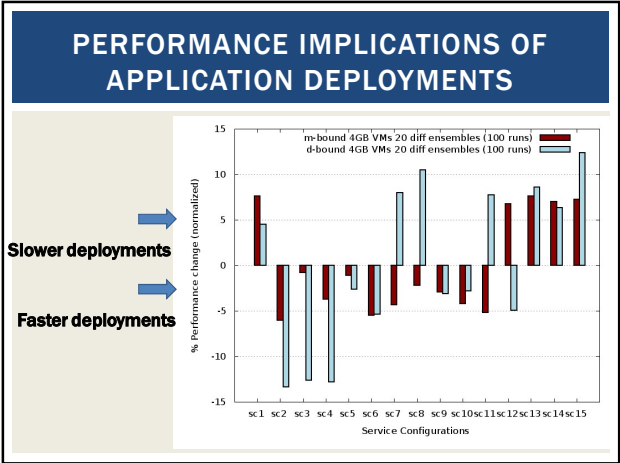
61



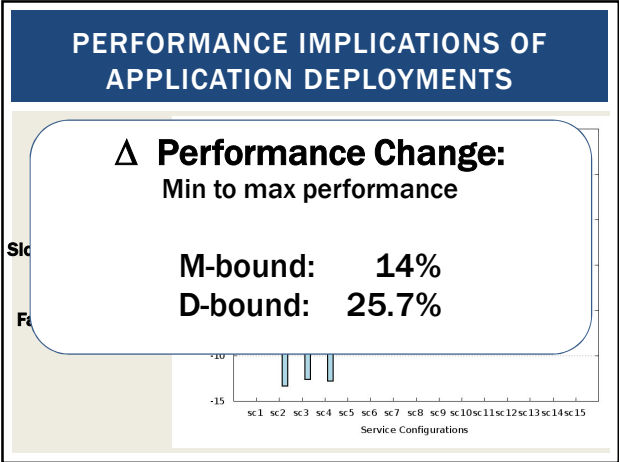
62



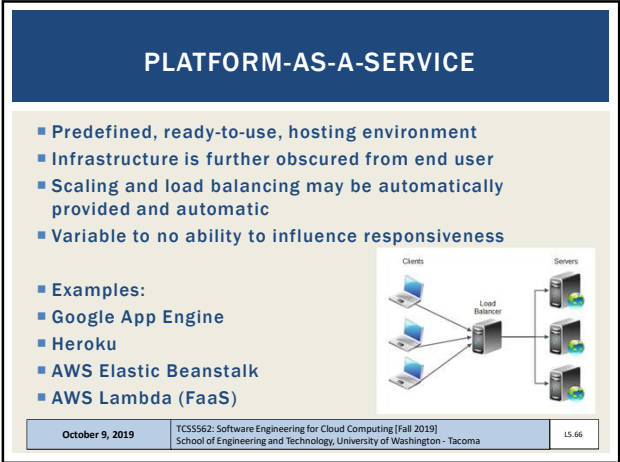
63



64



65



66

USES FOR PAAS

- Cloud consumer
 - Wants to extend on-premise environments into the cloud for "web app" hosting
 - Wants to entirely substitute an on-premise hosting environment
 - Cloud consumer wants to become a cloud provider and deploy its own cloud services to external users
- PaaS spares IT administrative burden compared to IaaS


October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.67

67

SERVERLESS COMPUTING



What is serverless?

Build and run applications without thinking about servers

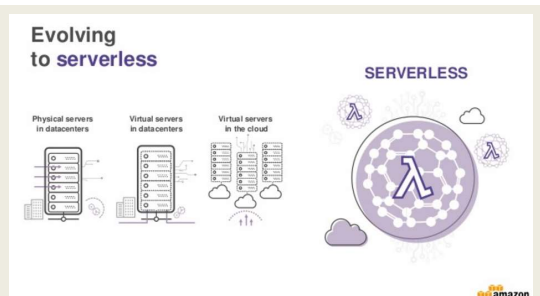
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.68

68

SERVERLESS COMPUTING - 2



Evolving to serverless

Physical servers in datacenters → Virtual servers in datacenters → Virtual servers in the cloud → **SERVERLESS**

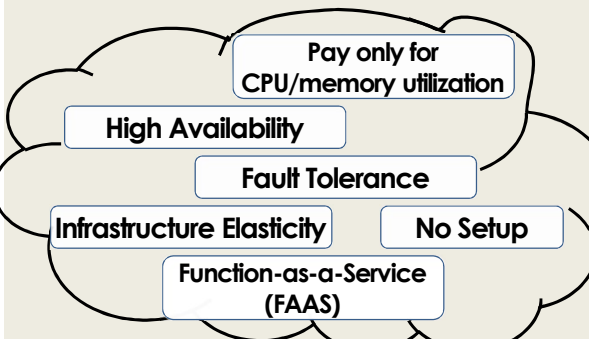
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.69

69

SERVERLESS COMPUTING



- Pay only for CPU/memory utilization
- High Availability
- Fault Tolerance
- Infrastructure Elasticity
- No Setup
- Function-as-a-Service (FAAS)

70

SERVERLESS COMPUTING

Why Serverless Computing?

Many features of distributed systems, that are challenging to deliver, are provided automatically

...they are built into the platform

71

SERVERLESS VS. FAAS

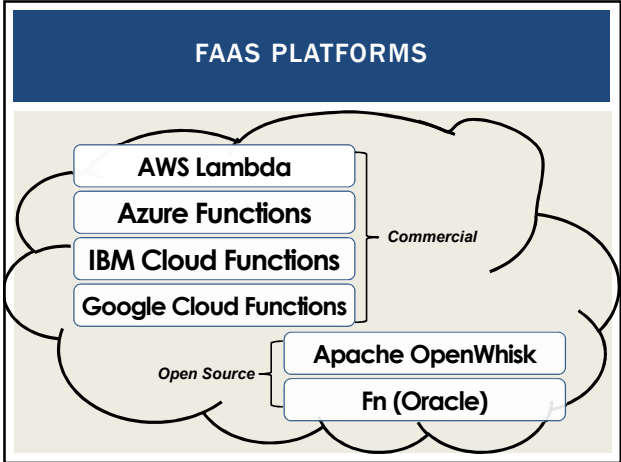
- Serverless Computing**
 - Refers to the avoidance of managing servers
 - Can pertain to a number of "as-a-service" cloud offerings
 - Function-as-a-Service (FaaS)
 - Developers write small code snippets (microservices) which are deployed separately
 - Database-as-a-Service (DBaaS)
 - Container-as-a-Service (CaaS)
 - Others...
- Serverless is a buzzword
- This space is evolving...

October 9, 2019

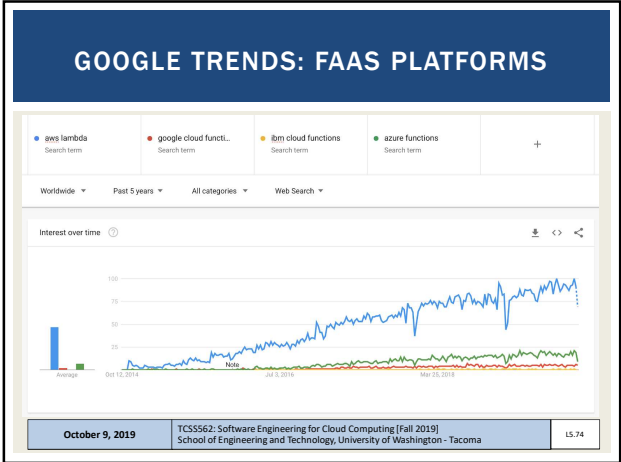
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.72

72



73



74

AWS LAMBDA

Using AWS Lambda

Bring your own code

- Node.js, Java, Python, C#
- Bring your own libraries (even native ones)

Simple resource model

- Select power rating from 128 MB to 3 GB
- CPU and network allocated proportionately

Flexible use

- Synchronous or asynchronous
- Integrated with other AWS services

Flexible authorization

- Securely grant access to resources and VPCs
- Fine-grained control for invoking your functions

75

FAAS PLATFORMS - 2

- New cloud platform for hosting application code
- Every cloud vendor provides their own:
 - AWS Lambda, Azure Functions, Google Cloud Functions, IBM OpenWhisk
- Similar to platform-as-a-service
- Replace opensource web container (e.g. Apache Tomcat) with abstracted vendor-provided **black-box** environment

76

FAAS PLATFORMS - 3

- Many challenging features of distributed systems are provided automatically
- **Built into the platform:**
- Highly availability (24/7)
- Scalability
- Fault tolerance

77

CLOUD NATIVE SOFTWARE ARCHITECTURE

■ Every service with a different pricing model

The diagram illustrates a weather application architecture. It starts with a user clicking a link to get local weather information. This triggers an API Gateway, which then calls a Lambda function (labeled 'Lambda is triggered'). The Lambda function interacts with a DynamoDB database to retrieve local weather information and returns the data back to the user. The architecture is shown with various cloud service icons and a pricing model for each service.

78

Slides by Wes J. Lloyd

L5.13


IAAS BILLING MODELS

- Virtual machines as-a-service at ¢ per hour
- No premium to scale:

1000 computers @ 1 hour

= 1 computer @ 1000 hours
- Illusion of infinite scalability to cloud user
- As many computers as you can afford
- Billing models are becoming increasingly granular
 - By the minute, second, 1/10th sec
- Auction-based instances: Spot instances →

Spot Instance Pricing History



Product: Linux/UNIX (Amazon VPC) Instance Type: c4.large

Time	Price
Oct 9	\$5.00
Oct 10	\$4.50
Oct 11	\$4.00
Oct 12	\$3.50
Oct 13	\$3.00
Oct 14	\$2.50
Oct 15	\$2.00
Oct 16	\$1.50
Oct 17	\$1.00
Oct 18	\$0.50
Oct 19	\$0.25
Oct 20	\$0.10
Oct 21	\$0.05
Oct 22	\$0.02
Oct 23	\$0.01
Oct 24	\$0.00
Oct 25	\$0.00
Oct 26	\$0.00
Oct 27	\$0.00
Oct 28	\$0.00
Oct 29	\$0.00
Oct 30	\$0.00
Oct 31	\$0.00

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.79

79

IAAS VS. FAAS COMPUTING BILLING MODELS

- AWS Lambda Pricing**
- FREE TIER:**

first 1,000,000 function calls/month → FREE

first 400,000 GB-sec/month → FREE
- Afterwards:** obfuscated pricing (AWS Lambda):

\$0.0000002 per request

\$0.000000208 to rent 128MB / 100-ms

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.80

80

WEBSERVICE HOSTING EXAMPLE

- Workload:** 1-month continuous 1-second service calls that fully utilize 3GB of RAM and two CPU cores
- ON AWS Lambda**
 - Each service call: 100% of 1 CPU-core
 - 100% of 3GB of memory
 - Workload: 2 continuous client threads
 - Duration: 1 month (30 days)
- ON AWS EC2:**
 - Amazon EC2 c4.large 2-vCPU VM@3.75GB
 - Hosting cost: \$72/month
 - c4.large: 10¢/hour, 24 hrs/day x 30 days
- How much would hosting this workload cost on AWS Lambda?**

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.81

81

PRICING OBFUSCATION

- Workload: 7,776,000 GB-sec
- FREE: - 400,000 GB-sec
- Ch...
- M...
- In...
- FF...
- AW...
- Ch...
- 1,776,000 calls
- Calls: \$0.32
- Total: \$123.28**
- BREAK-EVEN POINT = ~4,319,136 GB-sec-month**
For compute only, not considering cost of function calls = ~16.7 days

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.84

82

FAAS PRICING

- Break-even point is the point where renting VMs or deploying to a serverless platform (e.g. Lambda) is exactly the same.
- Our example is for one month
- Could also consider one day, one hour, one minute
 - What factors influence the break-even point for an application running on AWS Lambda?
 - What scenario would result in a 1-day break-even point where pricing for IaaS=FaaS?

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.83

83

FACTORS IMPACTING PERFORMANCE OF FAAS COMPUTING PLATFORMS

- Infrastructure elasticity
- Load balancing
- Provisioning variation
- Infrastructure retention: COLD vs. WARM
 - Infrastructure freeze/thaw cycle
- Memory reservation
- Service composition

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.84

84

FAAS CHALLENGES

- **Outline:**
- Vendor architectural lock-in – how to migrate?
- Pricing obfuscation – is it cost effective?
- Memory reservation – how much to reserve?
- Service composition – how to compose software?
- Infrastructure freeze/thaw cycle – how to avoid?

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.85

85

VENDOR ARCHITECTURAL LOCK-IN

- Cloud native (FaaS) software architecture requires external services/components

Example: Weather Application

The diagram shows a Client interacting with an S3 bucket (containing front-end code), an API Gateway, and a Lambda function. The Lambda function is triggered by a 35° C event and interacts with a DynamoDB database. The entire architecture is dependent on AWS services, leading to increased hosting costs.

■ Increased dependencies → increased hosting costs

Images credit: aws.amazon.com

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.85

86

PRICING OBFUSCATION

- **VM pricing:** hourly rental pricing, billed to nearest second is intuitive...
- **FaaS pricing:**

AWS Lambda Pricing

FREE TIER: first 1,000,000 function calls/month → FREE
first 400 GB-sec/month → FREE

■ Afterwards: \$0.0000002 per request
\$0.000000208 to rent 128MB / 100-ms

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.87

87

MEMORY RESERVATION QUEST

- Lambda memory reserved for functions
- UI provides “slider bar” to set function's memory allocation
- Resource capacity (CPU, disk, network) coupled to memory allocation: “every doubling of memory, doubles CPU...”
- But how much memory do model services require?

The screenshot shows the AWS Lambda console's 'Basic settings' tab. A red circle highlights the 'Memory (MB)' slider, which is set to 1536 MB. Below it, the 'Timeout' is set to 3 minutes. A red question mark icon is placed next to the 'Performance' label.

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.88

88

SERVICE COMPOSITION

- How should application code be composed for deployment to serverless computing platforms?

The diagram compares two deployment models. 'Monolithic Deployment' shows a single block representing the entire application. 'Serverless Composition' shows a flow of four functions (represented by Lambda icons) connected by arrows, with 'Client flow control' and 'Server flow control' labels. A red question mark icon is placed next to the 'Performance' label.

- Recommended practice: Decompose into many microservices
- Platform limits: code + libraries ~250MB
- How does composition impact the number of function invocations, and memory utilization?

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.89

89

INFRASTRUCTURE FREEZE/THAW CYCLE

- Unused infrastructure is deprecated
 - But after how long?
- Infrastructure: VMs, “containers”
- **Provider-COLD / VM-COLD**
 - “Container” images - built/transferred to VMs
- **Container-COLD**
 - Image cached on VM
- **Container-WARM**
 - “Container” running on VM

The image shows a highway with several potholes. A red question mark icon is placed next to the 'Performance' label.

Image from: Denver7 - The Denver Channel News

October 9, 2019

TCCS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.90

90



FUNCTION-AS-A-SERVICE

AWS
Lambda
Demo

91

SOFTWARE-AS-A-SERVICE

- Software applications as shared cloud service
- Nearly all server infrastructure management is abstracted away from the user
- Software is generally configurable
- SaaS can be a complete GUI/UI based environment
- Or UI-free (database-as-a-service)
- SaaS offerings
 - Google Docs
 - Office 365
 - Cloud9 Integrated Development Environment
 - Salesforce

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.92

92

The diagram illustrates the layers of cloud service models. At the top is the SaaS layer with three yellow circles labeled 'Cloud Service A', 'Cloud Service B', and 'Cloud Service C'. Below this is the PaaS layer with three boxes labeled 'Ready-Made Environment A', 'Ready-Made Environment B', and 'Ready-Made Environment C'. At the bottom is the IaaS layer with two boxes labeled 'Virtual Server A' and 'Virtual Server B', and one box labeled 'Physical Server A'. Dashed lines connect the cloud services to their respective ready-made environments, and the ready-made environments to the virtual servers. The virtual servers are connected to the physical server.

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.93

93

CONTAINER-AS-A-SERVICE

- Cloud service model for deploying application containers (e.g. Docker) to the cloud
- Deploy containers without worrying about managing infrastructure:
 - Servers
 - Or container orchestration platforms
- Container platform examples: Kubernetes, Docker swarm, Apache Mesos/Marathon, Amazon Elastic Container Service
- Container platforms support creation of container clusters on the using cloud hosted VMs
- CaaS Examples:
 - AWS Fargate
 - Azure Container Instances
 - Google KNative

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.94

94

OTHER CLOUD SERVICE MODELS

- IaaS
 - Storage-as-a-Service
- PaaS
 - Integration-as-a-Service
- SaaS
 - Database-as-a-Service
 - Testing-as-a-Service
 - Model-as-a-Service
- ?
 - Security-as-a-Service
 - Integration-as-a-Service

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.95

95

OBJECTIVES

- Cloud Computing Concepts and Models
 - Roles and boundaries
 - Cloud characteristics
 - Cloud delivery models
 - Cloud deployment models**

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.96

96

Cloud Deployment Models

- Distinguished by ownership, size, access
- Four common models
 - Public cloud
 - Community cloud
 - Hybrid cloud
 - Private cloud

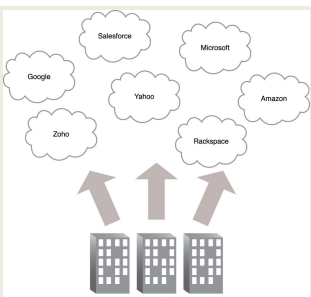
October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.97

97

Public Clouds



October 9, 2019

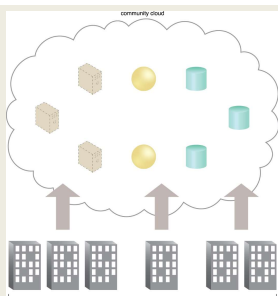
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.98

98

Community Cloud

- Specialized cloud built and shared by a particular community
- Leverage economies of scale within a community
- Research oriented clouds
- Examples:
 - Bionimbus - bioinformatics
 - Chameleon
 - CloudLab



October 9, 2019

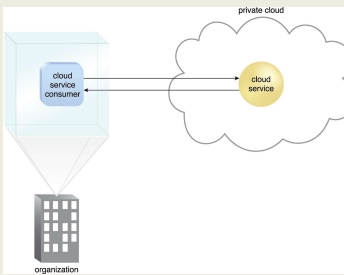
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.99

99

Private Cloud

- Compute clusters configured as IaaS cloud
- Open source software
 - Eucalyptus
 - Openstack
 - Apache Cloudstack
 - Nimbus
- Virtualization: XEN, KVM, ...



October 9, 2019

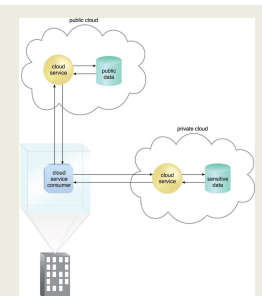
TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.100

100

Hybrid Cloud

- Extend private cloud typically with public or community cloud resources
- Cloud bursting: Scale beyond one cloud when resource requirements exceed local limitations
- Some resources can remain local for security reasons



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.101

101

Other Clouds

- Federated cloud
 - Simply means to aggregate two or more clouds together
 - Hybrid is typically private-public
 - Federated can be public-public, private-private, etc.
 - Also called inter-cloud
- Virtual private cloud
 - Google and Microsoft simply call these virtual networks
 - Ability to interconnect multiple independent subnets of cloud resources together
 - Resources allocated private IPs from individual network subnets can communicate with each other (10.0.1.0/24) and (10.0.2.0/24)
 - Subnets can span multiple availability zones within an AWS region


October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.102

102

TCSS 562
TERM PROJECT



October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.103

103

TCSS 562 TERM PROJECT

- Build a serverless cloud native application
- Application provides a case study to design trade-offs: Projects will compare and contrast one or more trade-offs:
- Service composition**
 - Switchboard architecture**
 - Address COLD Starts
 - Infrastructure Freeze/Thaw cycle of AWS Lambda (FaaS)
 - Full service isolation, full service aggregation
- Application flow control**
- Programming Languages**
- Alternate FaaS Platforms**
- Data provisioning**

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.104

104

EXTRACT TRANSFORM LOAD
DATA PIPELINE

- Service 1: TRANSFORM**
 - Read CSV file, perform some transformations
 - Write out new CSV file
- Service 2: LOAD**
 - Read CSV file, load data into relational database
 - Cloud DB (AWS Aurora), or local DB (Derby/SQLite)
 - Derby DB and/or SQLite code examples to be provided in Java

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.105

105

EXTRACT TRANSFORM LOAD
DATA PIPELINE 2

- Service 3: EXTRACT**
 - Using relational database, apply filter(s) and/or functions to aggregate data to produce sums, totals, averages
 - Output aggregations as JSON

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.106

106

SERVICE COMPOSITION

A	B	C	3 services Full Service Isolation
A	B	C	2 services
A	B	C	2 services
A	B	C	1 service Full Service Aggregation

Other possible compositions: group by library, functional cohesion, etc.

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.107

107

SWITCH-BOARD ARCHITECTURE

Single deployment package with consolidated codebase (Java: one JAR file)

Entry method contains "switchboard" logic
Case statement that route calls to proper service

Routing is based on data payload
Check if specific parameters exist, route call accordingly

Goal: reduce # of COLD starts to improve performance

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

L5.108

108

APPLICATION FLOW CONTROL

- **Serverless Computing:**
- AWS Lambda (FAAS: Function-as-a-Service)
- Provides HTTP/REST like web services
- Client/Server paradigm
- **Synchronous web service:**
- Client calls service
- Client blocks (freezes) and waits for server to complete call
- Connection is maintained in the "OPEN" state
- Problematic if service runtime is long!
 - Connections are notoriously dropped
 - System timeouts reached
- Client can't do anything while waiting unless using threads

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.109

109

APPLICATION FLOW CONTROL - 2

- **Asynchronous web service**
- Client calls service
- Server responds to client with OK message
- Client closes connection
- Server performs the work associated with the service
- Server posts service result in an external data store
 - AWS: S3, SQS (queueing service), SNS (notification service)

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.110

110

APPLICATION FLOW CONTROL - 3

The diagrams show different architectural patterns for microservices. (a) Client flow control: A Remote Client calls an API Gateway, which then calls multiple Microservices. (b) AWS Step Function: A Remote Client calls an AWS Step Function, which orchestrates multiple Microservices. (c) Microservice as controller: A Remote Client calls an API Gateway, which calls a Controller, which then calls multiple Microservices. (d) Asynchronous: A Remote Client calls an API Gateway, which calls Microservices. The Microservices use asynchronous calls and polling to interact with a Message Queue.

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.111

111

PROGRAMMING LANGUAGE

- Function-as-a-Service platforms support hosting services code in multiple languages
- AWS Lambda- common: Java, Node.js, Python
 - Plus others: Go, PowerShell, C#, and Ruby
- Also Runtime API ("BASH") which allows deployment of any binary executable in any programming languages
- Jackson D, Lynch G. An Investigation of the Impact of Language Runtime on the Performance and Cost of Serverless Functions. In Proc. Of the 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion) 2018 Dec 17 (pp. 154-160).
- <http://faculty.washington.edu/wlloyd/courses/tcss562/papers/AnInvestigationOfTheImpactOfLanguageRuntimeOnThePerformanceAndCostOfServerlessFunctions.pdf>

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.112

112

FAAS PLATFORMS

- Many commercial and open source FaaS platforms exist
- TCSS562 projects can choose to compare performance and cost implications of alternate platforms.
- Supported by SAAF:
- AWS Lambda
- Google Cloud Functions
- Azure Functions
- IBM Cloud Functions

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.113

113

DATA PROVISIONING

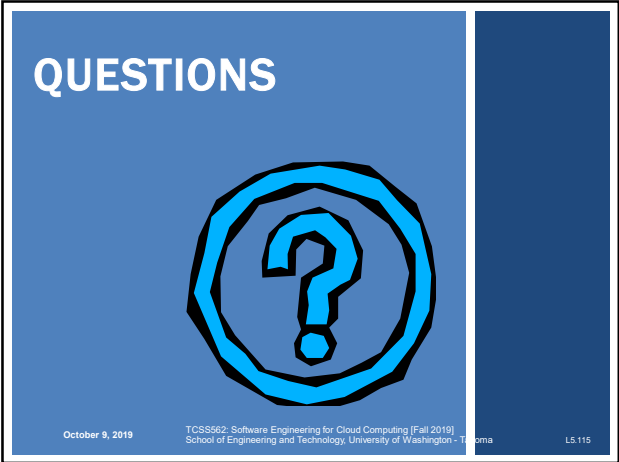
- Consider performance and cost implications of the data-tier design for the serverless application
- Use different tools as the relational datastore to support service #2 (LOAD) and service #3 (EXTRACT)
- **SQL / Relational:**
- Amazon Aurora (serverless cloud DB), Amazon RDS (cloud DB), DB on a VM (MySQL), DB inside Lambda function (SQLite, Derby)
- **NO SQL / Key/Value Store:**
- Dynamo DB, MongoDB, S3

October 9, 2019

TCSS562: Software Engineering for Cloud Computing [Fall 2019]
School of Engineering and Technology, University of Washington - Tacoma

LS.114

114



115