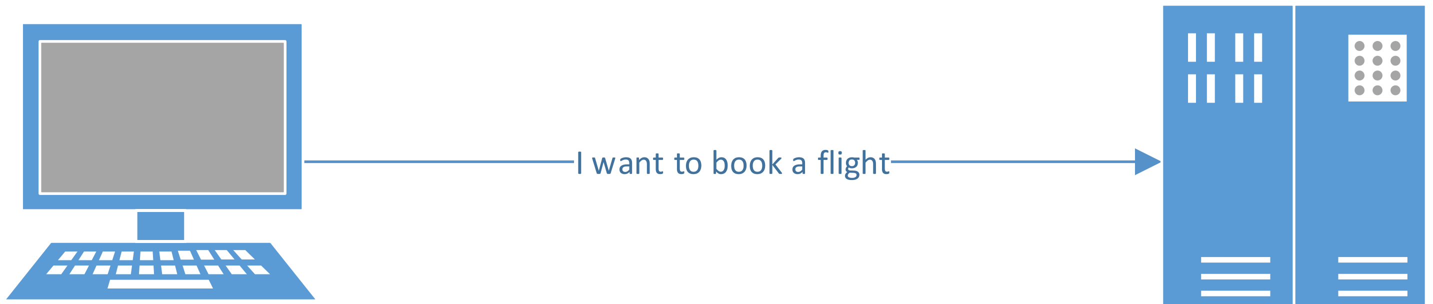
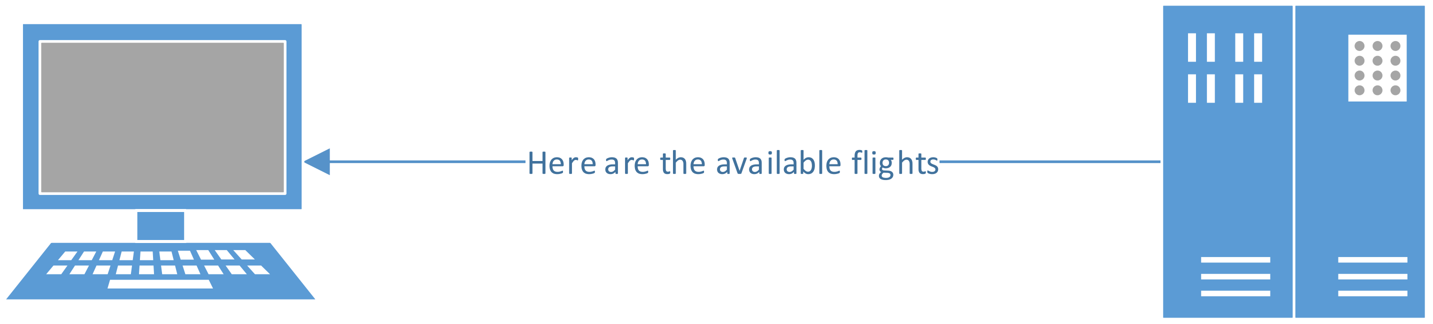
Microservice Architecture

Benefits vs. Monolithic Architecture

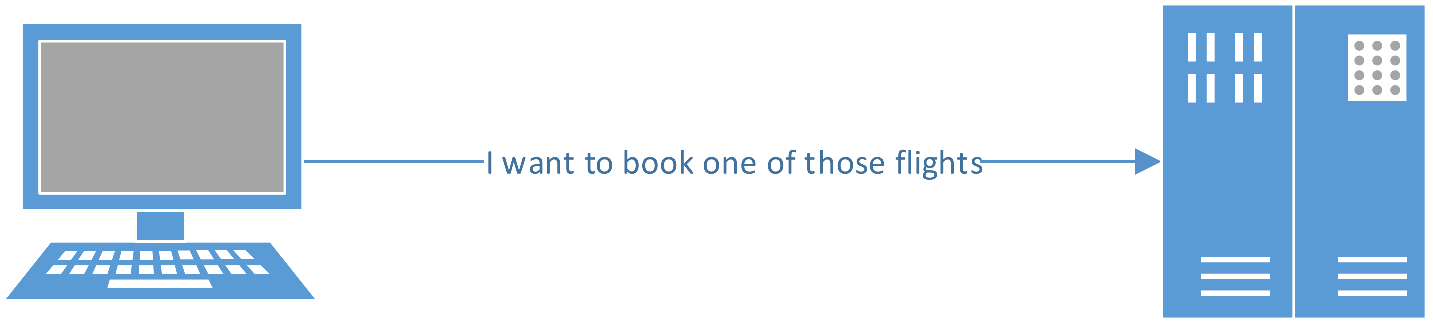
Monolithic Flow #1



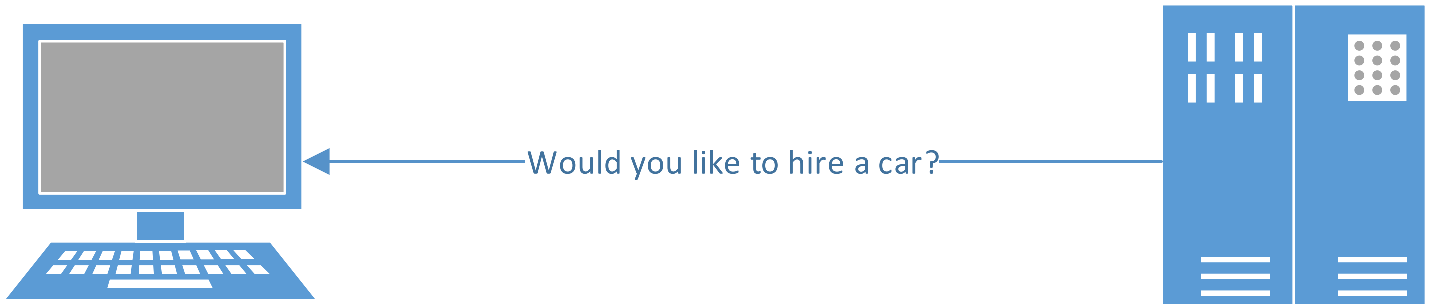
Monolithic Flow #2



Monolithic Flow #3



Monolithic Flow #4



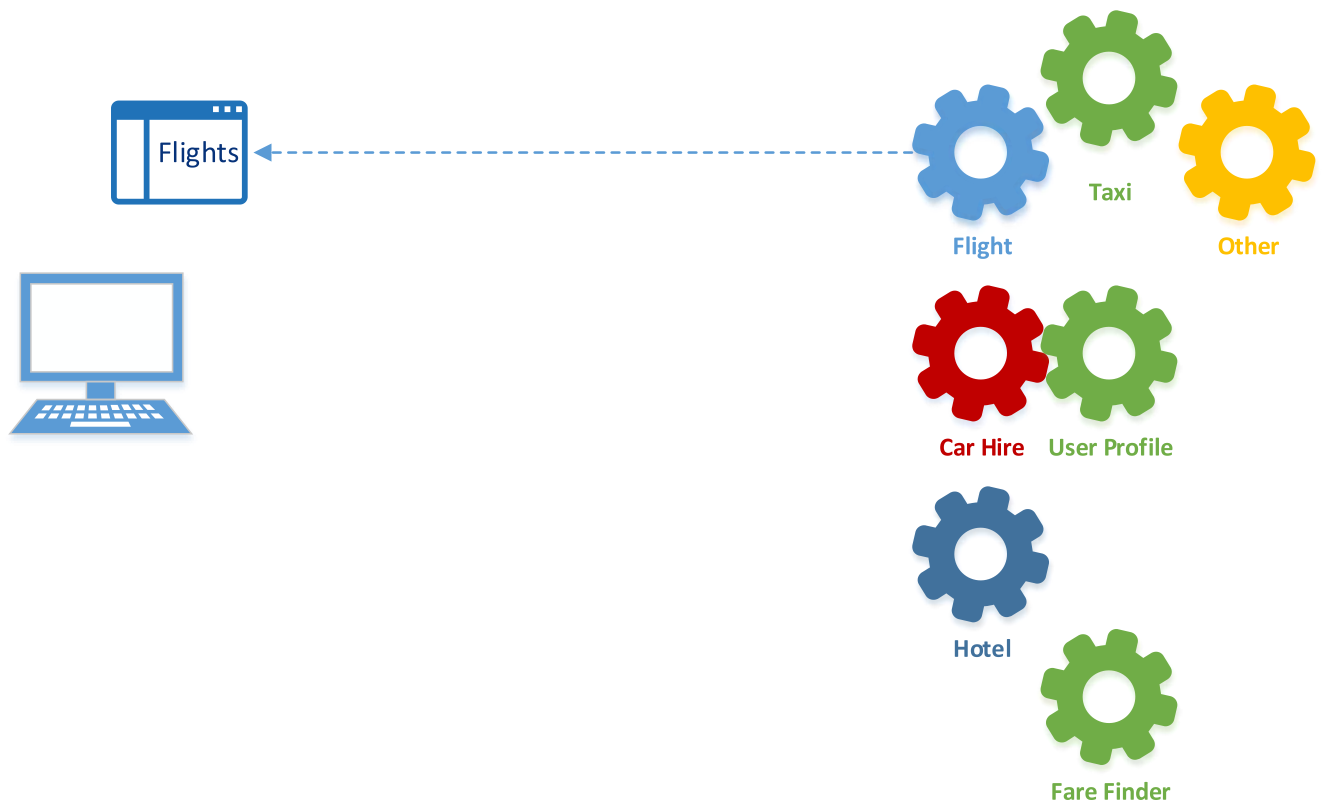
Business Drawbacks

- One-way communication
- Customer is in control
- Website is idle when user is idle
- Limited window of opportunity to interact
- Reduced scope for ancillary revenue

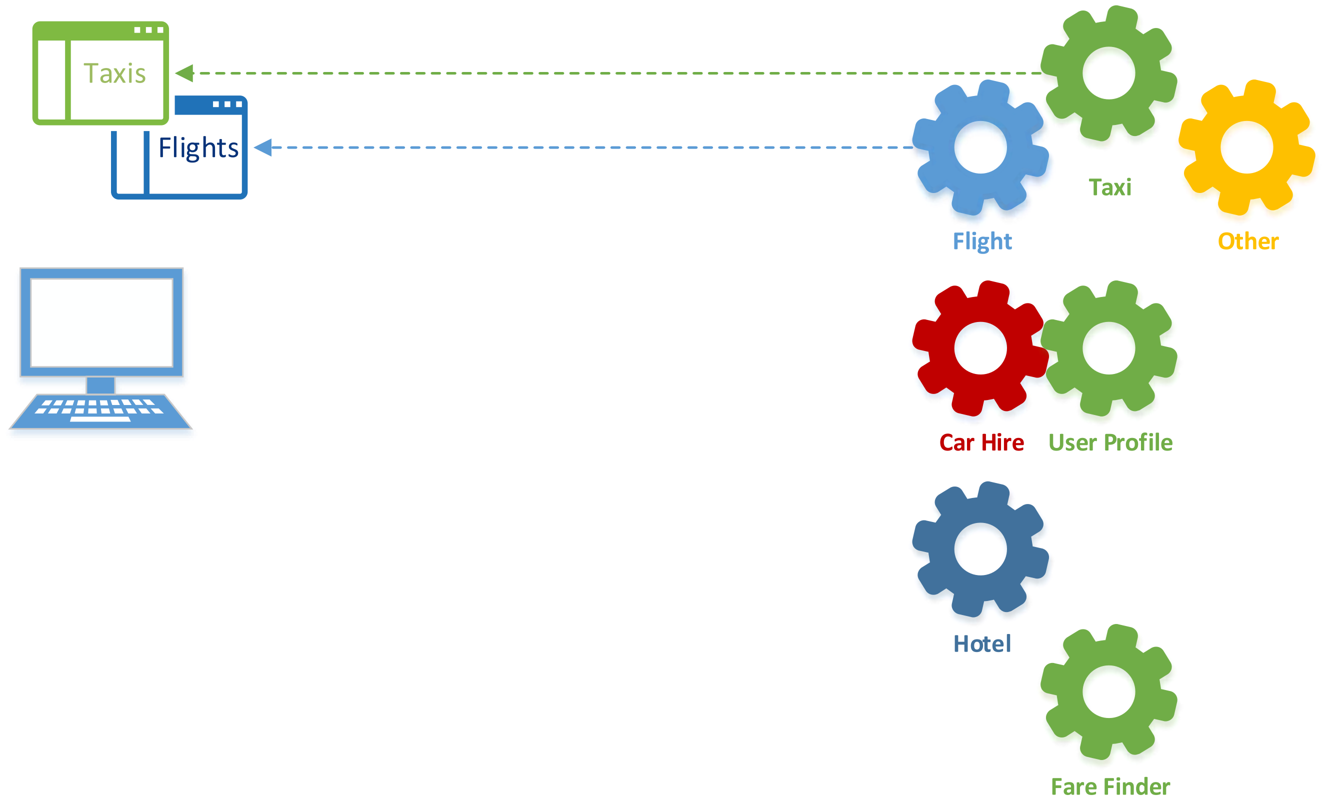
Technical Drawbacks

- Results in dependencies
- Failure affects everything
- Change is slow
- Scale is expensive (minor features require unilateral scale)
- Steep learning curve
- Technology stack is limited to specific skillsets
- Introduces legal pitfalls (PCI DSS, Compliance)
- Duplicated components due to lack of explicit boundaries
- Rigid – likely to break under pressure

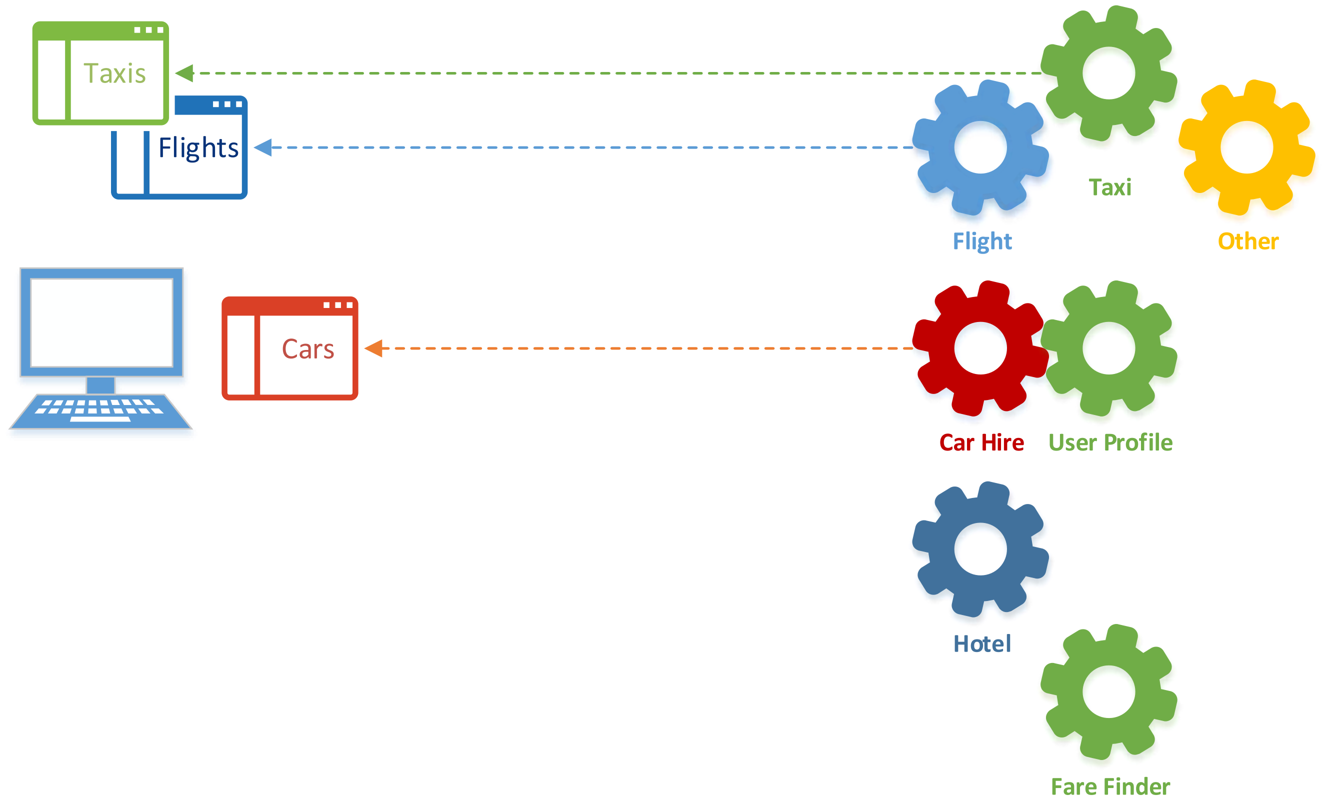
Microservice Flow #1



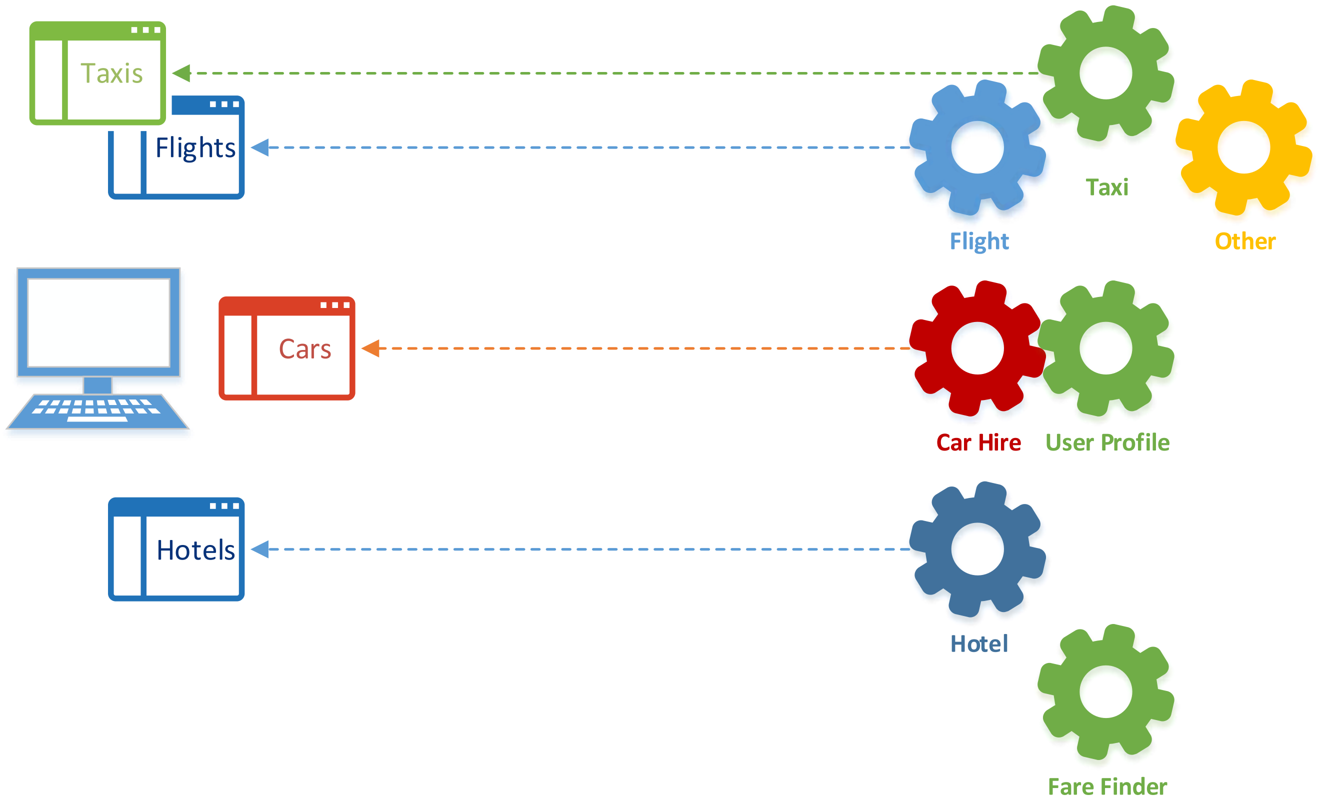
Microservice Flow #2



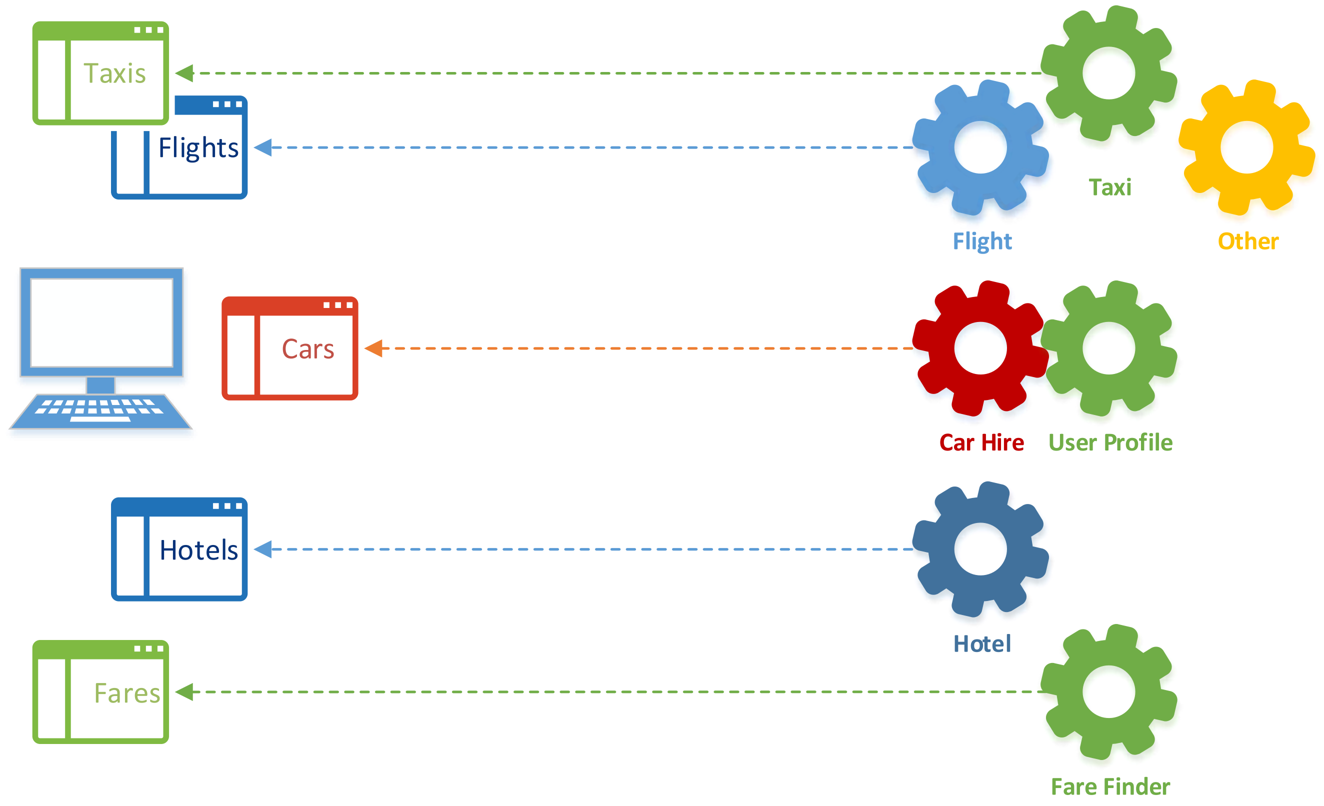
Microservice Flow #3



Microservice Flow #4



Enhanced Flow Step #5



Business Benefits

- Two-way communication
- We’re in control (think Google)
- APIs are always working
- Unlimited opportunities to interact
- Broader scope for ancillary revenue

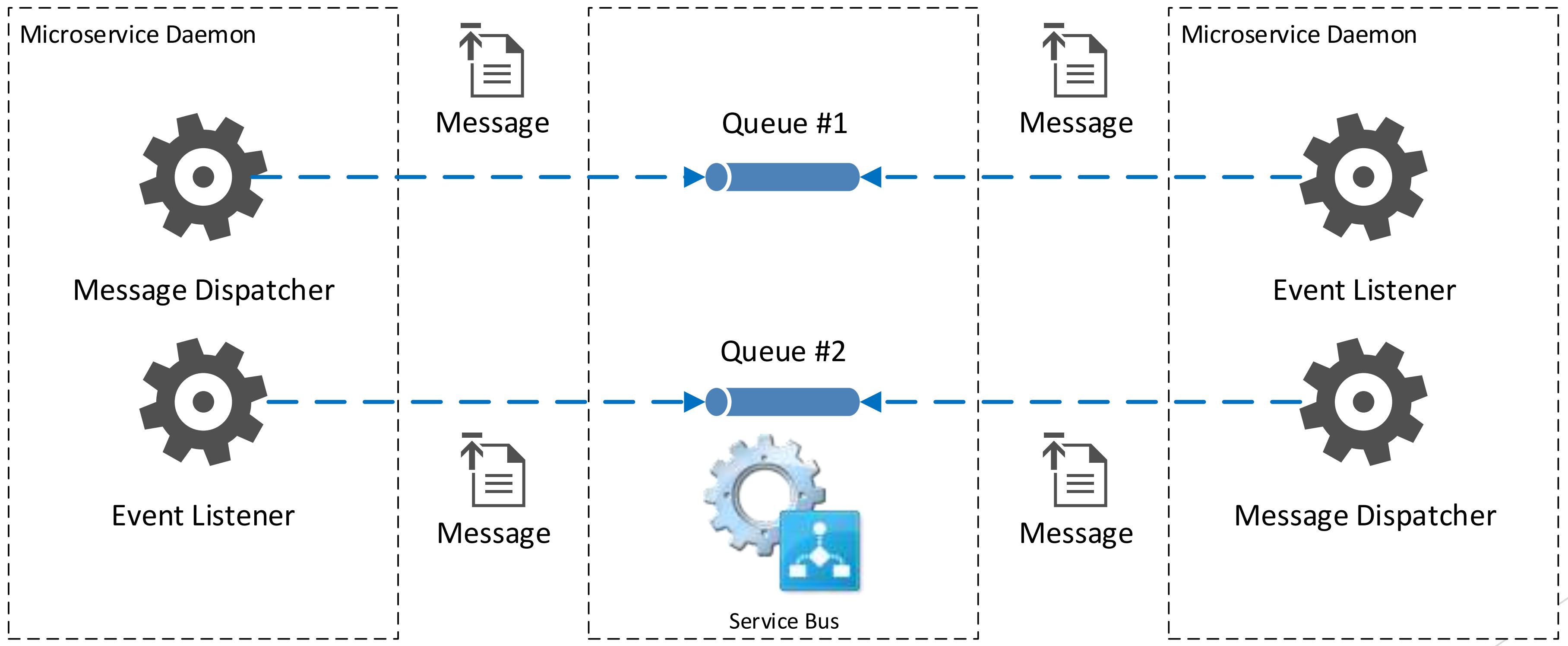
Technical Benefits

- Eliminates dependencies
- Failure is isolated
- React to change quicker
- Scale is less expensive (APIs scale individually)
- More intuitive learning curve
- Technology stack is not limited to specific skillsets
- Shielded from legal pitfalls
- Reusable components
- Flexible – will bend rather than break under pressure

Anatomy of a Microservice

- Decoupled Middleware design pattern
- Microservices communicate across a Service Bus (Kafka, RabbitMQ, NATS.io)
- Service Bus is centralised
- Microservices are distributed
- TCP communication is generally favoured
- Microservices do 1 thing only, and they do it very well
- Not restricted to a specific technology
- Facilitates Circuit Breaker, Bulkhead, and Handshaking design patterns
- Avoids cascading failure

Anatomy of a Microservice



References

- http://insidethecpu.com/2015/05/22/microservices-with-c-and-rabbitmq/
- http://martinfowler.com/articles/microservices.html
- http://microservices.io/
- http://cdn.oreillystatic.com/en/assets/1/event/79/Stability%20Patterns%20Presentation.pdf

Questions

- How do we achieve Continuous Integration/Deployment?
- Monitoring sounds complicated
- Why now? Is there a reason this hasn’t been done up until now?
- Can we deploy segment-by-segment?
- Which brokers offer message-durability?
- How will this affect UI development?
- How do we manage the extra overhead involved in multiple service calls?