TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Cloud Enabling Technology III
Tutorial 5 Demo

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

1

OFFICE HOURS – FALL 2023

- **THIS WEEK**
- Campus is closed Friday November 10, due to the Veteran's Day holiday
- **Tuesdays:**
  - 2:30 to 3:30 pm - CP 229
- *** Thursday ***
  - 6:00 pm to 7:00 pm – CP 229 and via Zoom
- Or email for appointment

> Office Hours set based on Student Demographics survey feedback

2

NEXT WEEK

- Tuesday November 14th
- The class will meet and begin with the usual review, but instead of new lecture, we will focus on tutorial demonstrations and questions to provide a catch-up day
- Thursday November 16th
- We will feature lecture on containerization

3

OBJECTIVES – 11/9

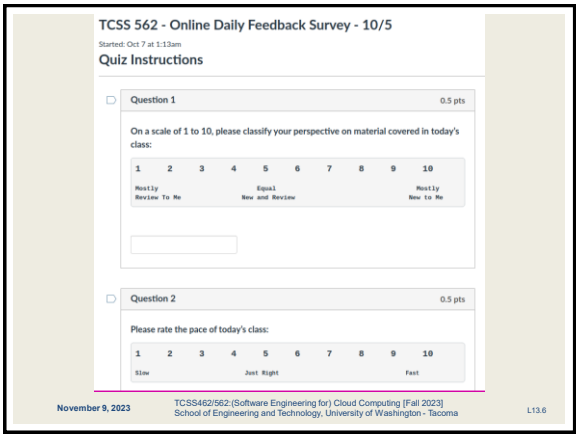
- **Questions from 11/7**
- Tutorials Questions
- Class Presentations: Cloud Technology or Research Paper Review
- Ch. 5: Cloud Enabling Technology
- Tutorial 5 Demo

4

ONLINE DAILY FEEDBACK SURVEY

- Daily Feedback Quiz in Canvas – Take After Each Class
- Extra Credit for completing

5



6

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (**50** respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.20 (↓ - *previous* 6.11)**
- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.82 (↓ - previous 5.52)**
- **Response rates:**
- TCSS 462: 32/44 – 72.72%
- TCSS 562: 18/25 – 72.00%

7

FEEDBACK FROM 11/7

- ***OS containers still remain unclear to me. How do they differ from a VM ?***
- Linux containers differ than VMs in that all instances share the same Linux operating system kernel
- All processes and threads across all running OS containers on the host must be scheduled through the same Linux kernel
- If all containers on a host share the same Linux kernel, how is isolation different than sharing the host using virtual machines?
- When you run Windows and Virtual Box Linux on the same computer, do the operating systems share the same kernel?
- Containers partition the Linux host into distinct sand boxes so that each container has a private view of only its resources

8

FEEDBACK - 2

- ***Can OS containers perform the same operations as a VM ?***
- OS containers run a full set of OS processes to mimic a Virtual Machine
- With many OS container instances, common OS processes are duplicated in every container increasing memory consumption and overhead
  - How many processes does your Linux Virtual Box VM run when booted? Check with 'top'
  - Each OS container runs a few hundred processes like a VM
- With more processes on the host, there is more context switching between processes on the CPU, and more performance overhead
  - More processes may also lead to more memory page faults

9

AWS CLOUD CREDITS UPDATE

- AWS CLOUD CREDITS ARE NOW AVAILABLE FOR TCSS 462/562
- Credits provided on request with expiry of Sept 30, 2024
- Credit codes must be securely exchanged
- Request codes by sending an email with the subject "**AWS CREDIT REQUEST**" to **wlloyd@uw.edu**
- Codes can also be obtained in person (or zoom), in the class, during the breaks, after class, during office hours, by appt
  - 61 credit requests fulfilled as of Nov 6 @ 11:59p
- Codes not provided using discord

10

OBJECTIVES – 11/9

- Questions from 11/7
- **Tutorials Questions**
- Class Presentations:
  Cloud Technology or Research Paper Review
- Ch. 5: Cloud Enabling Technology
- Tutorial 5 Demo

11

TUTORIAL 0

- Getting Started with AWS
- http://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2023_tutorial_0.pdf
- Create an AWS account
- Create account credentials for working with the CLI
- Install awsconfig package
- Setup awsconfig for working with the AWS CLI

12

Don't Forget to Terminate (Shutdown) all EC2 instances for Tutorials 3

Spot instances:
c5d.large instance @ ~2 cents / hour

$0.48 / day
$3.36 / week
$14.60 / month
$175.20 / year

AWS CREDITS → → → → → → → →

13

TUTORIAL 4 – DUE NOV 7

- Introduction to AWS Lambda with the Serverless Application Analytics Framework (SAAF)
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2023_tutorial_4.pdf (link to be posted)
- Obtaining a Java development environment
- Introduction to Maven build files for Java
- Create and Deploy "hello" Java AWS Lambda Function
  - Creation of API Gateway REST endpoint
- Sequential testing of "hello" AWS Lambda Function
  - API Gateway endpoint
  - AWS CLI Function invocation
- Observing SAAF profiling output
- Parallel testing of "hello" AWS Lambda Function with faas_runner
- Performance analysis using faas_runner reports
- Two function pipeline development task

14

TUTORIAL 4 - RESUBMISSION

- Sometimes students get unexpected results when adding **Thread.sleep(10000)** to the hello Lambda function:
- It is possible that:

1. Results from the SAAF Report Generator were from a test run before the **Thread.Sleep()** statement was added to the code
   - OR -
2. The **Thread.Sleep()** statement was added in the incorrect location of the code
   - OR -
3. When opening the CSV output from the Report Generator, the file separator characters were set incorrectly.

- The only separator for a CSV file is the comma ","
  Be sure to correctly open the CSV file in the spreadsheet.
  Columns can be offset resulting in the wrong answers being provided for Question 6.

15

TUTORIAL 4 – RESUBMISSION - 2

- The sleep statement must go between the START FUNCTION and END FUNCTION comments in the handleRequest() method specified as the AWS Lambda function's handler under runtime settings in the AWS Lambda GUI.

```
//***************START FUNCTION IMPLEMENTATION*************************
    try
    {
        Thread.sleep(10000);
    }
    catch (InterruptedException ie)
    {
        System.out.println("Interruption occurred while sleeping.");
    }
//***************END FUNCTION IMPLEMENTATIO N**************************
```

- Code must be recompiled and redeployed after modification

16

TUTORIAL 4 – RESUBMISSION - 3

- **SANITY CHECK:** consider that adding 10 seconds of sleep to your AWS Lambda function will cause the function to run for at least 10 seconds. This will impact the outputs requested for Question 6:

- **avg_runtime** is the server-side (cloud) runtime of the function
- This is the time it takes for the function to run on AWS Lambda (cloud)
- Adding sleep of 10 seconds should increase a function's **avg_runtime**

- **avg_roundTripTime** is the total time for a request from a client (laptop?) to travel to the server (cloud), make the function call, and return.
- If trying to make 50 calls at once on a laptop with a small # of CPU cores this time may be slow
- Adding sleep of 10 seconds should increase a function's **avg_roundTripTime**

17

TUTORIAL 4 – RESUBMISSION - 4

- **avg_cpuIdleDelta** time is the amount of time the Lambda function's Firecracker two vCPUs are idle during the function call on the server measured in centiseconds:

100 centiseconds = 1 second
100 centiseconds = 1000 milliseconds

- By default, AWS Lambda functions with 512 MB run in a runtime environment with access to two vCPU cores
- This is the total vCPU idle time for both cores (it is doubled)
- Adding sleep of 10 seconds should increase your function's avg_cpuIdleDelta

- **How much should avg_cpuIdleDelta increase ?**

18

TUTORIAL 5 – DUE NOV 14

- Introduction to Lambda II: Working with Files in S3 and CloudWatch Events
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2023_tutorial_5.pdf
- Customize the Request object (add getters/setters)
  - Why do this instead of HashMap ?
- Import dependencies (jar files) into project for AWS S3
- Create an S3 Bucket
- Give your Lambda function(s) permission to work with S3
- Write to the CloudWatch logs
- Use of CloudTrail to generate S3 events
- Creating CloudWatch rule to capture events from CloudTrail
- Have the CloudWatch rule trigger a target Lambda function with a static JSON input object (hard-coded filename)
- **Optional**: for the S3 PutObject event, dynamically extract the name of the file put to the S3 bucket for processing

19

TUTORIAL 6 – NOV 21

- Introduction to Lambda III: Serverless Databases
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2023_tutorial_6.pdf
- Create and use Sqlite databases using sqlite3 tool
- Deploy Lambda function with Sqlite3 database under /tmp
- Compare in-memory vs. file-based Sqlite DBs on Lambda
- Create an Amazon Aurora "Serverless" v2 MySQL database
- Using an ec2 instance in the same VPC (Region + availability zone) connect and interact with the database using the mysql CLI app
- Deploy an AWS Lambda function that uses the MySQL "serverless" database

20

TUTORIAL 7

- Introduction to Docker
- (to be posted) https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2023_tutorial_7.pdf
- Must complete using Ubuntu 22.04 (for cgroups v2)
- Use DOCX file for copying and pasting Docker install commands
- Topics:
  - Installing Docker
  - Creating a container using a Dockerfile
  - Using cgroups virtual filesystem to monitor CPU utilization of a container
  - Persisting container images to Docker Hub image repository
  - Container vertical scaling of CPU/memory resources
  - Testing container CPU and memory isolation

21

TUTORIAL COVERAGE

- **Docker CLI → Docker Engine (dockerd) → containerd → runc**
- Working with the docker CLI:
- docker run create a container
- docker ps -a list containers, find CONTAINER ID
- docker exec --it run a process in an existing container
- docker stop stop a container
- docker kill kill a container
- docker help list available commands
- man docker Docker Linux manual pages

22

```
Commands:
  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  deploy      Deploy a new stack or update an existing stack
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  images      List images
  import      Import the contents from a tarball to create a filesystem image
  info        Display system-wide information
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  login       Log in to a Docker registry
  logout      Log out from a Docker registry
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  run         Run a command in a new container
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  search      Search the Docker Hub for images
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  version     Show the Docker version information
  wait        Block until one or more containers stop, then print their exit codes
```

Docker CLI

23

TUTORIAL 7

- Tutorial introduces use of two common Linux performance benchmark applications
- stress-ng
- 100s of CPU, memory, disk, network stress tests
- Sysbench
- Used in tutorial for memory stress test

24

OBJECTIVES – 11/9

- Questions from 11/7
- Tutorials Questions
- **Class Presentations:**
  Cloud Technology or Research Paper Review
- Ch. 5: Cloud Enabling Technology
- Tutorial 5 Demo

25

GROUP PRESENTATION

- TWO OPTIONS:
- ***Cloud technology presentation***
- ***Cloud research paper presentation***
  - Recent & suggested papers will be posted at:
    http://faculty.washington.edu/wlloyd/courses/tcss562/papers/
- Submit presentation type and topics (paper or technology) with desired dates of presentation via Canvas by:
  Friday November 17th @ 11:59pm
- Presentation dates:
  - Tuesday November 28, Thursday November 30
  - Tuesday December 5, Thursday December 7

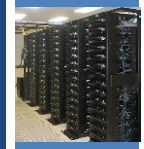
26

OBJECTIVES – 11/9

- Questions from 11/7
- Tutorials Questions
- Class Presentations:
  Cloud Technology or Research Paper Review
- **Ch. 5: Cloud Enabling Technology**
- Tutorial 5 Demo

27

CLOUD ENABLING TECHNOLOGY



28

CLOUD ENABLING TECHNOLOGY

- Broadband networks and internet architecture
- Data center technology
- **Virtualization technology**
- Multitenant technology
- Web/web services technology

29

VIRTUALIZATION MANAGEMENT

- Virtual infrastructure management (VIM) tools
- Tools that manage pools of virtual machines, resources, etc.
- Private cloud software systems can be considered as a VIM
- Considerations:
- Performance overhead
  - Paravirtualization: custom OS kernels, I/O passed directly to HW w/ special drivers
- Hardware compatibility for virtualization
- Portability: virtual resources tend to be difficult to migrate cross-clouds

30

VIRTUAL INFRASTRUCTURE MANAGEMENT (VIM)

- Middleware to manage virtual machines and infrastructure of IaaS "clouds"
- Examples
  - OpenNebula
  - Nimbus
  - Eucalyptus
  - OpenStack

31

VIM FEATURES

- Create/destroy VM Instances
- Image repository
  - Create/Destroy/Update images
  - Image persistence
- Contextualization of VMs
  - Networking address assignment
    - DHCP / Static IPs
  - Manage SSH keys

32

VIM FEATURES - 2

- Virtual network configuration/management
  - Public/Private IP address assignment
  - Virtual firewall management
    - Configure/support isolated VLANs (private clusters)
- Support common virtual machine managers (VMMs)
  - XEN, KVM, VMware
  - Support via libvirt library

33

VIM FEATURES - 3

- Shared "Elastic" block storage
  - Facility to create/update/delete VM disk volumes
    - Amazon EBS
    - Eucalyptus SC
    - OpenStack Volume Controller

34

CONTAINER ORCHESTRATION FRAMEWORKS

- Middleware to manage Docker application container deployments across virtual clusters of Docker hosts (VMs)
- Considered Infrastructure-as-a-Service
- **Opensource**
- Kubernetes framework
- Docker swarm
- Apache Mesos/Marathon
- **Proprietary**
- Amazon Elastic Container Service

35

CONTAINER SERVICES

- **Public cloud container cluster services**
- Azure Kubernetes Service (AKS)
- Amazon Elastic Container Service for Kubernetes (EKS)
- Google Kubernetes Engine (GKE)
- **Container-as-a-Service**
- Azure Container Instances (ACI – April 2018)
- AWS Fargate (November 2017)
- Google Kubernetes Engine Serverless Add-on (alpha-July 2018)

36

CLOUD ENABLING TECHNOLOGY

- *Adapted from Ch. 5 from Cloud Computing Concepts, Technology & Architecture*
- Broadband networks and internet architecture
- Data center technology
- Virtualization technology
- **Multitenant technology**
- Web/web services technology

37

4. MULTITENANT APPLICATIONS

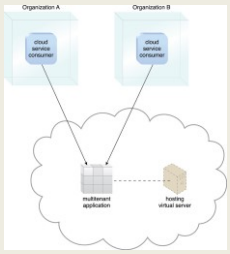
- Each tenant (like in an apartment) has their own view of the application
- Tenants are unaware of their neighbors
- Tenants can only access their data, no access to data and configuration that is not their own

- Customizable features
  - UI, business process, data model, access control
- Application architecture
  - User isolation, data security, recovery/backup by tenant, scalability for a tenant, for tenants, metered usage, data tier isolation

38

MULTITENANT APPS - 2

- Forms the basis for SaaS (applications)



39

CLOUD ENABLING TECHNOLOGY

- *Adapted from Ch. 5 from Cloud Computing Concepts, Technology & Architecture*
- Broadband networks and internet architecture
- Data center technology
- Virtualization technology
- Multitenant technology
- **Web/web services technology**

40

5. WEB SERVICES/WEB

- Web services technology is a key foundation of cloud computing's "**as-a-service**" cloud delivery model
- SOAP – "Simple" object access protocol
  - First generation web services
  - WSDL – web services description language
  - UDDI – universal description discovery and integration
  - SOAP services have their own unique interfaces
- REST – instead of defining a custom technical interface REST services are built on the use of HTTP protocol
- HTTP GET, PUT, POST, DELETE

41

HYPERTEXT TRANSPORT PROTOCOL (HTTP)

- An ASCII-based request/reply protocol for transferring information on the web
- HTTP request includes:
  - request method (GET, POST, etc.)
  - Uniform Resource Identifier (URI)
  - HTTP protocol version understood by the client
  - headers—extra info regarding transfer request
- HTTP response from server
  - Protocol version & status code →
  - Response headers
  - Response body

HTTP status codes:
2xx — *all is well*
3xx — *resource moved*
4xx — *access problem*
5xx — *server error*

42

REST: REPRESENTATIONAL STATE TRANSFER

- Web services protocol
- *Supersedes SOAP* – Simple Object Access Protocol
- Access and manipulate web resources with a predefined set of stateless operations (known as web services)
- Requests are made to a URI
- Responses are most often in JSON, but can also be HTML, ASCII text, XML, no real limits as long as text-based
- HTTP verbs: GET, POST, PUT, DELETE, …

43

```
// SOAP REQUEST

POST /InStock HTTP/1.1
Host: www.bookshop.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Body xmlns:m="http://www.bookshop.org/prices">
  <m:GetBookPrice>
    <m:BookName>The Fleamarket</m:BookName>
  </m:GetBookPrice>
</soap:Body>
</soap:Envelope>
```


44

```
// SOAP RESPONSE
POST /InStock HTTP/1.1
Host: www.bookshop.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Body xmlns:m="http://www.bookshop.org/prices">
  <m:GetBookPriceResponse>
    <m: Price>10.95</m: Price>
  </m:GetBookPriceResponse>
</soap:Body>
</soap:Envelope>
```


45

```
// WSDL Service Definition
<?xml version="1.0" encoding="UTF-8"?>
<definitions  name ="DayOfWeek"
  targetNamespace="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl"
  xmlns:tns="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <message name="DayOfWeekInput">
    <part name="data" type="xsd:data"/>
  </message>
  <message name="DayOfWeekResponse">
    <part name="dayOfWeek" type="xsd:string"/>
  </message>
  <portType name="DayOfWeekPortType">
    <operation name="GetDayOfWeek">
      <input message="tns:DayOfWeekInput"/>
      <output message="tns:DayOfWeekResponse"/>
    </operation>
  </portType>
  <binding name="DayOfWeekBinding" type="tns:DayOfWeekPortType">
    <soap:binding style="document"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetDayOfWeek">
      <soap:operation soapAction="getdayofweek"/>
      <input>
        <soap:body use="encoded"
          namespace="http://www.roguewave.com/soapworx/examples"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </input>
      <output>
        <soap:body use="encoded"
        namespace="http://www.roguewave.com/soapworx/examples"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </output>
    </operation>
  </binding>
  <service name="DayOfWeekService" >
    <documentation>
      Returns the day-of-week name for a given date
    </documentation>
    <port name="DayOfWeekPort" binding="tns:DayOfWeekBinding">
      <soap:address location="http://localhost:8090/dayofweek/DayOfWeek"/>
    </port>
  </service>
</definitions>
```


46

REST CLIMATE SERVICES EXAMPLE

- USDA Lat/Long Climate Service Demo
- Just provide a Lat/Long

```
// REST/JSON
// Request climate data for Washington

{
 "parameter": [
  {
    "name": "latitude",
    "value":47.2529
  },
  {
    "name": "longitude",
    "value":-122.4443
  }
  ]
}
```


47

REST - 2

- App manipulates one or more types of resources.
- Everything the app does can be characterized as some kind of operation on one or more resources.
- Frequently services are CRUD operations (create/read/update/delete)
  - Create a new resource
  - Read resource(s) matching criterion
  - Update data associated with some resource
  - Destroy a particular a resource
- Resources are often implemented as objects in OO languages

48

REST ARCHITECTURAL ADVANTAGES

- **Performance**: component interactions can be the dominant factor in user-perceived performance and network efficiency
- **Scalability**: to support large numbers of services and interactions among them
- **Simplicity**: of the Uniform Interface
- **Modifiability**: of services to meet changing needs (even while the application is running)
- **Visibility**: of communication between services
- **Portability**: of services by redeployment
- **Reliability**: resists failure at the system level as redundancy of infrastructure is easy to ensure

49

WE WILL RETURN AT ~4:50 PM

50

OBJECTIVES – 11/9

- Questions from 11/7
- Tutorials Questions
- Class Presentations:
  Cloud Technology or Research Paper Review
- Ch. 5: Cloud Enabling Technology
- **Tutorial 5 Demo**

51

QUESTIONS

52