COMS W4156 Advanced Software Engineering (ASE)

September 16, 2021

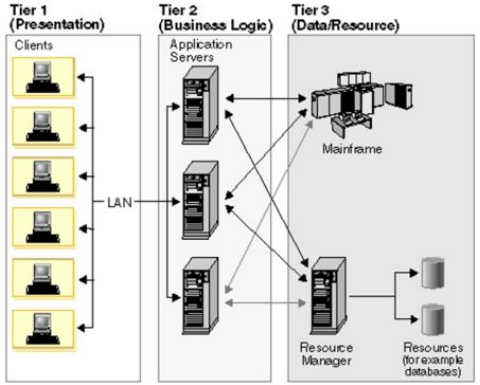
Lecture Notes linked on Courseworks Calendar

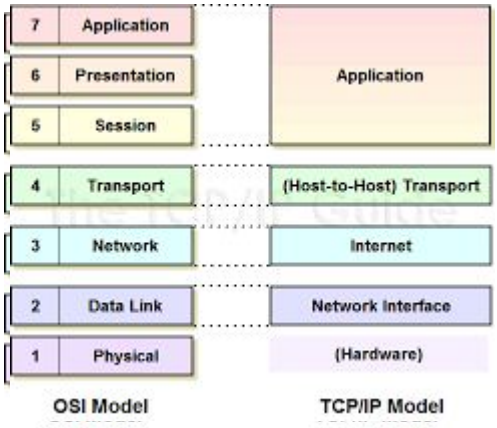
These are “live” links, so possibly changing up to and even during class

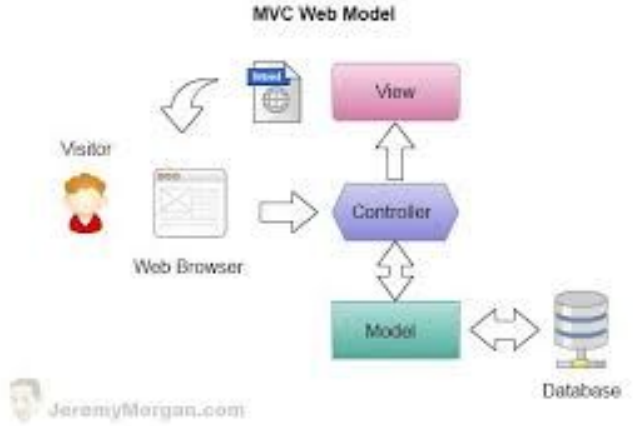
Questions and Comments during Class

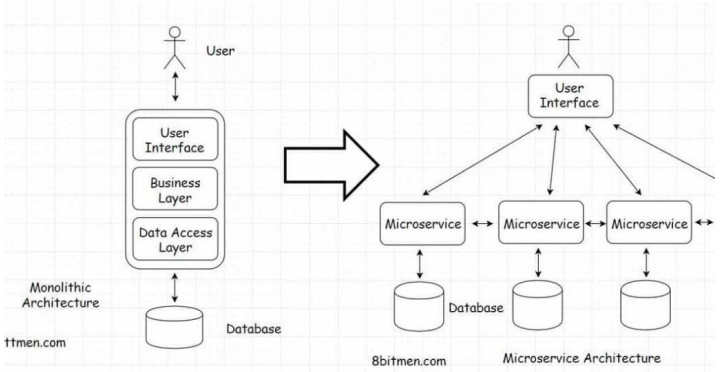
We will start with Piazza Live Q&A but may switch to something else

Software Architectures



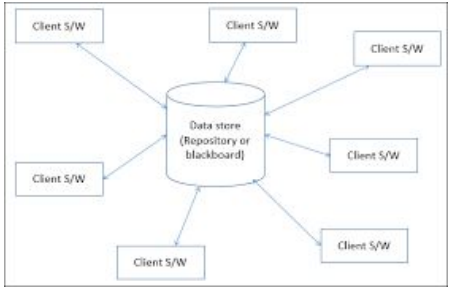


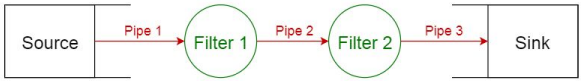


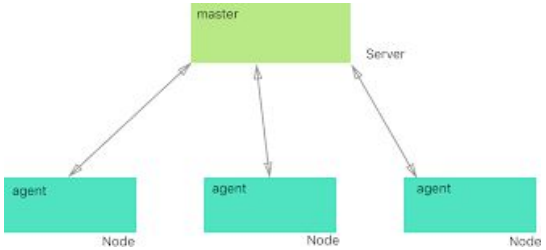


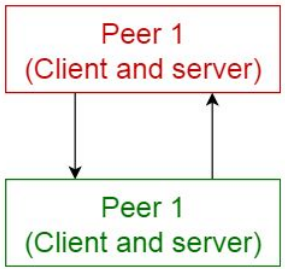
From monolithic to microservices

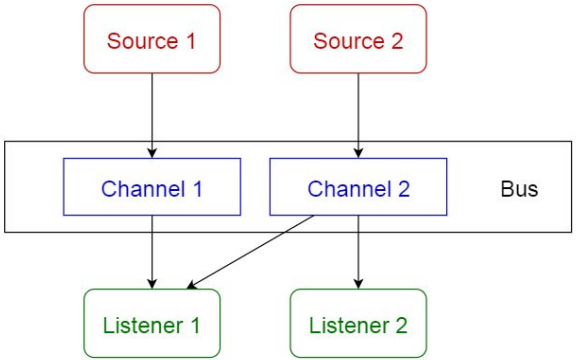
More Software Architectures

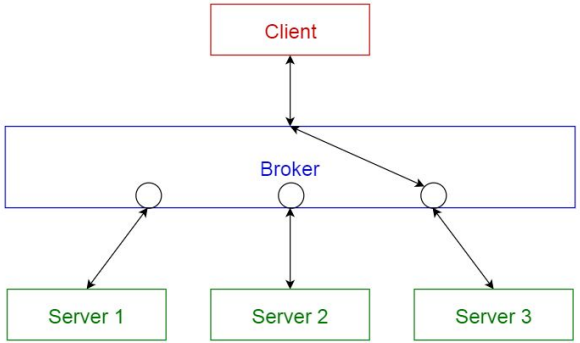












Architectural Components need to Communicate

Protocol - the rules for initiating, continuing and completing communications among components and how the communication messages are formatted in terms of bytes (e.g., SSH, HTTP)

API (Application Programming Interface) - the set of function calls that a given component understands (e.g., Java SE and JDK, Facebook APIs)

We are concerned in this class primarily with APIs, but you will need to use some protocol to send API calls across processes and machines

Some APIs are designed to access data, not code, but here we focus on code

API Communication (Meta-)Patterns

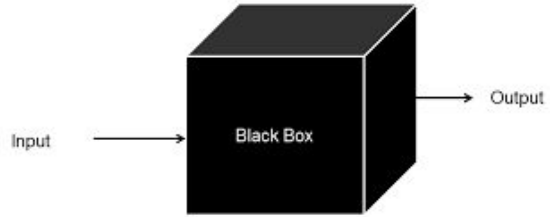
Library = shared code that you compile or bundle into your application, usually runs in same process so might or might not be considered a separate architectural component

Service = a shared capability that you access from your application, usually runs in another process, often on another machine

Frameworks - shared code, and sometimes shared capabilities, for building applications, might include libraries and/or services

Most APIs should be treated as “blackbox”

Use the interface provided by the box but do not peek inside the box (even when the box is open-source) and be careful about depending on how the box works



Internal behavior of the code is unknown

[Do not copy/paste, use the API!]

EVERY CHANGE BREAKS SOMEONE'S WORKFLOW.

Libraries

Most programming languages have “standard libraries” that come with the language or the platform (e.g., JVM, Linux)

- Typically includes I/O, string processing, math, “system calls”
- In most cases, just call the function like any other call - and remember to check status codes (when applicable)
- If you’ve written a program that does pretty much anything, you’ve already used some standard libraries

Many other libraries also implement reusable functionality
Download manually or via a build tool / package manager ecosystem (e.g., Maven Central, PyPI) to bundle with your program

“Third-Party” Code

Your team can develop your own libraries, services and frameworks that are used only by your team or maybe within your organization (e.g., creating a statically linked library)

But many libraries, services and frameworks are third-party - developed and maintained by someone else

Who are the three parties?

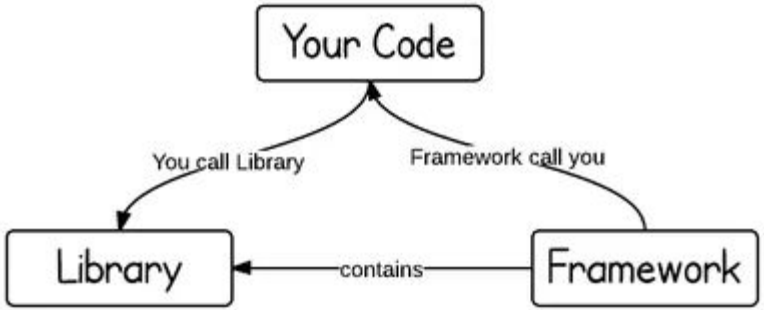
1. Your team
2. Your customers and users
3. Any other source of software besides your team (or your organization)

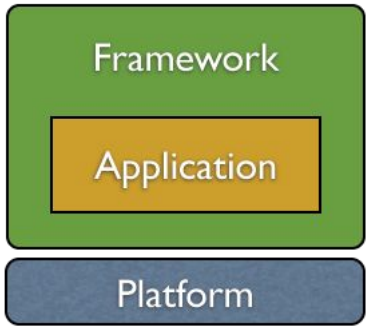
When you break up your own application into services intended to only be used within your organization, they are typically referred to as “microservices”; when you use a third-party service, it’s just a “service”

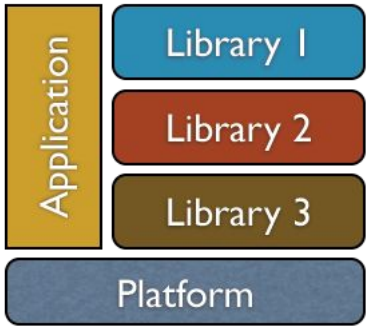
How are Frameworks different from Libraries?

"*Inversion of control*" Hollywood Principle = don't call us, we'll call you

- When your code calls a library or a service, your code is in control
- In contrast, the framework calls your code, so the framework is in control
- The framework plays the role of the main program in coordinating and sequencing application activity and you fill in the blanks







Frameworks

Set of generic components (some of which may be libraries or services) that work together to accomplish common development tasks and functionalities for a popular target domain, such as web or mobile apps

Most frameworks define an application skeleton with places to put your code for the framework to call

The context of those places (e.g., callbacks or event handlers) comprises the framework's API

Most web apps are built on some MVC-oriented framework (e.g., flask, spring)

Services

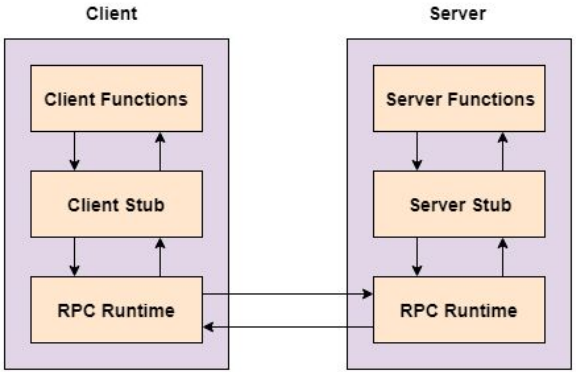
Some services are very specific to a particular task or domain (e.g., https://ipinfo.io/ maps an IP address to its geographic location, carrier, whether using VPN, etc.)

Some services interface to a particular application or ecosystem (e.g., Twitter API)

Lots of public services listed here and here

Services might be called via what appear to be regular functions calls (*RPC* = remote procedure call) or invoked via publish/subscribe (events), with middleware interprocess plumbing (e.g., ROS)

Concerned now with calls via REST APIs



REST APIs

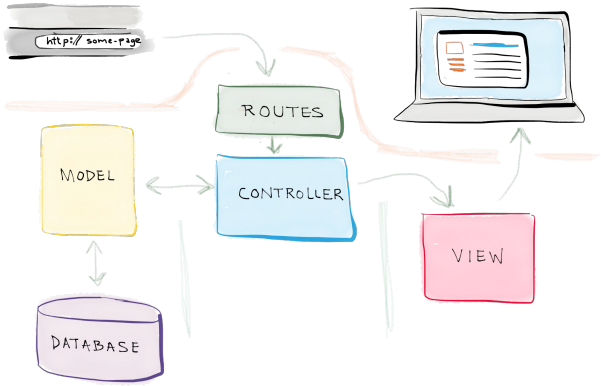
A service's REST API (or "RESTful" API) is usually accessed via http (or https) with a predefined set of URLs called "*routes*"

A REST API consists of a set of HTTP method calls:

HTTP METHOD endpoint1 { optional body with data }
...
HTTP METHOD endpointN { optional body with data }

where "endpointN" is the last part of the route URL and corresponds to the API function to call

REST API terminology uses the term "method" to apply to the HTTP operations, usually only GET and POST (others listed here)



Simplest REST API Calls

Consider the URL http://example.com/path/to/resource

If you click a link like this one on a page in your browser, the browser will use HTTP to connect to the server at example.com (domain) and access the /path/to/resource file directory path from the webroot

This typically fetches some content located at /path/to/resource, which is then returned to and rendered by your browser

Or this might run a program located at /path/to/resource and then return the results of running that program, with the results (not the program) rendered by your browser

The ‘rendering’ could give you text, image, audio, video, code, or any other data

What do these REST API calls return?

REST = REpresentational State Transfer because accessing the resource *transfers* to the caller a *representation* of the current *state* of the resource

The HTTP caller (requester) can be any program, it does not need to be a web browser

And the response does not have to come from a conventional web server

When the HTTP caller is your program, you are using a REST API

What is a “Representation”?

The same resource (or resource state) may have many different representations (variants)

- A page of text might be presented in HTML or plain text
- The words in the text might be written in English or Chinese
- The characters in the text might be encoded in ASCII or UTF-8
- The page might be organized differently for a PC versus a mobile device

The requester might specify the representation(s) it can handle by including optional fields in the HTTP header or the responder might send multiple representations and the requester uses those it can handle

Determining what to send/receive is called *content negotiation*

REST APIs with Basic Parameters

http://example.com/path/to/resource?query

A query string is something like field1=value1&field2=value2&field3=value3

If you click a query URL like this one in your browser, the browser will use HTTP to connect to the server at example.com, tell the program at the /path/to/resource endpoint to run, and give it the parameters value1, value2 and value3

The browser then renders the results of running that program with the parameters

Again, accessing the resource *transfers* to the caller a *representation* of the *state* of the resource

Where does the Query come from?

Usually a query URL would not be sitting there embedded in a page with the field values already filled in

Instead the values of the fields would likely be computed somehow, possibly by showing the user a form to fill in - then the form displays the fields and the user enters the values

Again the HTTP caller can be any program, it does not need to be a web browser

When the caller is your program, you are again using a REST API - your program computes the values for the fields and the recipient server responds with the results from calling the resource program with those parameters

Where are the HTTP ‘Methods’?

Most REST APIs only use GET and POST

A URL like http://example.com/path/to/resource would presumably not change the state of the resource (except maybe to increment metadata like how many times accessed or update the timestamp of most recent access)

If your API call only intends to fetch something with no side-effects, use GET

A URL query like
http://example.com/path/to/resource?field1=value1&field2=value2&field3=value3
should also not have side-effects

If your API call only intends to fetch something, e.g., the field values will be used for a search, then again use GET - in this case with the query fields/values included in the HTTP header

Many Services Have State

Let’s say we wanted to update some data to set field1=value1&field2=value2&field3=value3

That is, your API call intends to *change* the state of the resource based on the provided field values (e.g., from an HTML form), then use POST and put the fields/values in the body of the HTTP request rather than the header

http://example.com/path/to/resource goes in the http header and the query string field1=value1&field2=value2&field3=value3 goes in the http body rather than the header

Many Services Compute State

Let's say we wanted to update some data based on the result of running the resource program with parameters field1=value1&field2=value2&field3=value3

Again, your API call intends to *change* the state of the resource based on the provided field values (e.g., from an HTML form), then use POST and put the fields/values in the body of the HTTP request rather than the header

Again http://example.com/path/to/resource goes in the http header and the query string field1=value1&field2=value2&field3=value3 goes in the http body rather than the header

Now the code that implements the endpoint is more complicated, but it's just like any other code you write except it probably uses json, xml, or some other easily serializable format

Using a REST API in Your Code

- Your individual mini-project will fill in the code that implements the endpoints for the Connect-4 game
- And also add some other code, see the places marked in the skeleton
- These are all in *.py files as explained in assignments.txt, do not change any other files (you can add new files)
- If you change any other files, you will probably break the game!

➢ Your team project will need to implement a service (or set of services) with some API, not necessarily a REST API, with functionality much more sophisticated than Connect-4

Individual Mini-Project

Three parts:

1. Implementing a simple game (intentionally due September 22, the day after program change period ends)
2. Testing the game (due September 29)
3. Saving game state (due October 6)

Note we added the requirement to submit a <= 2-minute demo video for each part (this will be used only for grading)

See Connect Four

Reminder: “Homework Zero”

https://courseworks2.columbia.edu/courses/135182/assignments/680926

Due six days after you joined the class if you were not enrolled on the first day of class (past due for those who joined the class at the beginning)