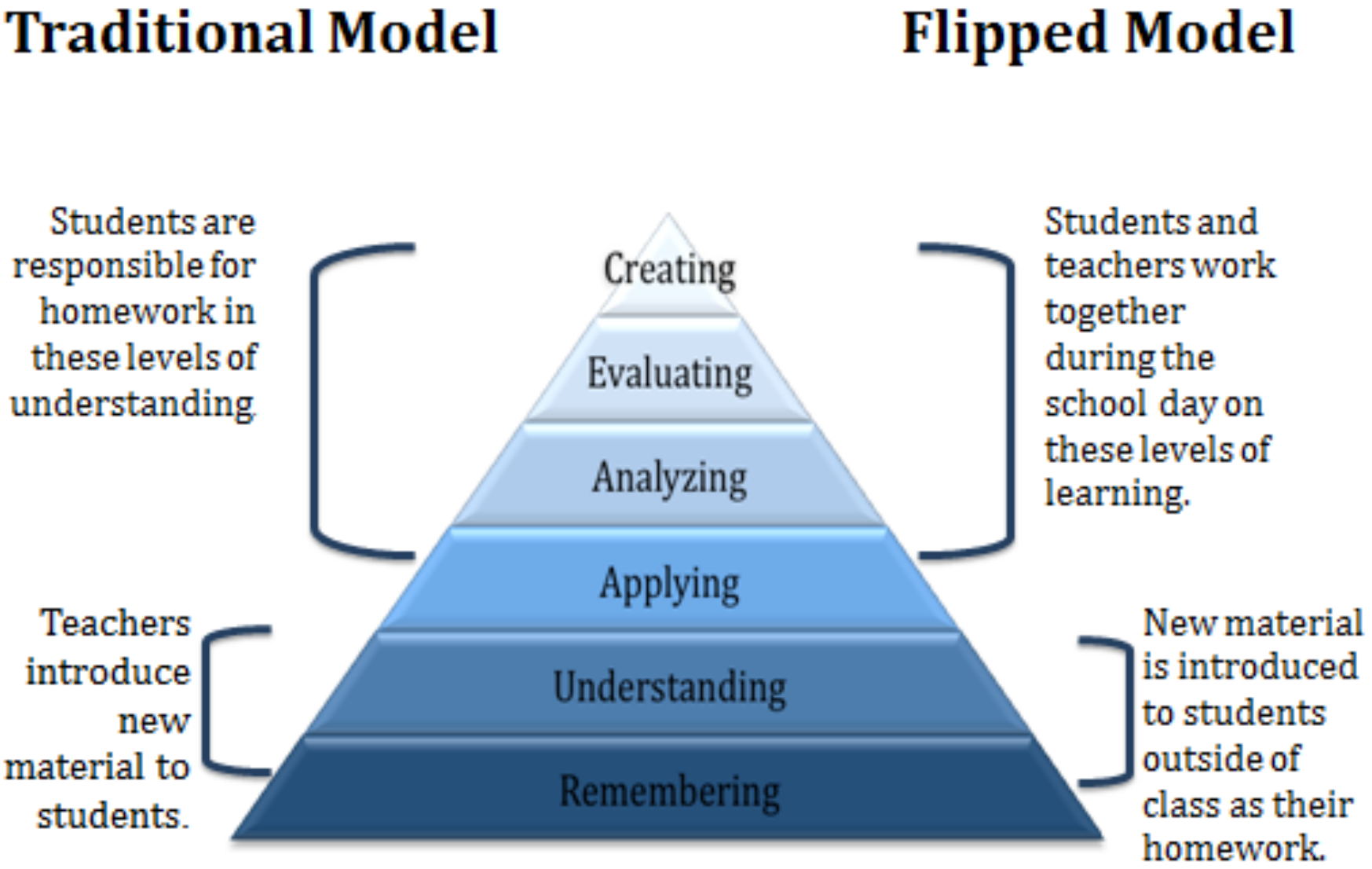
Assignment 1: Description of “microservices”

Marlon Pierce, Suresh Marru

CSCI-B 649 Science Gateway Architectures

INDIANA UNIVERSITY BLOOMINGTON
SCHOOL OF INFORMATICS AND COMPUTING

“Flipped Classroom”

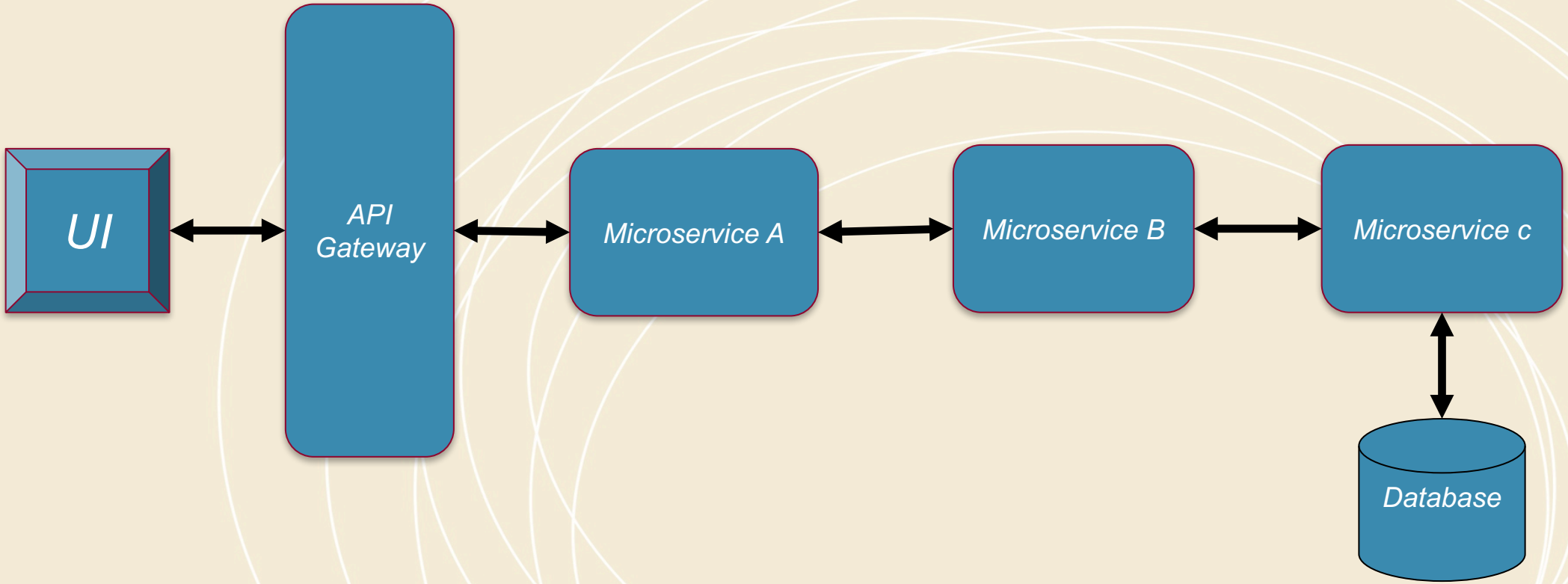


“this pattern of teaching also involves giving students the task of reading from a textbook or practicing a concept by working on a problem set, for example, outside school” - Wikipedia

Rethink if this course is right for you

Implement a small full stack “micro service” architecture

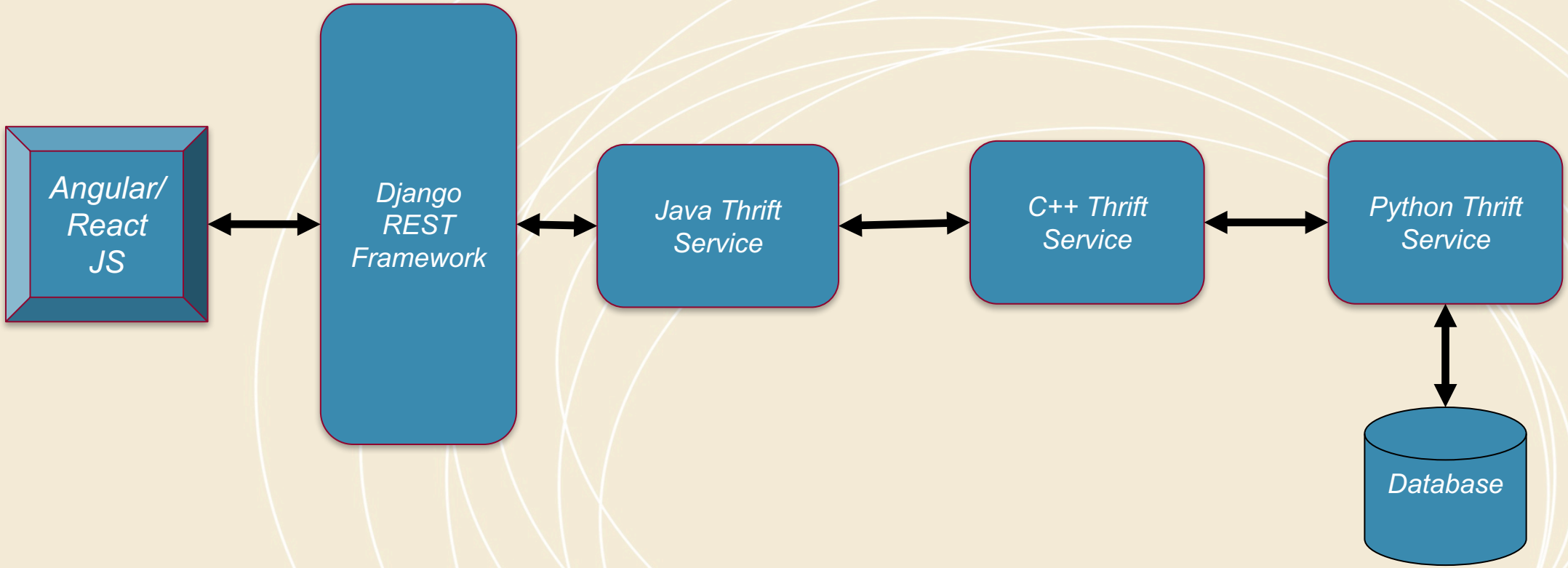
Sample Architecture



Technology Choices

- We will not be prescriptive but can make suggestions.
- Need to choose at least 3 programming languages.
- All components (including UI) need to use a build framework.
  - Make, Maven, Bower……
- Required to have a README instructing how to checkout, build, run, verify.

Example Choices



Example Use Case

- A simplified personal weather predictor
  - Mock the implementations to avoid getting distracted.
  - Focus on the Science Gateway Architecture, a special case of “Distributed System”.

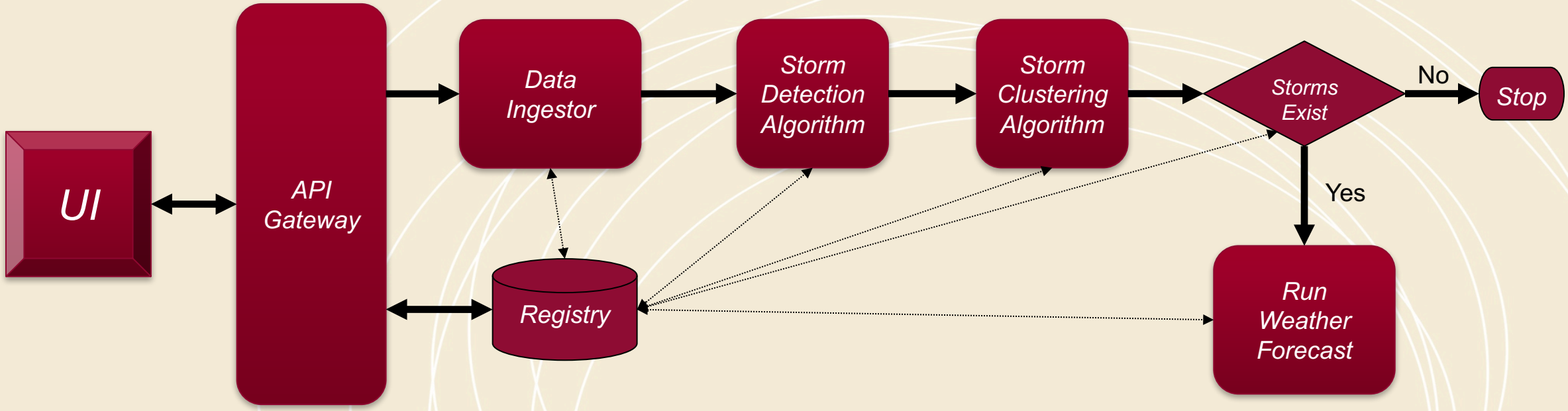
Weather Forecasting Summary

- Current weather determined by observations is the initial state.
- The atmosphere is a physical system governed by the laws of physics
  - these laws are expressed as mathematical equations.
  - models start from initial state (observations) and calculate state changes over time.
  - Models are very complicated (non-linear) and require supercomputers to do the calculations.
- Forecast duration defines temporal boundary conditions
  - the accuracy decreases as the range increases; there is an inherent limit of predictability.

Assignment 1 Preparation

- Learn how to write API's in REST or Apache Thrift or ProtoBuff
- Decide on your Programming Languages.
- Decide on your Web Framework.
- Learn how to use build systems like Apache Maven.
- Test-Driven Development

Implement "mock" services



User Interface

- Pick your Favorite web framework/language
- Have a user management, ok to use cloud services, but preferably open source software.
- Milestone 1: User triggers “diagnose current atmospheric conditions”
  - Provide input of Date, Time and NEXRAD station name (http://www.nws.noaa.gov/tg/pdf/wsr88d-radar-list.pdf)
  - List all interactions queried from a database.

Microservice A – Registry

- Persist all actions of the science gateway and show a queriable audit trails.
- Log all requests, responses and times and display them through API.

Microservice B - Data Ingestor

1. Accept users input and return an acknowledgement.
2. Outputs a Data file URL
   - Refer to https://aws.amazon.com/noaa-big-data/nexrad/
     - /<Year>/<Month>/<Day>/<NEXRAD Station>/<filename>
     - <filename> is the name of the file containing the data (compressed with gzip). The file name has more precise timestamp information.
3. Advanced Track
   - Real Time triggers using Amazon Simple Queue Service or Amazon Lambda NoOps.

Microservice C – Storm Detection

- Detect 3D storm characterized by the reflectivity over a given threshold.
- Basic Track will mock it up and output dummy kml.
- Advanced Track will port an existing C++ library to “Big Data” compatible techniques.
- Advanced++ Track will compare and contrast with other approaches like “Connected Component Analysis”.

Microservice D – Storm Clustering

- Group the storm events detected into spatial clusters using Density based clustering algorithm.
- Basic Track will mock the application and return dummy clusters.
- Advanced Track will port the existing C++ library.
- Advance Track will use EC2 “Big Data” pipelines and services like Kinesis.

Microservice E – Forecast Trigger

- Make Decision on to run forecasts or not.
- Basic Track can mock the decisions but show both stopping and moving foreword of control.
- Advance Track will use real decisions.

Microservice F – Run Weather Forecast

- Basic Track will mock it up and return dummy forecast outputs.
- Advanced Track will invoke Apache Airavata API to launch a WRF application and track progress.

Questions

Marlon Pierce, Suresh Marru

{marpierc, smarru}@iu.edu