Course Recap

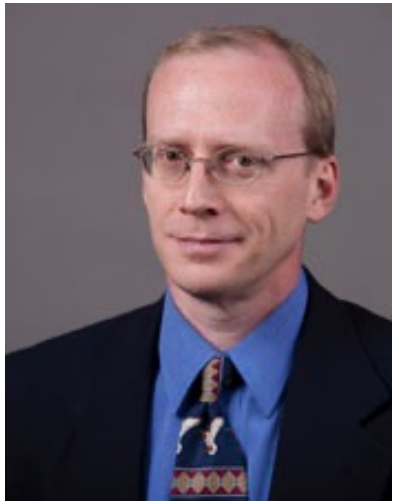
April 28th 2022

Cyberinfrastructure Integration Research Center (CIRC)

CIRC's core mission is to accelerate research, discovery and collaboration through the creation, integration and operation of user-centric cyberinfrastructure that benefits scientific communities.

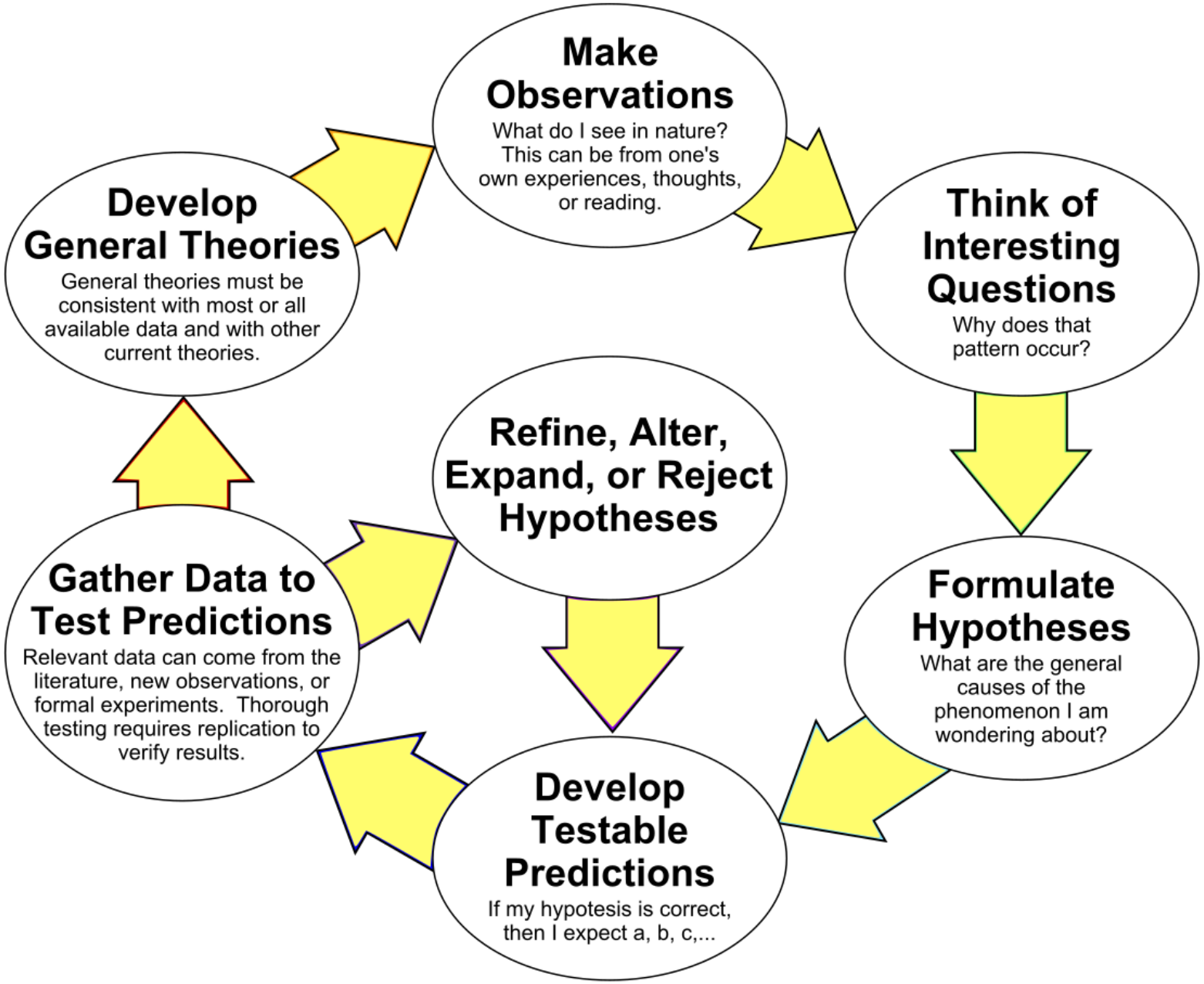
Course Instructors

Marlon Pierce



Suresh Marru

The Scientific Method as an Ongoing Process



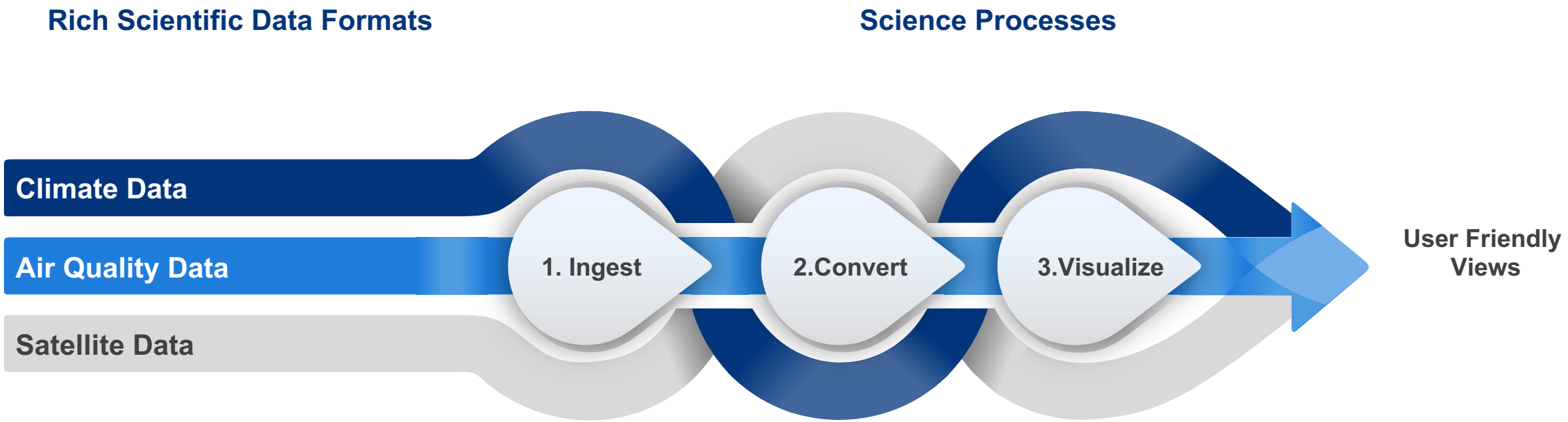
What did we expect you to get out of this class?

A fusion of conceptual skills and “scientific way” of making choices.

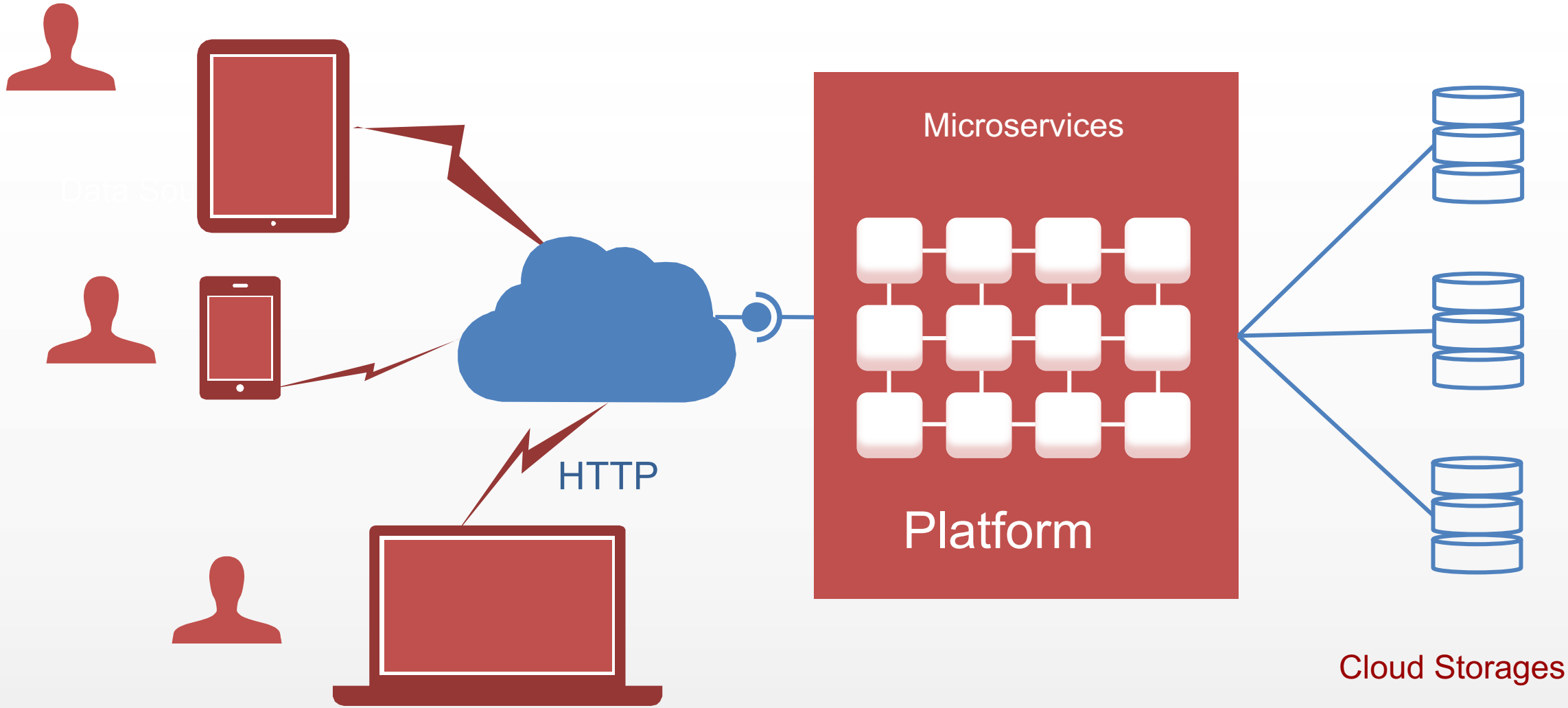
The course is tailored to use tools and technologies relevant in 2022 but our expectation is you will learn how to make choices not necessarily be a tutorial on a buzzy technology.

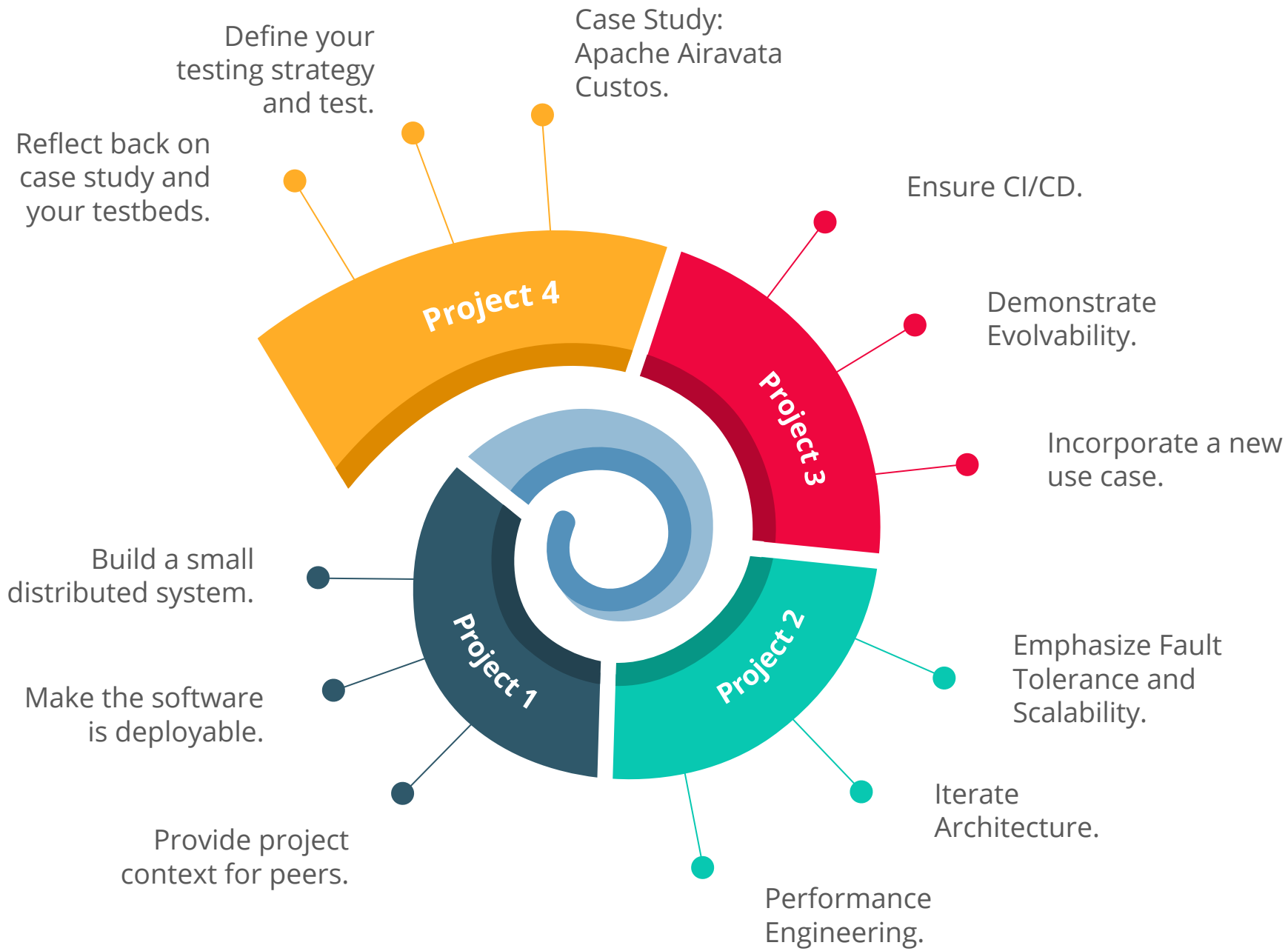
Our definition of a good student is someone who understand the difference between the two.

Use case for Distributed Systems



Testbed for Learning: full stack distributed system

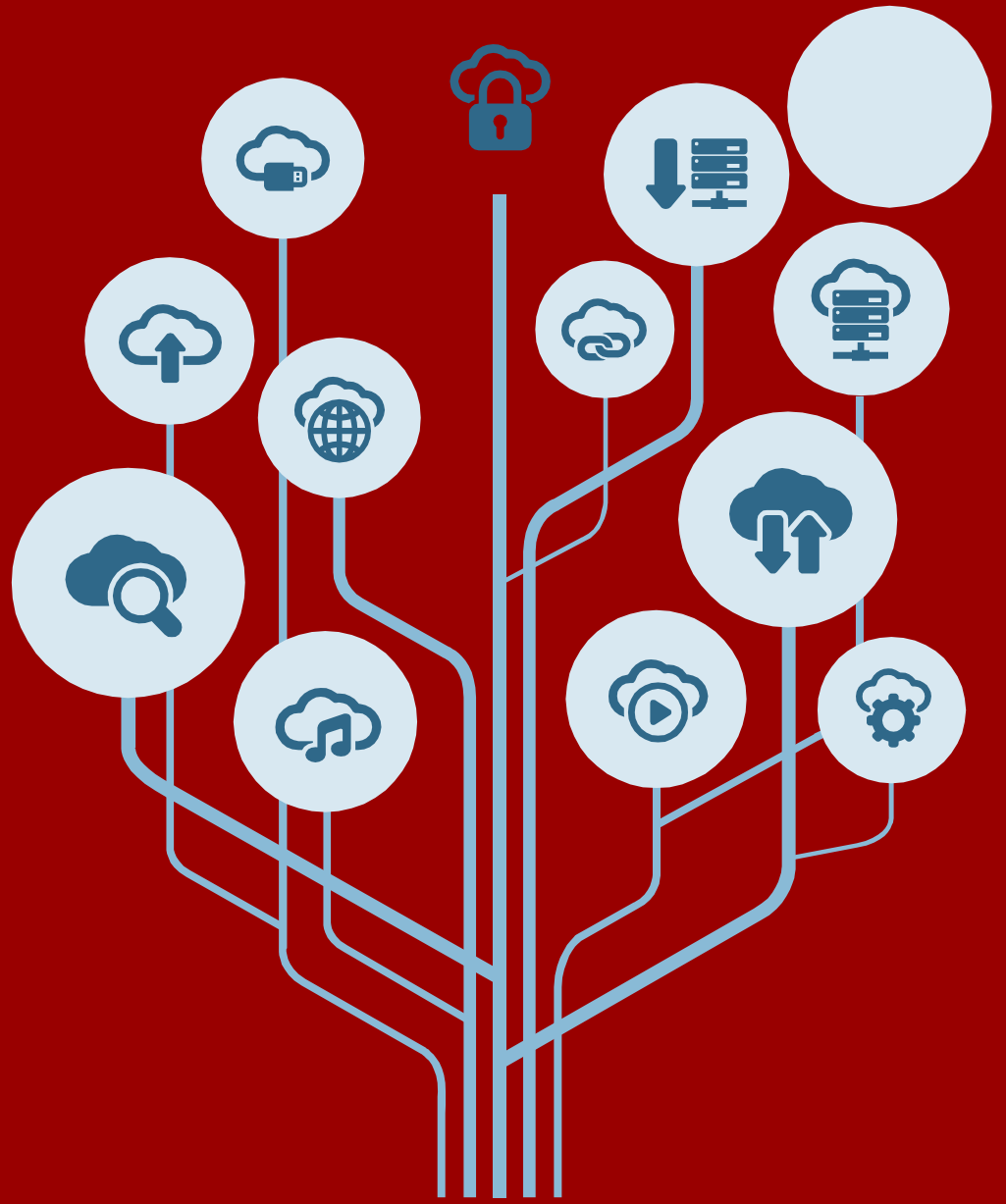


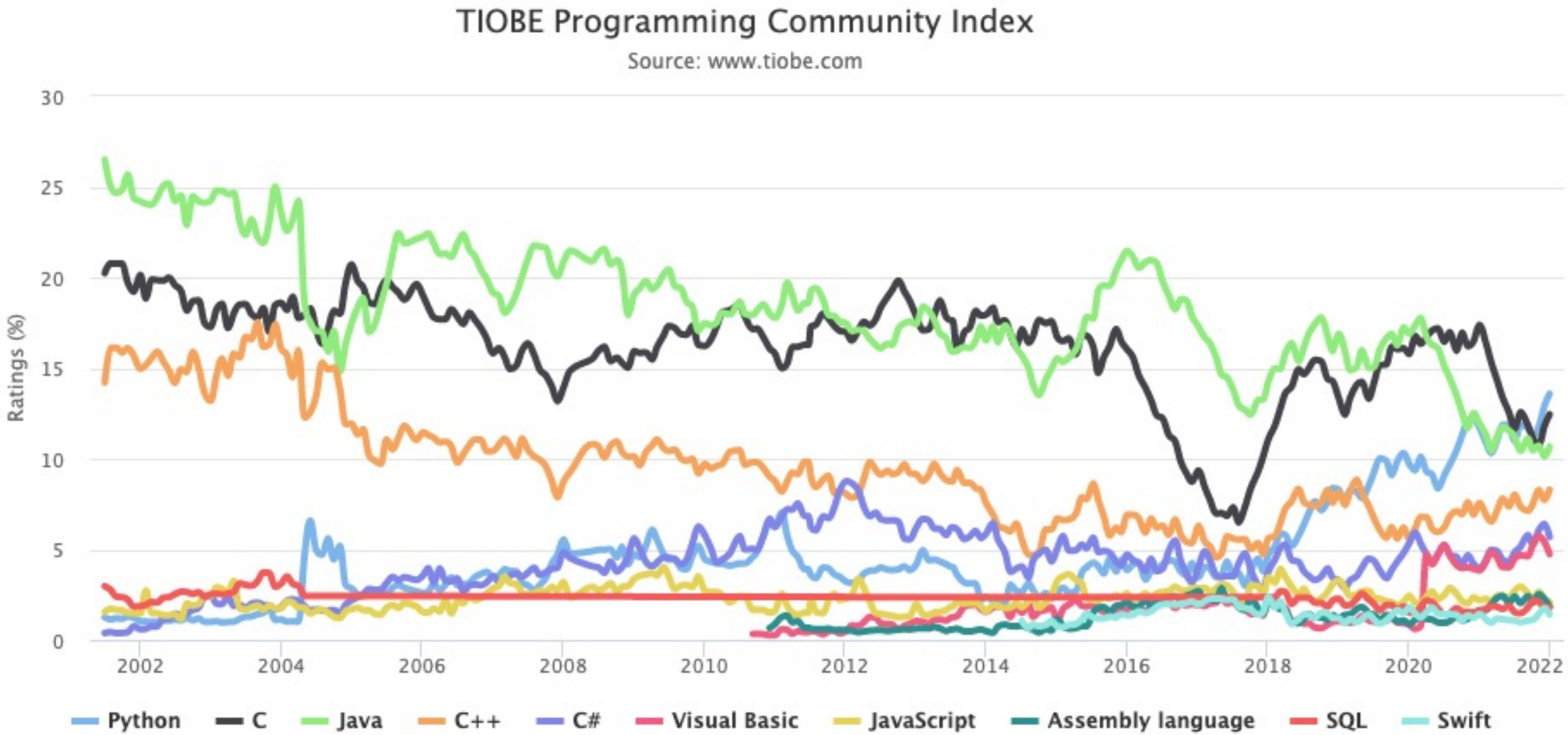

Define your testing strategy and test.
Case Study: Apache Airavata Custos.
Reflect back on case study and your testbeds.
Ensure CI/CD.
Project 4
Project 3
Demonstrate Evolvability.
Incorporate a new use case.
Build a small distributed system.
Project 1
Project 2
Emphasize Fault Tolerance and Scalability.
Make the software is deployable.
Iterate Architecture.
Provide project context for peers.
Performance Engineering.

Cloud-Native Architecture Principles

Each service is broken by a functional capability

Services should be able to evolve independently, scale independently.





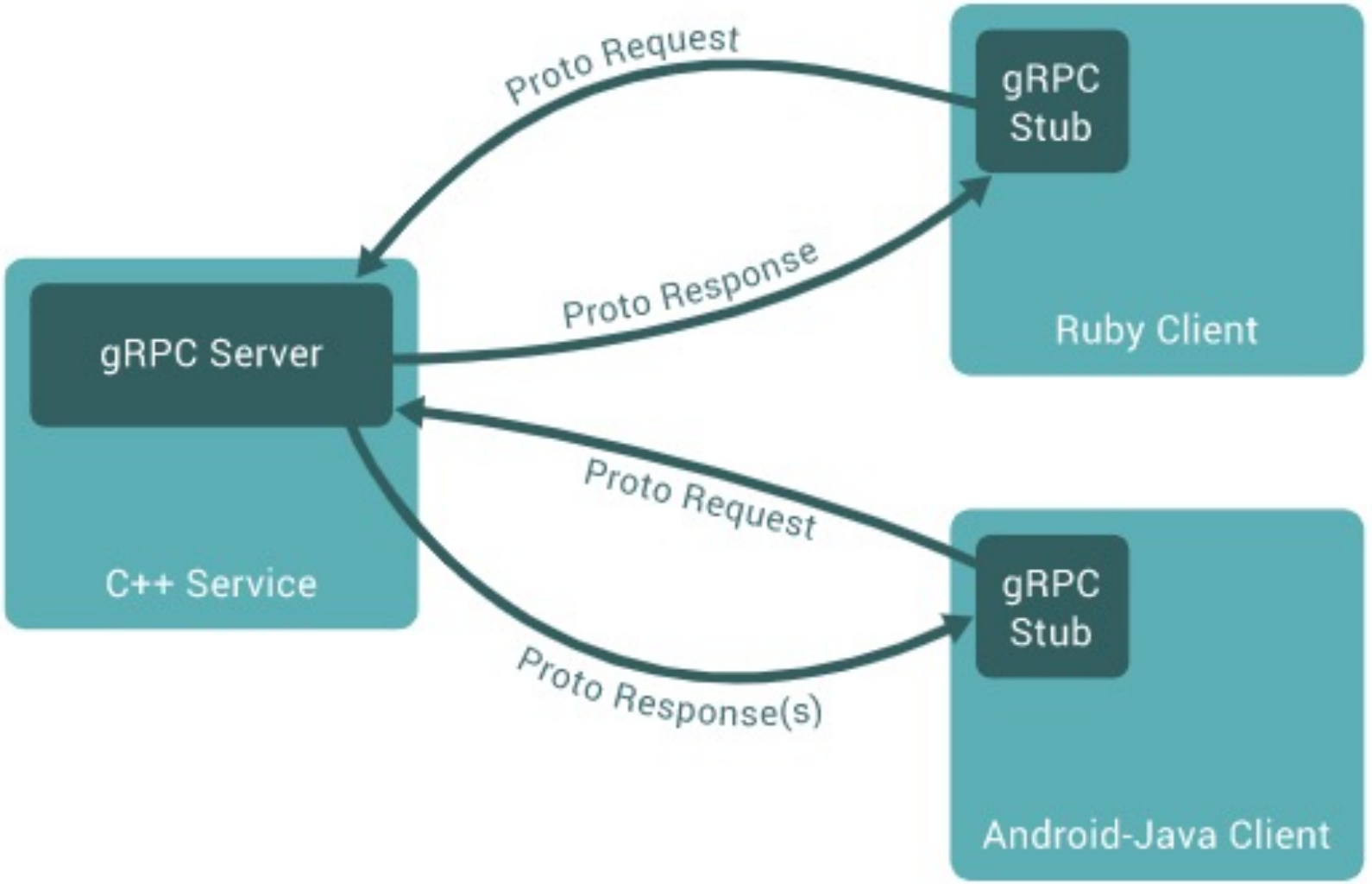
Programming Language “polyglotism”

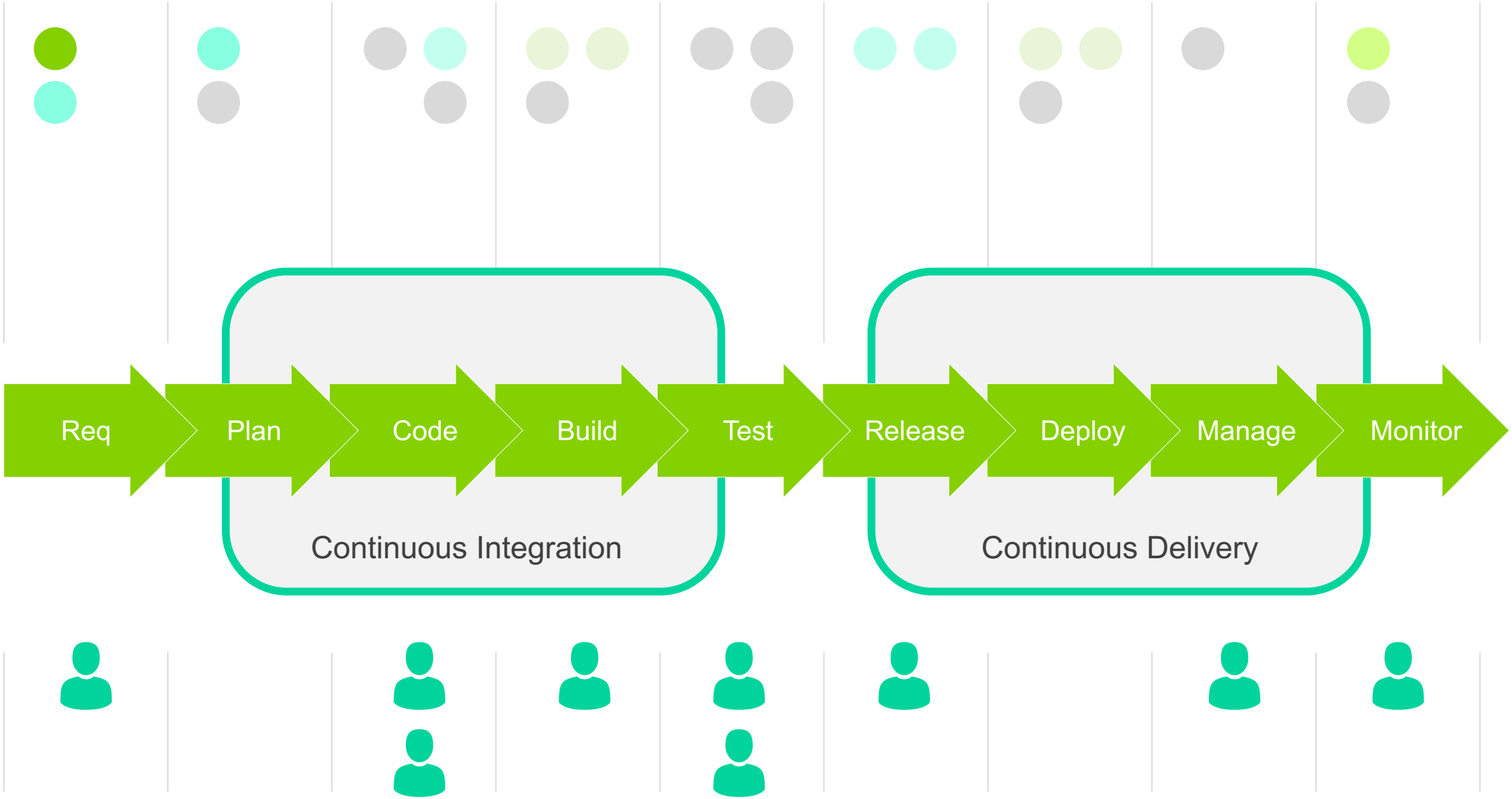
Cyber Security at all layers:

Go beyond Authentication and Authorization

Securing all communications

gRPC, Thrift, Protocol Buffers

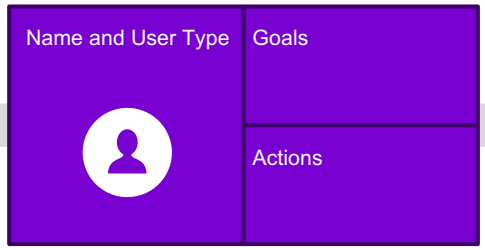



Req
Plan
Code
Build
Test
Release
Deploy
Manage
Monitor
Continuous Integration
Continuous Delivery

Essential Components

README

Describe the project overview, your team introduction in README in your git repo.



Napkin Diagram

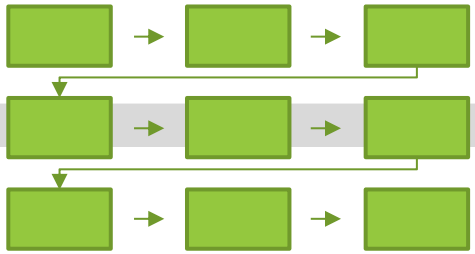
Articulate the project as a user story.
Add this diagram to README and describe it in words.

Outcome

A user-centric understanding of the project.

Information Architecture

Organise discover, explore options, develop wireframes and prototypes



Flow Charts

Sketching
Wireframes
Journey Mapping
Add this to your README

Outcome

Solution Exploration

UX Design

Visual articulation of the solution, validation of ideas and concepts, test with users

Mockups

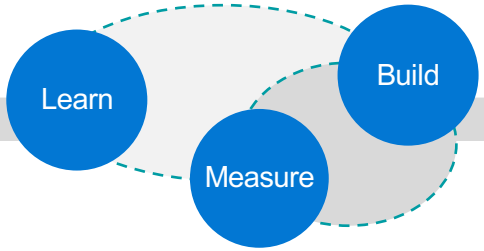
High-Fidelity Visual Design
Rapid Prototyping
Mockups
A/B Testing
Add this to your GitHub WIKI

Outcome

Solution Validation

Peer Review

Validate, learn, plan for the next iteration



Methods

Accessibility
Usability Testing
Feedback Integration
Interactive Design

Outcome

Solution Scalability

Pragmatic Innovation

Inspiration

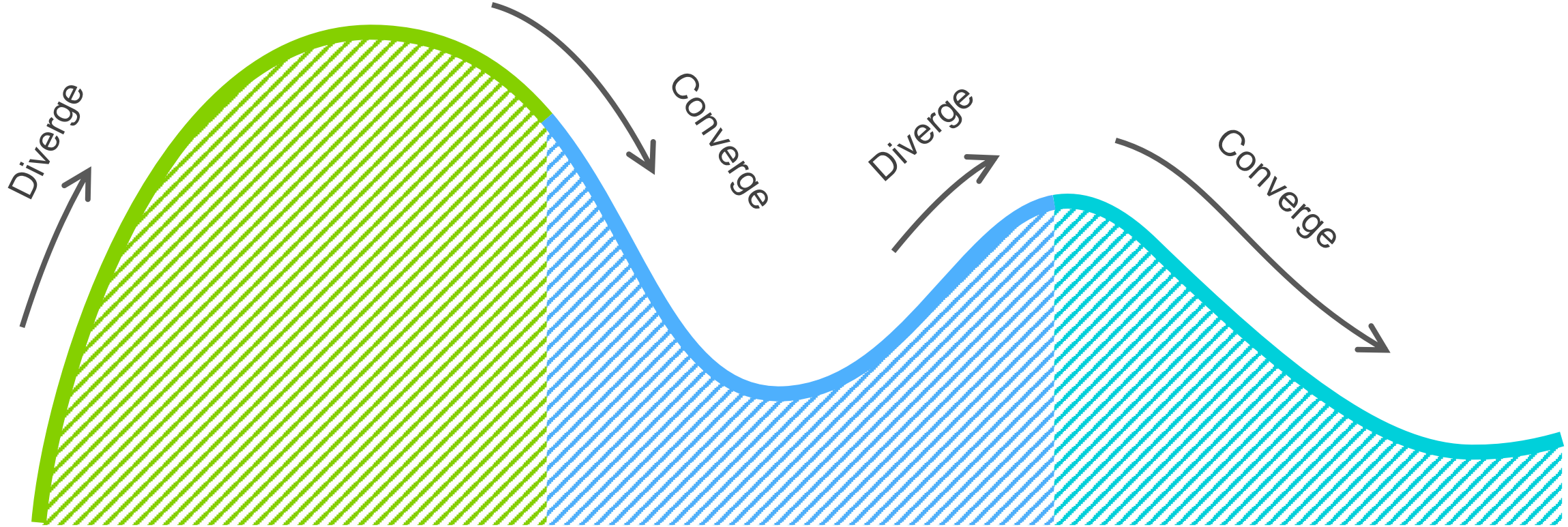
Design Challenge
You should let all kind of ideas float.
Dream Big.

Ideation

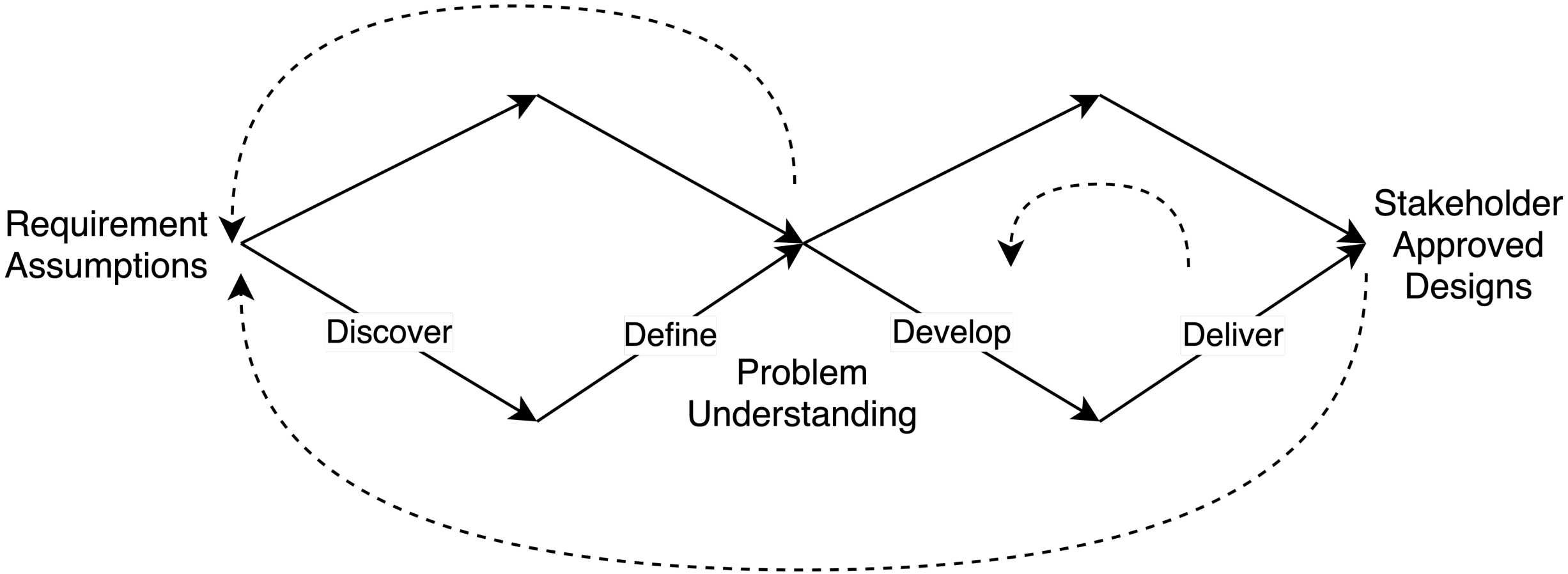
New Opportunity for Design
Get realistic.
Do not loose your ambitious thoughts.
Plan on “evolution”.

Implementation

Innovative Solution
If you shoot for the moon, you will at least reach the roof.
You should not stop at the roof and still plan to launch a rocket.



Double Diamond Design Process



Docker is a shipping container system for code

Multiplicity of Stacks

Static website

User DB

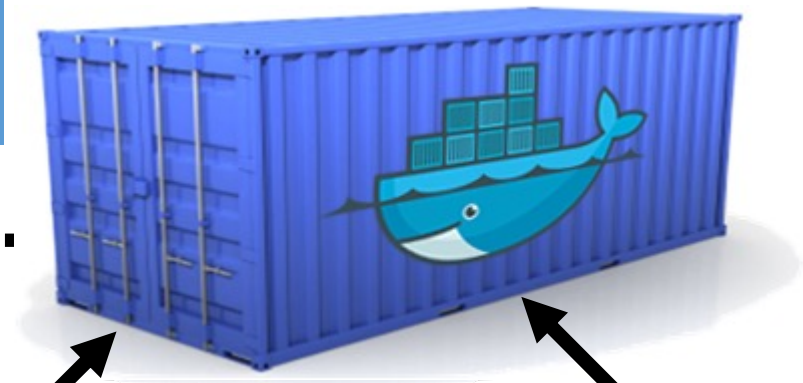
Web frontend

Queue

Analytics DB

Do services and apps interact appropriately?

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container…



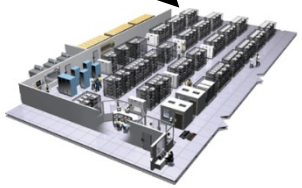
…that can be manipulated using standard operations and run consistently on virtually any hardware platform

Multiplicity of hardware environments

Development VM

QA server

Customer Data Center



Public Cloud

Production Cluster

Contributor's laptop

Can I migrate smoothly and quickly

Container recap

- “Containers encapsulate the application environment, abstracting away many details of machines and operating systems from the application developer and the deployment infrastructure.”
- “Because well-designed containers and container images are scoped to a single application, managing containers means managing applications rather than machines. This shift of management APIs from machine-oriented to application oriented dramatically improves application deployment and introspection.”

Infrastructure as Code

- In short, Docker lets you define in script files everything about each of your microservices.
- Combine this with CI/CD systems to deploy EACH microservice.
  - Your development to test to production environments should be identical and reproducible.
  - Testing and production deployments for each service should be infinitely clone-able.
- This is not elasticity, but it is a prerequisite.
- Docker and other containers have much less overhead

Applied Learning