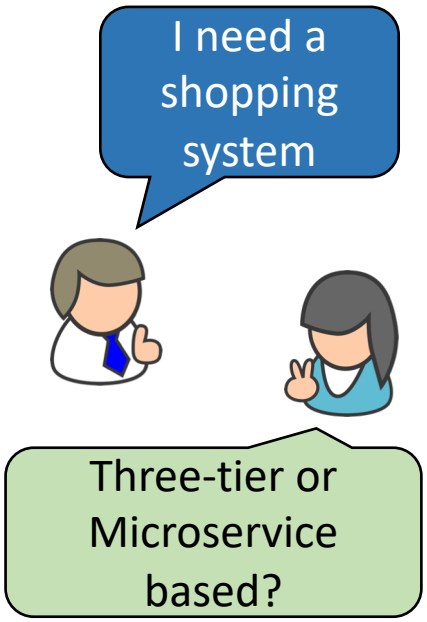
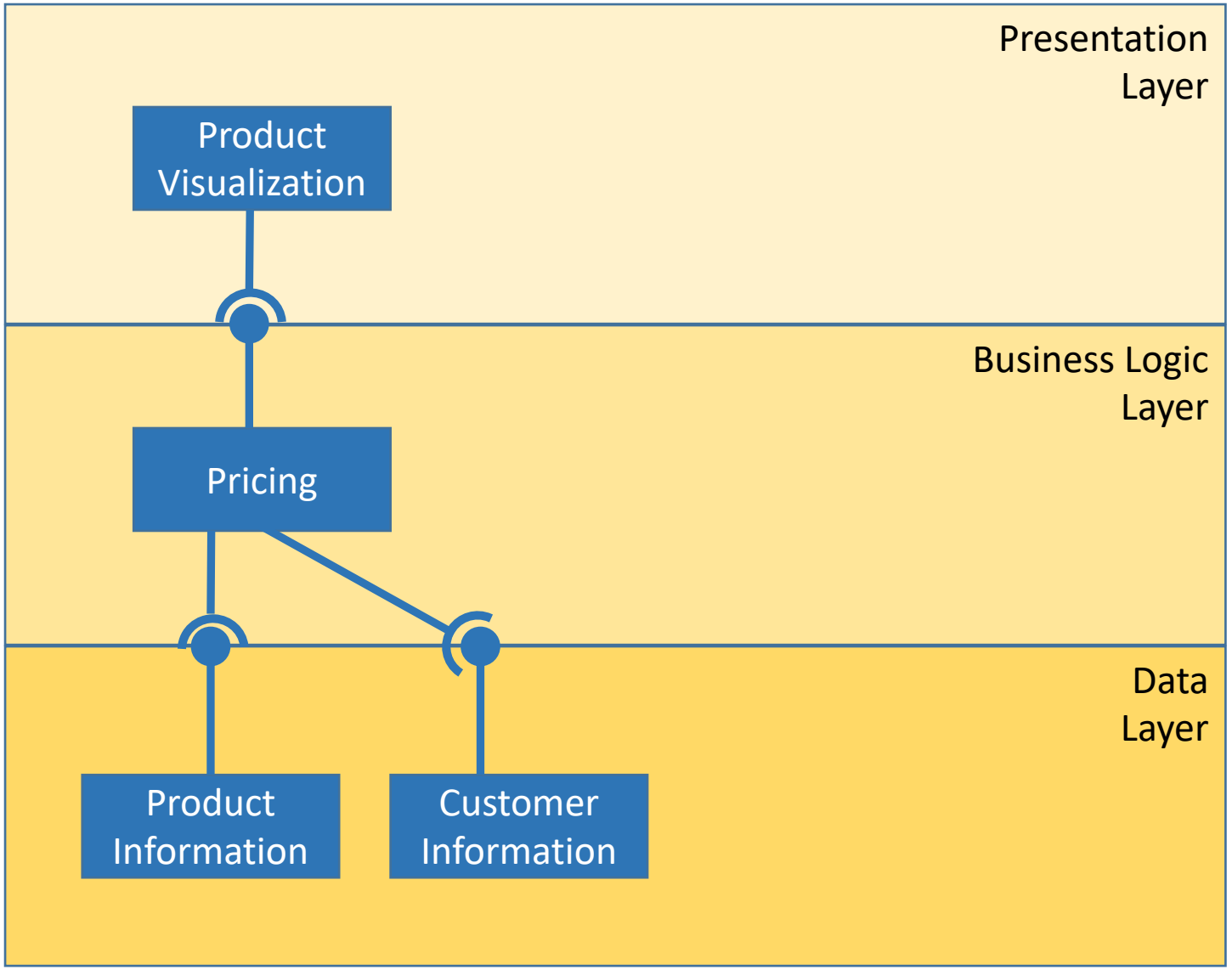
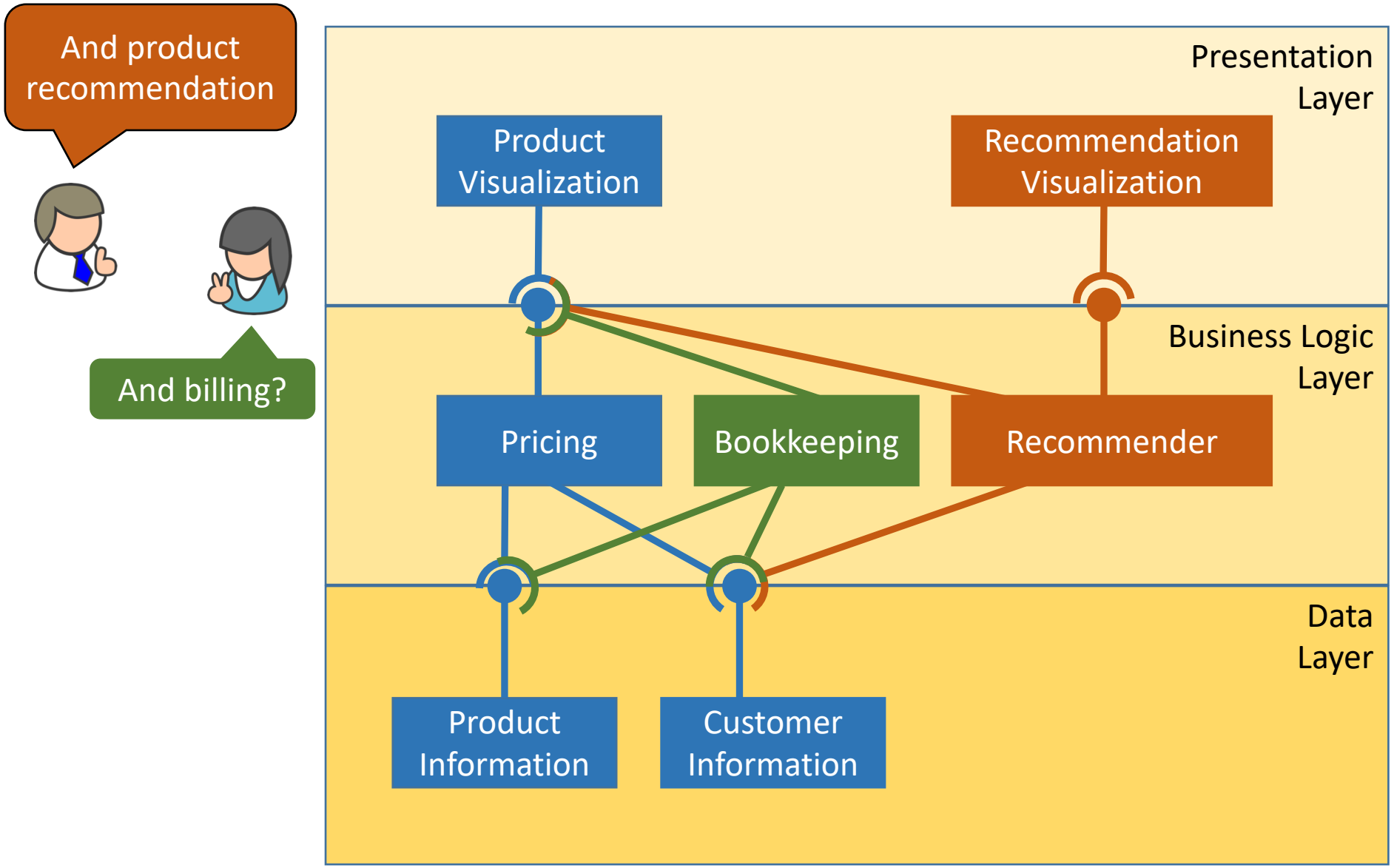
A Class Concept for Microservices to Manage Dynamic Complexity and Code Reuse

Lukas Reinhardt, Marcus Hilbrich

Operating Systems Group
TU Chemnitz

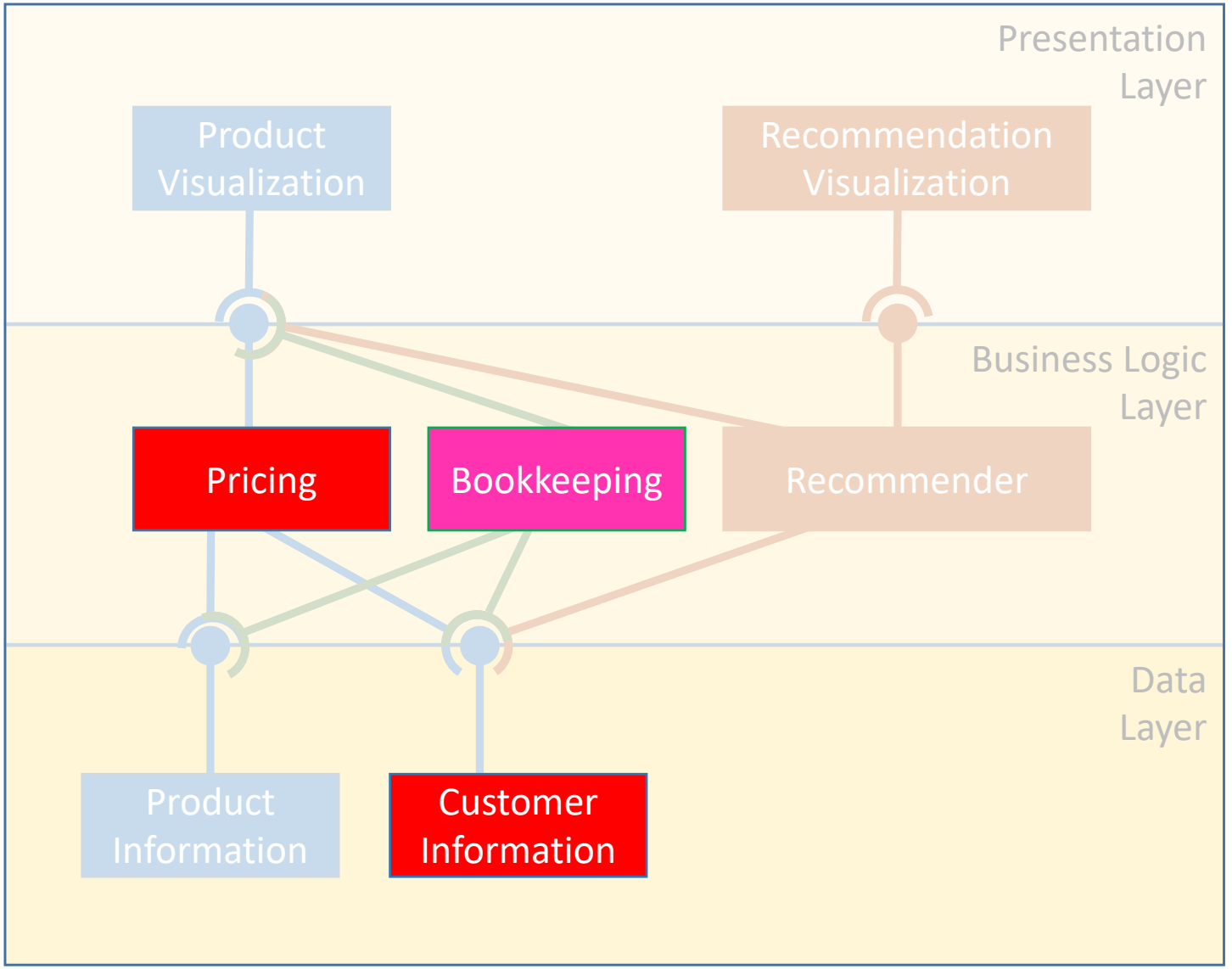

I need a shopping system
Three-tier or Microservice based?


Presentation Layer
Business Logic Layer
Data Layer
Product Visualization
Pricing
Product Information
Customer Information

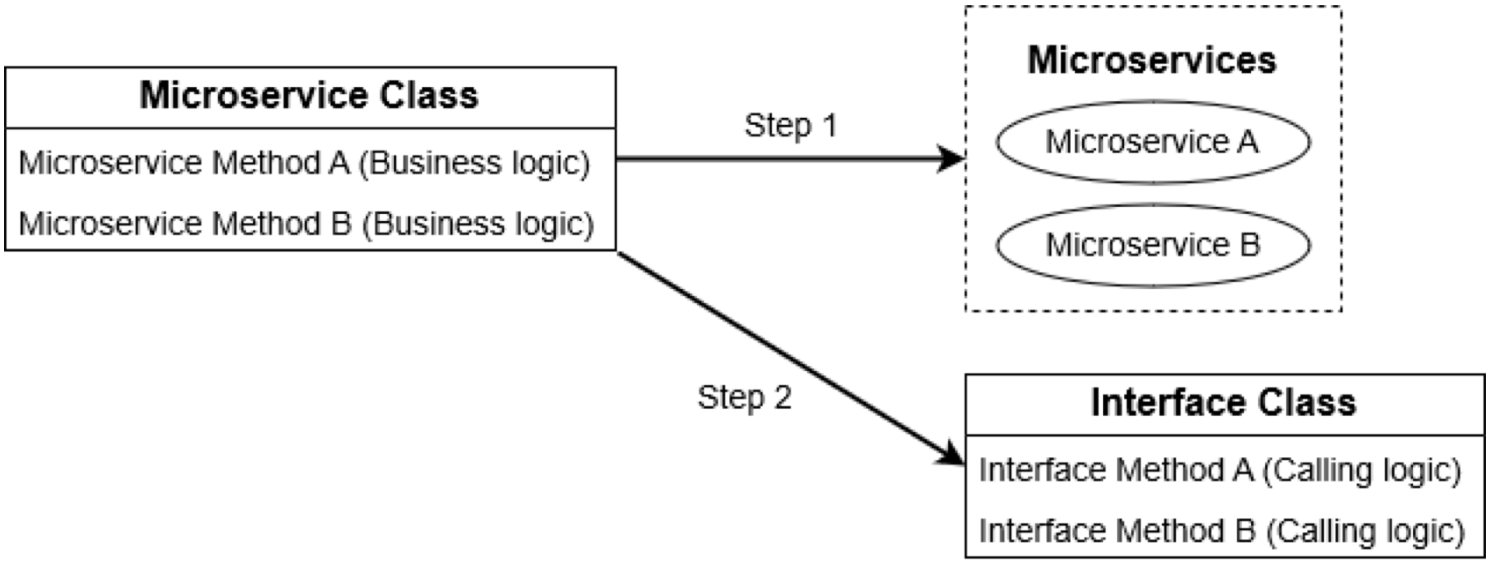

And product recommendation
And billing?
Presentation Layer
Product Visualization
Recommendation Visualization
Business Logic Layer
Pricing
Bookkeeping
Recommender
Data Layer
Product Information
Customer Information

Challenge I: Pricing and Costumer Storage as part of multiple microservices

Challenge II: Bookkeeping represents the state of physical products that are not represented as a service



Concept


Microservice Class
Microservice Method A (Business logic)
Microservice Method B (Business logic)
Step 1
Microservices
Microservice A
Microservice B
Step 2
Interface Class
Interface Method A (Calling logic)
Interface Method B (Calling logic)

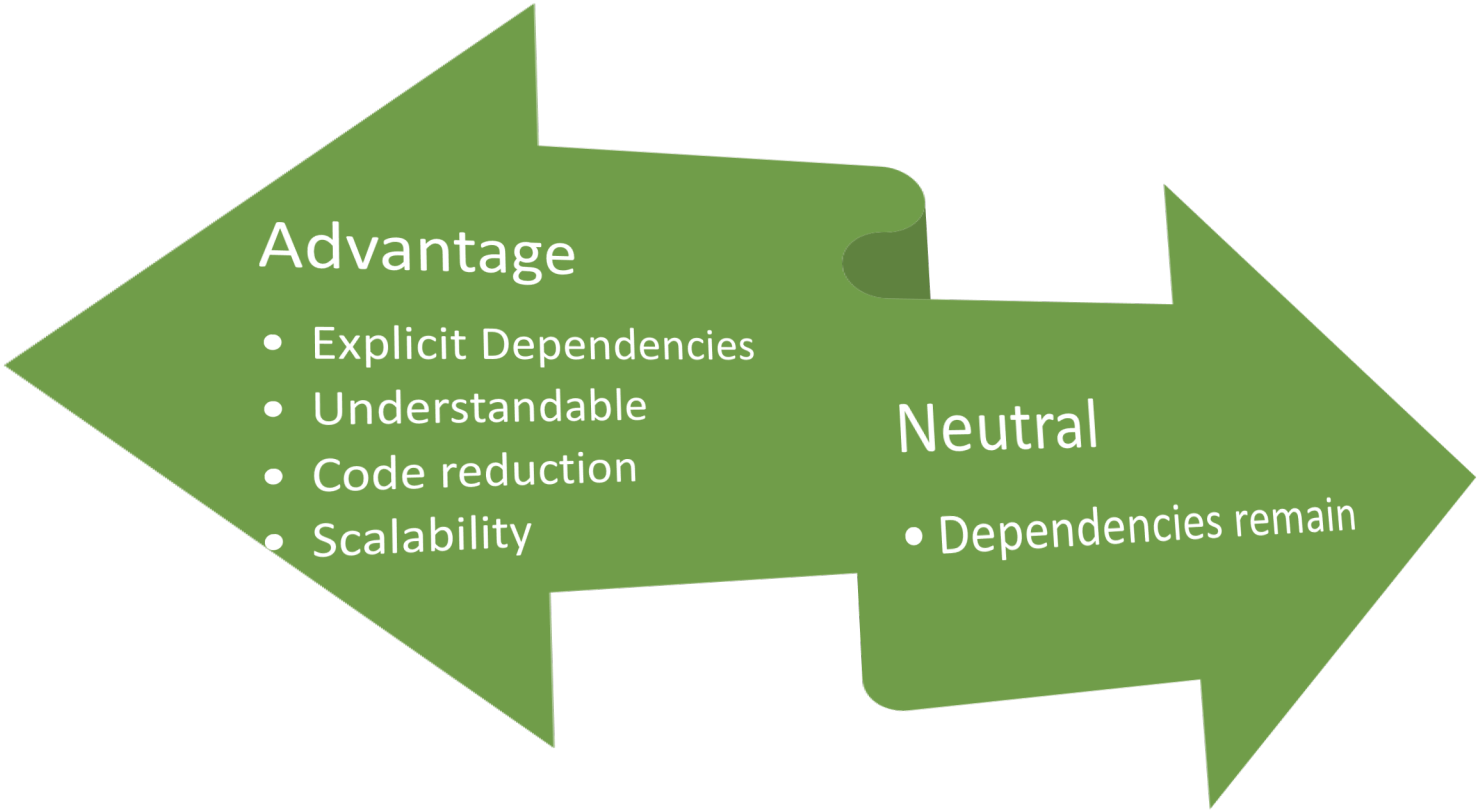
Challenge I: Manage Dependent Services

- Design time
  - Combine dependent services
  - Explicitly managed dependencies
- Compile time
  - Generate microservices
- Deployment
  - No changes

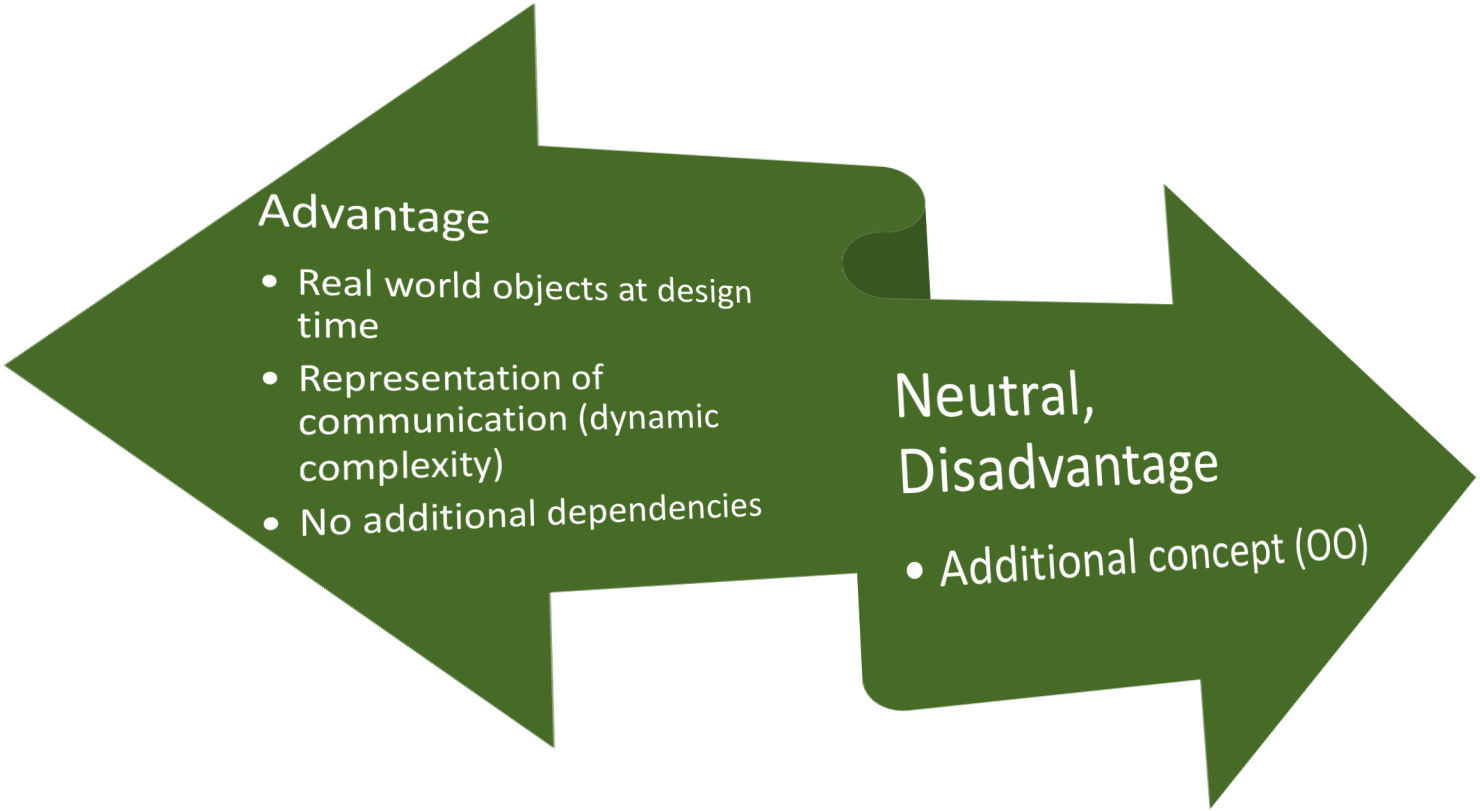
Challenge II: Representation of Real World Objects

- Design Time
  - Real world objects -> classes
- Compile time
  - Microservices <- methods
  - Communication of services <- objects
- Deployment
  - No changes

Challenge I: Review



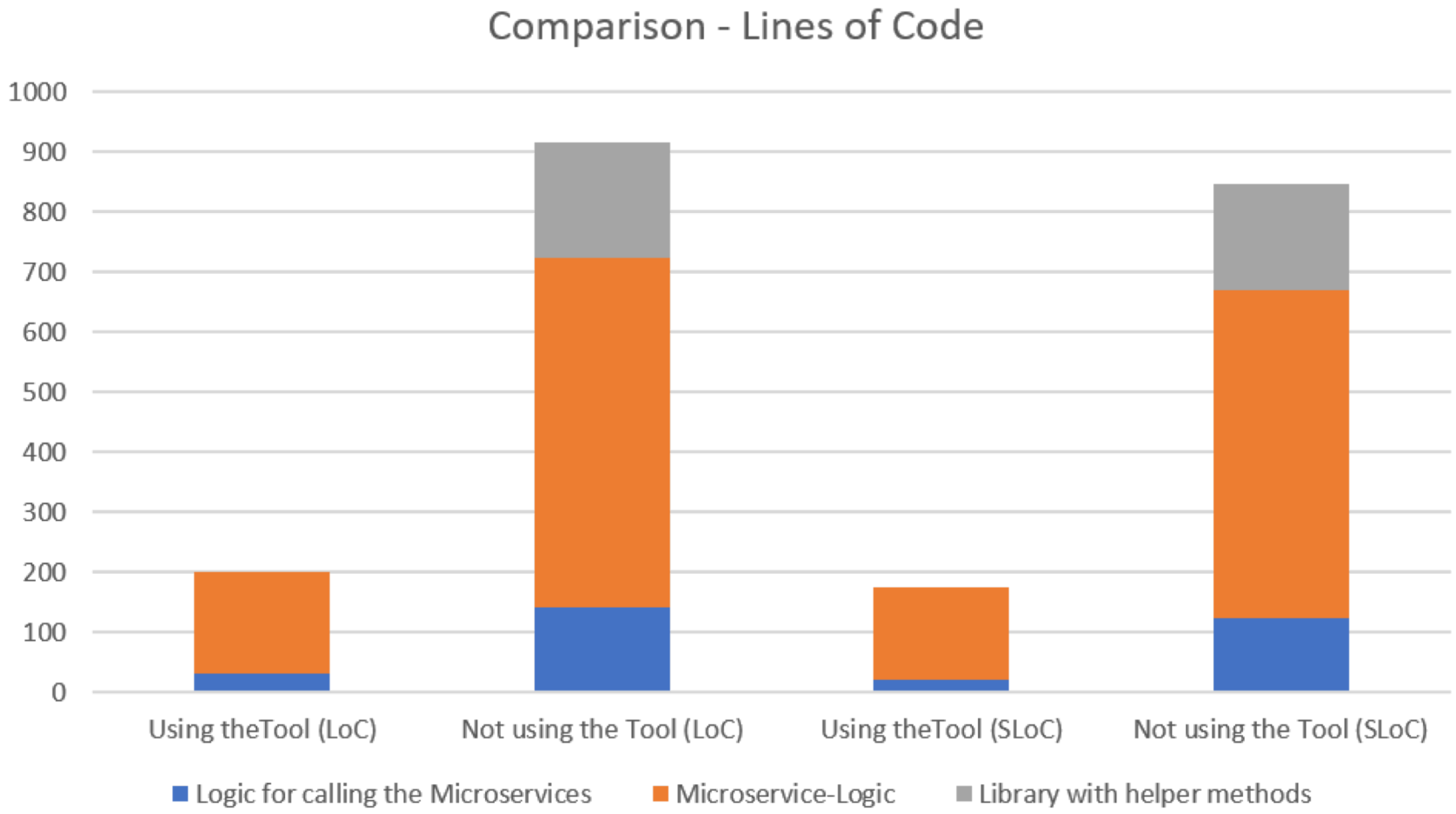
Challenge II: Review



Prototype realization

- Transformation of classes to microservices
- Generation of communication logic
- Technology
  - C#
  - Azure

First Results



Thanks for Listening