CS 575: Software Design

More on Modeling

Why is the ability to reconstruct an architecture important?

Its important to be able to reconstruct an architecture to reason about its design for a number of reasons…

- **Program Understanding** – we need to make modifications to the system, where would the modifications go, how long would they take, how much would they cost
- **Explaining the operation of the system to stakeholders** – how does the system work, what are its major features, why should you use this system versus a competitors solution
- **Providing a common document** to capture the high-level design of the system that can be used to make important management and technical decisions
- **Measuring technical health and technical debt** – how far does the "as built" system deviate from the "as designed" architecture?

Where to Start? Consider what needs to be modeled!

The hardest part of reconstructing an architecture is selecting what needs to be modeled …

- **Components** – hierarchical description of the major subsystems
- **Connectors** – connections between the components
- **Interfaces** – the protocols governing the connectors, or the properties managed by the clients
- **Patterns and Styles** – are there any interesting patterns or styles used in the architecture that should be documented?
- **Rationale** – reasoning behind decisions of the aspects we chose to model – are they important to stakeholders? are they critical to the understanding of the system?
- **Constraints** – what dependencies don't we want to have in the solution

Where to Start? Consider what needs to be modeled!

Think about what is important to show to promote an understanding of the architecture…

- Static aspects are things that do not change as the system runs, and
- Dynamic aspects are things that do change, manage important state, or are sequence dependent
- Platform aspects are things where the runtime or deployment decisions play an important part of the architecture – load balancers, proxy servers, etc.

Where to Start? Consider what needs to be modeled!

How do we find things that we want to model…

- Look at the code
  - Source code analysis tools
  - Class/package structure
  - Build / configuration management information
  - Test cases – what is being tested – that must be important
  - Directory structures
  - Naming conventions

Where to Start? Consider what needs to be modeled!

How do we find things that we want to model…

- What do we know about the system
  - What are the main features
  - How are they exposed to the users
  - Build / configuration management information
  - Test cases – what is being tested – that must be important
  - Directory structures
  - Naming conventions
  - Non-functional aspects

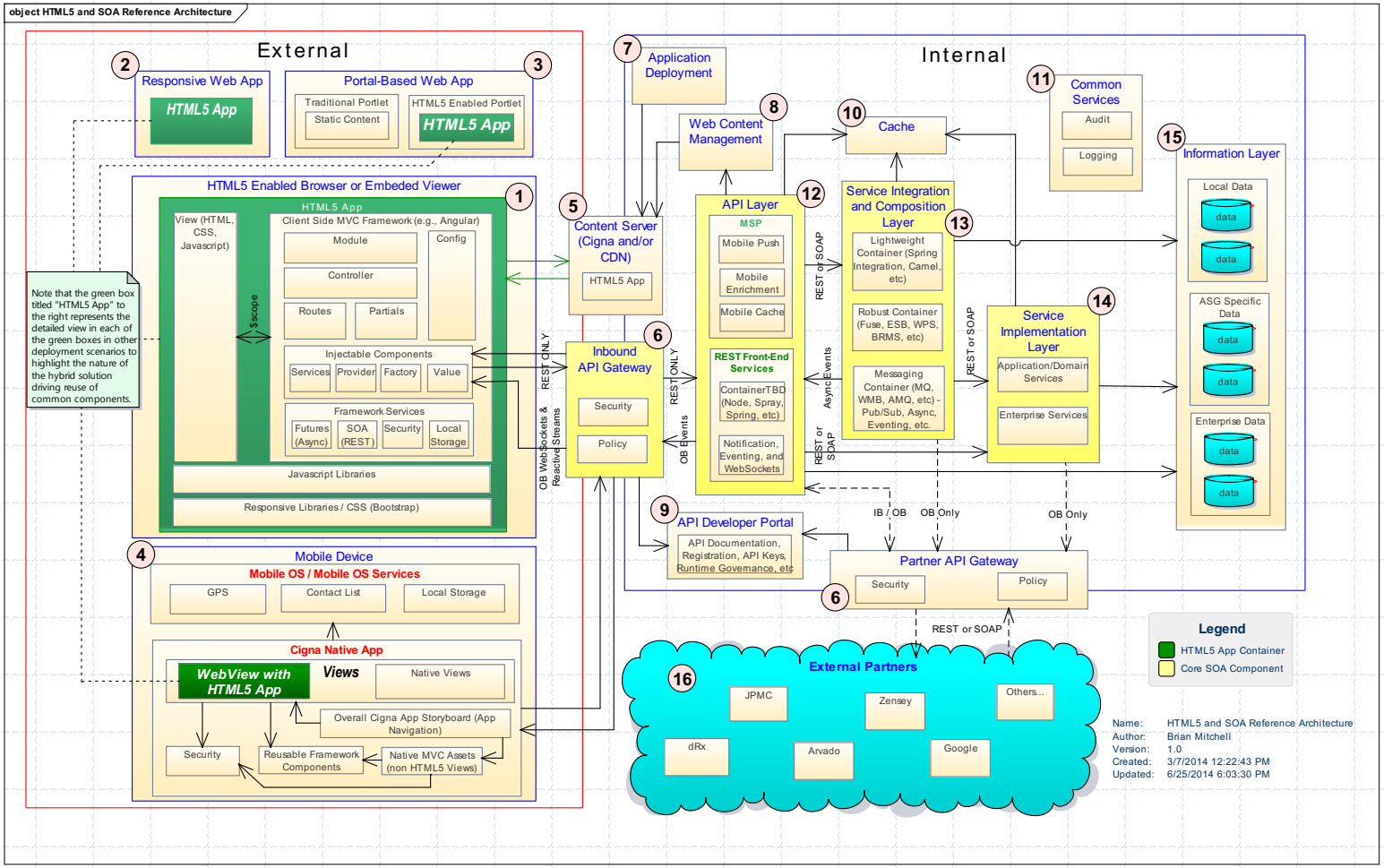
Think about what story that you want to tell

- Think about how you were taught to write stories in elementary school:
  - **Who?** Who is your stakeholder or target audience
  - **What?** What is the key message you are trying to get across
  - **Why?** Why is it important
  - **Where?** Where are the most important parts – think scale, security, etc
  - **When?** When do key events happen?
  - **How?** How does it work?
  - **How Much?** How much impact does the system have on an existing landscape?

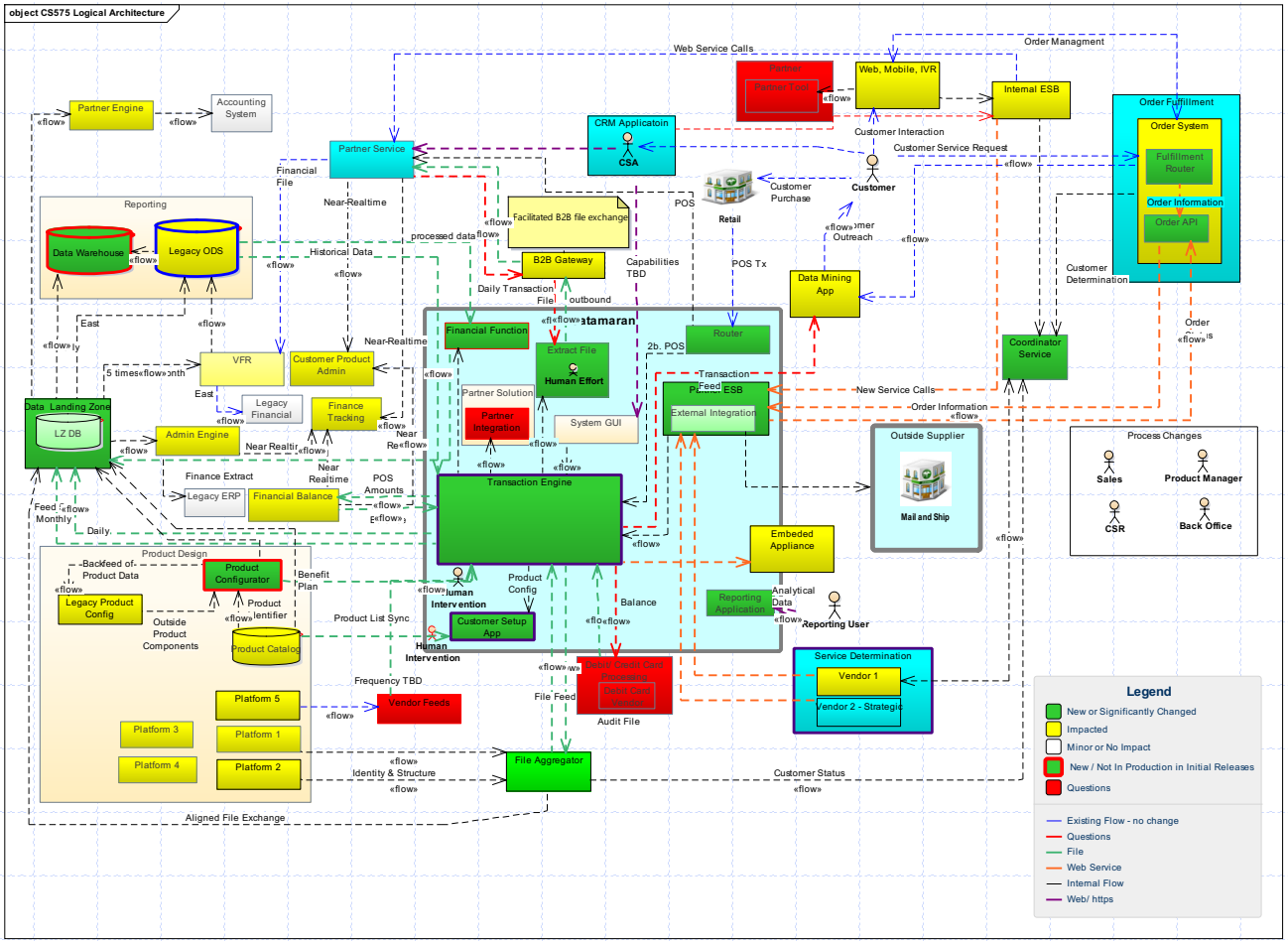
And the outcome you want to achieve

- Why do I create a Model / View:
  - Drive stakeholder clarity
  - Making quality / informed decisions
  - Forcing others to make decisions
  - Being transparent around the solution – what it is and what it is not
  - Being opinionated around constraints
  - Show alignment, and possible misalignment with enterprise or industry standards

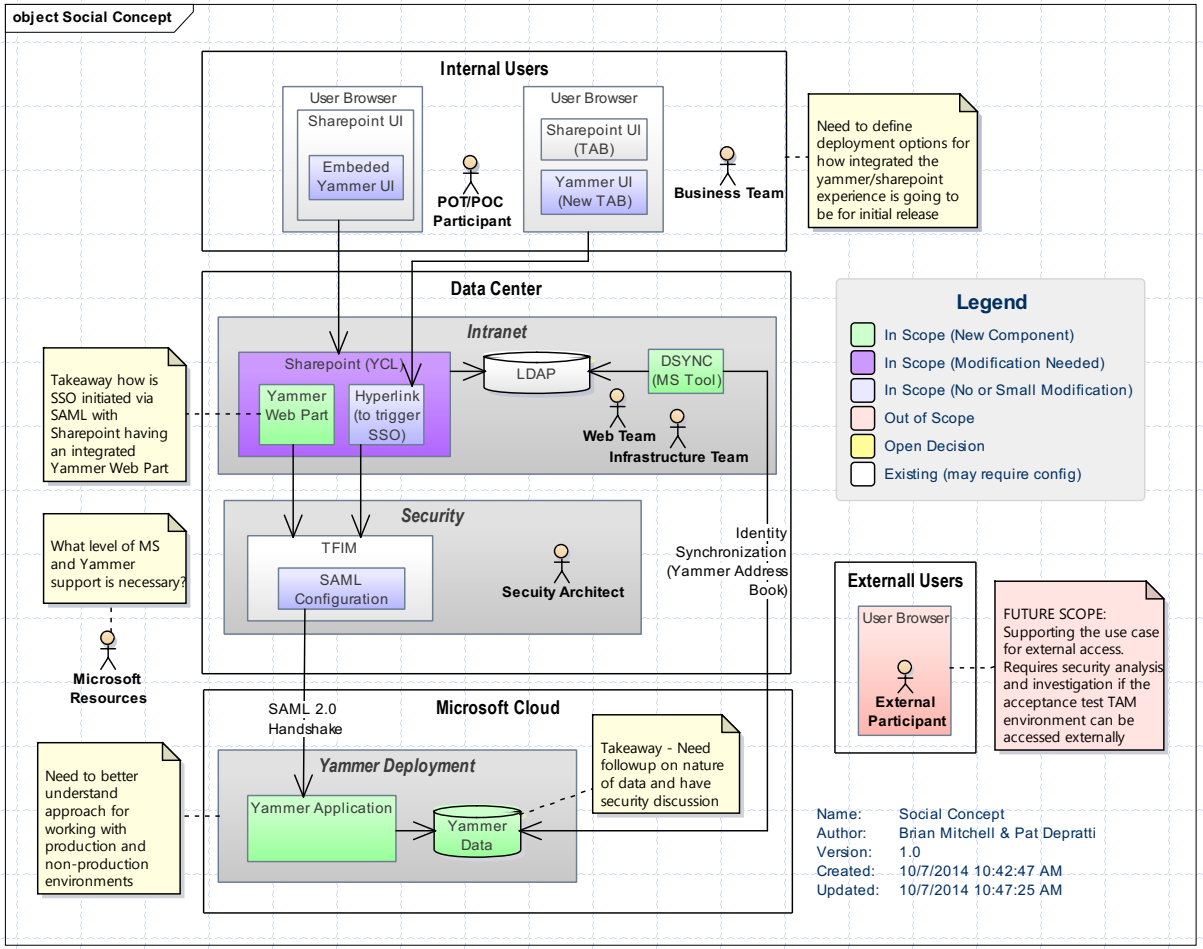
Example – What are the major pieces?



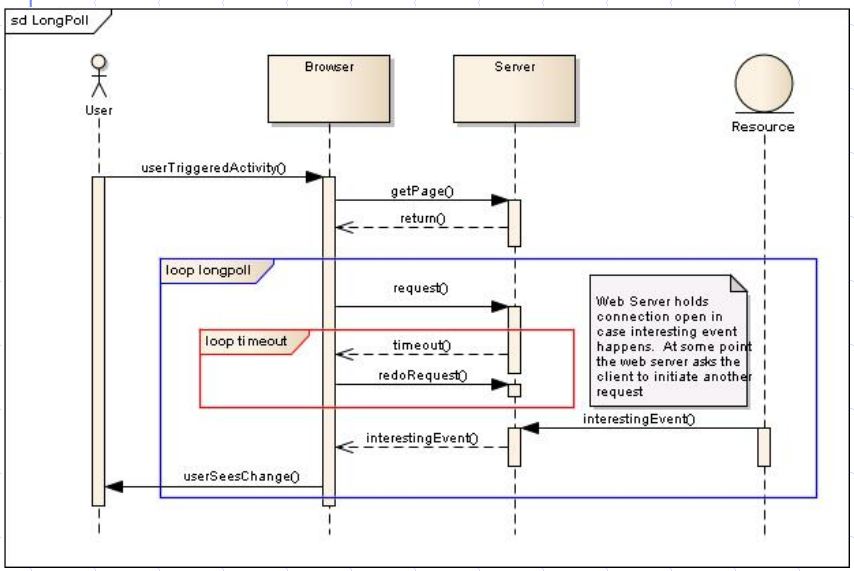
Example – What is impacted?

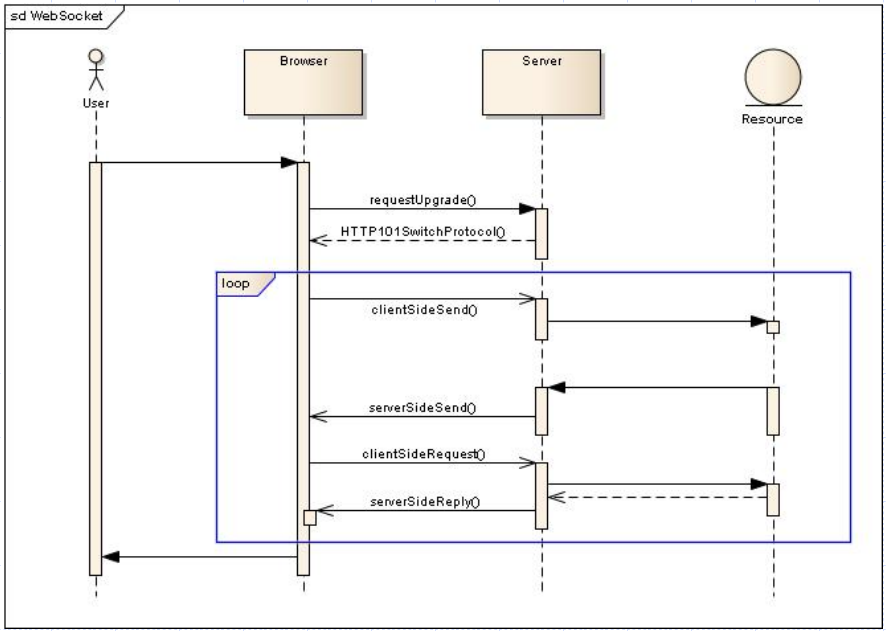


Example - Who needs to weigh in?

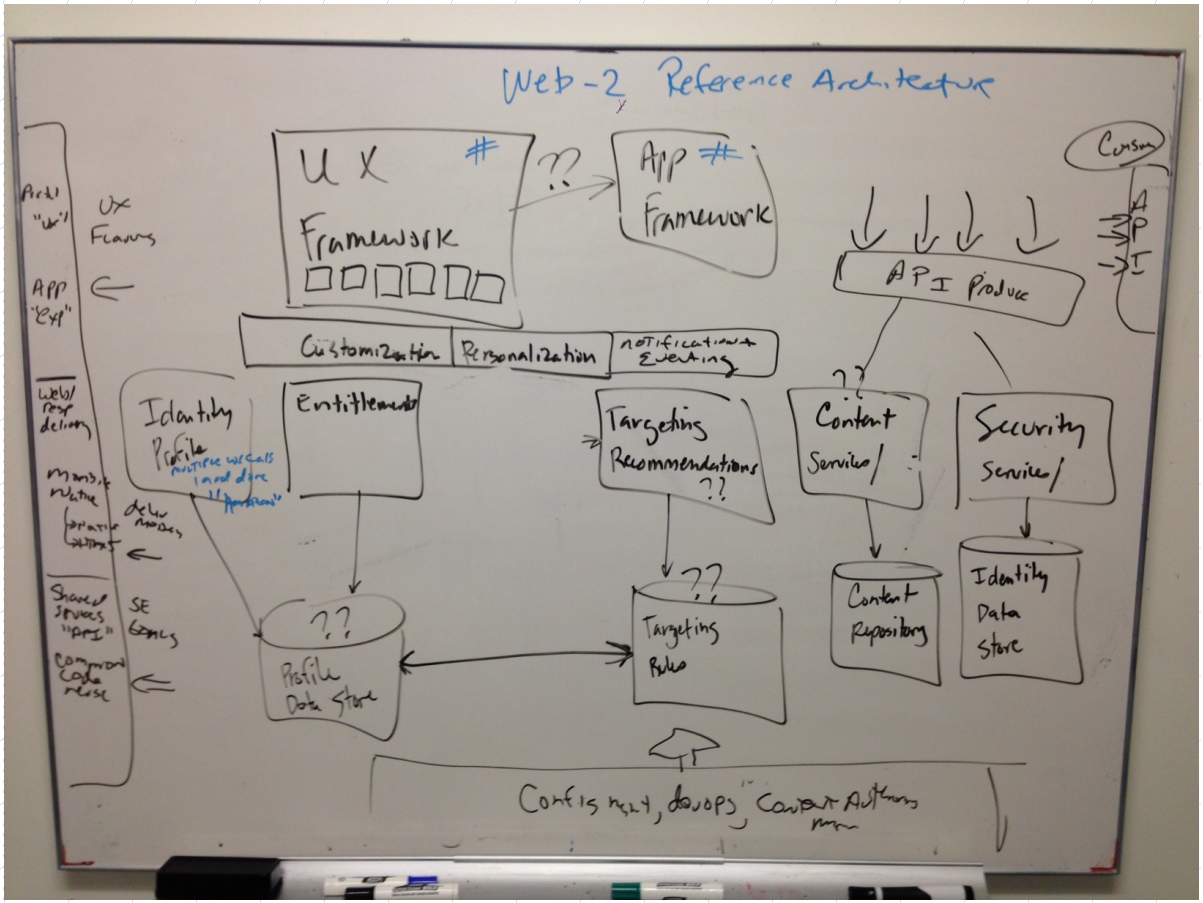


Example – How does it work?





Example – How do we explain what we want to change?



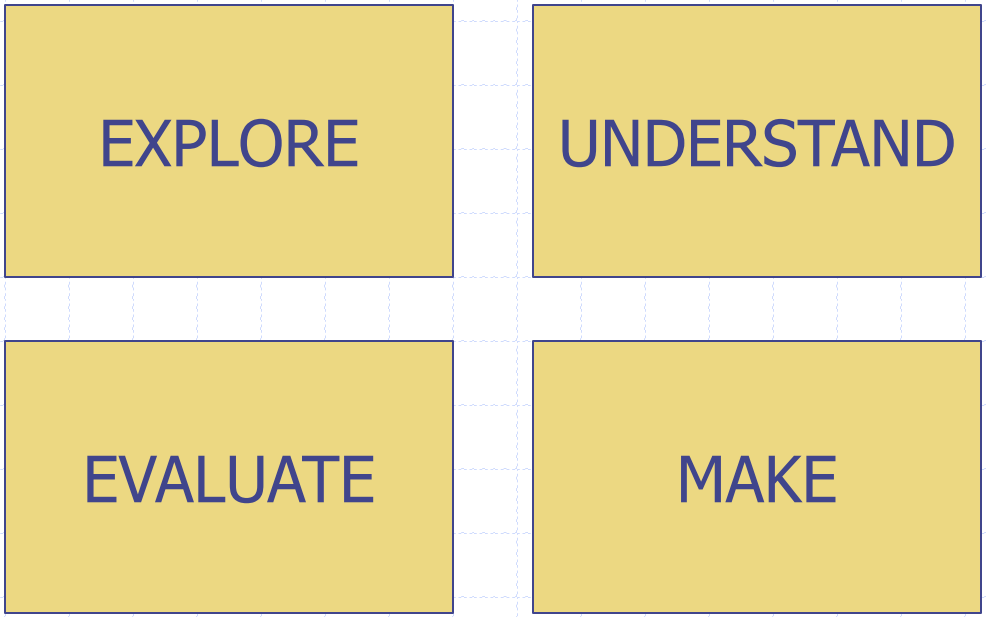
Example – Is this a feasible idea? –

(Inspiration during a camping trip…. sorry, blurred on purpose)

Don’t worry about the notation to get started

- Architecture and design models can be expressed using
  - Formal architecture description languages
  - UML or a variant of UML (Archimate, SysML, SoaML, etc)
  - Simple lines and boxes
  - Semiformal models like C4

A Conceptual Process to Get Started



Source: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436496

A Conceptual Process to Get Started

UNDERSTAND

Actively seek information from stakeholders and work to (re)frame the problem.

Source: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436496

A Conceptual Process to Get Started

EXPLORE

Use generative thinking to identify design concepts and engineering approaches.

Source: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436496

A Conceptual Process to Get Started

Realize design concepts by creating them in the real world as a model, prototype, program, or other artifact.

Source: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436496

A Conceptual Process to Get Started

Determine the fitness of design decisions and decide whether to revisit other modes.

EVALUATE

Source: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436496

Types of architecture models / views

- **Reference Architecture** – Provides a view across a particular domain or problem space
- **Solution Architecture** – Provide an understandable picture of the overall purpose of the proposed solution. Focus on the right level of abstraction
- **Physical Architecture** – Shows physical components such as servers, firewalls, network equipment, storage engines, etc. Generally captures location and function – ie "A Web Server"

Some best practices for modeling architectures from Simon Brown

YouTube

https://www.youtube.com/watch?v=x2-rSnhpw0g

Companion Material

https://architectis.je/

A copy of Simon's visualizing materials is also under the /Misc directory of the course materials on GitHub