

TCSS 360: SOFTWARE DEVELOPMENT AND QUALITY ASSURANCE

The Architecture of SAAS Applications

Wes J. Lloyd
 Institute of Technology
 University of Washington - Tacoma



FEEDBACK - JAN 9TH

- What point(s) remain least clear to you?
 - Frameworks vs. libraries?
 - Design patterns
 - Service oriented architecture
 - PaaS v.s. SaaS
- Project:
 - How are groups formed?
... semi-balanced teams geography & exp.
 - Requirements for project? forthcoming...

January 9, 2017	TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma	L2.2
-----------------	---	------

FEEDBACK - 2

- SOAP Services
 - SOAP is its own protocol like HTTP
 - Protocol runs atop TCP/IP to provide a dialect for web services
- REST Services
 - Rather than invent a new protocol, REST is implemented on top of HTTP
- Virtualization vs. Containerization
- I would like to better understand containers
- All the slides don't seem to fit together:
 - We are jumping between Software Engineering and SOA/SaaS/Cloud

January 9, 2017	TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma	L2.3
-----------------	---	------

```
// 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>
```

January 9, 2017	TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma	L2.4
-----------------	---	------

```
// 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>
```

January 9, 2017	TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma	L2.5
-----------------	---	------

```
// WSDL Service Definition
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="DayOfWeek"
targetNamespace="http://www.roguewave.com/soapwzr/examples/DayOfWeek.wsdl"
xmlns:tns="http://www.roguewave.com/soapwzr/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="date" type="xsd:date"/>
  </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/soapwzr/examples"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
        </input>
      <output>
        <soap:body use="encoded"
      namespace="http://www.roguewave.com/soapwzr/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>
```

January 9, 2017	TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma	L2.6
-----------------	---	------

```

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

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

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.7

OBJECTIVES

- Software-as-a-Service architecture
- TCP/IP, HTTP, REST/JSON
- Examples

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.8

SAAS ARCHITECTURE

\$2.1 100,000 feet
 • Client-server (vs. P2P)
\$2.2 50,000 feet
 • HTTP & URIs
\$2.3 10,000 feet
 • XHTML & CSS
\$2.4 5,000 feet
 • 3-tier architecture
 • Horizontal scaling
\$2.5 1,000 feet—Model-View-Controller
 (vs. Page Controller, Front Controller)
\$2.6 500 feet: Active Record models (vs. Data Mapper) • Active Record • REST • Template View
\$2.7 500 feet: RESTful controllers (Representational State Transfer for self-contained actions) • Data Mapper • Transform View
\$2.8 500 feet: Template View (vs. Transform View)

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.9

TCP/IP PROTOCOLS

- IP (Internet Protocol) address identifies a physical network interface with four octets, e.g. 128.32.244.172 (ipv4)
- Special address 127.0.0.1 is "this computer", named **localhost**, even if not connected to the Internet!
- TCP/IP (Transmission Control Protocol/Internet Protocol)
- IP**: no-guarantee, best-effort service that delivers packets from one IP address to another
- TCP**: make IP reliable by detecting "dropped" packets, data arriving out of order, transmission errors, slow networks, etc., and respond appropriately
- TCP ports** allow multiple TCP apps on same computer

GET /bears/ → GET /bears/
 HTTP/0.9 200 OK ← HTTP/0.9 200 OK

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.10

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

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.11

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, ...

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma L2.12

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

January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.13

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

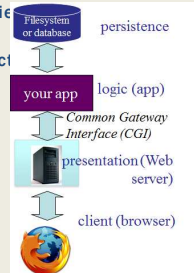
January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.14

PROGRAMMING SAAS

- Programming frameworks and libraries support common tasks
 - “map” URI to correct program & function
 - pass arguments
 - invoke program on server
 - handle persistent storage
 - handle cookies
 - handle errors
 - package output back to user

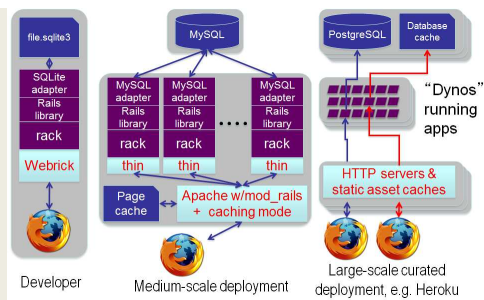


January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.15

SAAS DEPLOYMENTS



January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.16

COOKIES

- Observation: HTTP is stateless
- Early Web 1.0 problem: how to guide a user “through” a flow of pages?
- Use IP addresses to identify returning user?
 - ✘ public computers (e.g. library), users sharing single IP
- Embed per-user junk into URI query string?
 - ✘ breaks caching
 - This was done (and still occasionally is) ...
- Quickly superseded by cookies
- Watch: screencast.saasbook.info – Screencast 2.2.1
- Most frameworks manage tamper-evident cookies for you

January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.17

IN SUMMARY...

- **SAAS Architecture:** Harnesses the Model View Controller design pattern to provide separation of concerns to enable building decoupled scalable software deployable using modern computing infrastructure (e.g. clusters, clouds, containers, VMs)
- **TCP/IP, HTTP, REST/JSON:** Stateless services are supported using HTTP on top of TCP which are simple to write, use, change, deploy, scale, and combine.

January 9, 2017

TCSS360: Software Development and Quality Assurance [Winter 2017]
 Institute of Technology, University of Washington - Tacoma

L2.18

COOKIES EXAMPLE

- cloud9

Example

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma L2.19


JAVA WEB SERVICES EXAMPLE

- Java, Netbeans, Maven, REST/JSON

Example

January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma L2.20

QUESTIONS



January 9, 2017 TCSS360: Software Development and Quality Assurance [Winter 2017] Institute of Technology, University of Washington - Tacoma L23.21