สยามชำนาญกิจ
SIAM CHAMNAN KIT

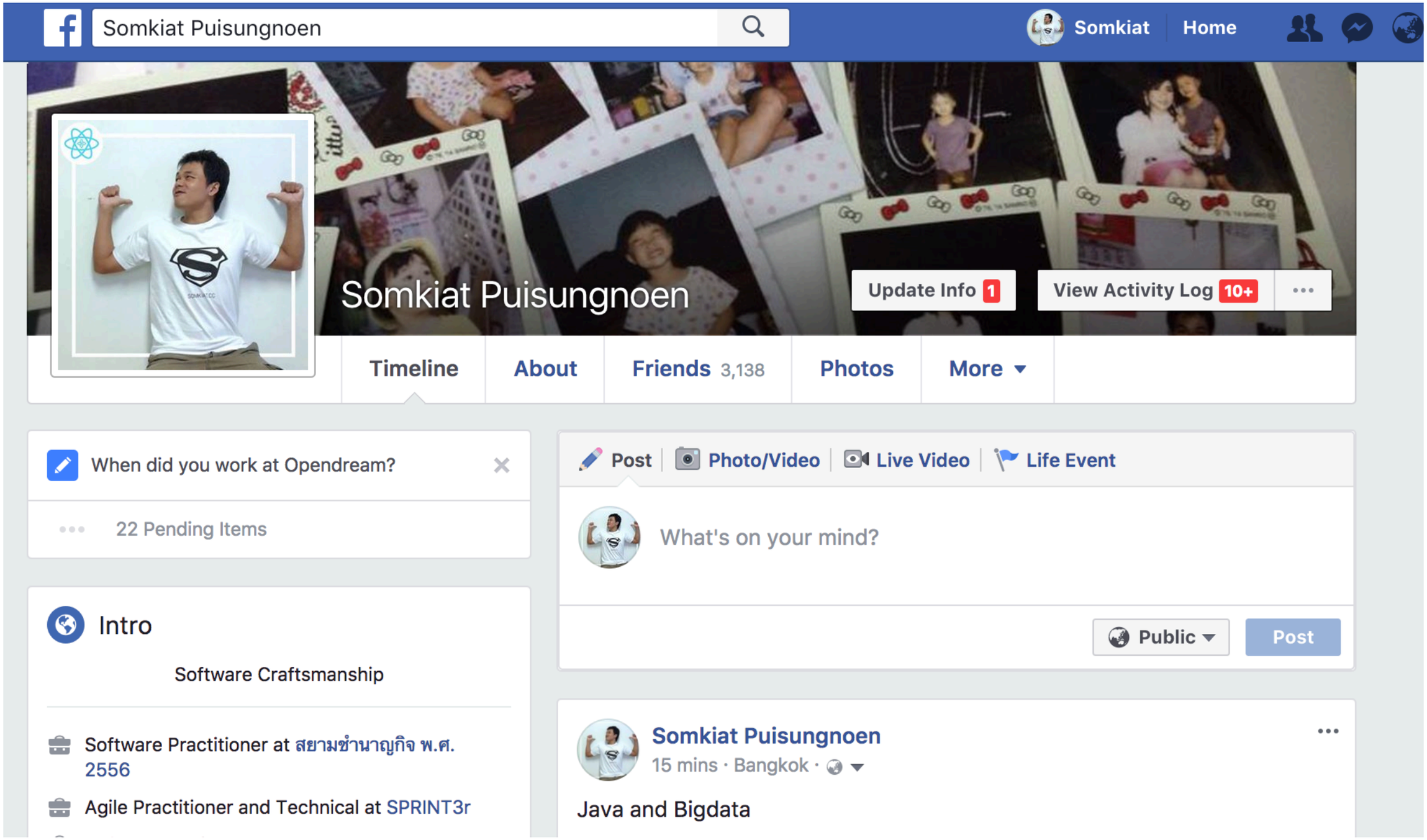
Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

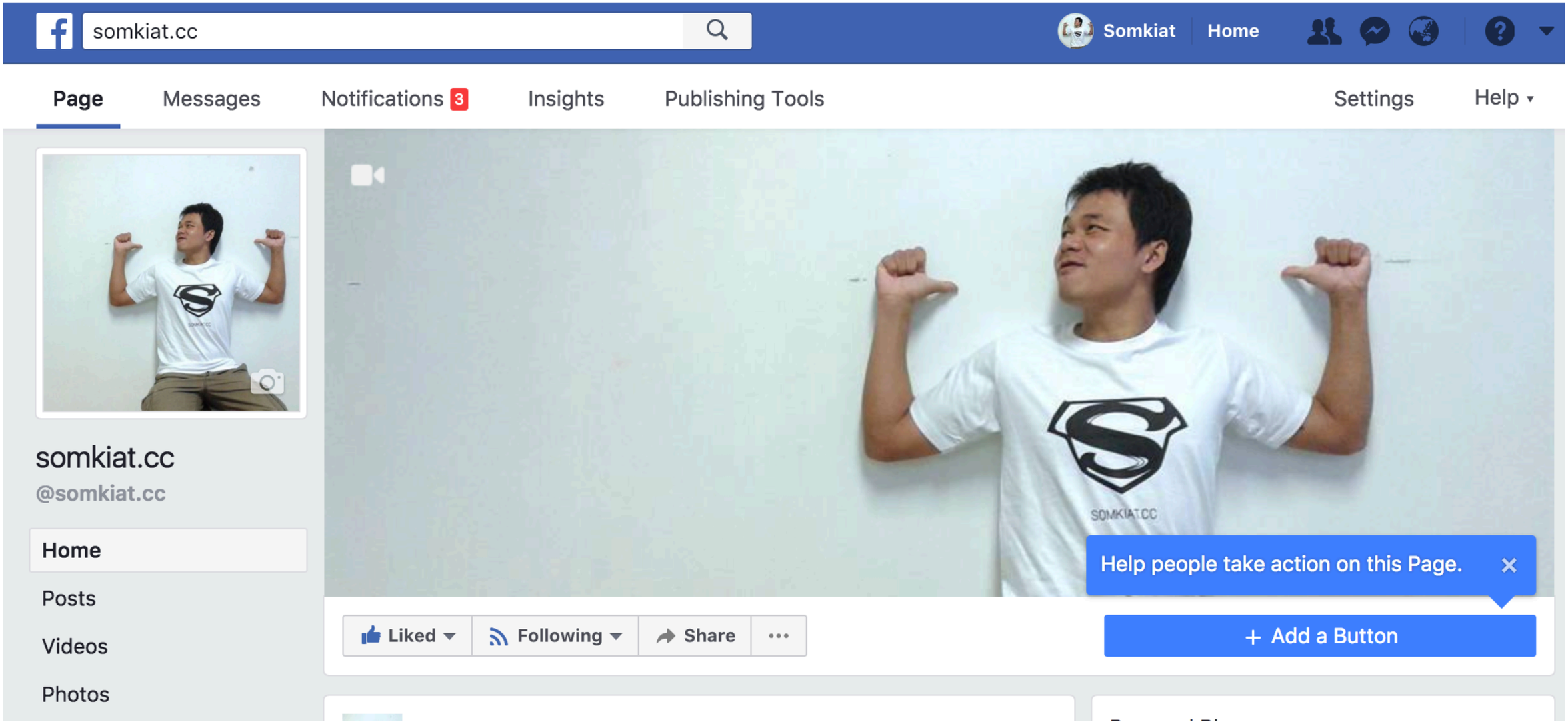
สยามชำนาญกิจ
SIAM CHAMNAN KIT

SHU HA RI
SOFTWARE DEVELOPMENT SCHOOL

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Topics

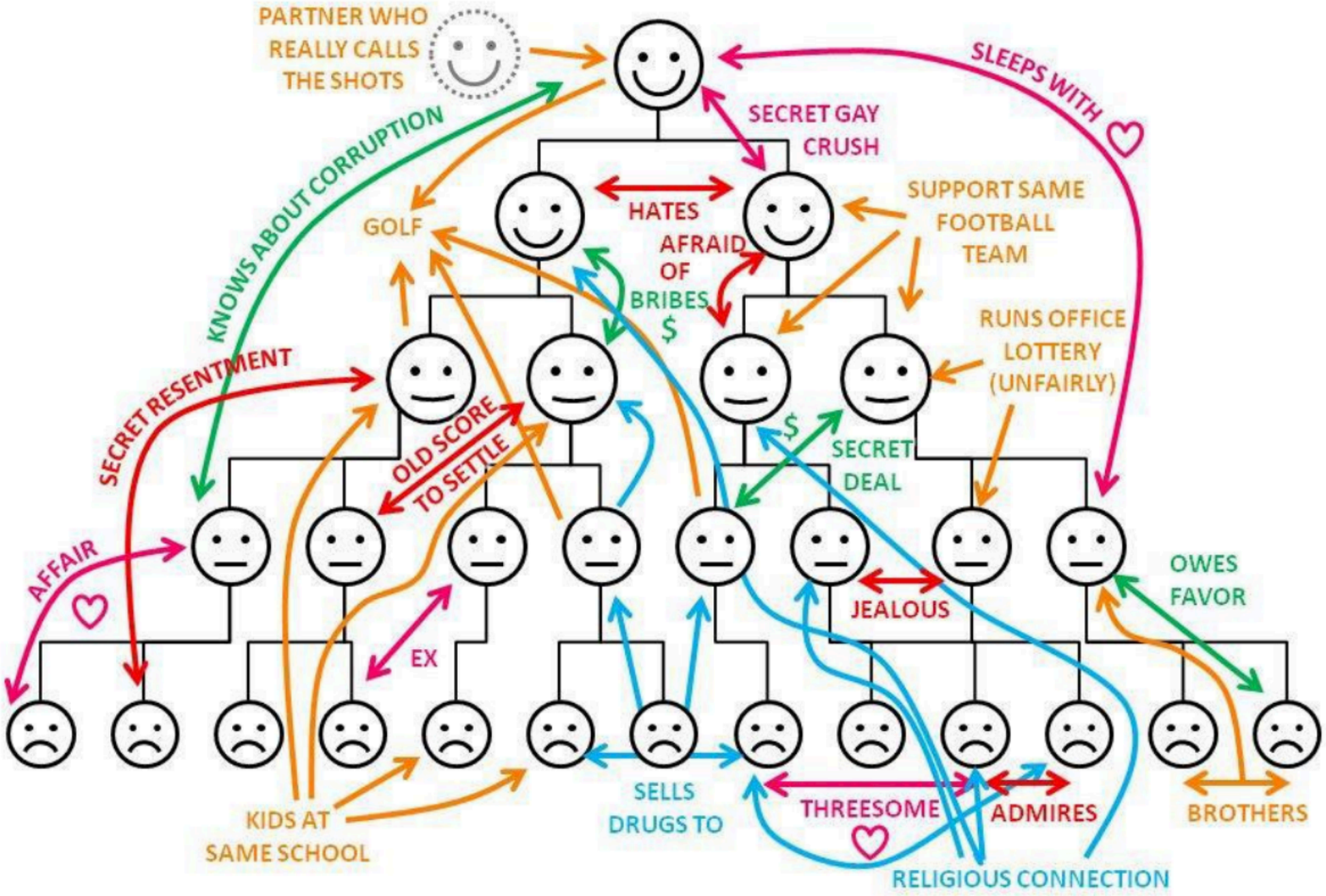
Cloud Native Application
Evolution of architecture
Microservice architecture
How to decompose app to Microservice
Communication between services
Data consistency
Workshop

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

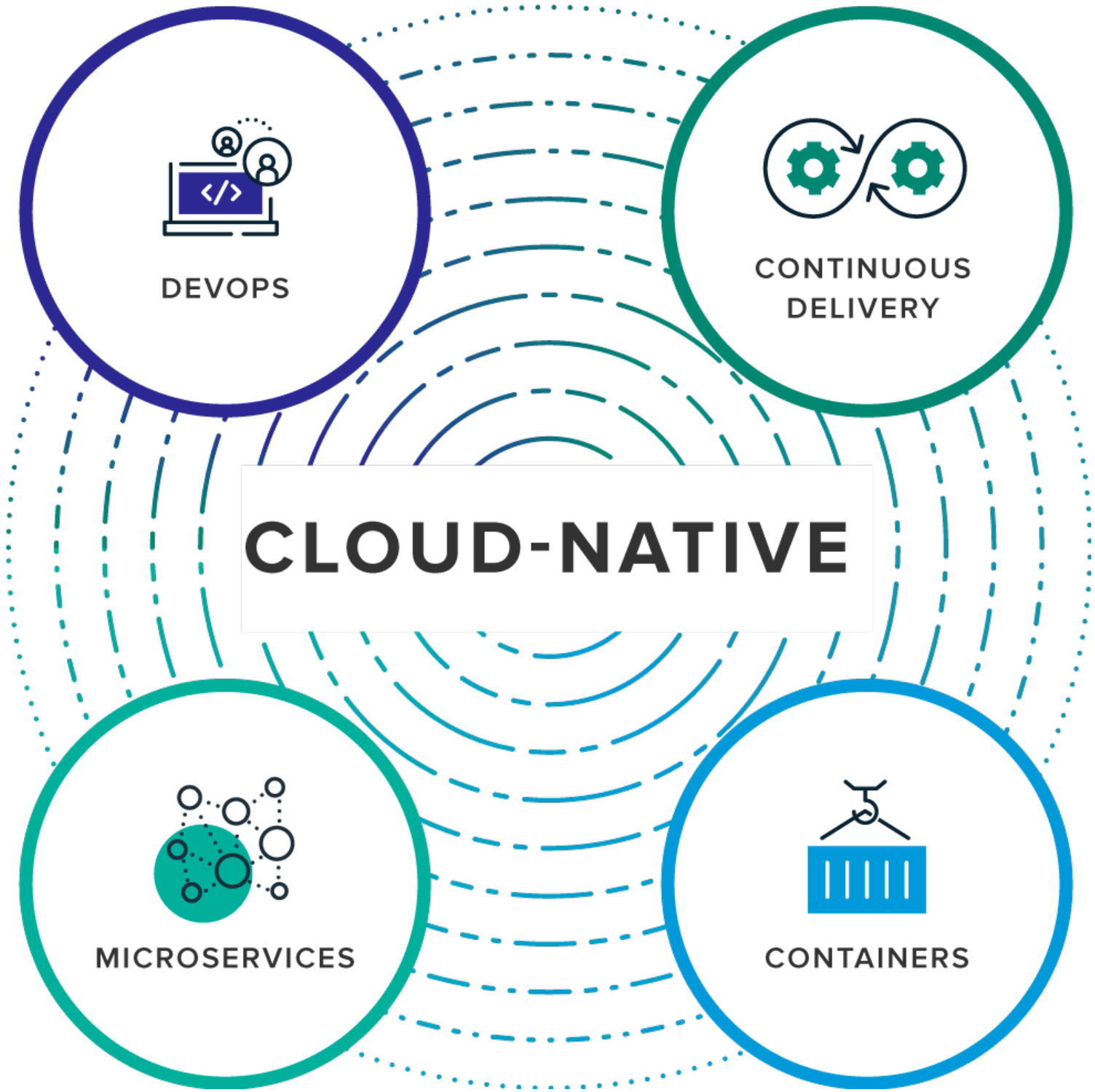
”organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”

- Melvin Conwey, 1968 -

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



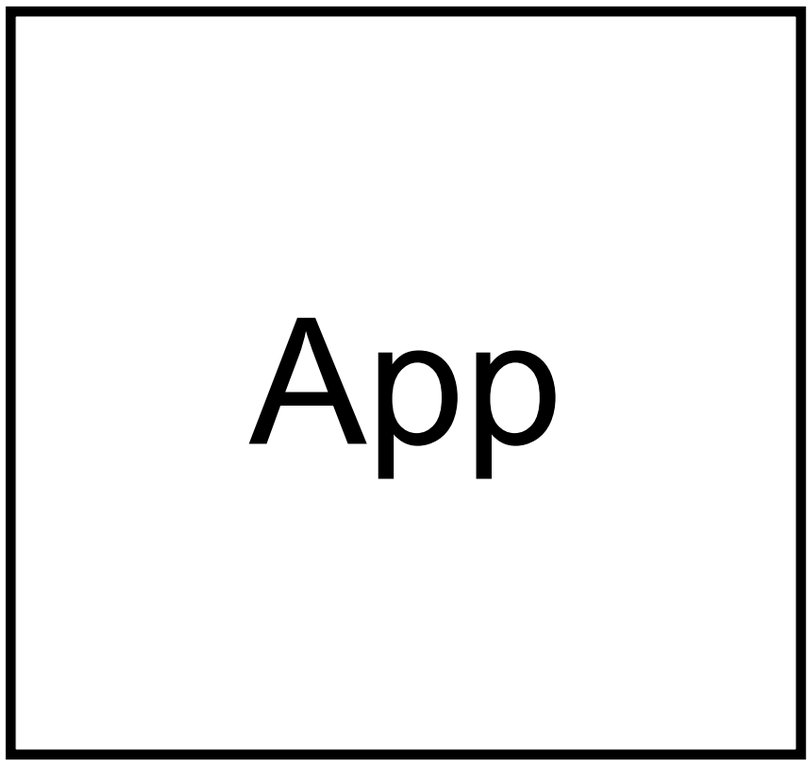
https://pivotal.io/cloud-native

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Evolution of Architecture

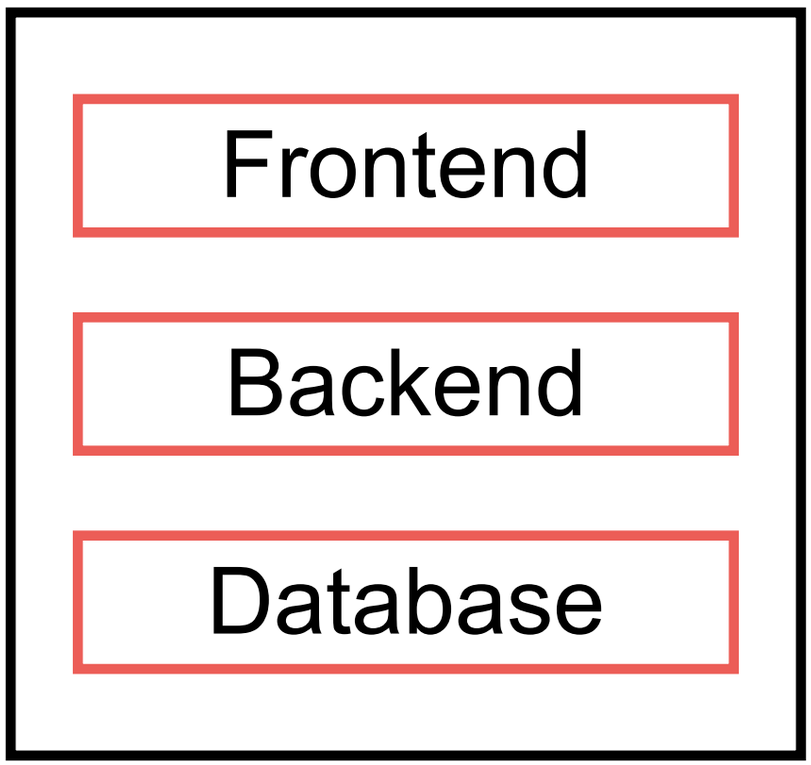
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Monolith



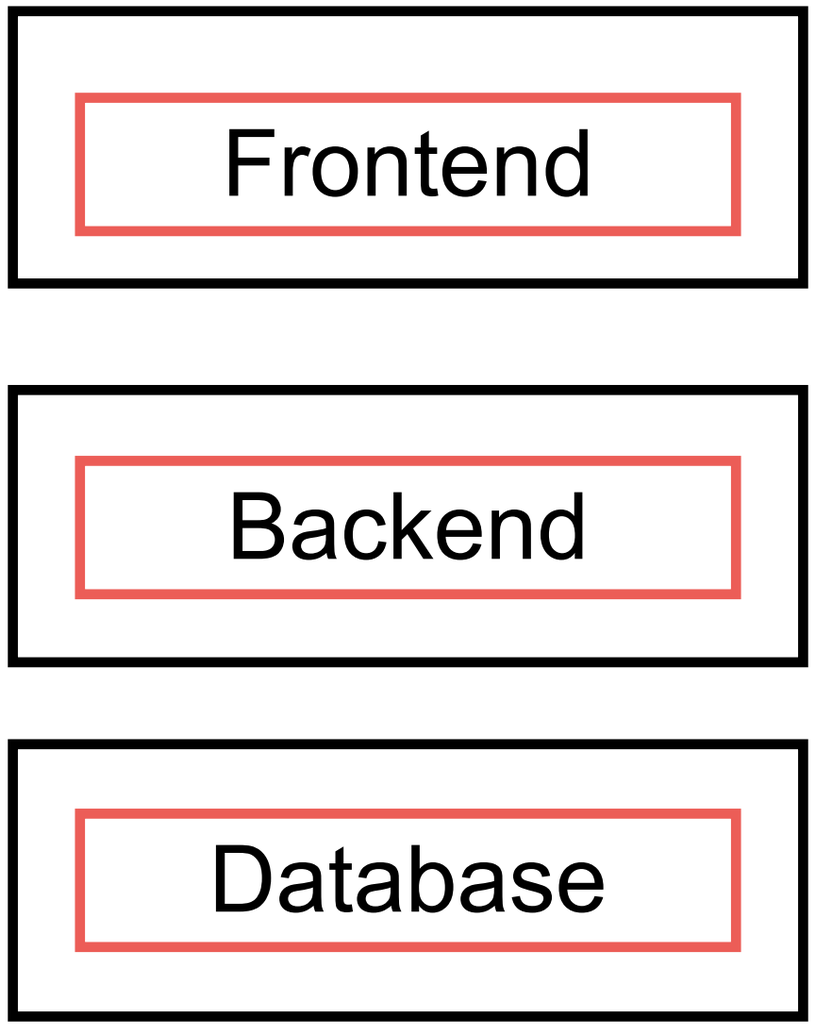
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Layer



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Layer



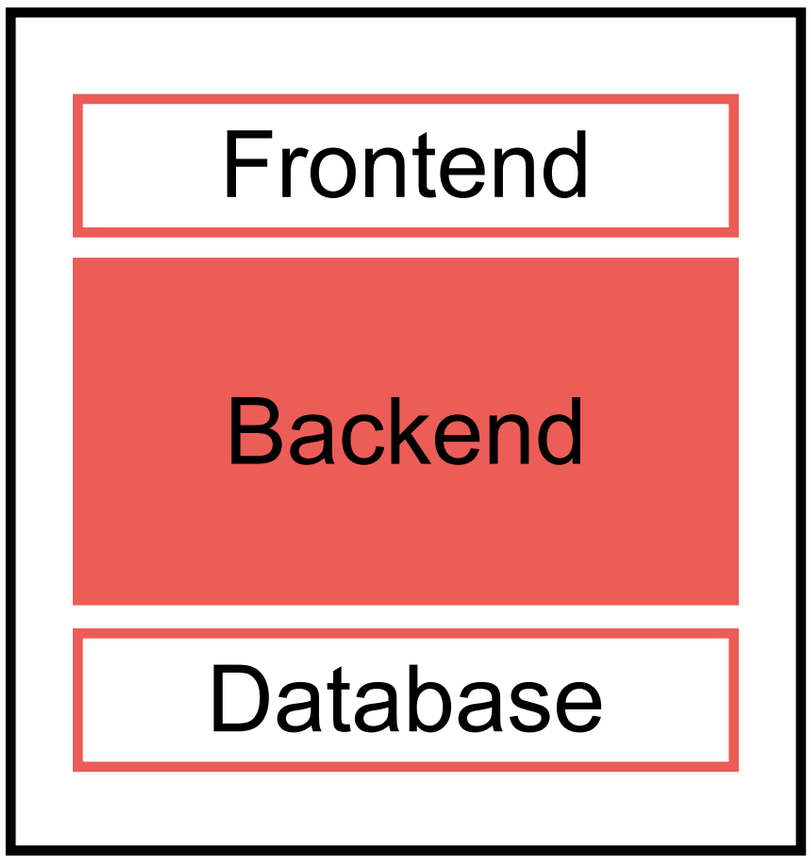
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Monolith

Easy to develop
Easy to change
Easy to testing
Easy to deploy
Easy to scale

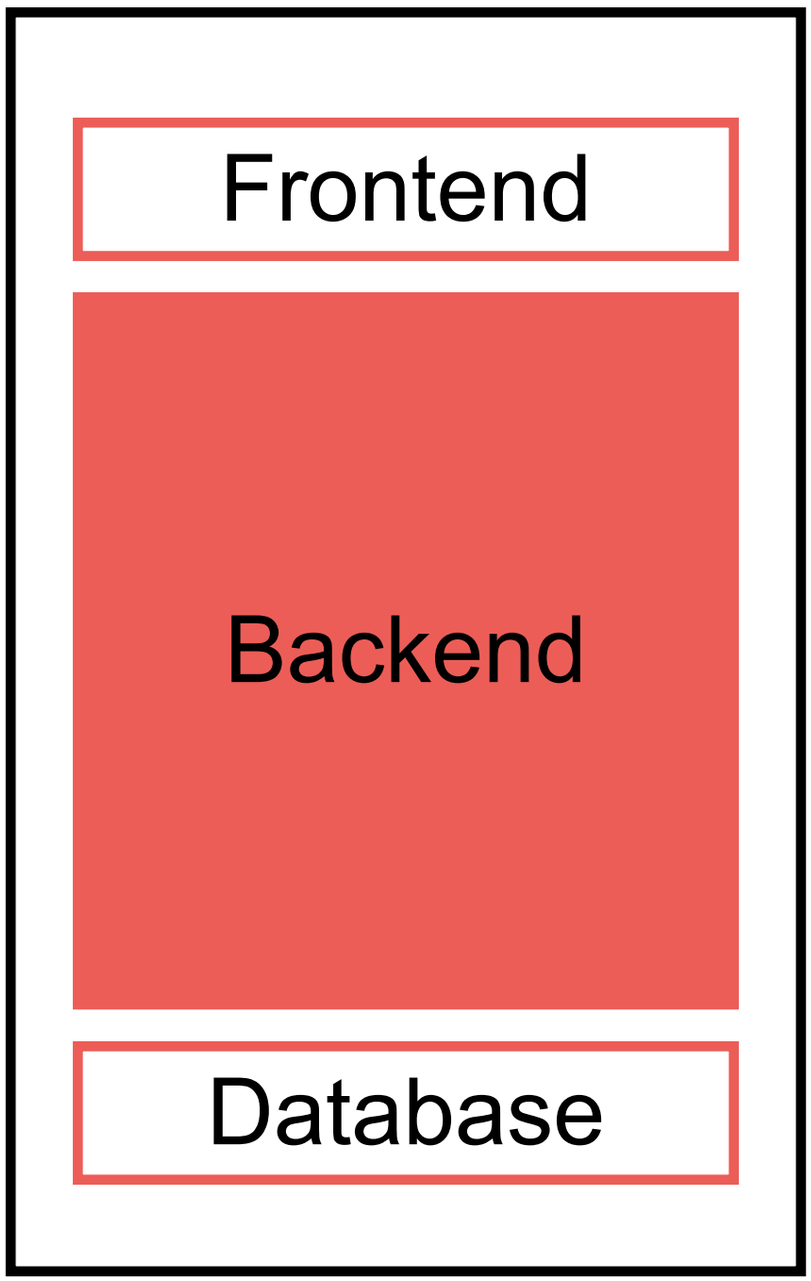
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

More features …



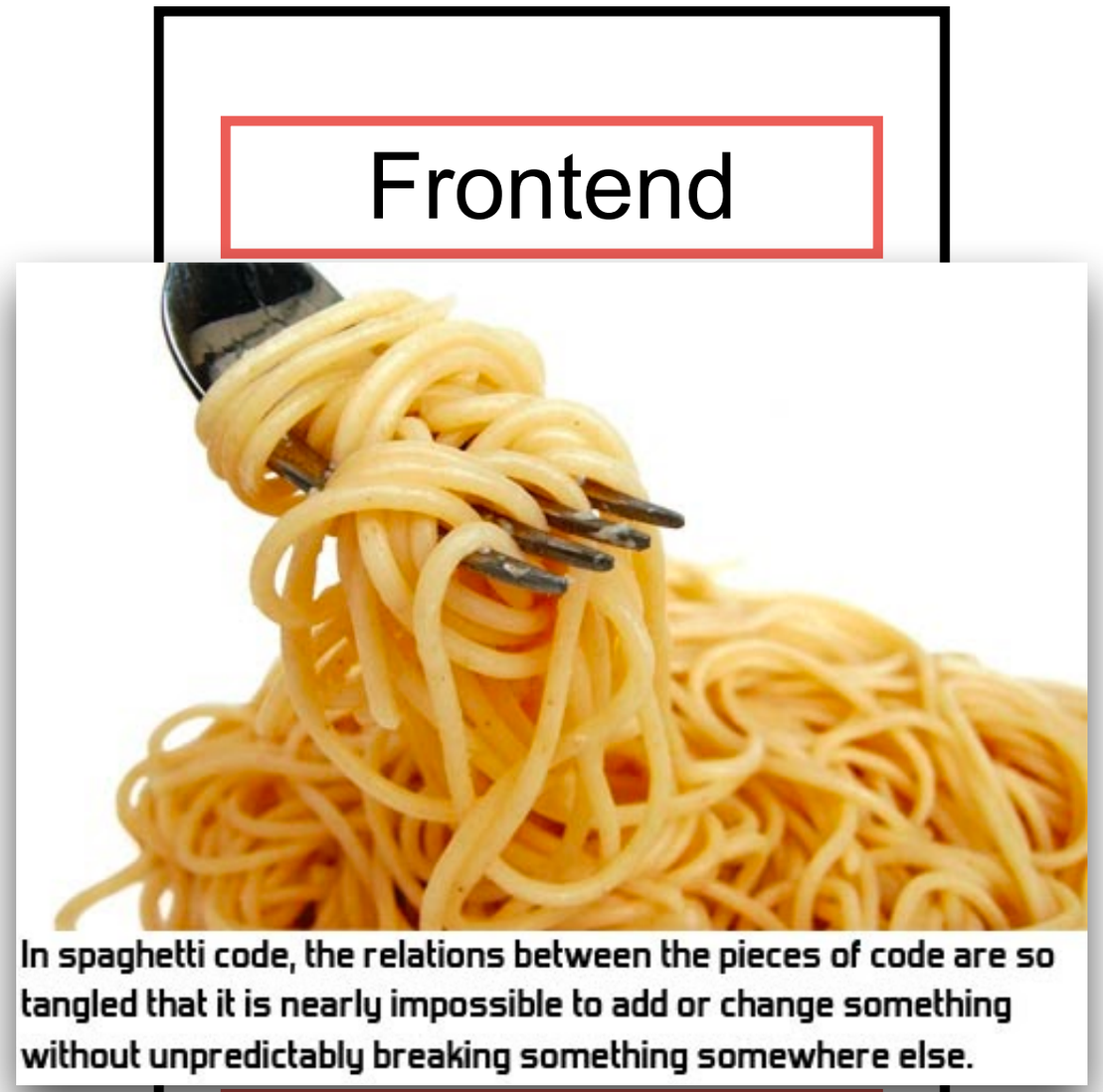
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

More features ...



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

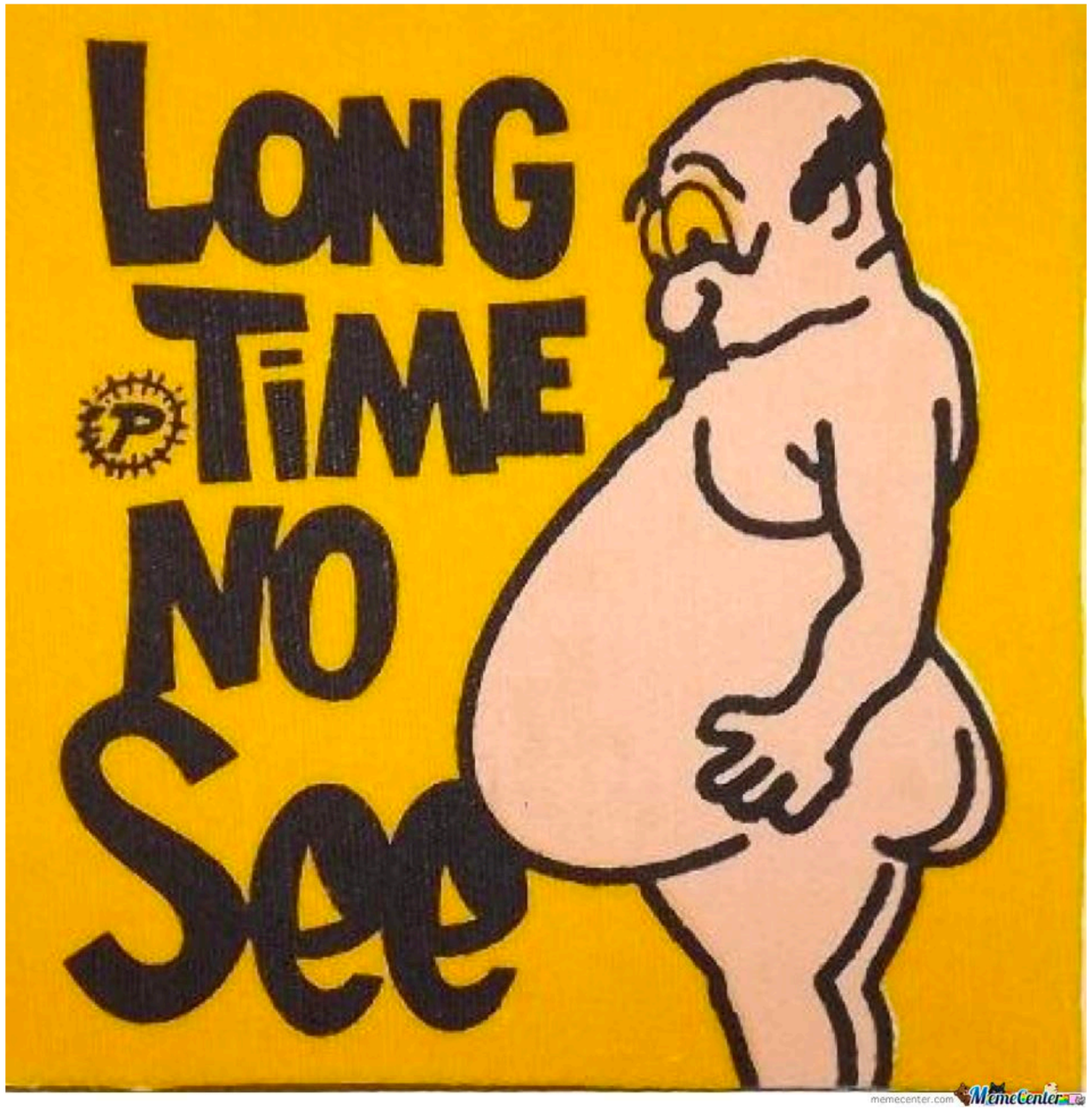
More features …



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

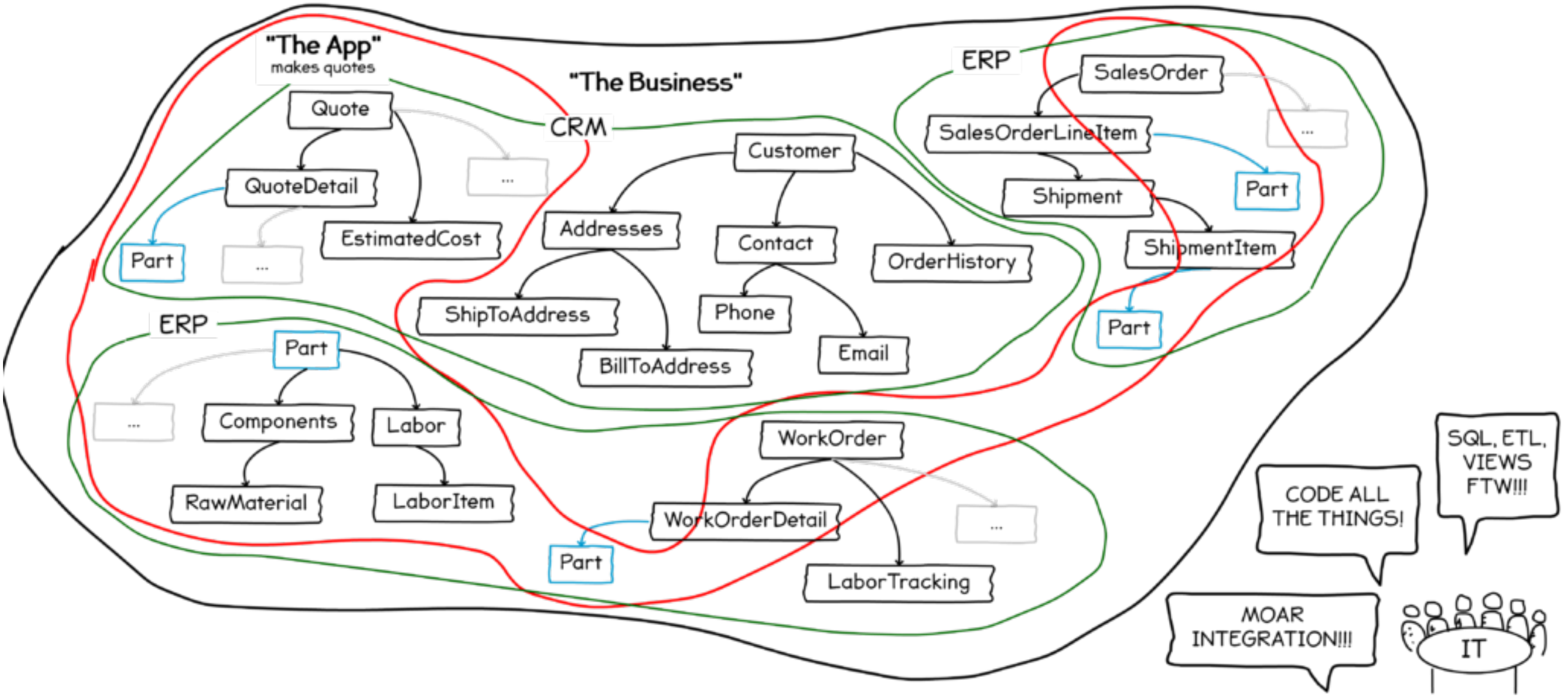
More features …

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Monolith Hell

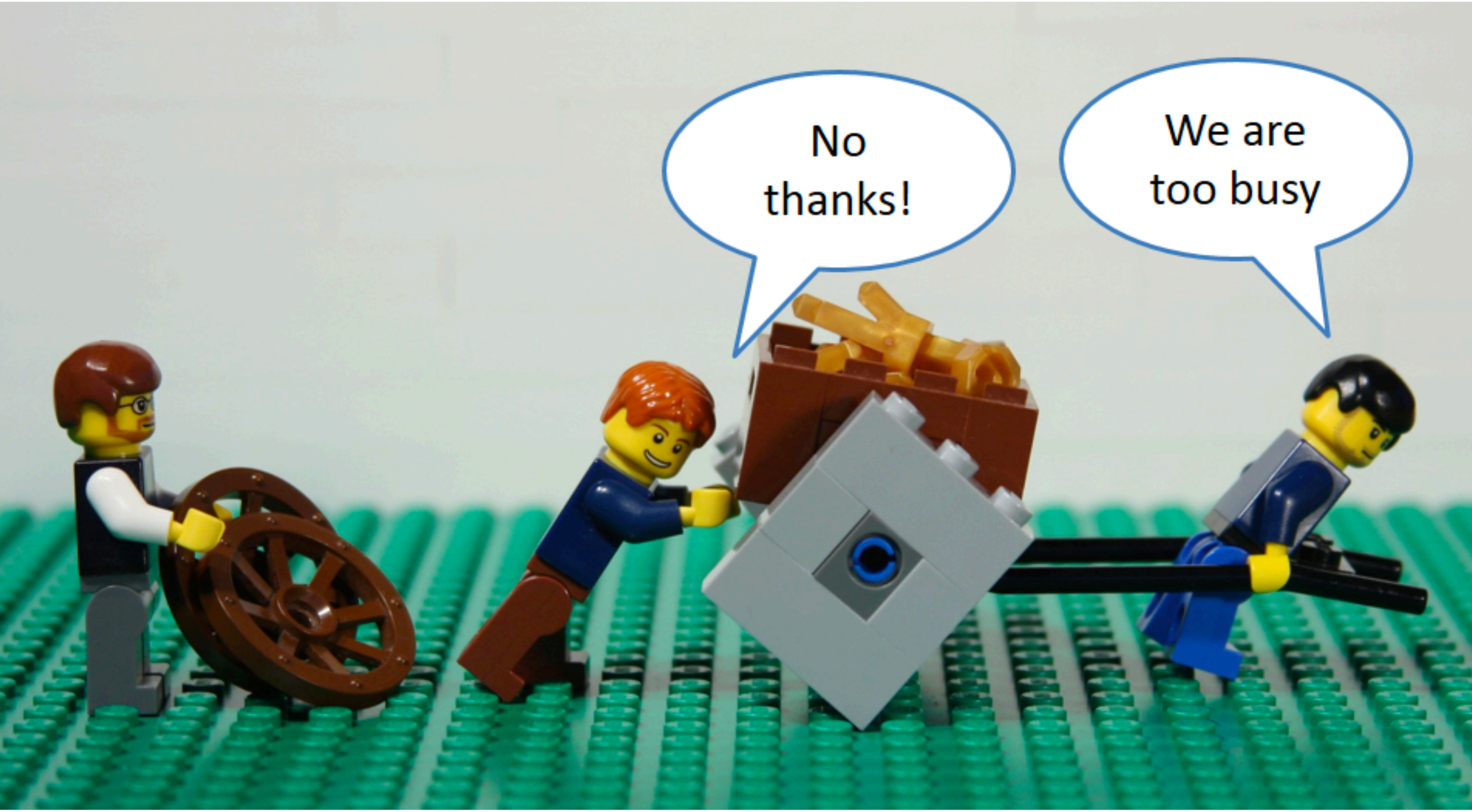


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

We need to improve

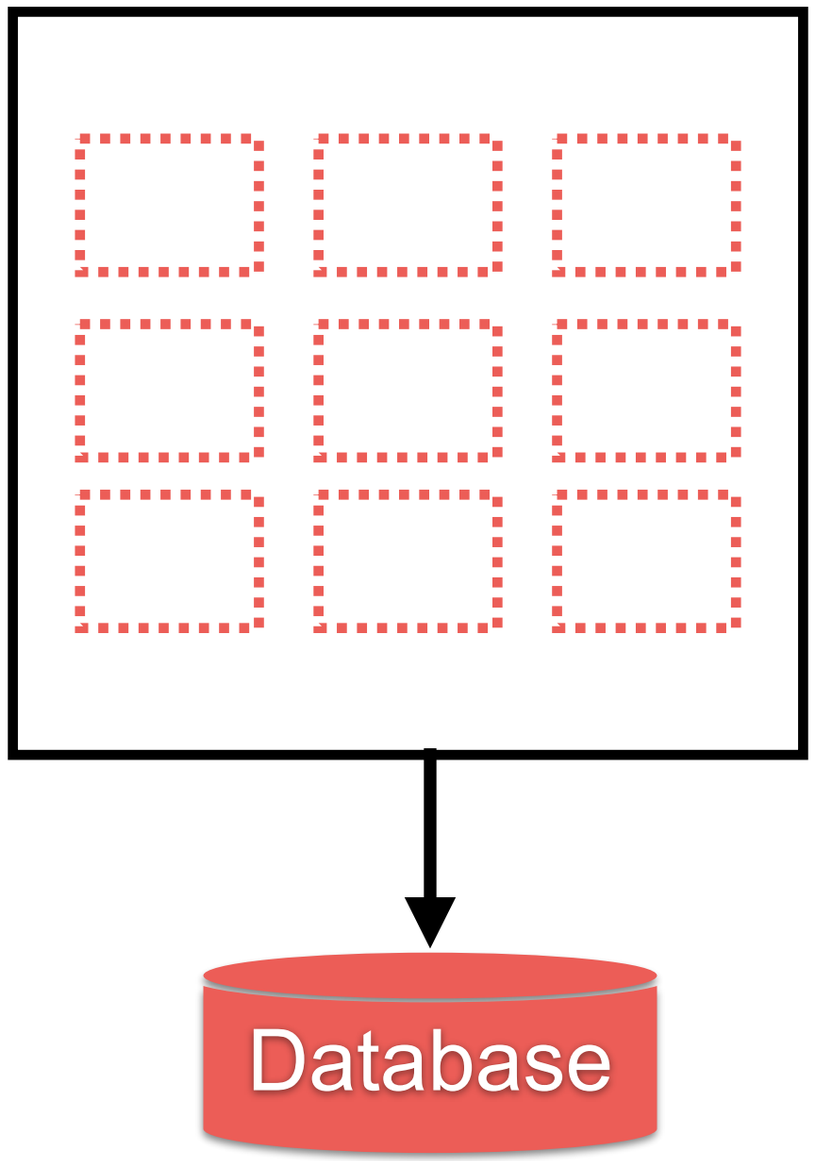
We need to get better

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



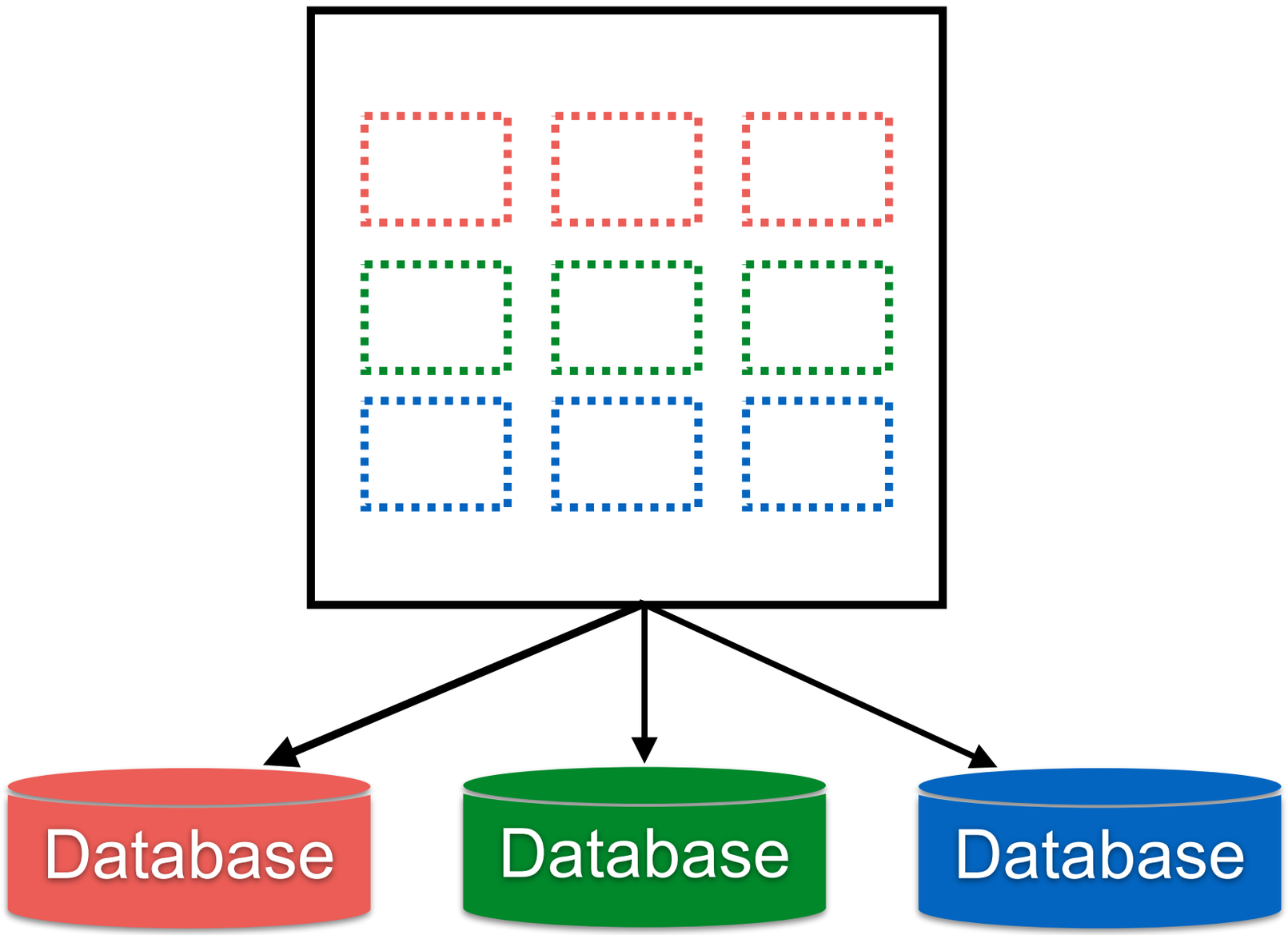
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Modular monolith



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Modular monolith with a decompose database



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

How to scale ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

What if we don’t use all modules equally ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

How can we update individual database ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Do all modules need to use the same Database, Language and Runtime … ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

We need to improve

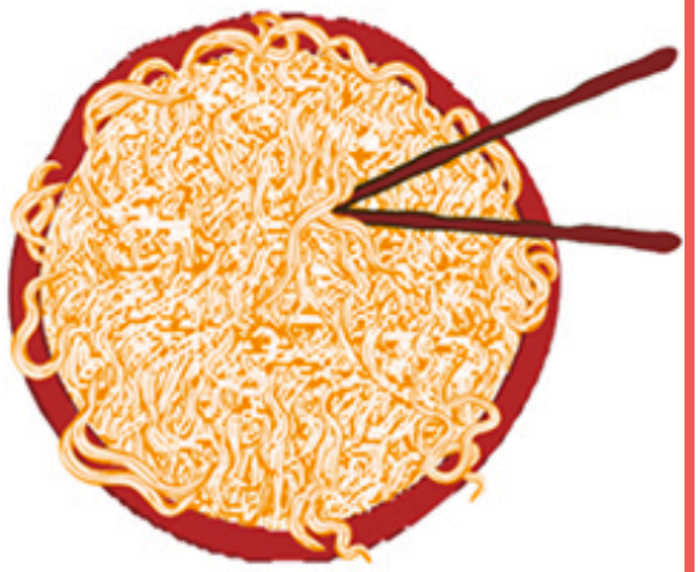
We need to get better

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

SOA

1990s and earlier

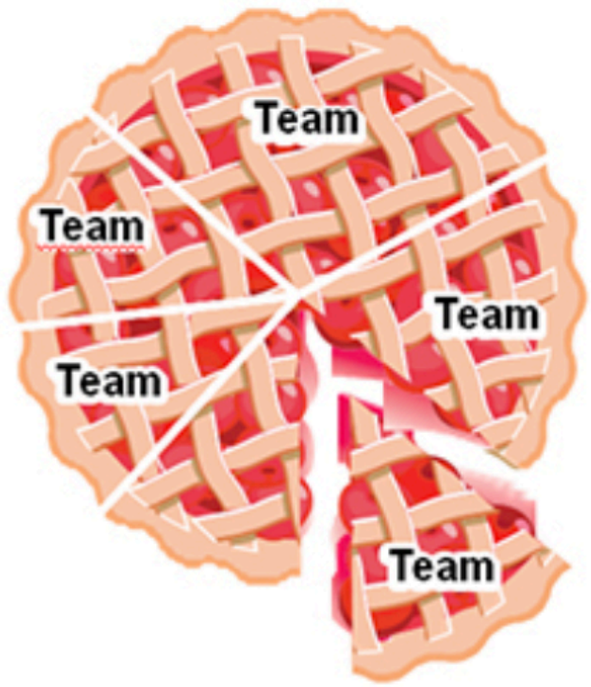
Pre-SOA (monolithic)
Tight coupling



For a monolith to change, all must agree on each change. Each change has unanticipated effects requiring careful testing beforehand.

2000s

Traditional SOA
Looser coupling



Elements in SOA are developed more autonomously but must be coordinated with others to fit into the overall design.

2010s

Microservices
Decoupled

Developers can create and activate new microservices without prior coordination with others. Their adherence to MSA principles makes continuous delivery of new or modified services possible.

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

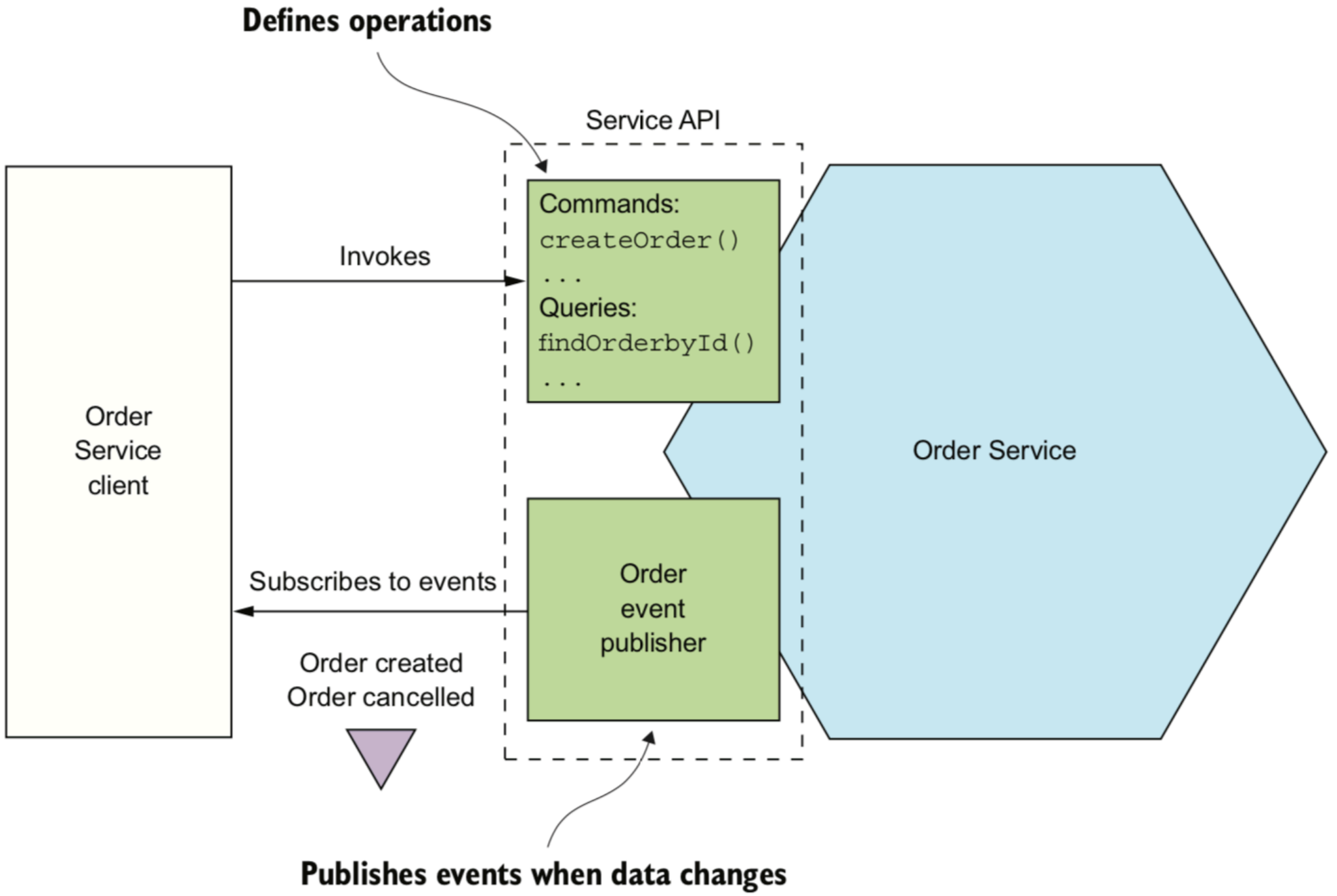
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

What is service ?

Standalone and loosely couple
Independently deployable software component
Implement some useful functionality

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example of service



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

SOA Manifesto

Service orientation is a paradigm that frames what you do.
Service-oriented architecture (SOA) is a type of architecture
that results from applying service orientation.

We have been applying service orientation to help organizations consistently deliver sustainable business value, with increased agility and cost effectiveness, in line with changing business needs.

http://www.soa-manifesto.org/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

Through our work we have come to prioritize:

Business value over technical strategy

Strategic goals over project-specific benefits

Intrinsic interoperability over custom integration

Shared services over specific-purpose implementations

Flexibility over optimization

Evolutionary refinement over pursuit of initial perfection

http://www.soa-manifesto.org/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

Through our work we have come to prioritize:

Business value over technical strategy

Strategic goals over project-specific benefits

Intrinsic interoperability over custom integration

Shared services over specific-purpose implementations

Flexibility over optimization

Evolutionary refinement over pursuit of initial perfection

http://www.soa-manifesto.org/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

Through our work we have come to prioritize:

Business value over technical strategy

Strategic goals over project-specific benefits

Intrinsic interoperability over custom integration

Shared services over specific-purpose implementations

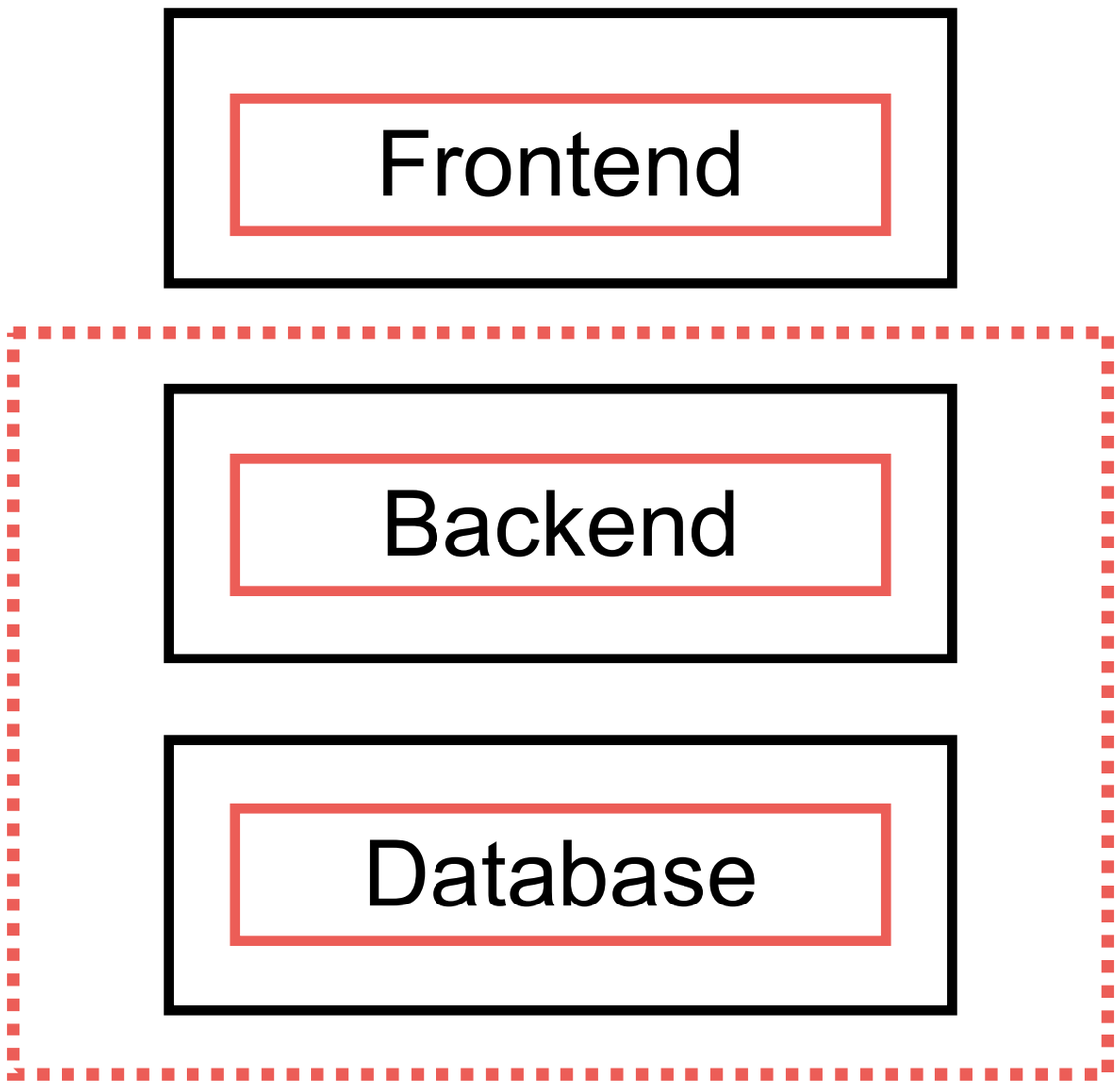
Flexibility over optimization

Evolutionary refinement over pursuit of initial perfection

http://www.soa-manifesto.org/

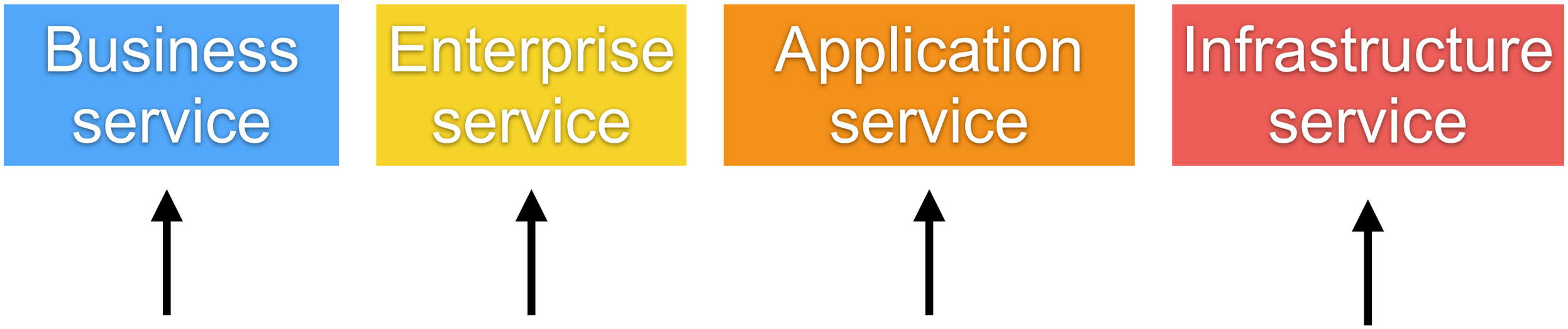
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Focus on services



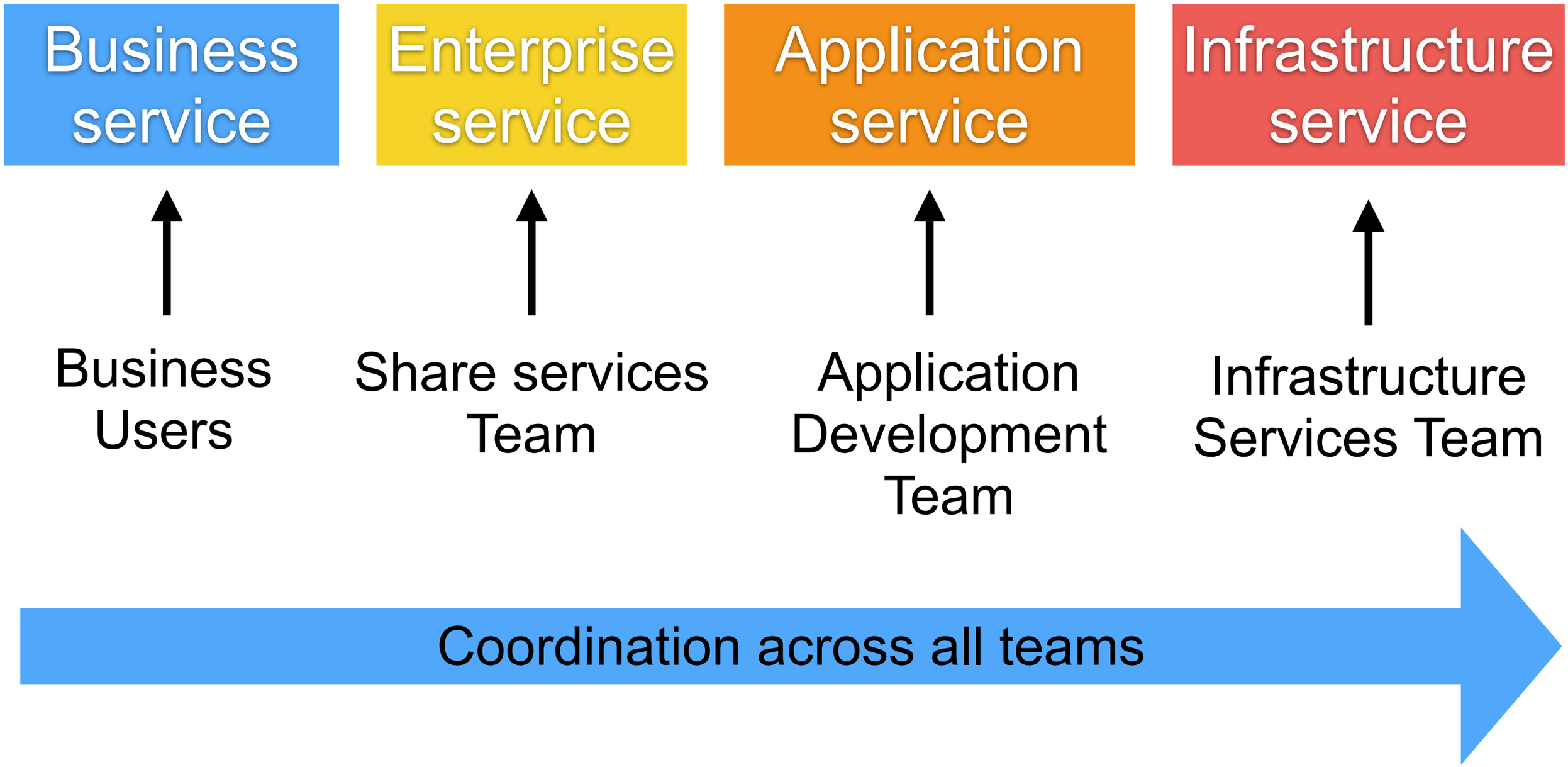
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture



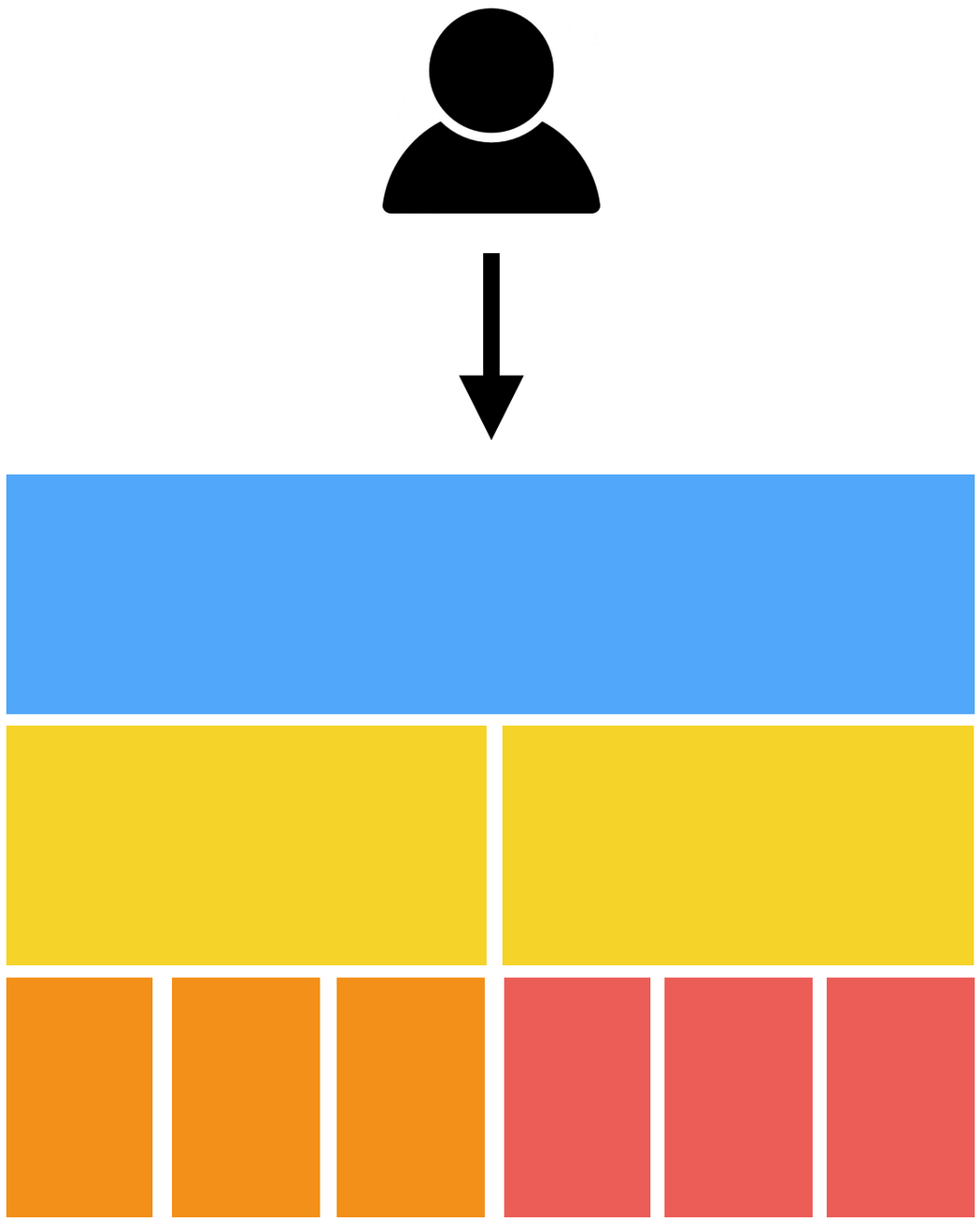
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture

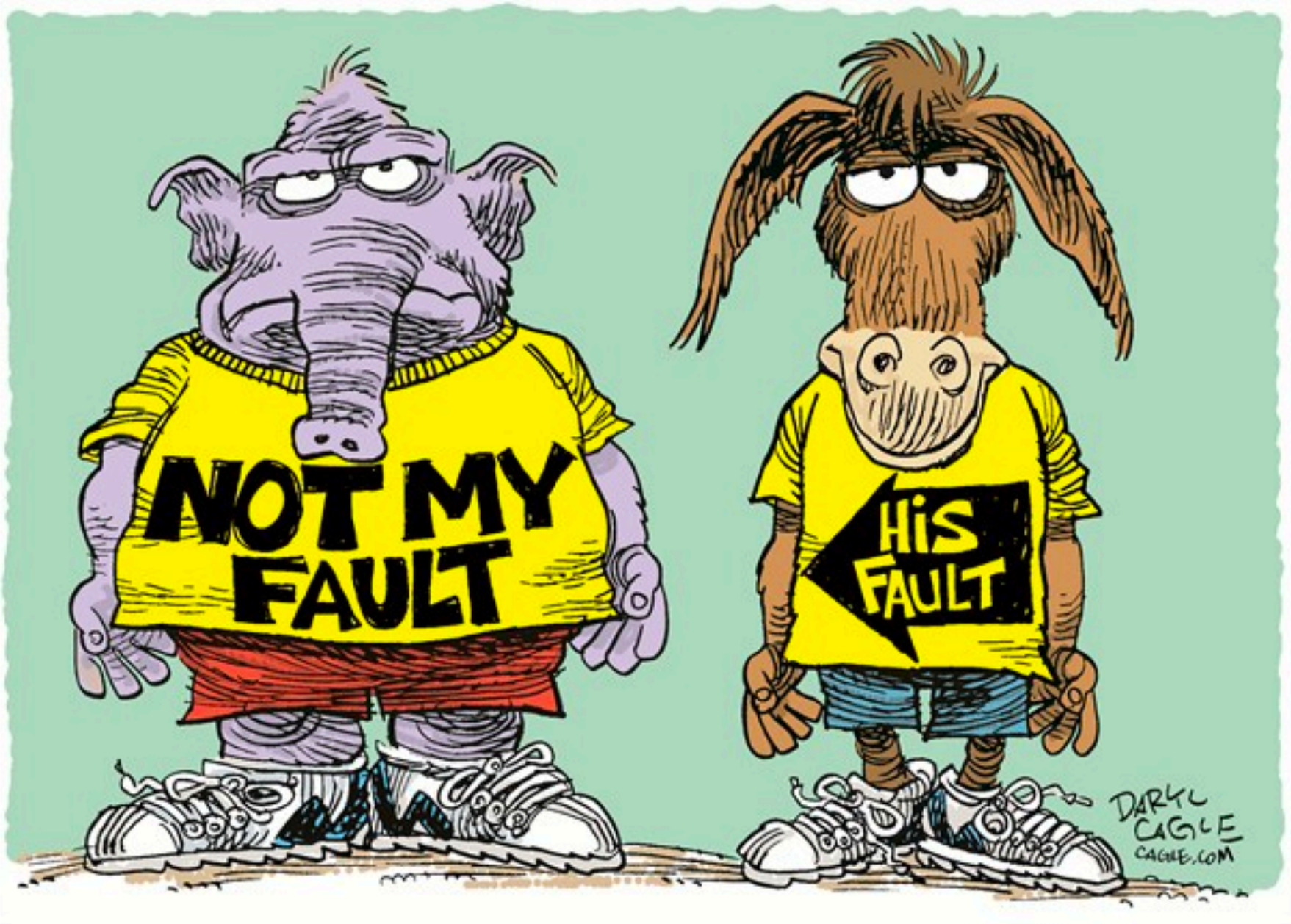


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service-Oriented Architecture !!



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

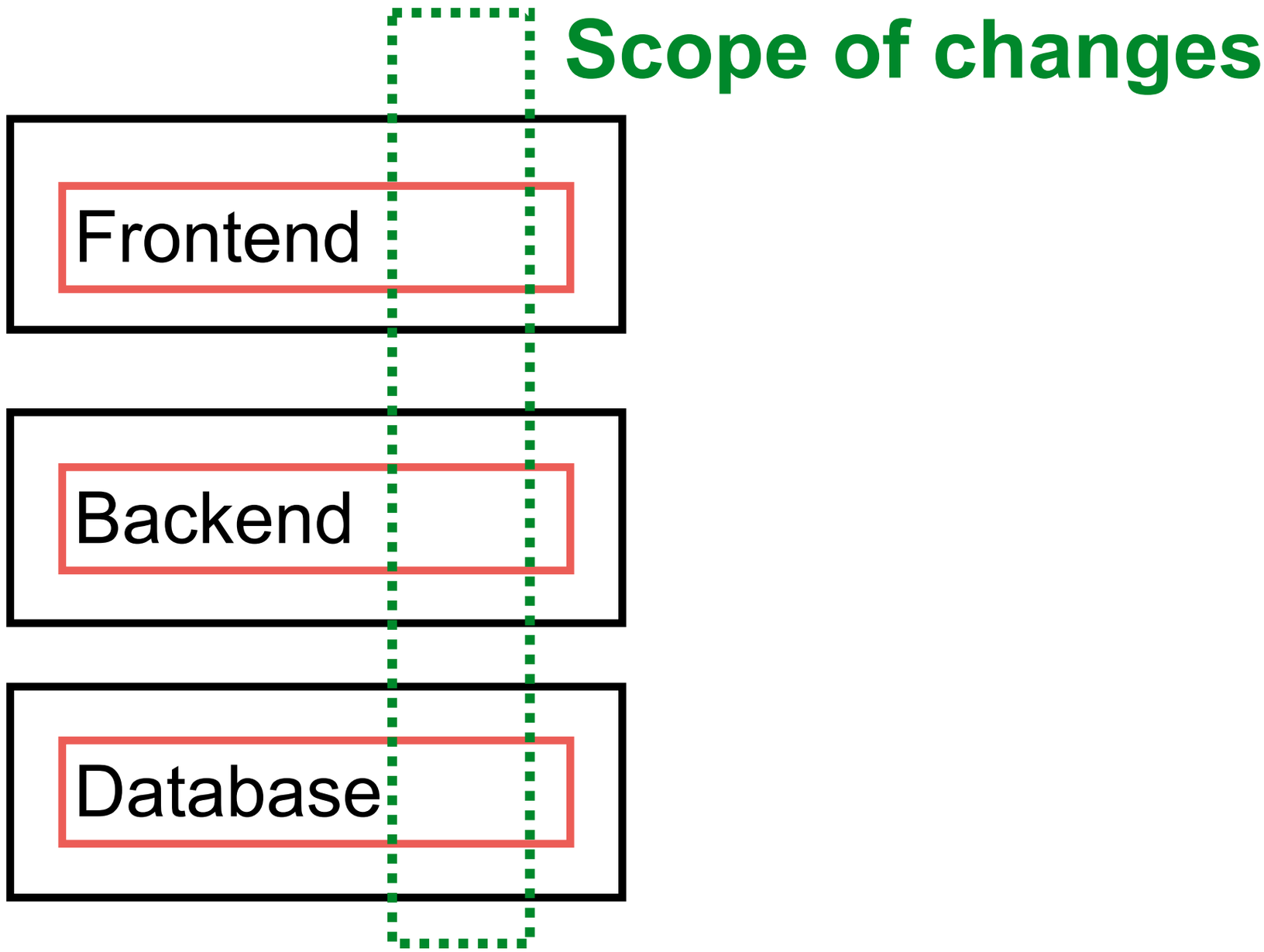


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice

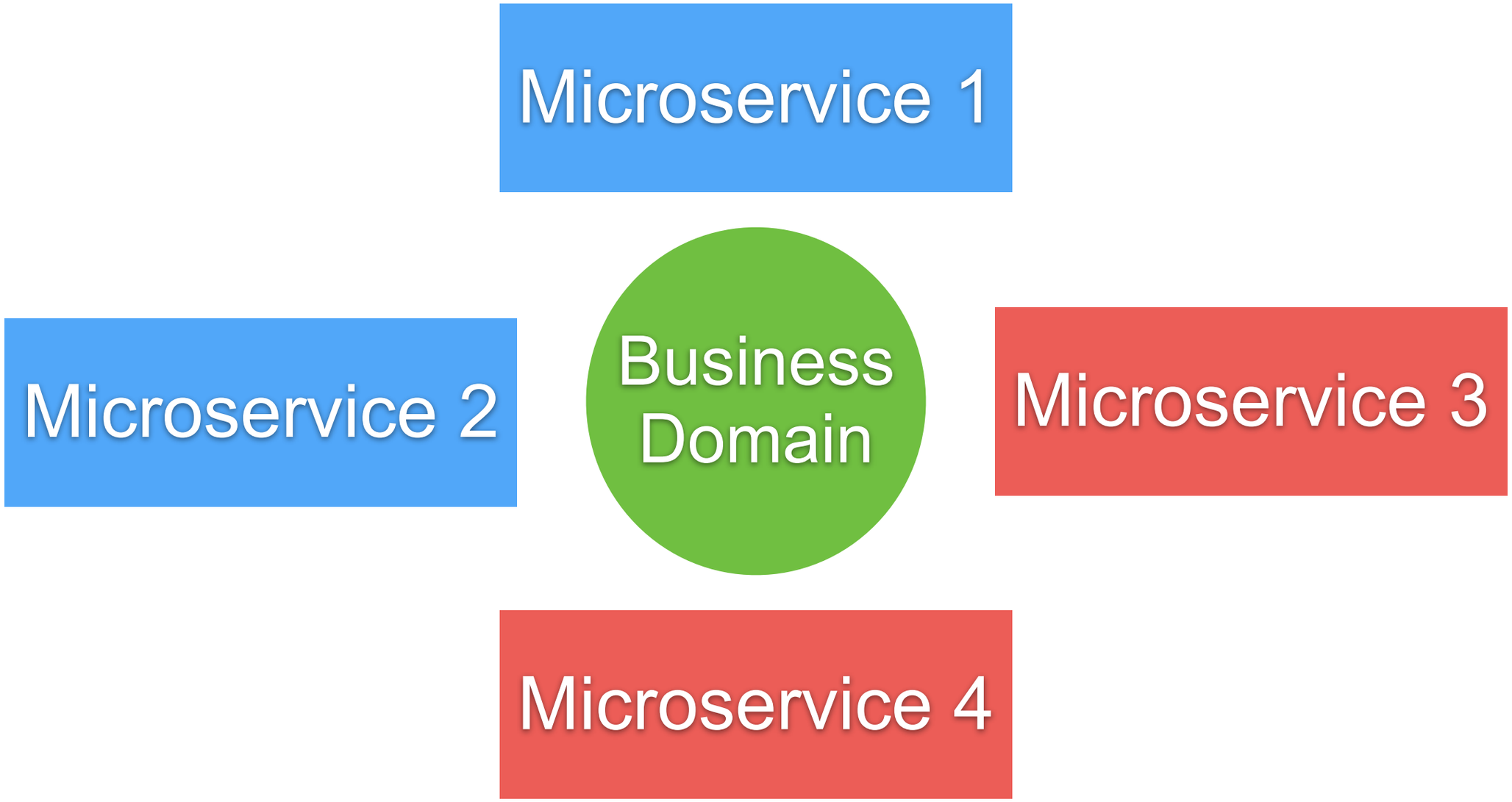
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Focus on all tiers/layers



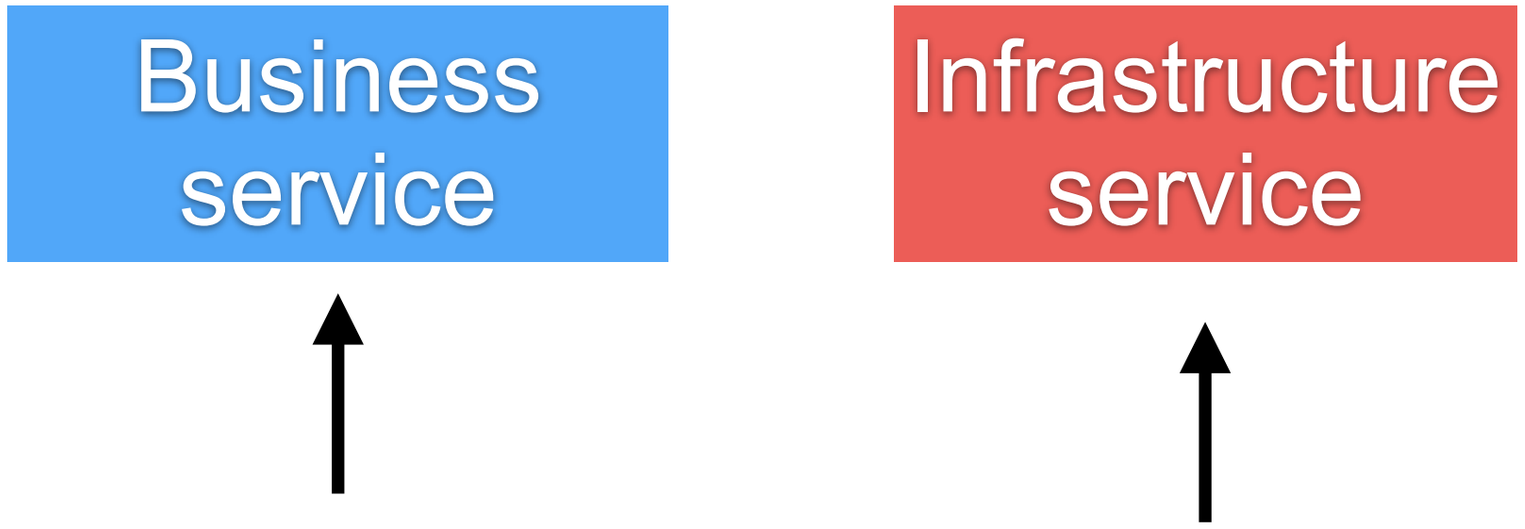
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice

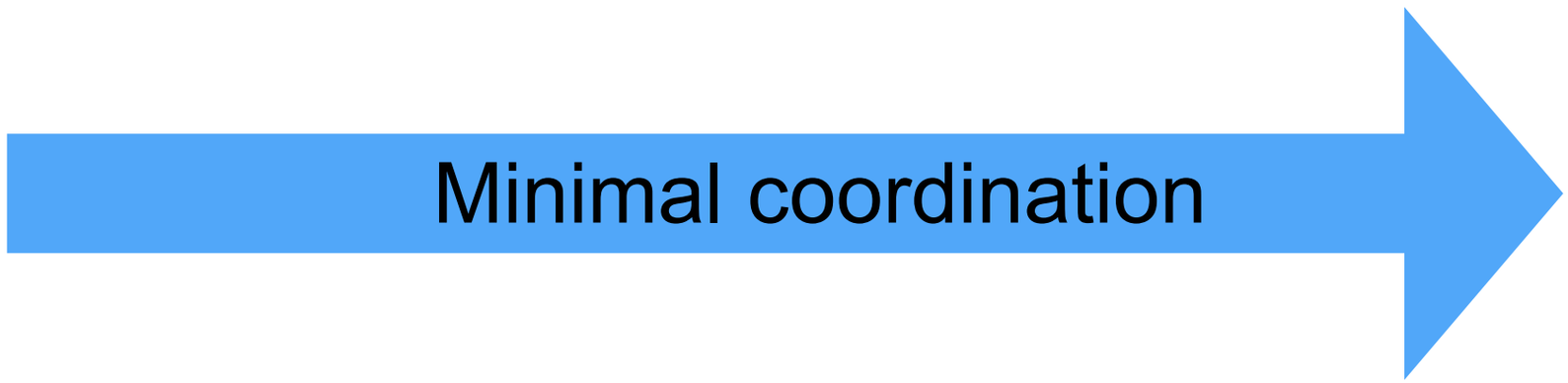


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice

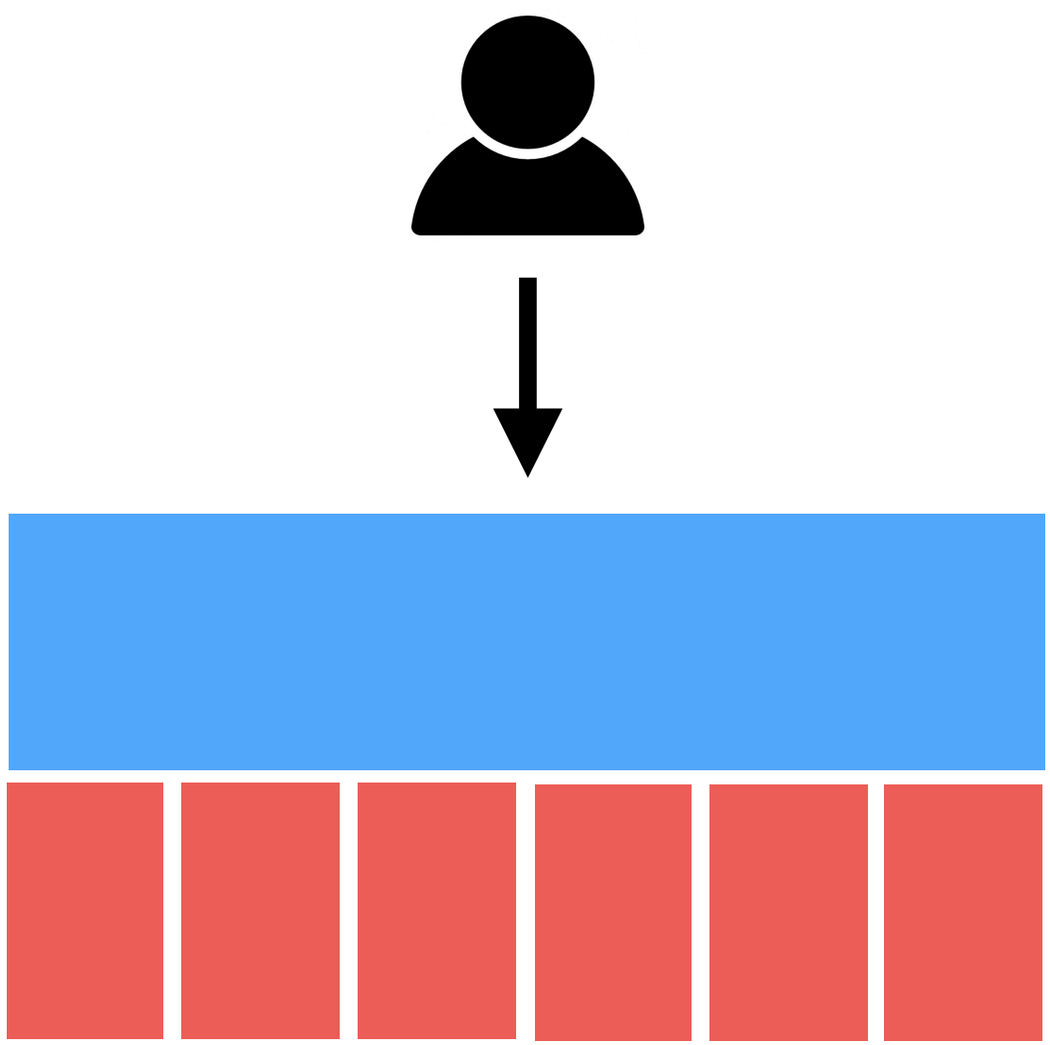


Application Development Team

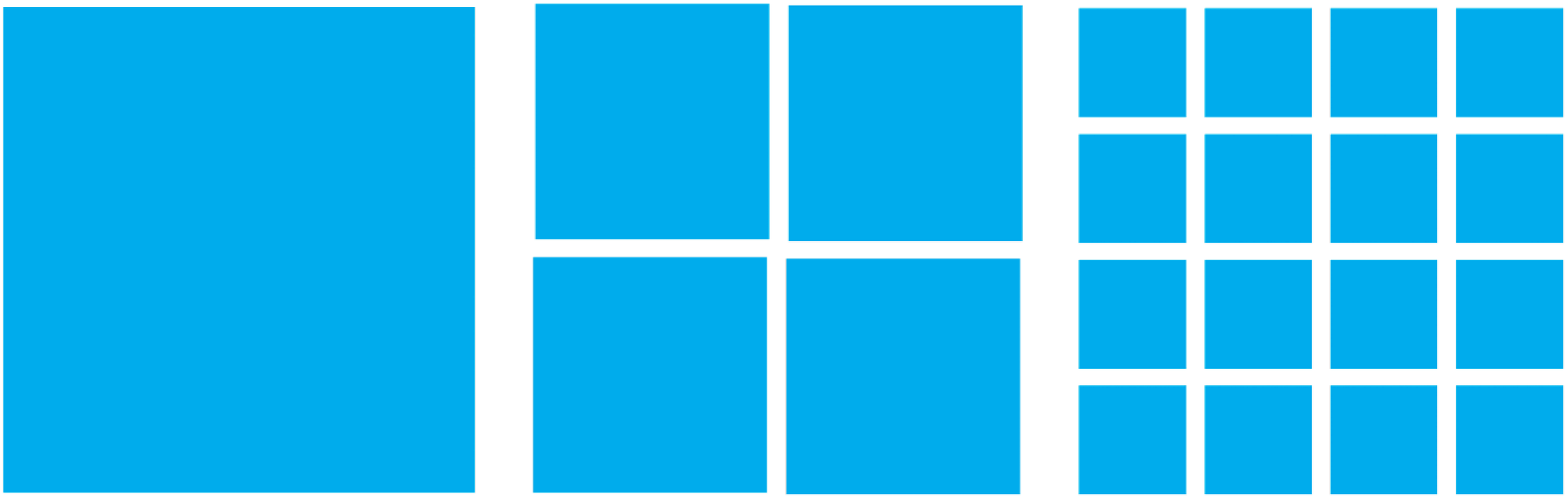


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



Monolithic

Service Oriented

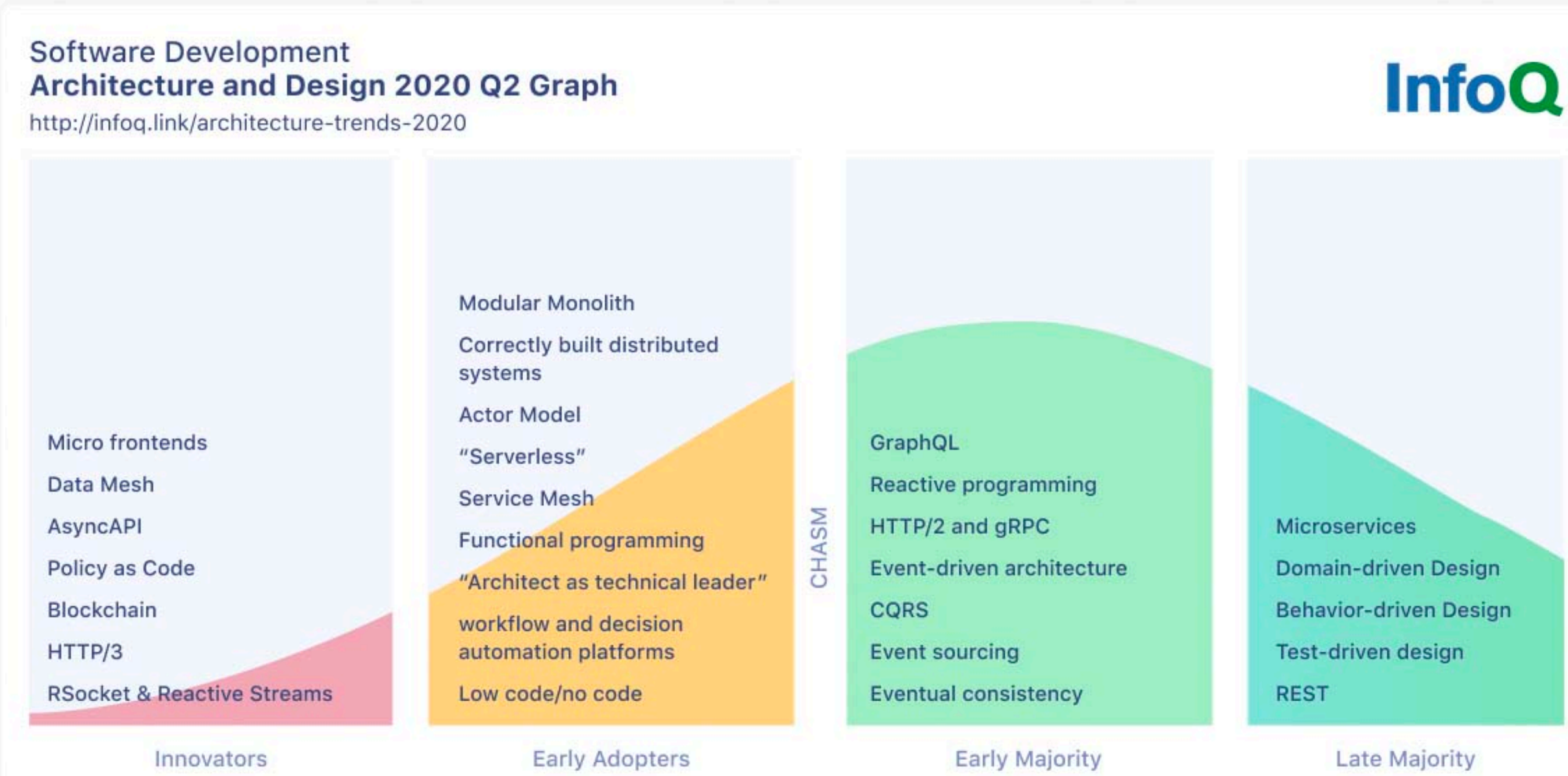
Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice history

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice 2020



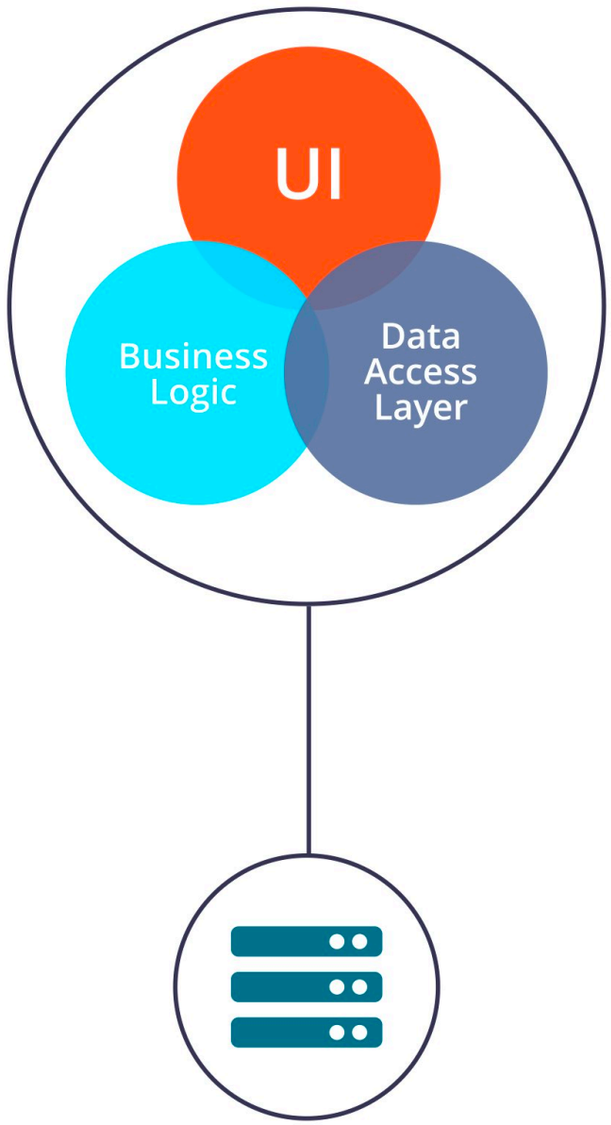
https://www.infoq.com/articles/architecture-trends-2020/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

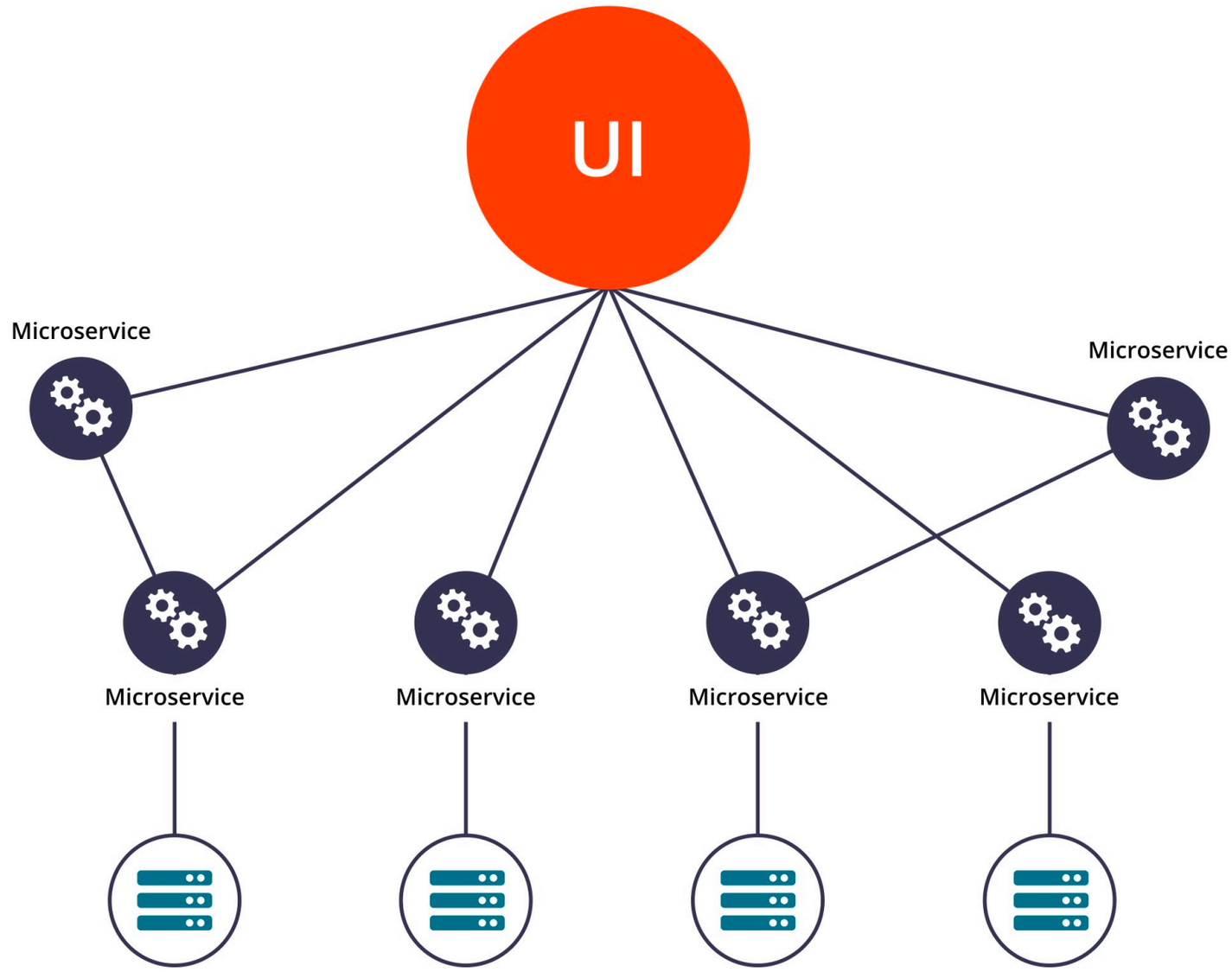
Microservice

Small, Do one thing (Single Responsibility)
Modular and **loosely coupled**
Easy to **understand**
Easy to **develop** by small team
Deploy independently
Focus on **business** first
Scale independently

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



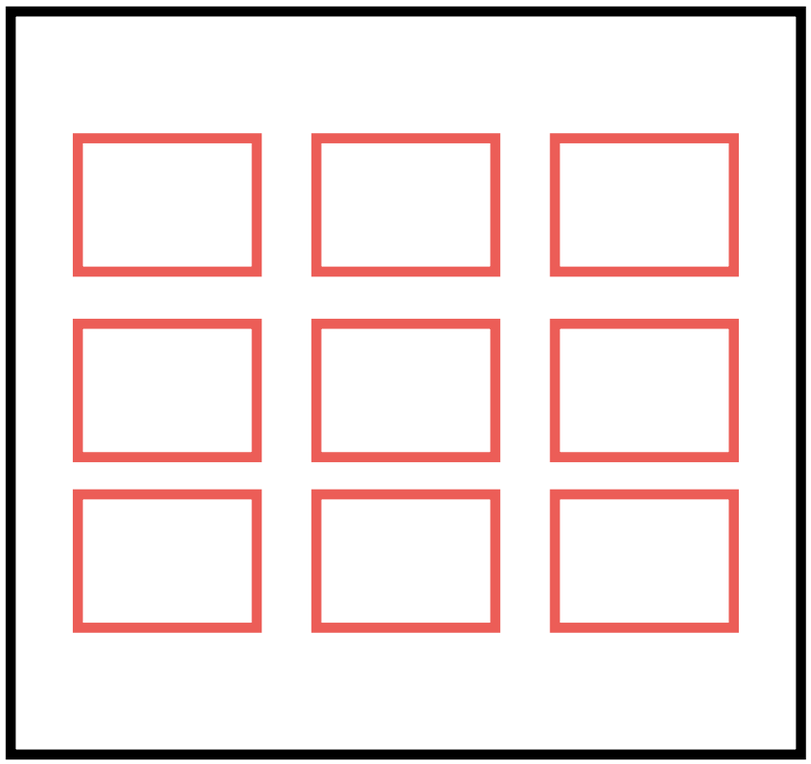
Monolithic Architecture



Microservice Architecture

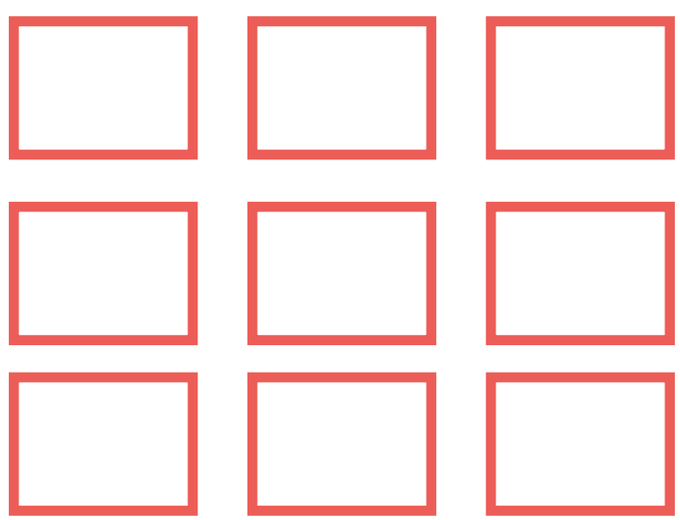
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice



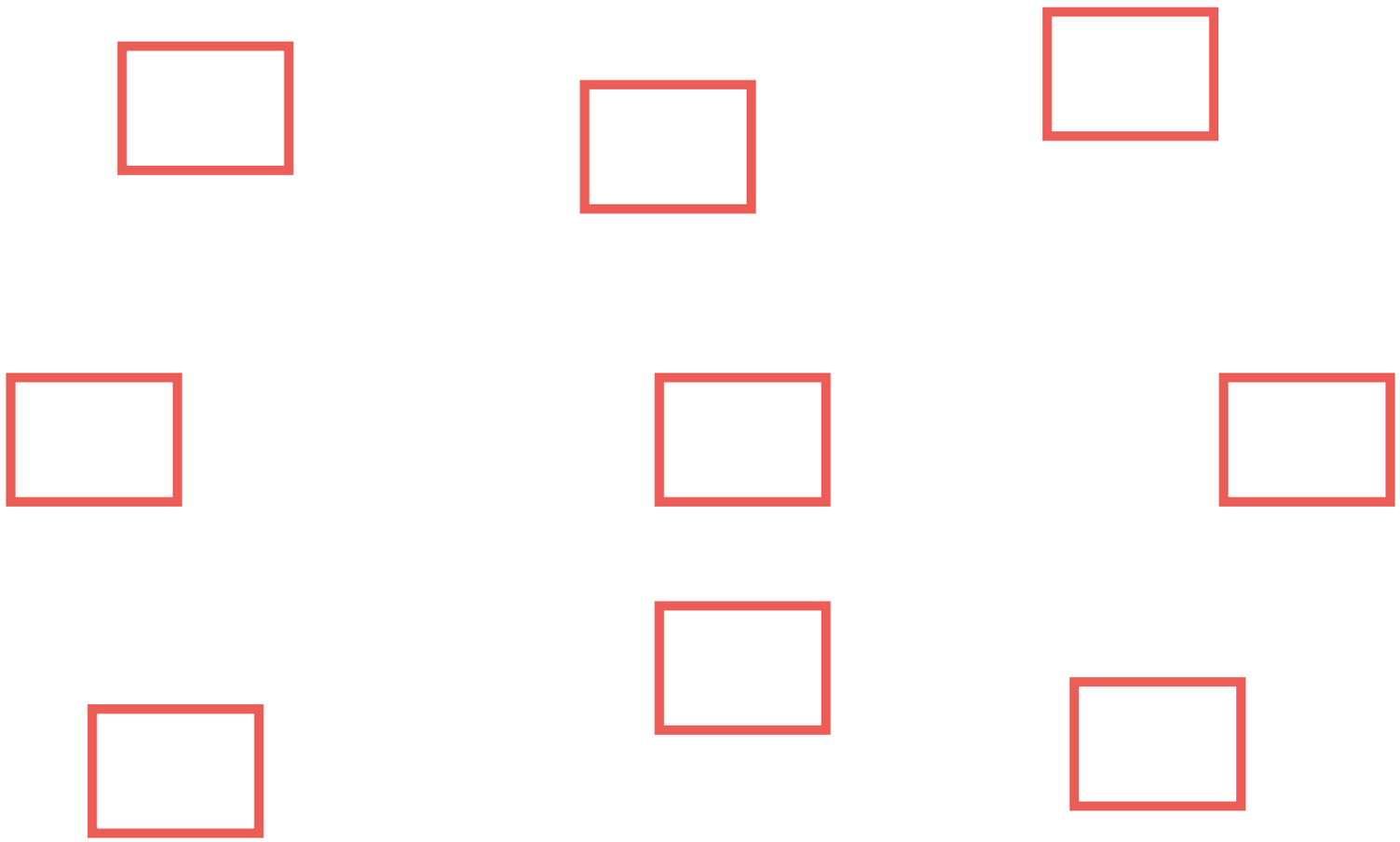
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice



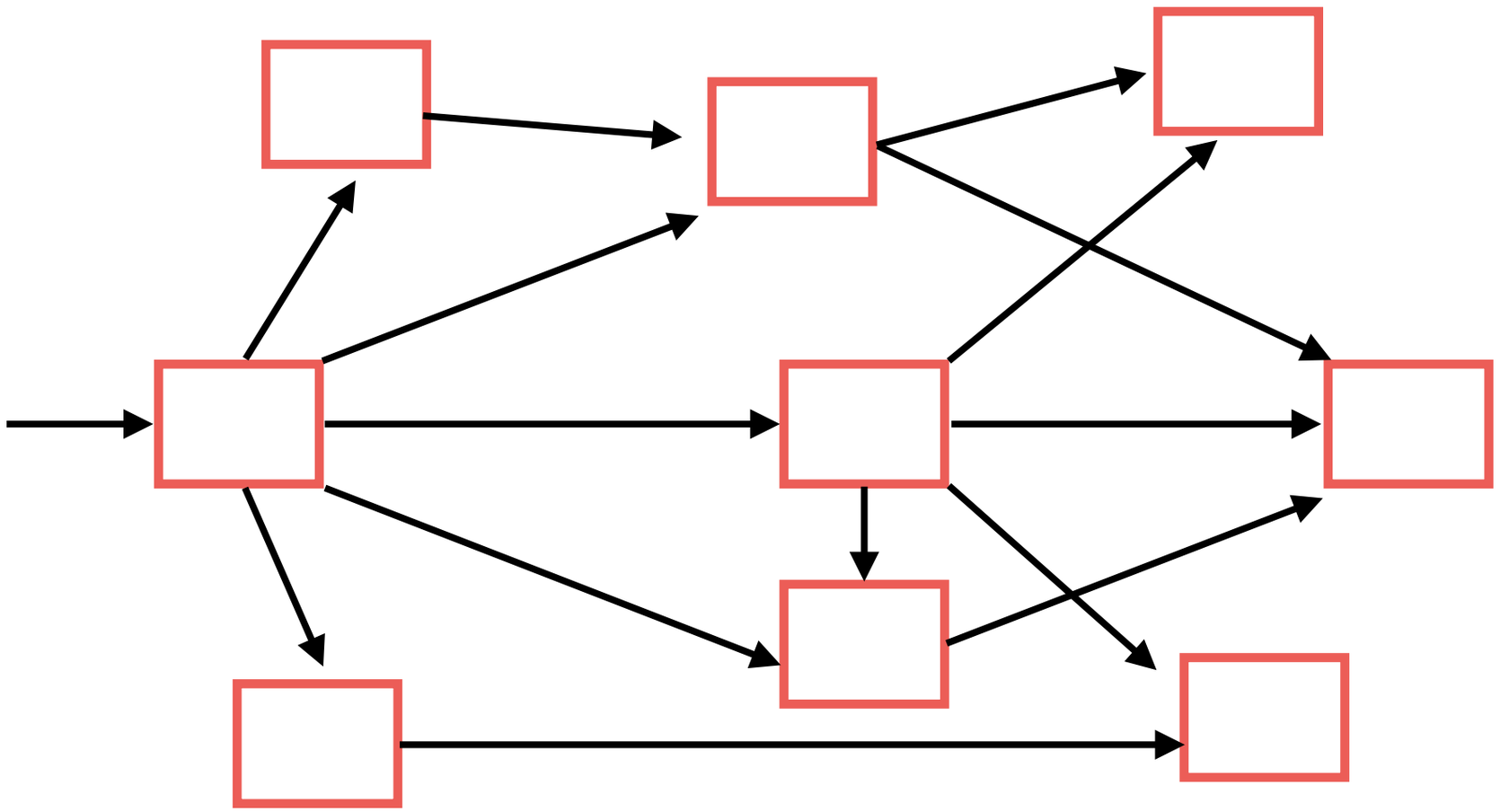
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice



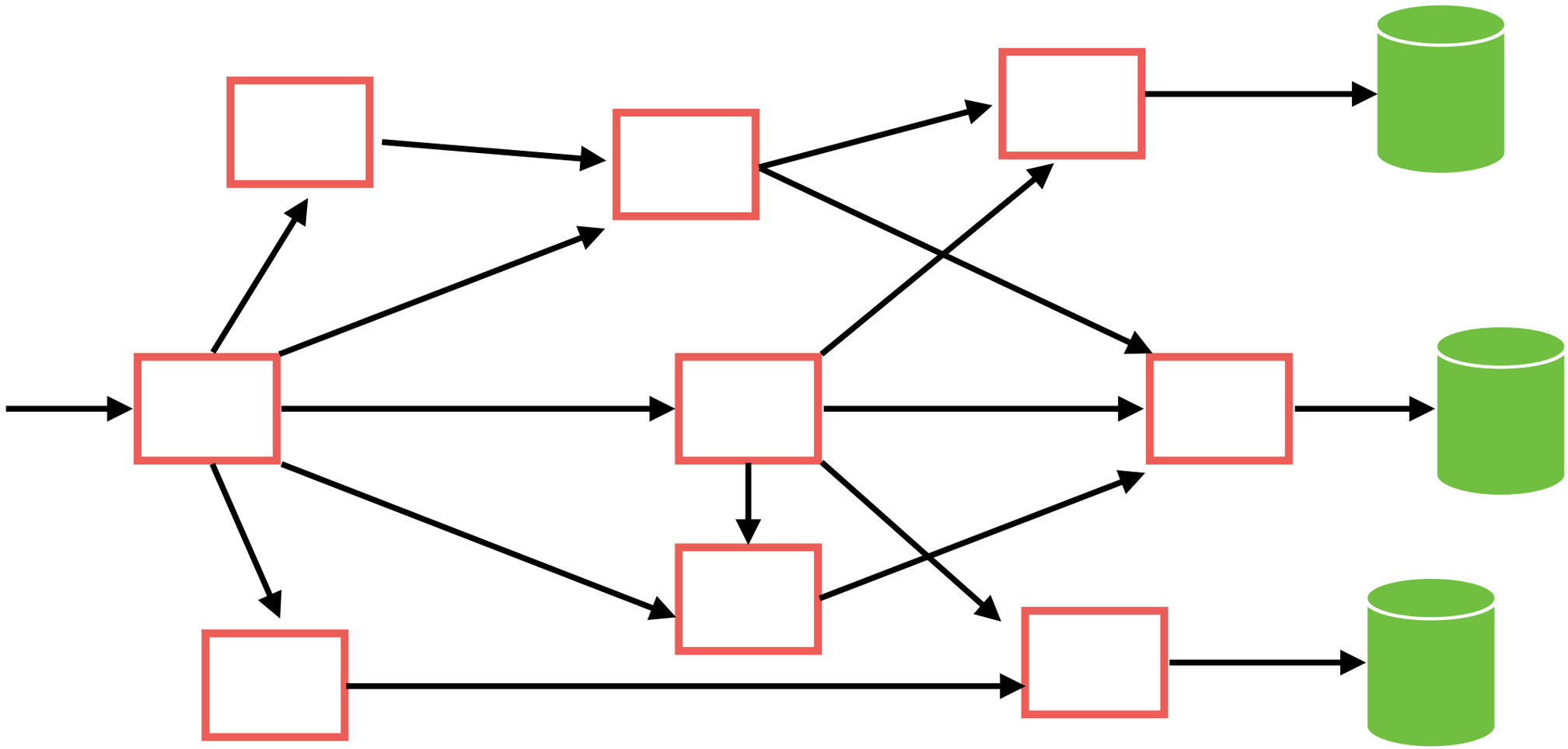
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice == Distributed



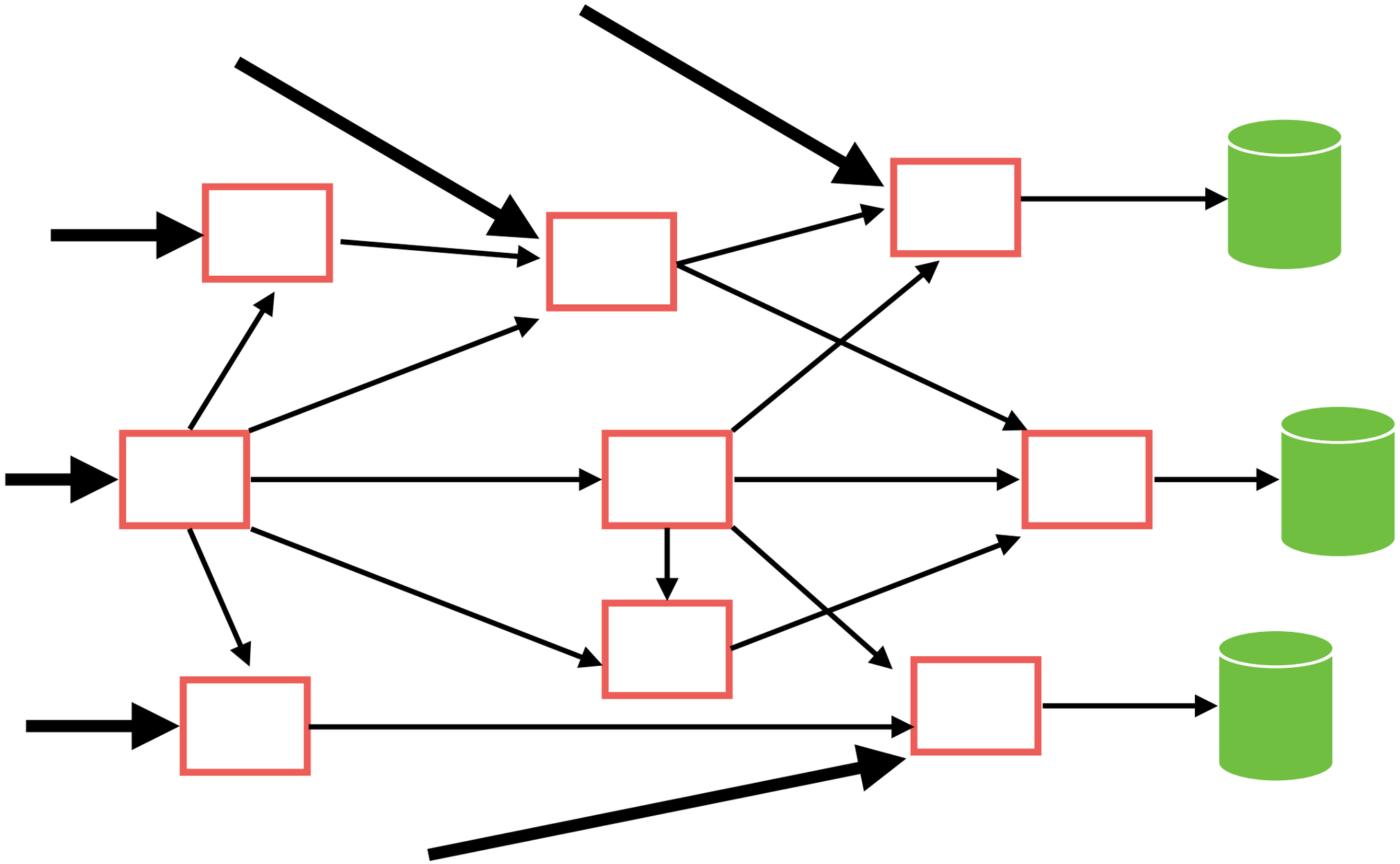
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Own data



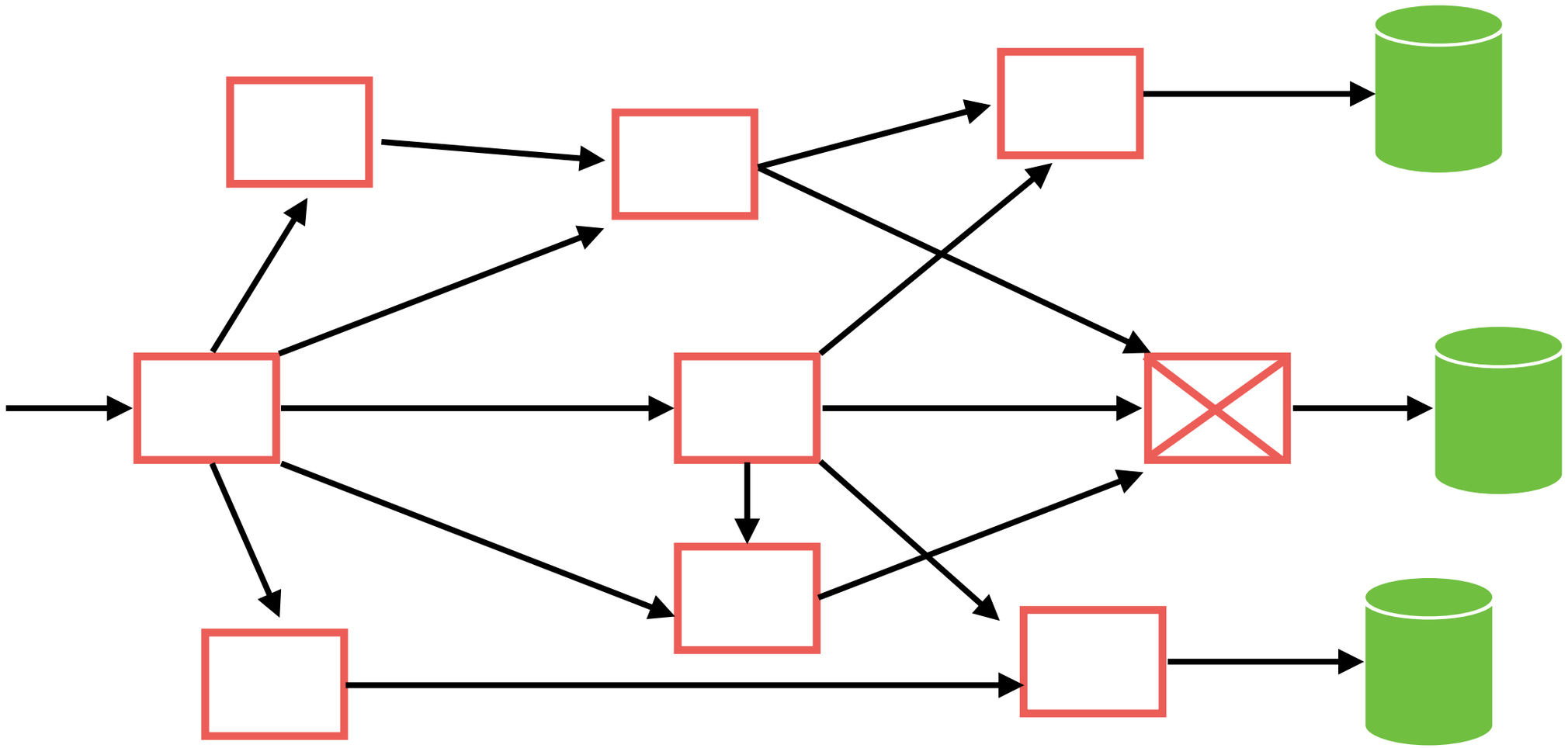
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Multiple entry points



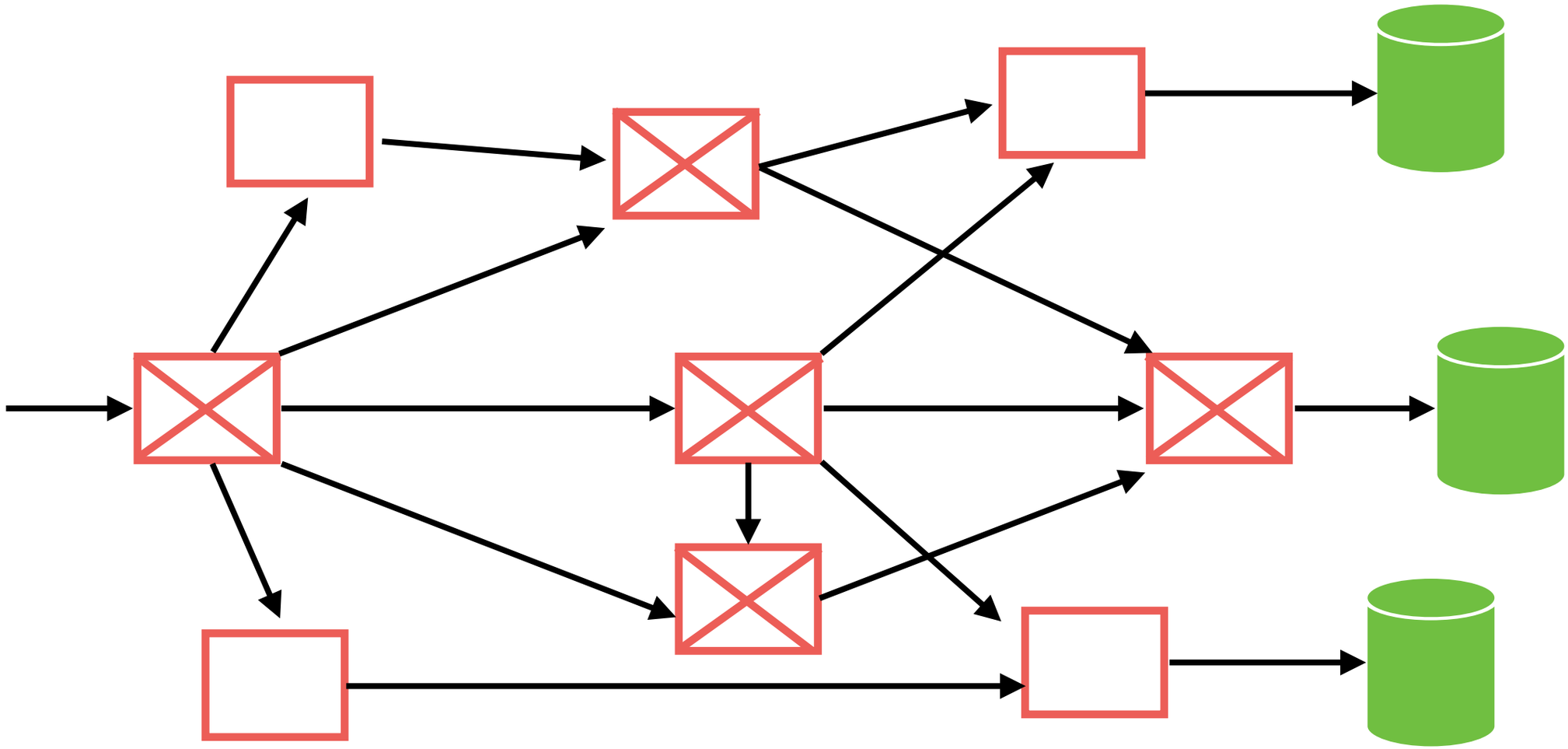
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Failure !!



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Cascading Failure !!



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Don’t design for failure

Design for recovery

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

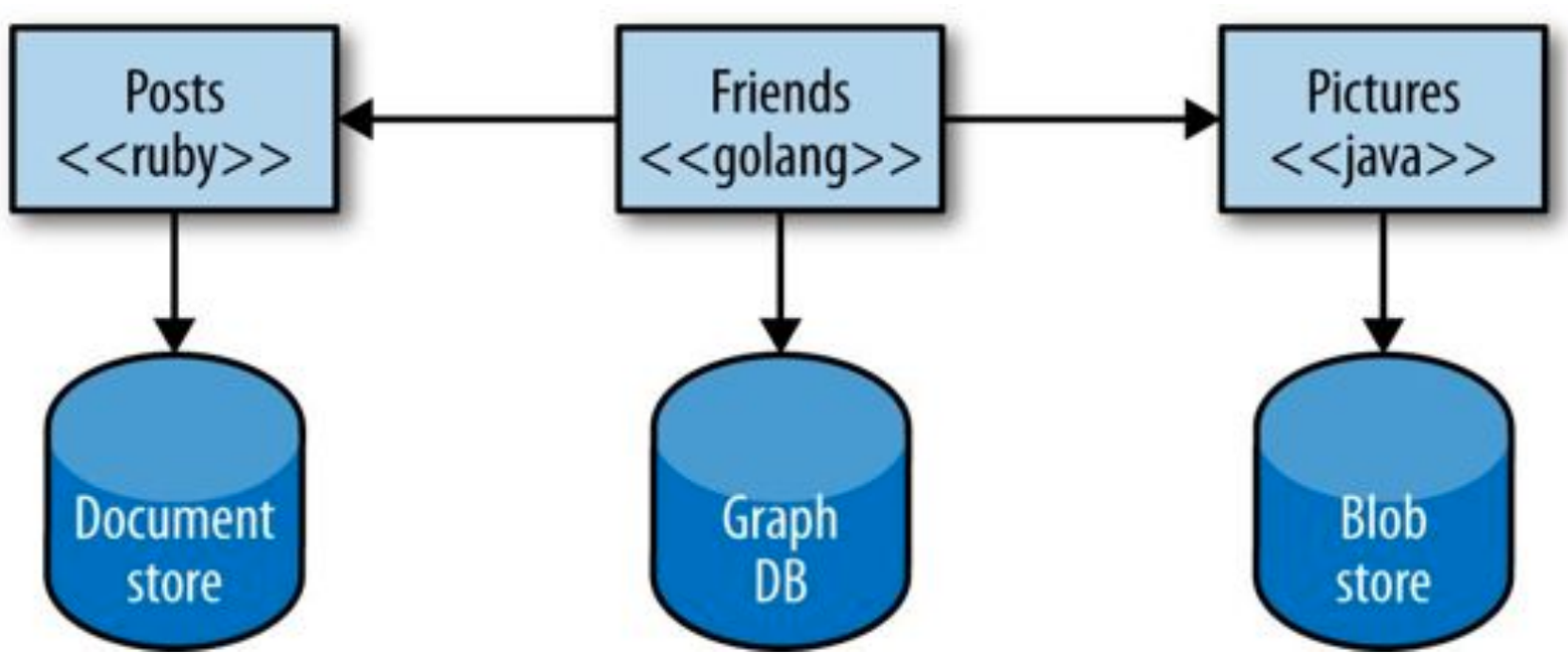
Key Benefits

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

1. Technology heterogeneous

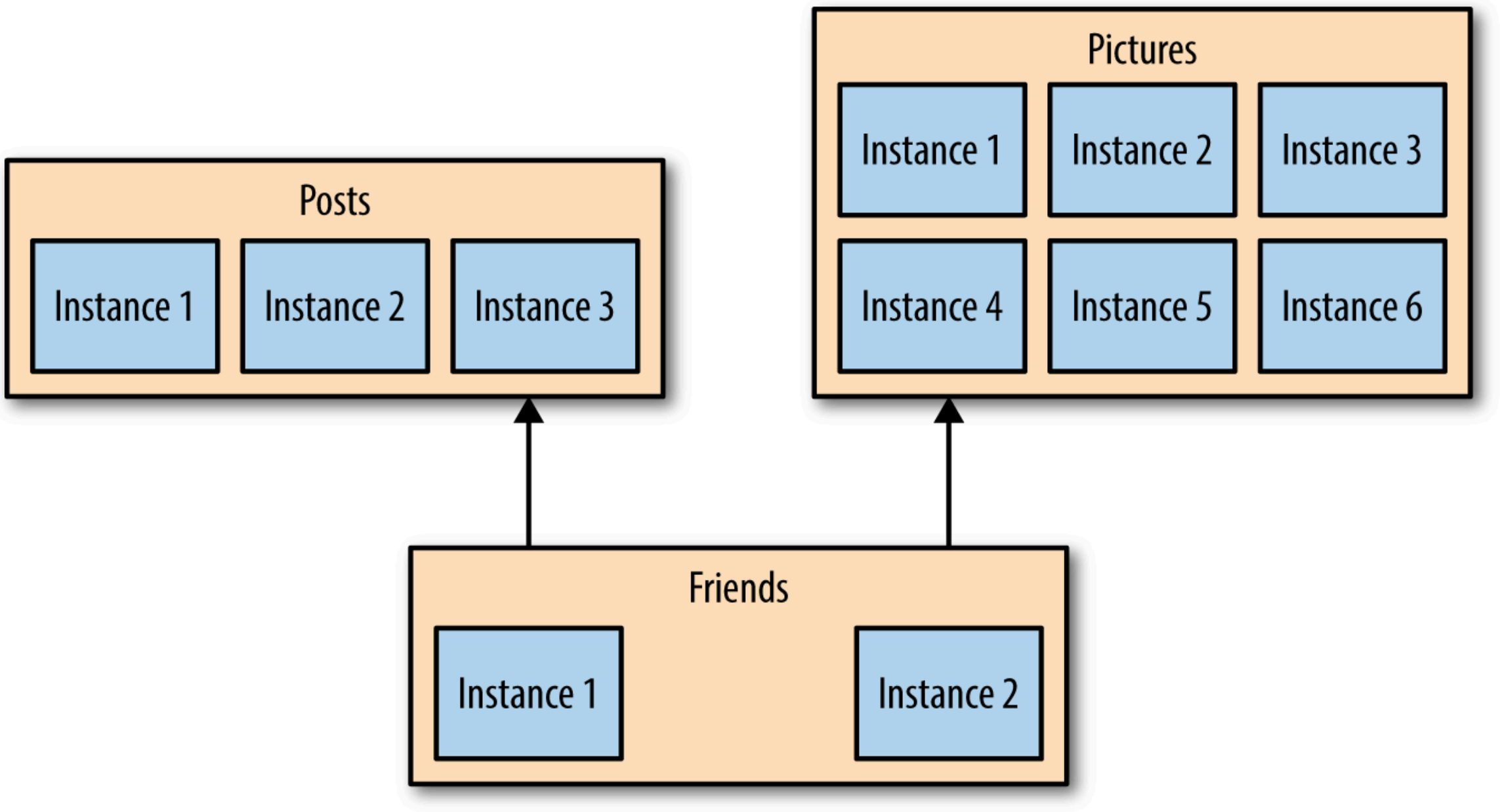
The right tool for each job

Allow easy experimenting and adoption of new technology



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

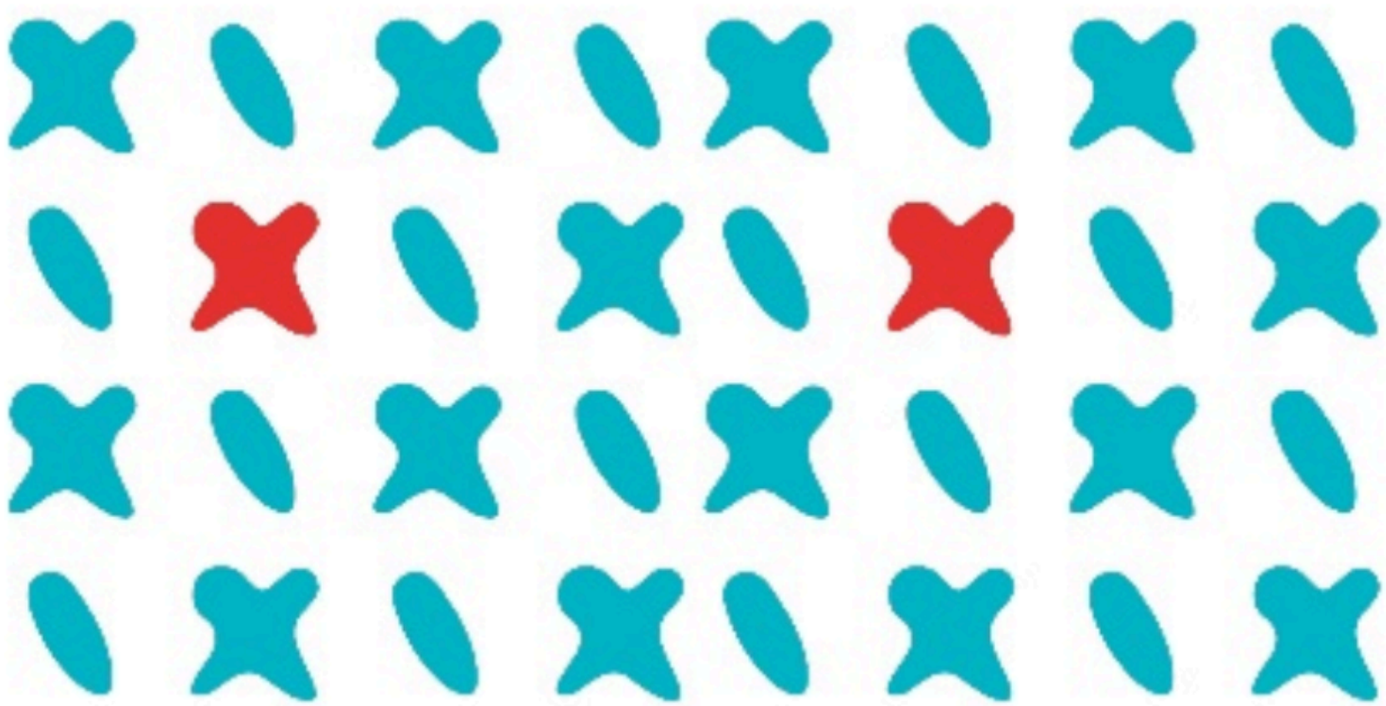
2. Scaling



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

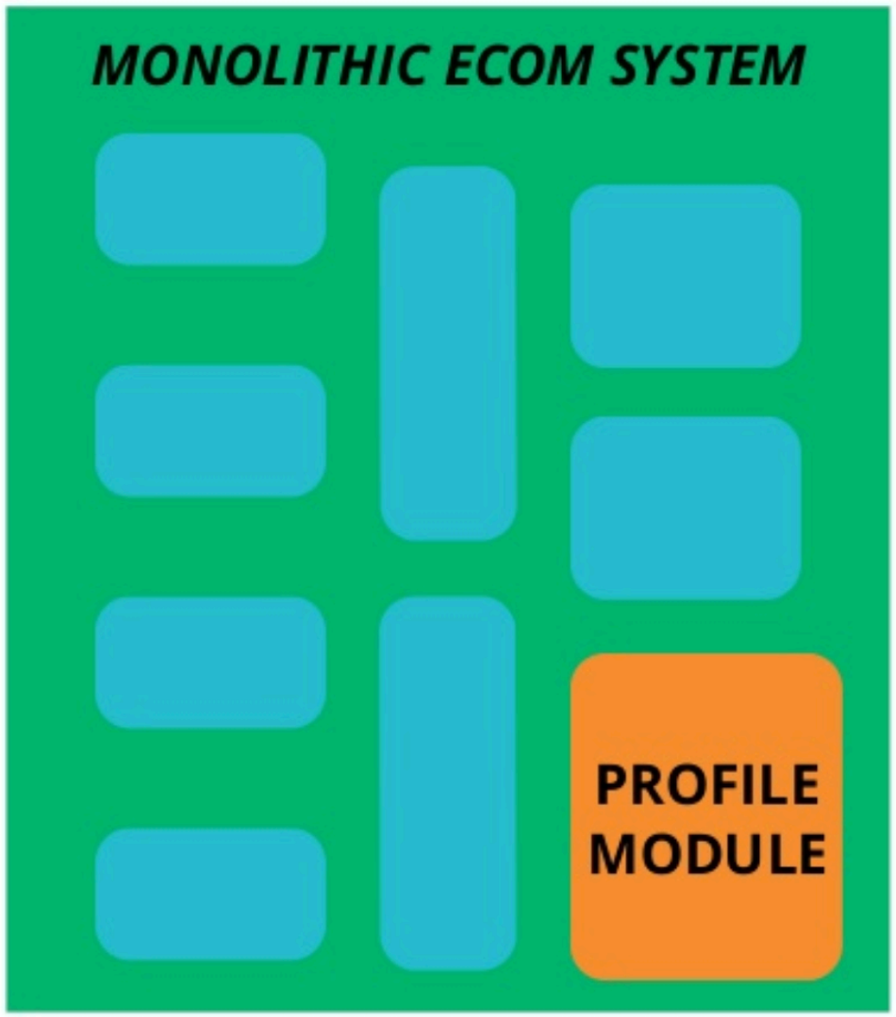
3. Ease of deployment

Deploys are faster, independent and problems can be isolated more easily



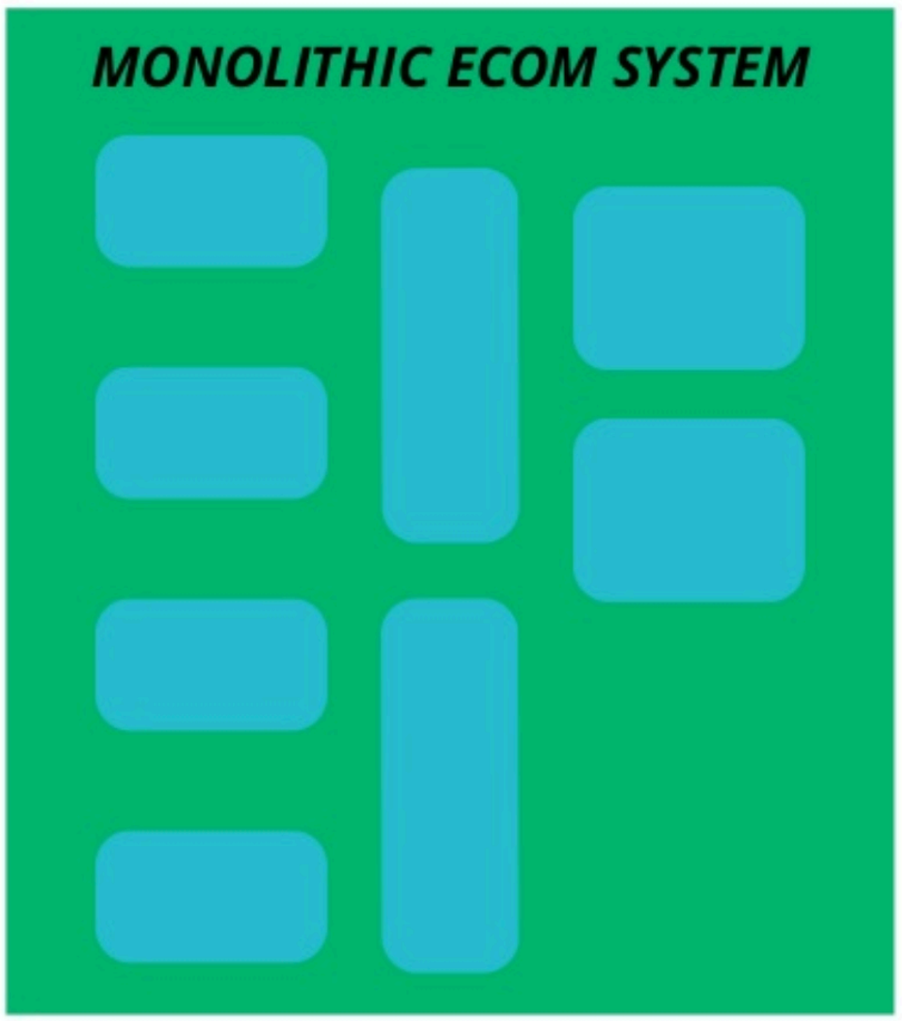
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4. Composability and replaceability



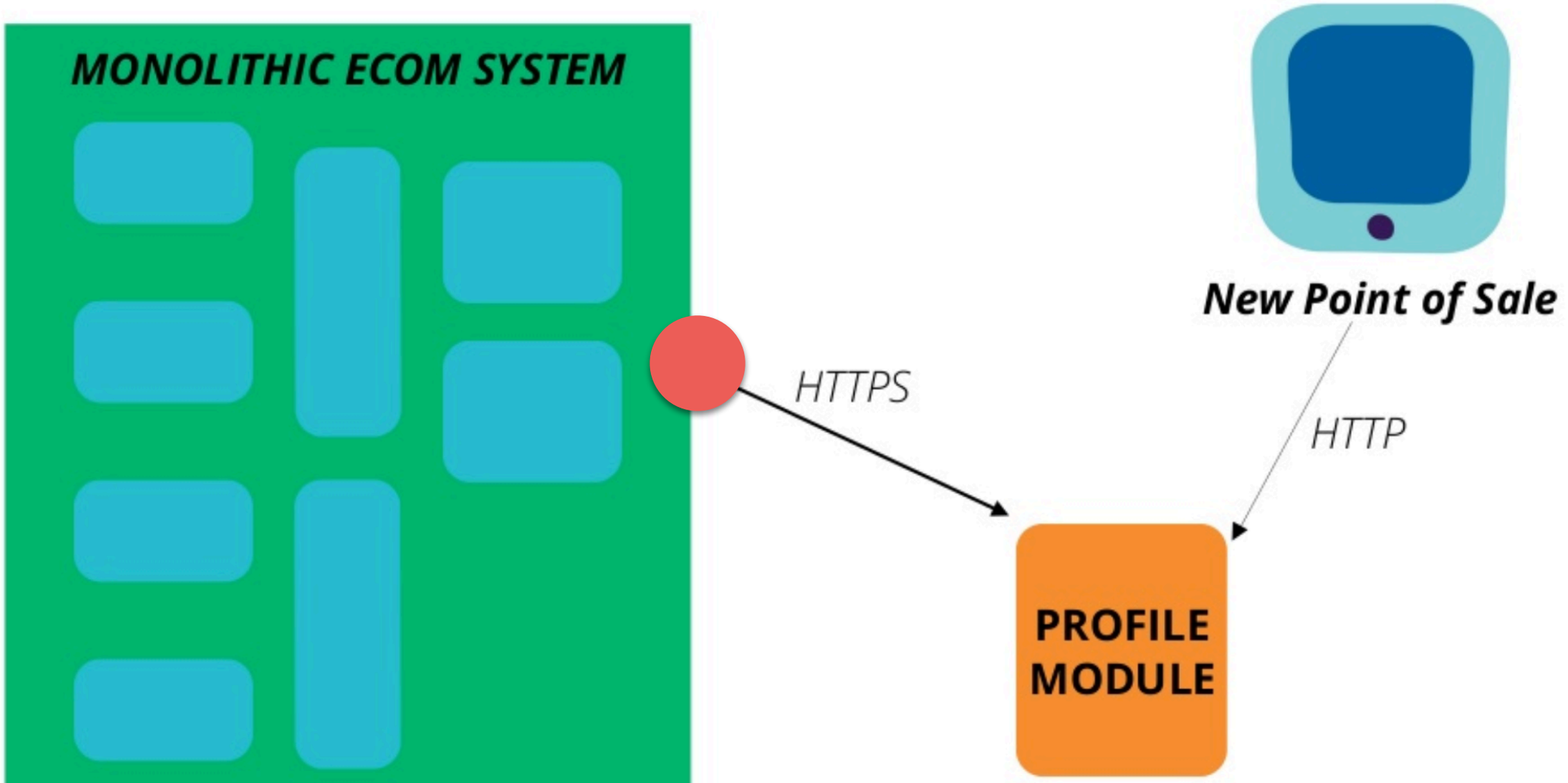
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4. Composability and replaceability



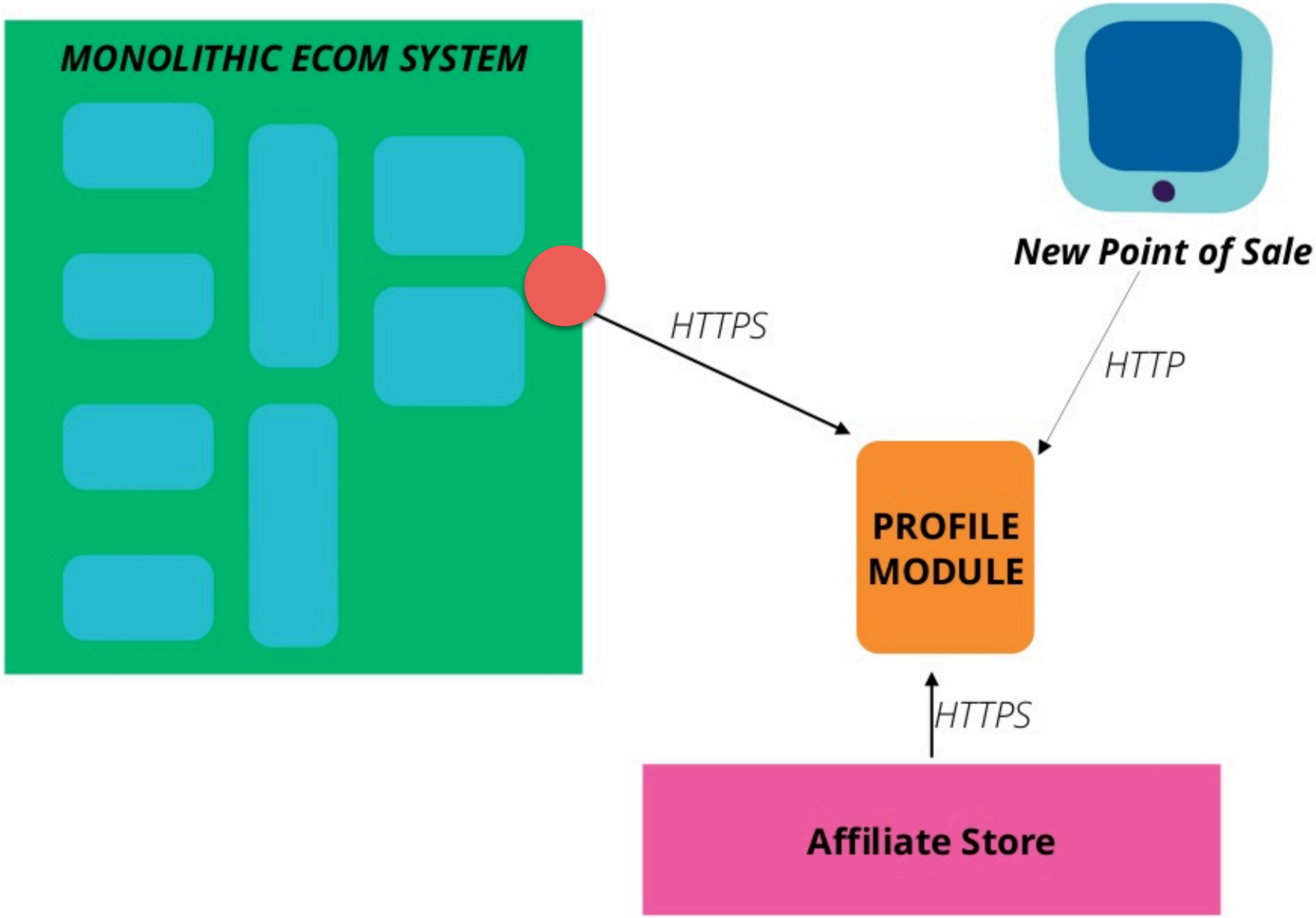
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4. Composability and replaceability



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4. Composability and replaceability



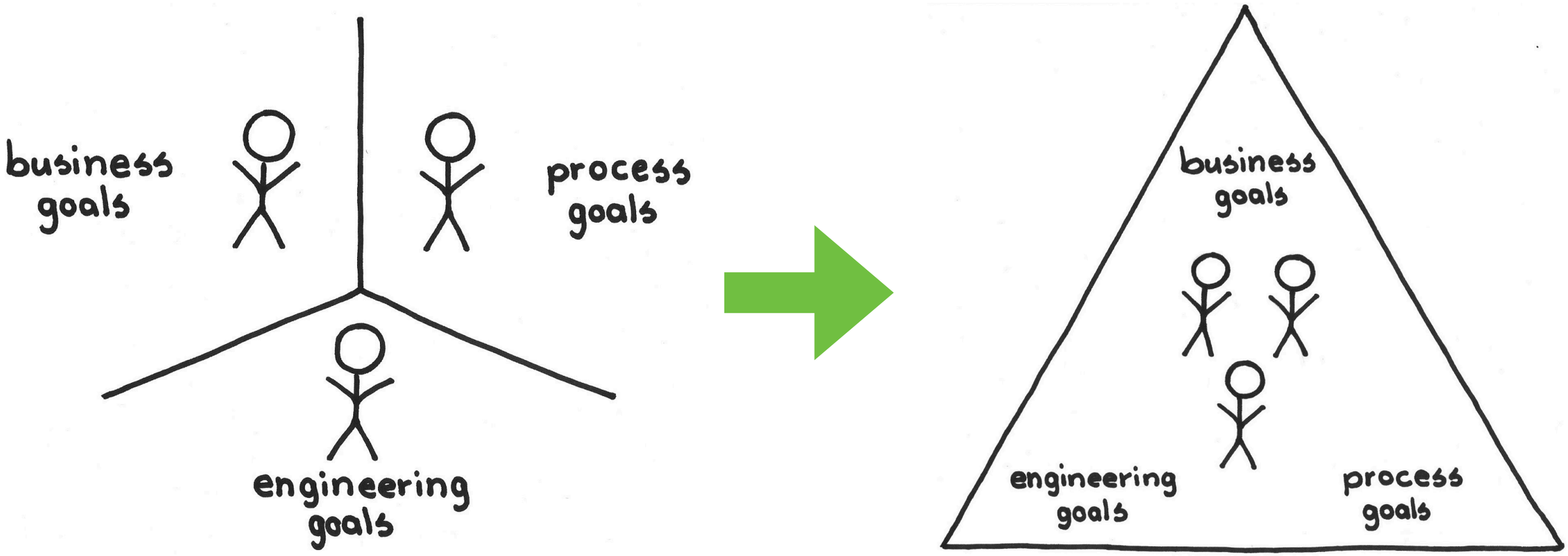
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

5. Organization alignment

Small teams and smaller codebases

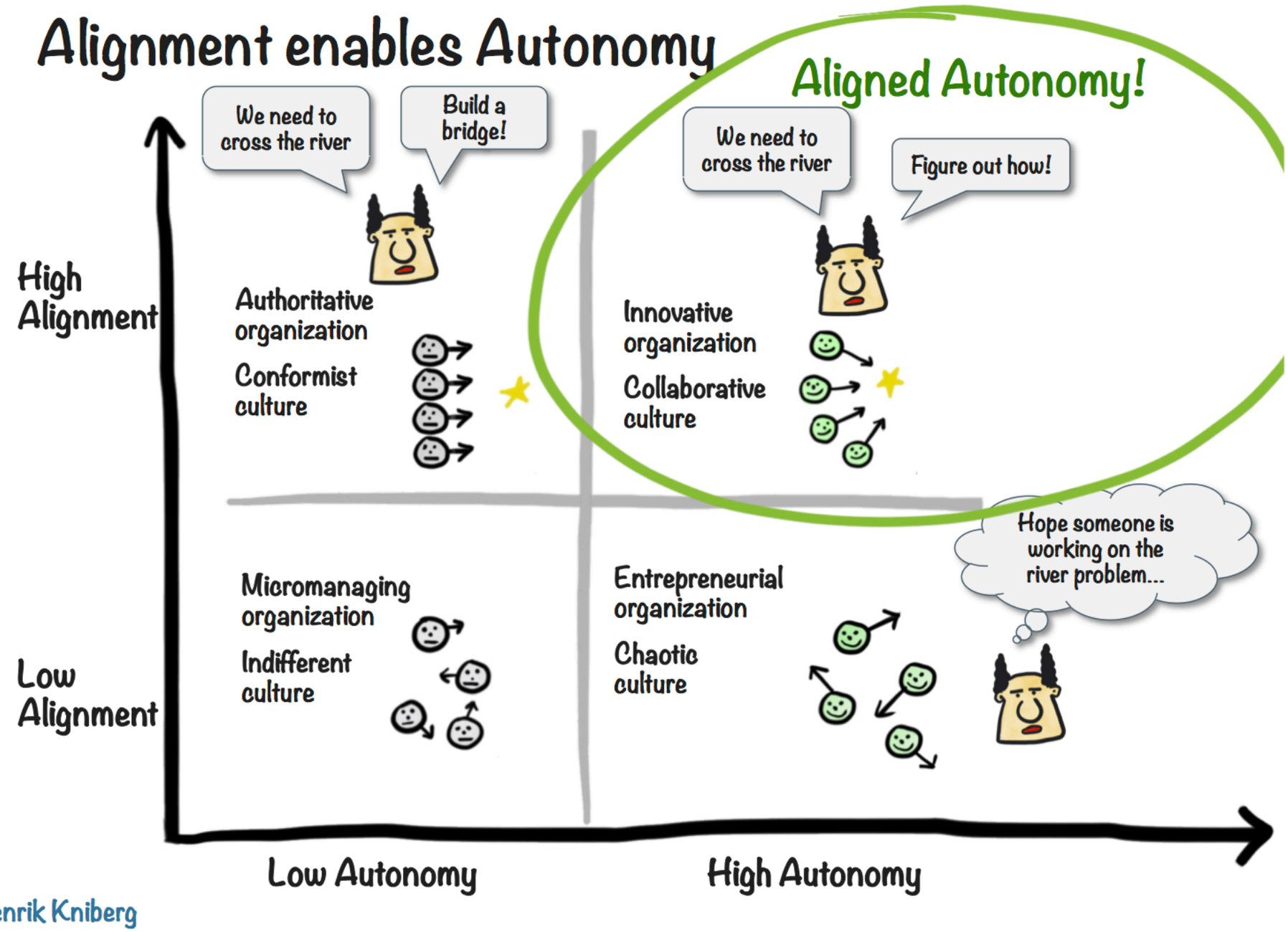
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Autonomous team



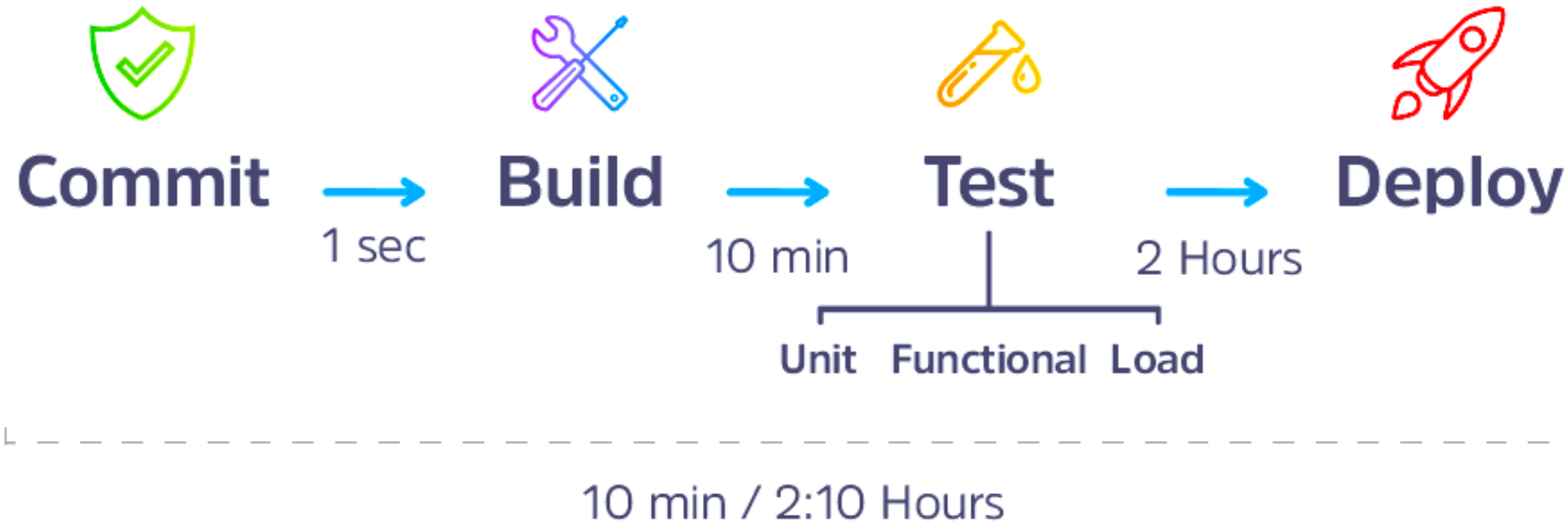
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Autonomous team



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Autonomous team

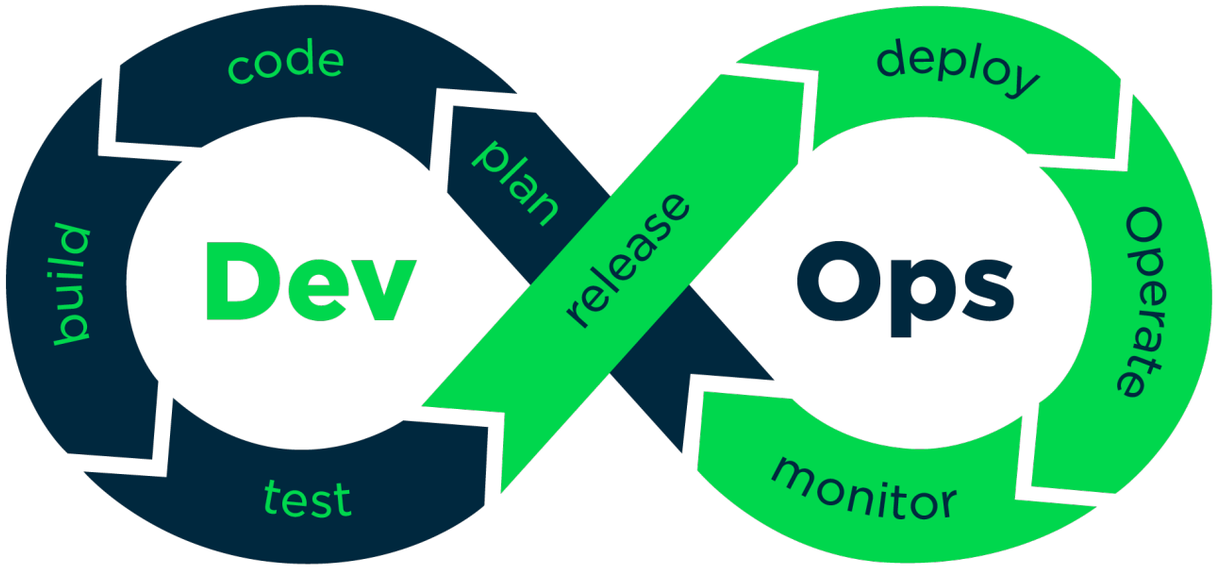


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

6. Enable Continuous Delivery

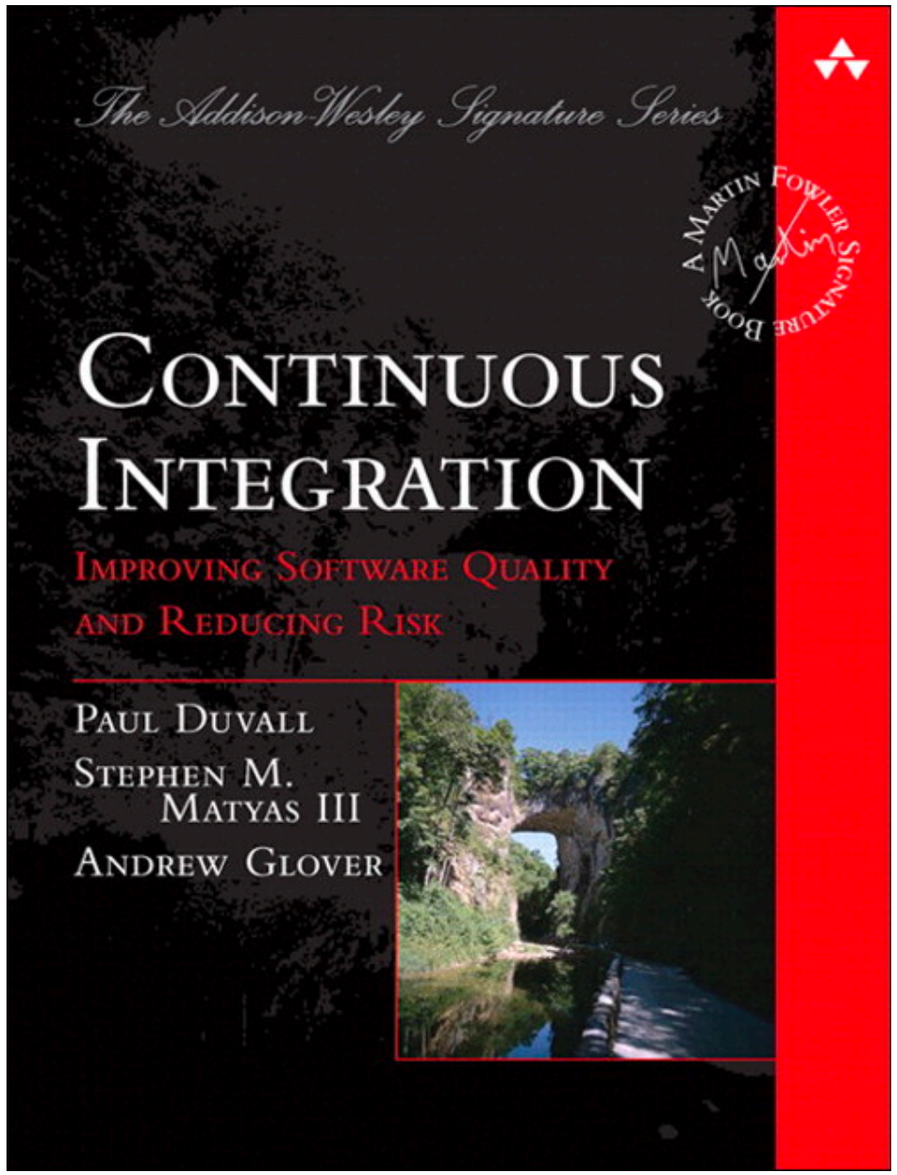
A part of **DevOps**

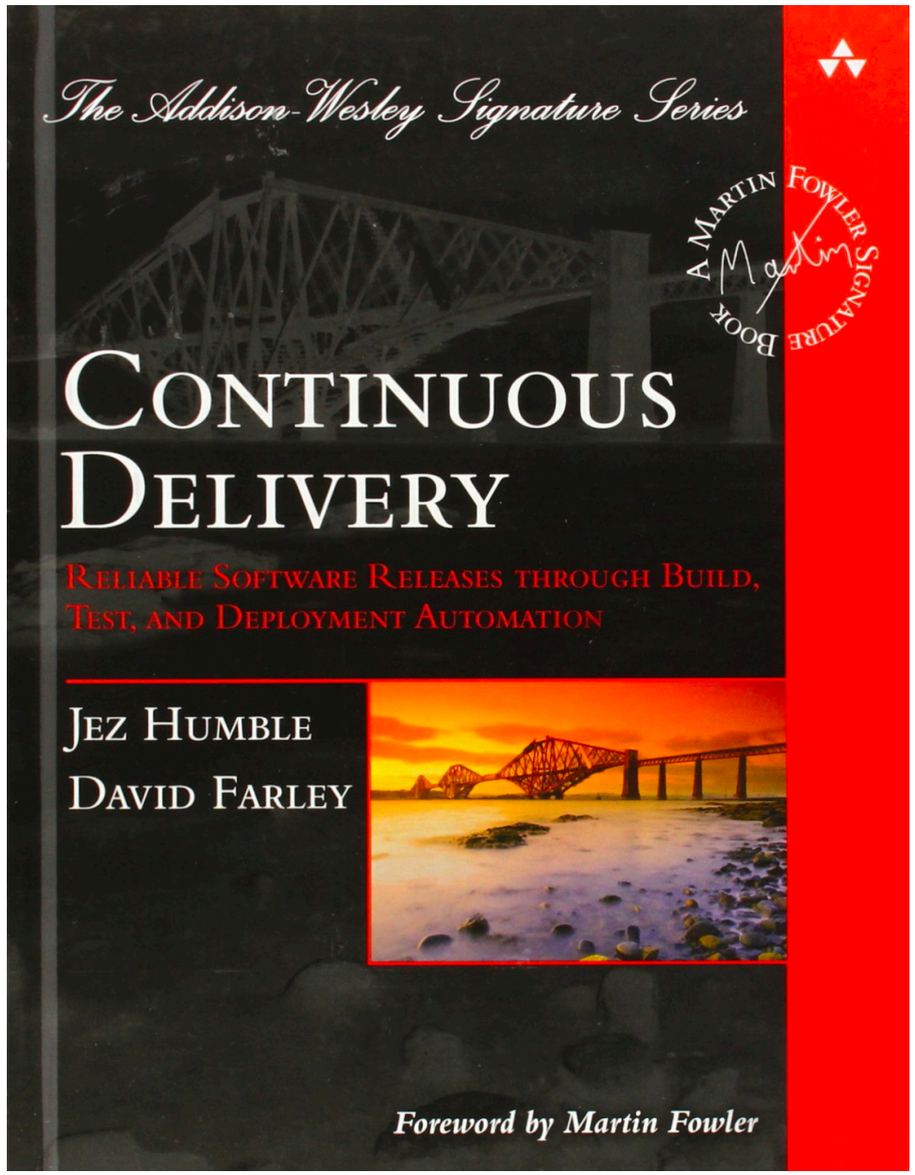
Set of practices for the **rapid**, **frequent** and **reliable delivery** software



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Continuous Delivery/Deployment





© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Continuous Delivery/Deployment

Testability
Deployability
Autonomous team and loose coupling

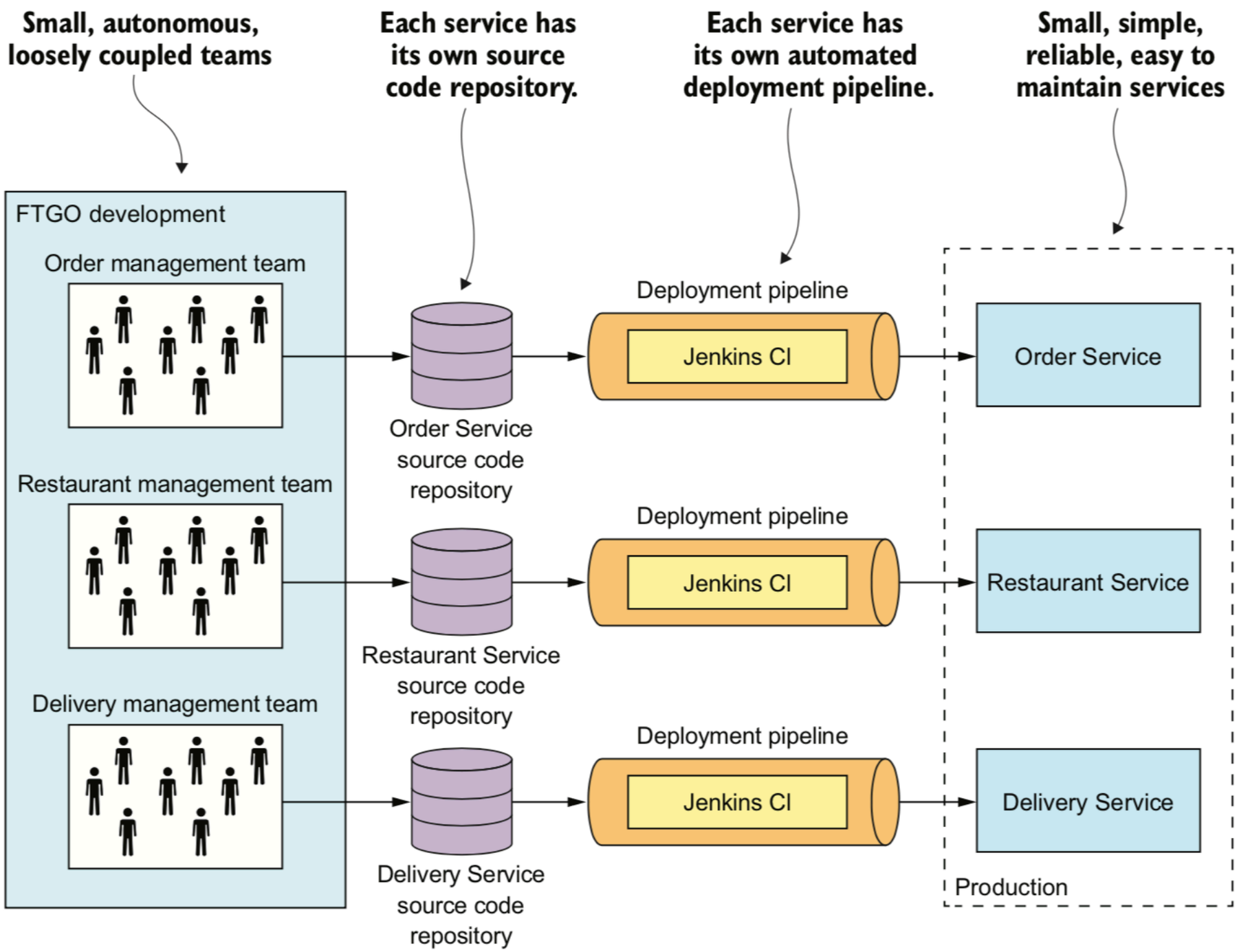
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Benefits of Continuous Delivery

Reduce time to market
Enable **business** to provide reliable service
Employee satisfaction is higher

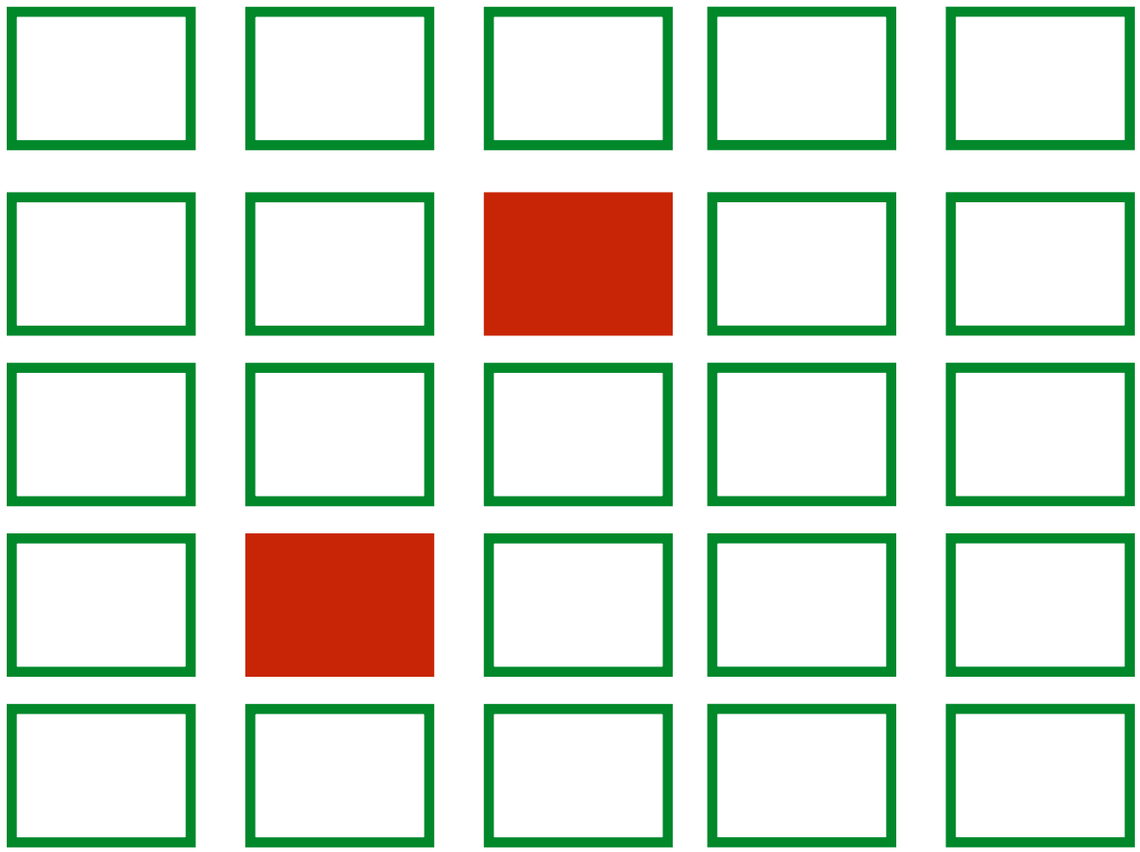
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Continuous Delivery for service



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

7. Better Fault isolation



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks of Microservice

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

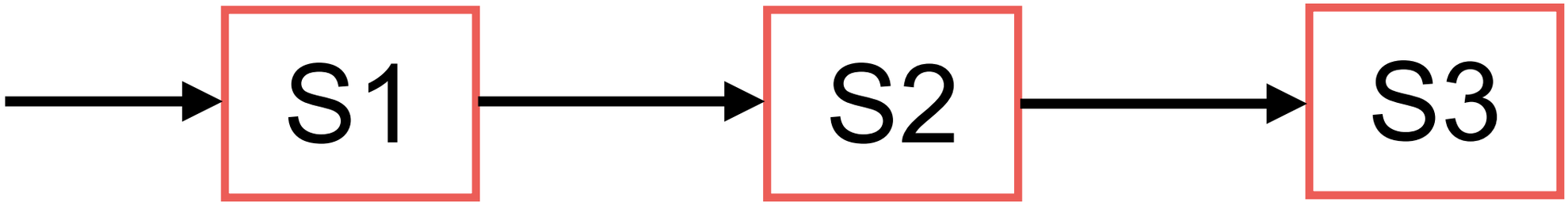
Drawbacks of Microservice

Find the right set of services
Distributed systems are complex
How to develop, testing and deploy ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks of Microservice

Deploy feature that required multiple service ?



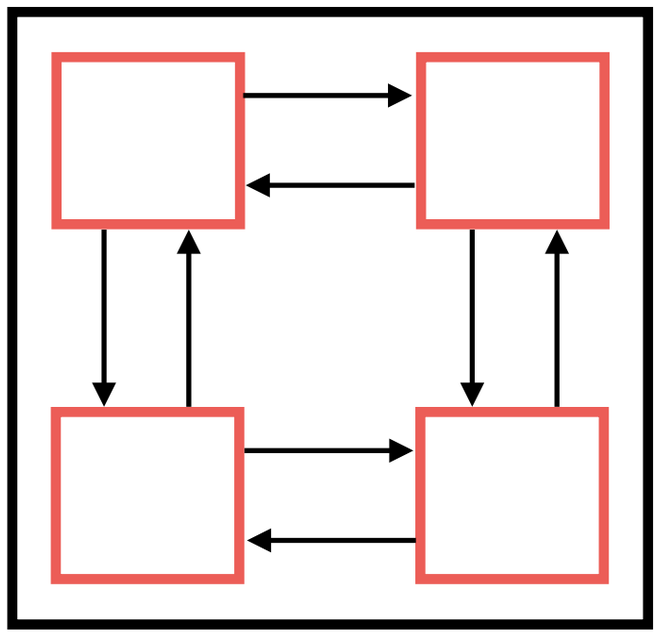
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

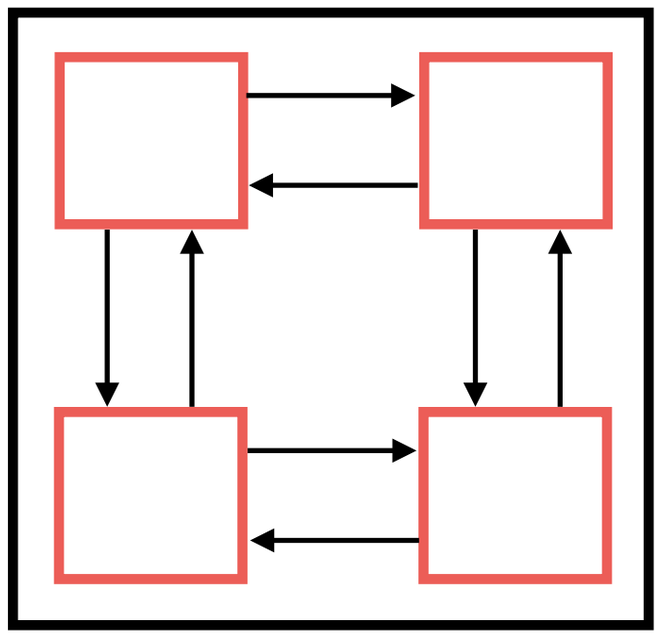
Coupling and Cohesion

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Cohesion

The code that changes together,
stay together

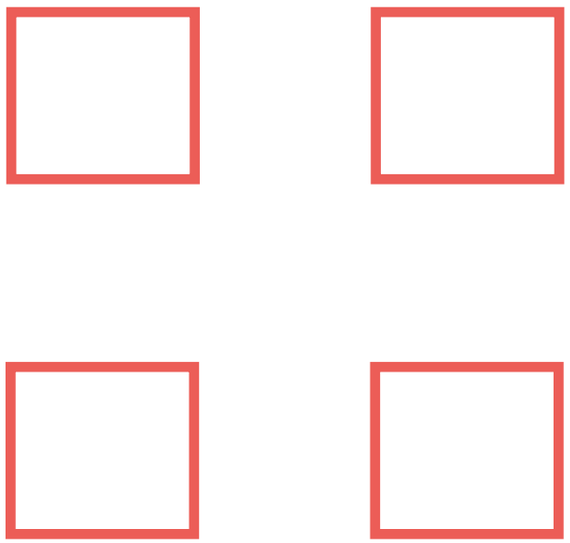




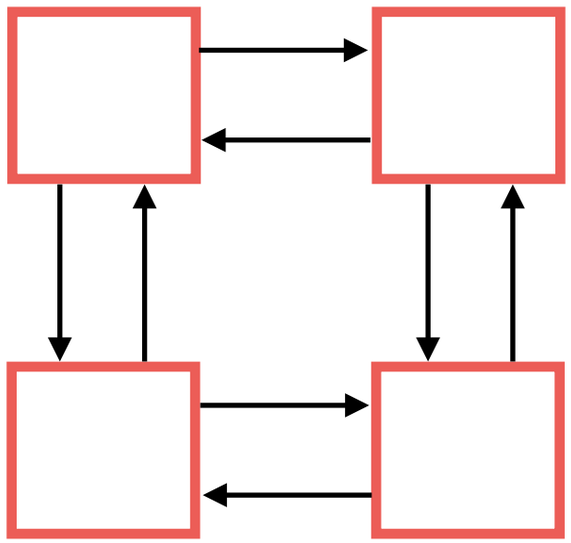
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Coupling

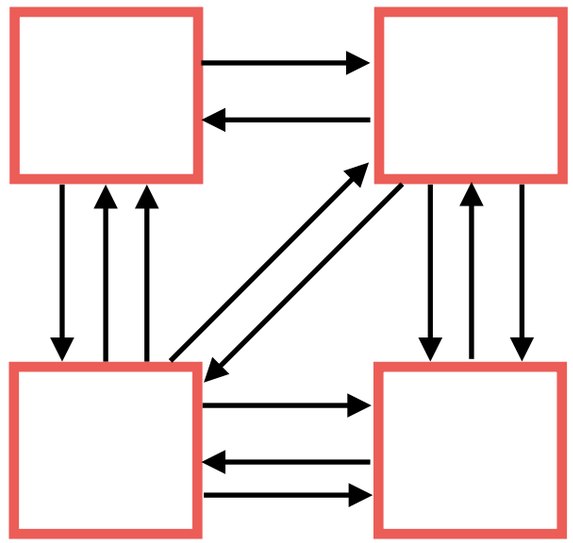
Degree of interdependence between services



Uncoupled



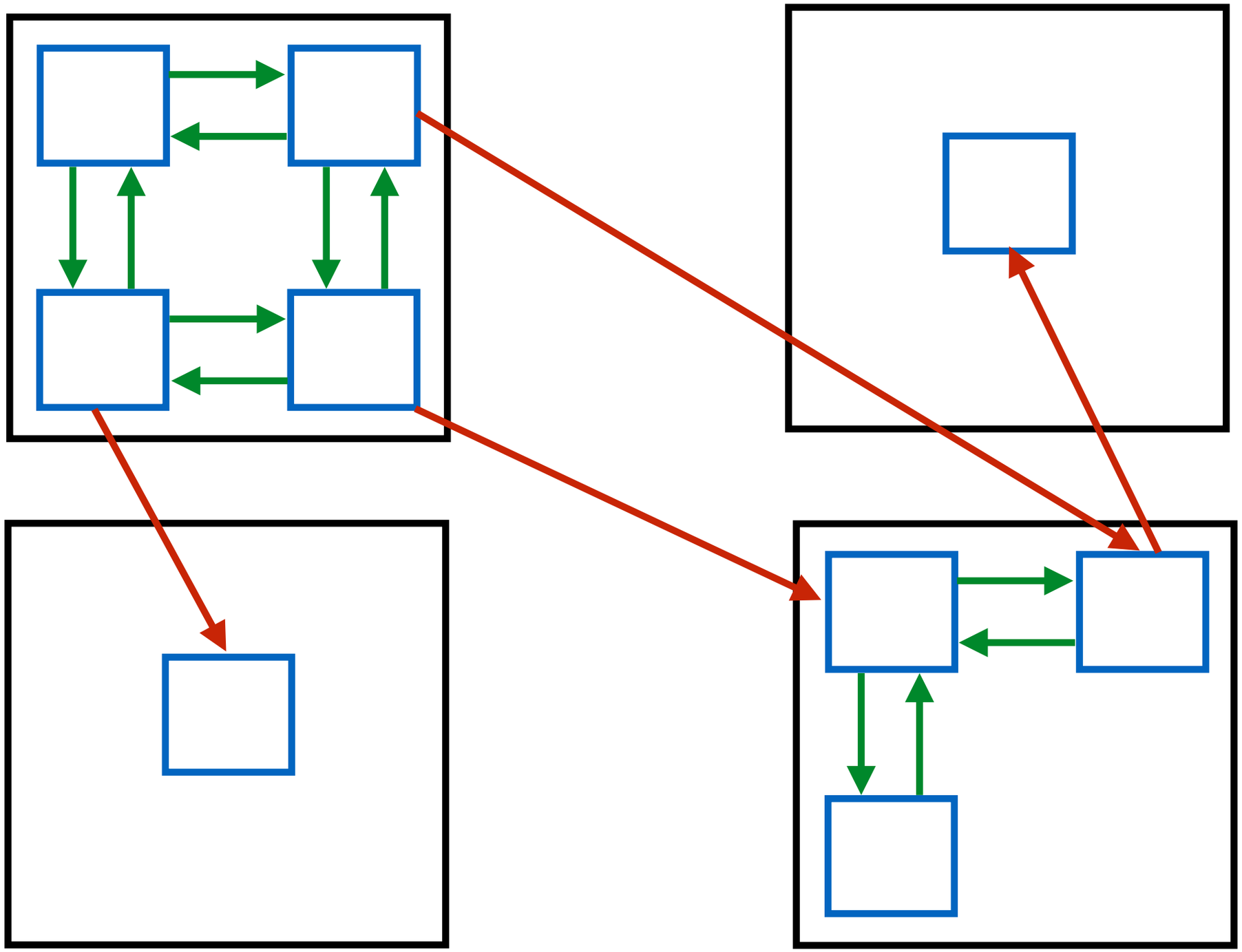
Loosely coupled



Tightly coupled

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Cohesion and Coupling

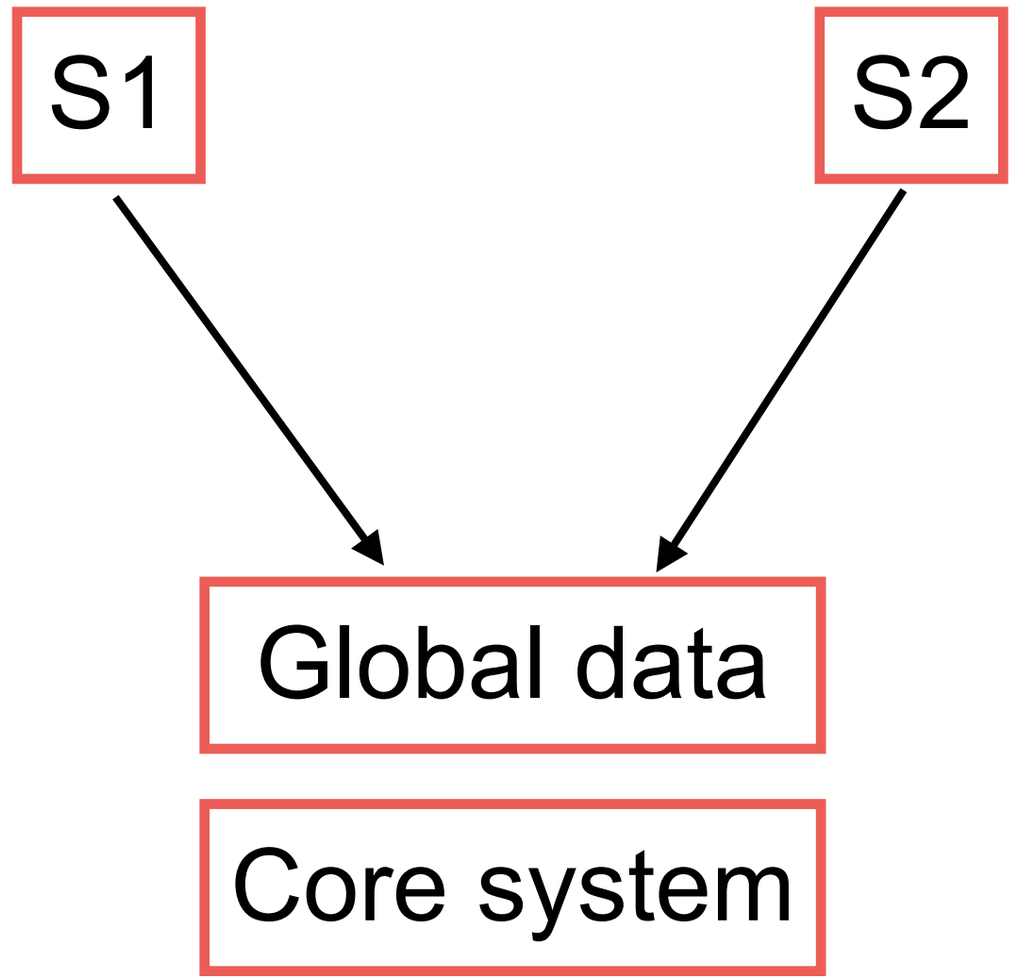


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Good system
Low Coupling
High Cohesion

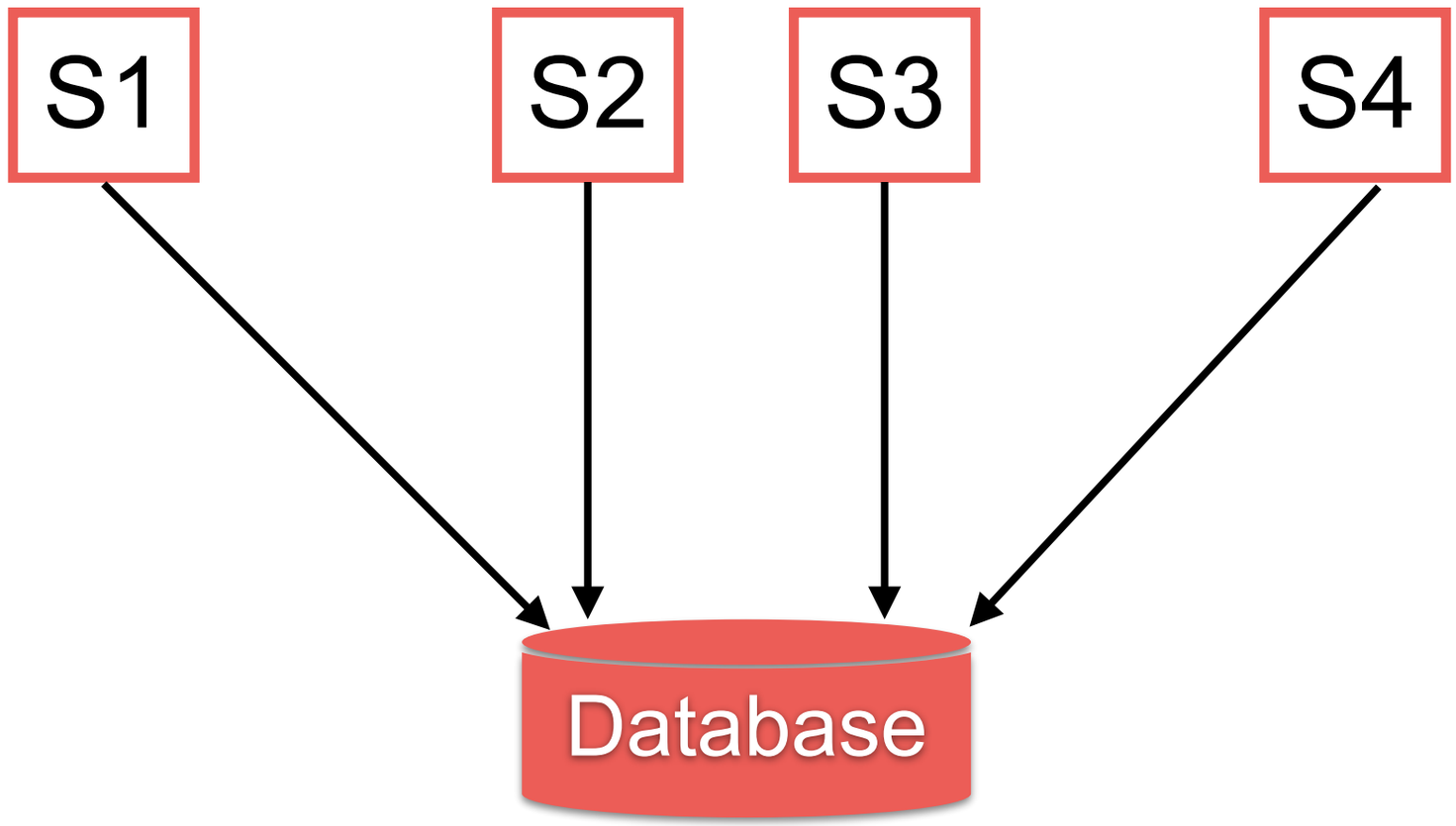
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Common Coupling



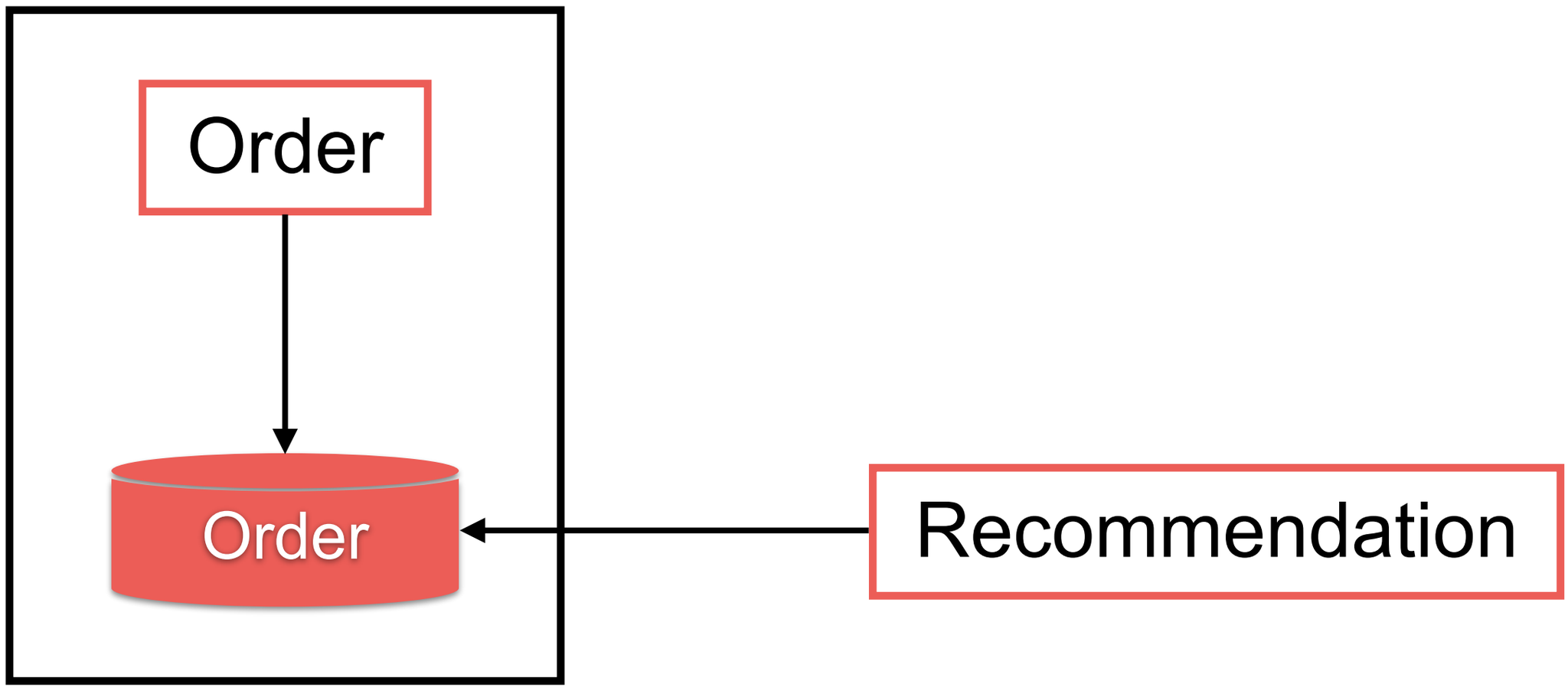
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Database Coupling



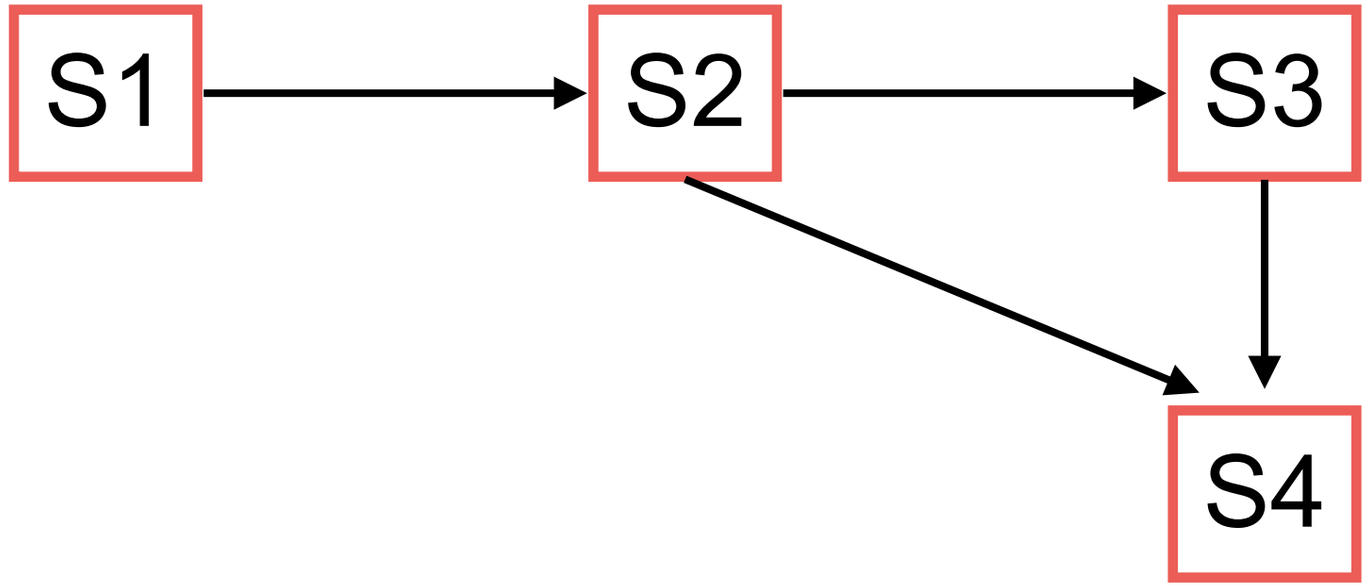
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example of problem ?



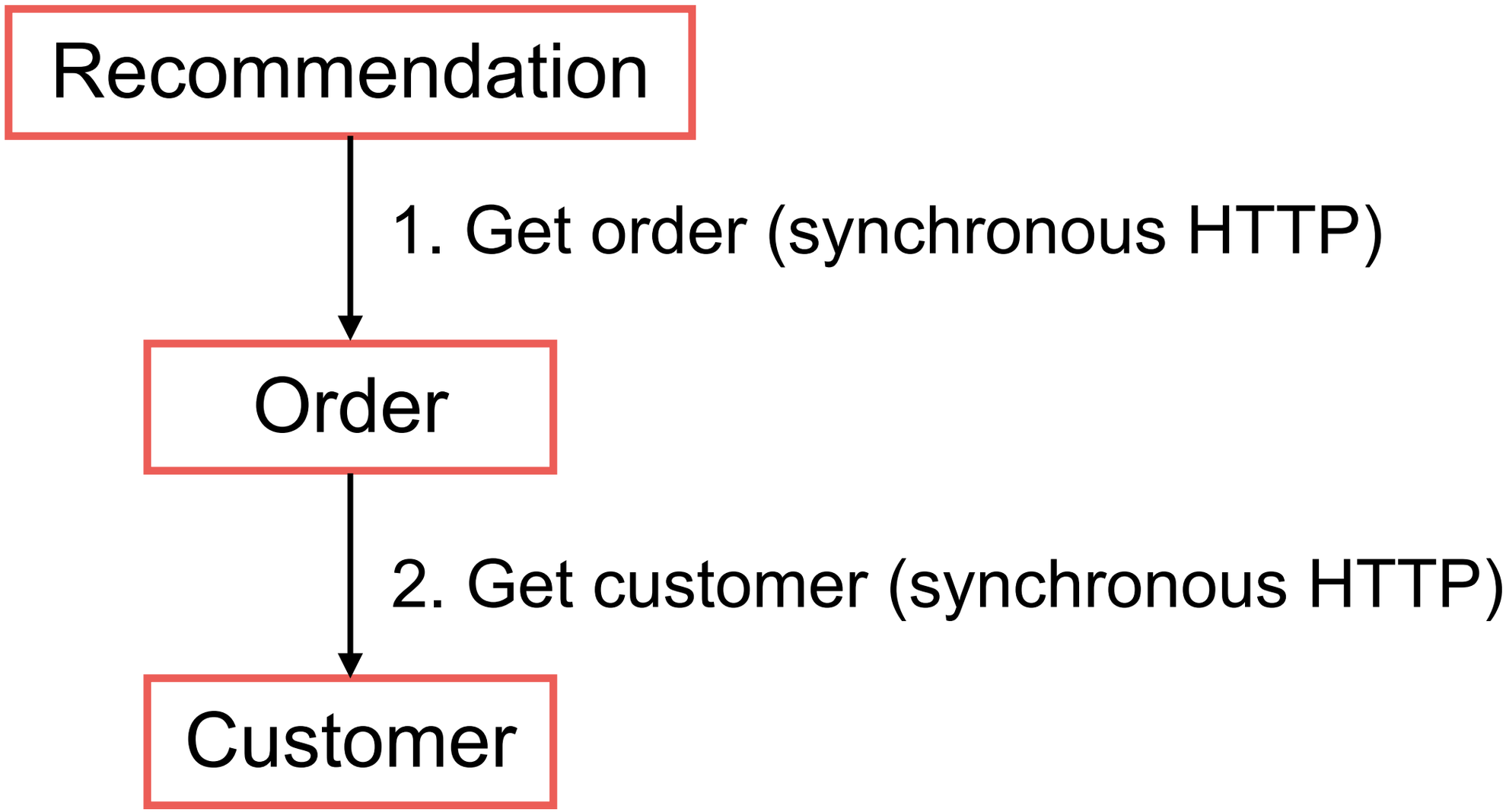
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Communication Coupling



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example of problem ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

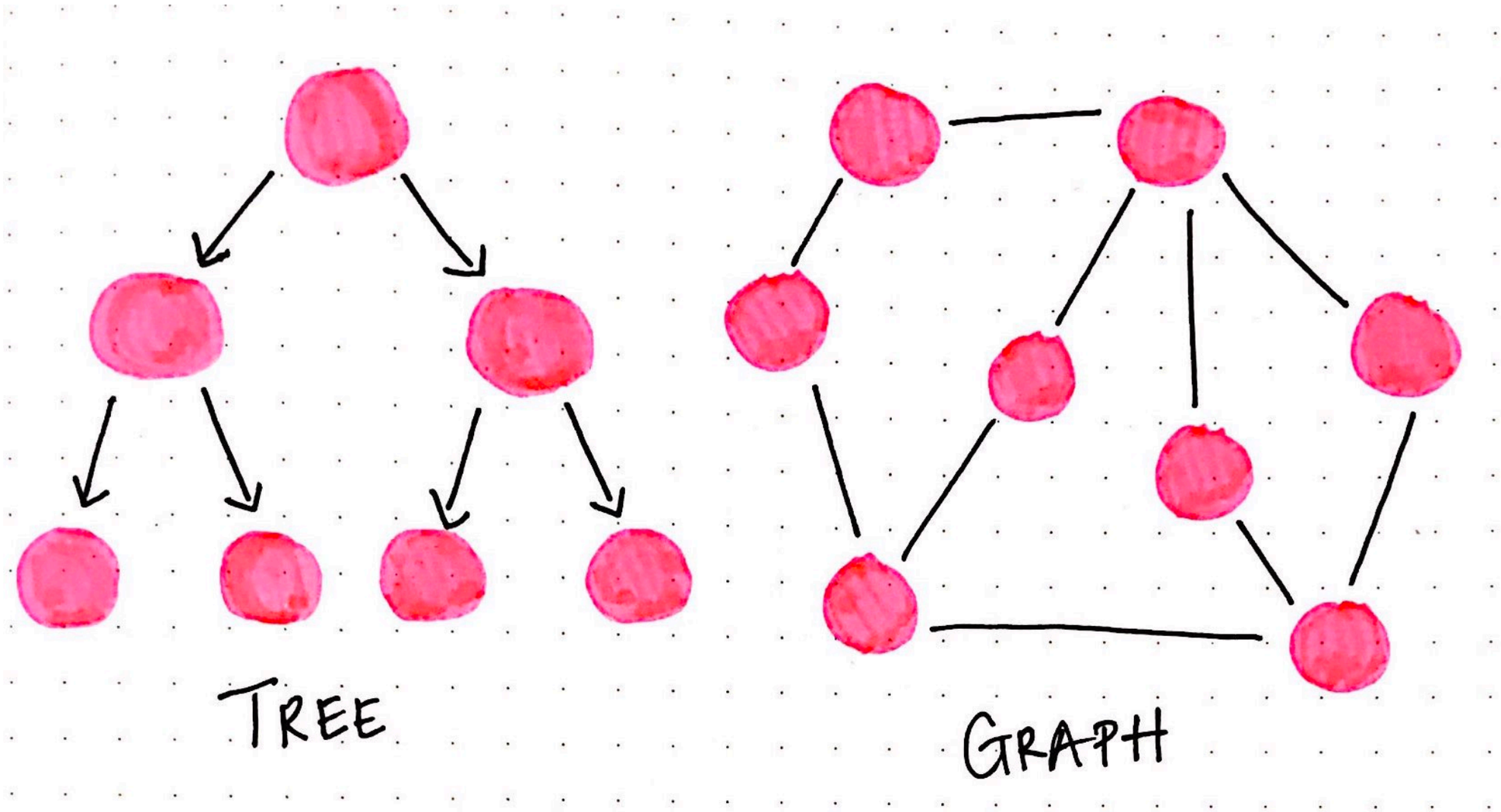
Deployment Coupling

Reduce deployment coupling not require Microservice

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service Principles

Minimize the depth of the service call-graph

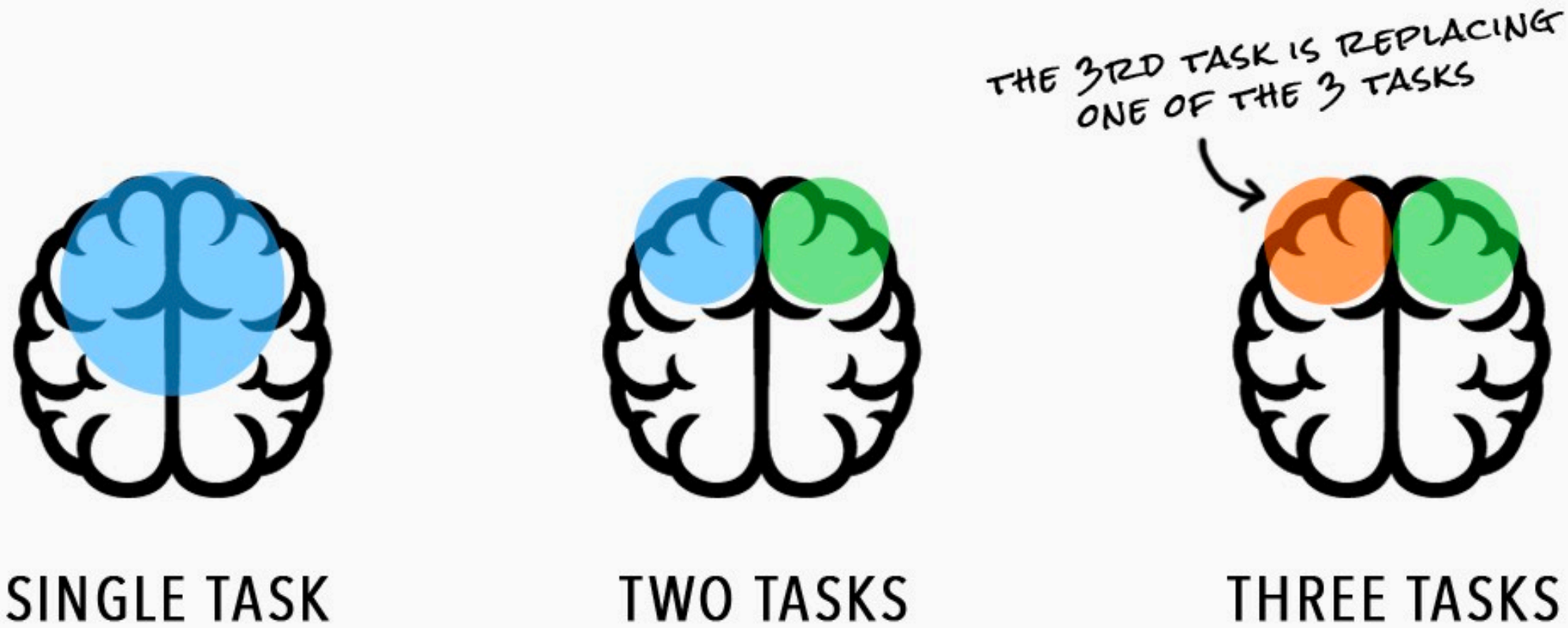


https://github.com/Yelp/service-principles

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service Principles

Minimize the number of services owned by your team

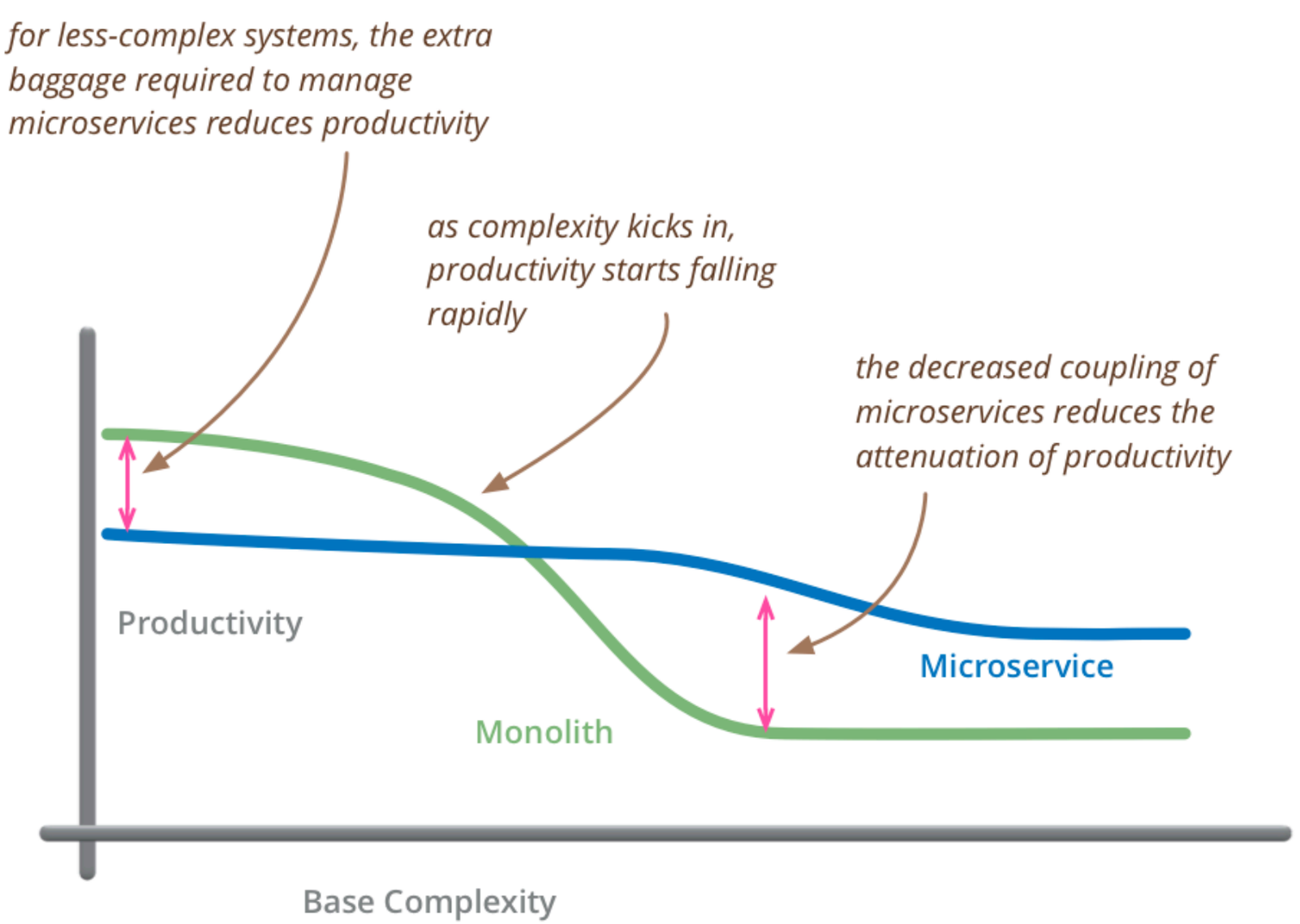


https://github.com/Yelp/service-principles

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks of Microservice

Decide to use when adopt is difficult !!



but remember the skill of the team will outweigh any monolith/microservice choice

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

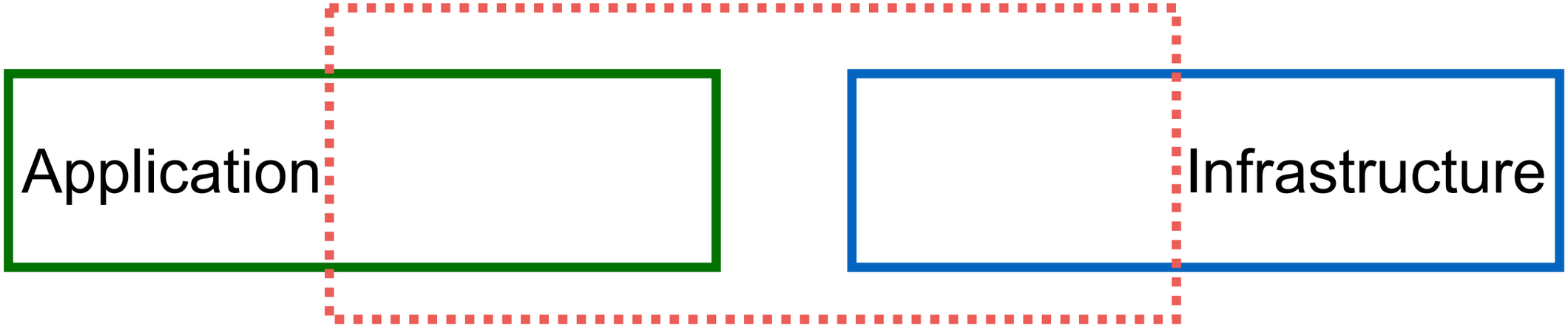
Microservice architecture patterns

https://microservices.io/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

3 Layers of service patterns

Application patterns
Application infrastructure pattern
Infrastructure pattern



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Patterns for Microservice

Decompose application into services
Communication patterns
Data consistency patterns
Service deployment patterns
Observability patterns
Automated testing of services
Security patterns

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Decompose application into services

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Premature splitting is the root of all **evil**.

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Every time you make the decision to split out a new microservice, there's a **risk** of ending up with a bloated app.

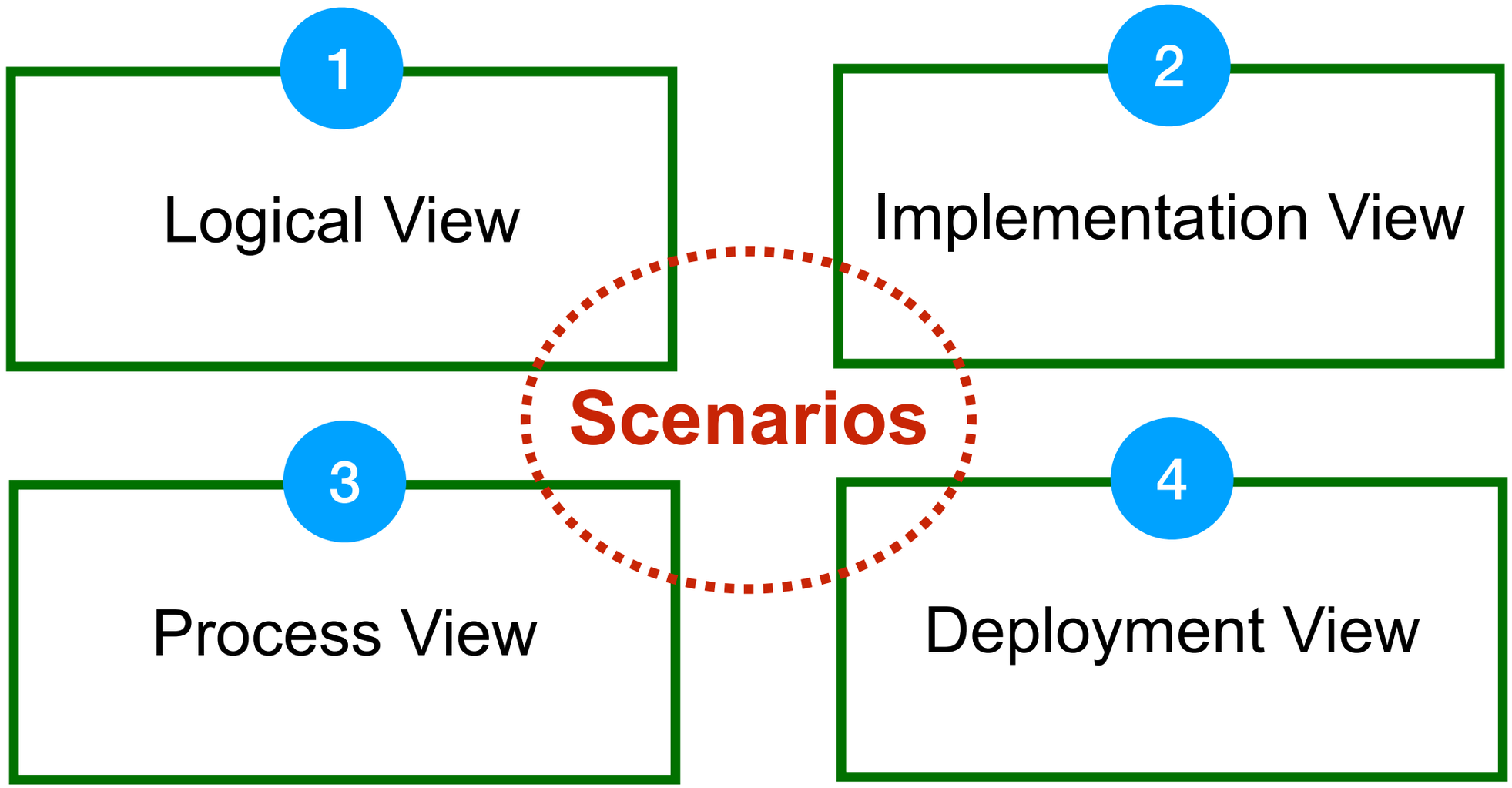
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4 View model of Software Architecture

1. Logical View
2. Implementation View
3. Process View
4. Deployment View

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

4 View model of Software Architecture



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Decompose application into services

By business capability ?
By subdomain/technical ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Step to define services

1. Identify system operations
2. Identify services
3. Define service APIs and collaboration

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

1. Identify system operations

Start with functional requirements

User story

1

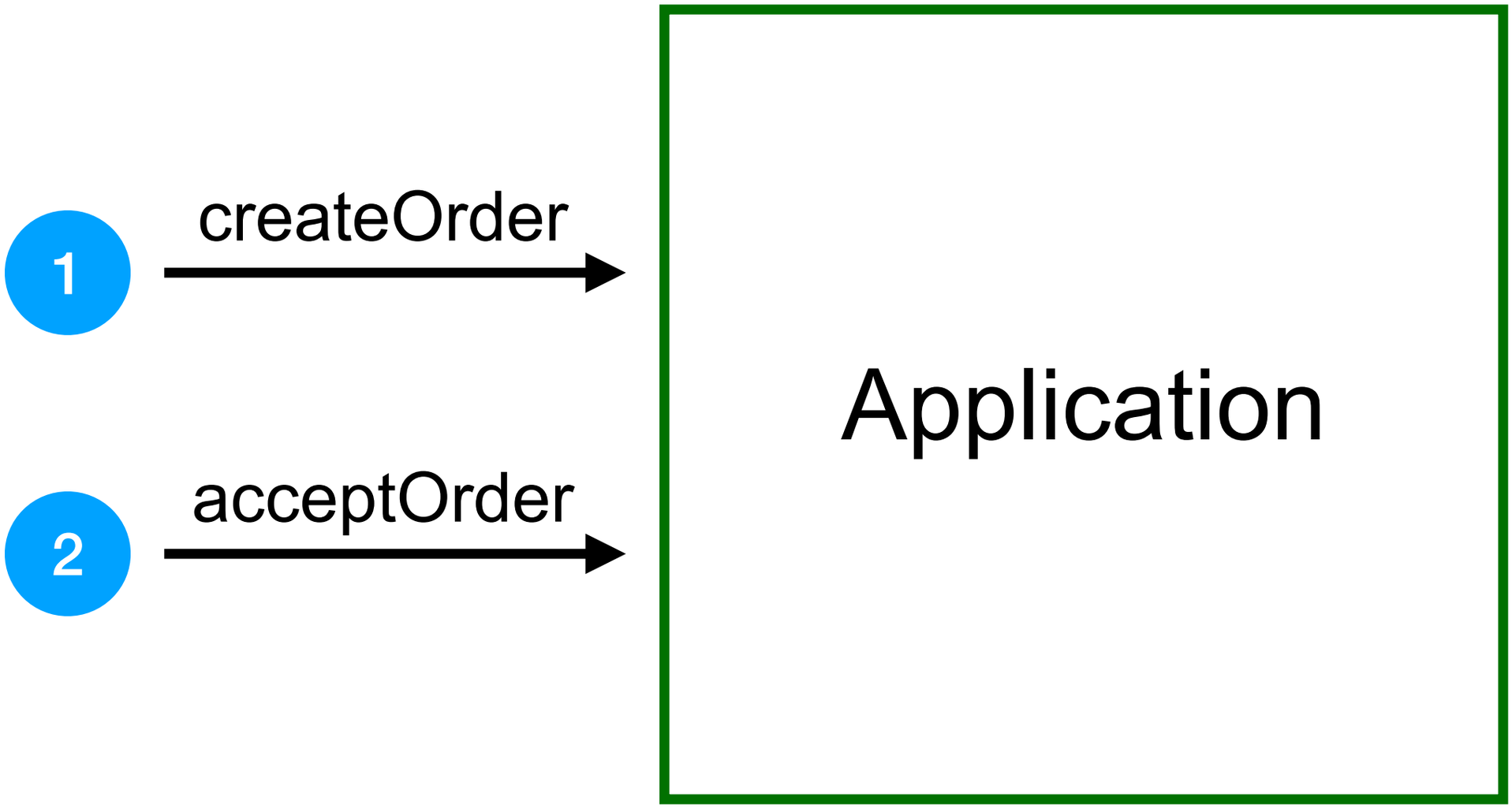
ลูกค้าต้องการสั่งซื้ออาหาร

2

ร้านอาหารรับคำสั่งซื้อ

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

1. Identify system operations



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

2. Identify services

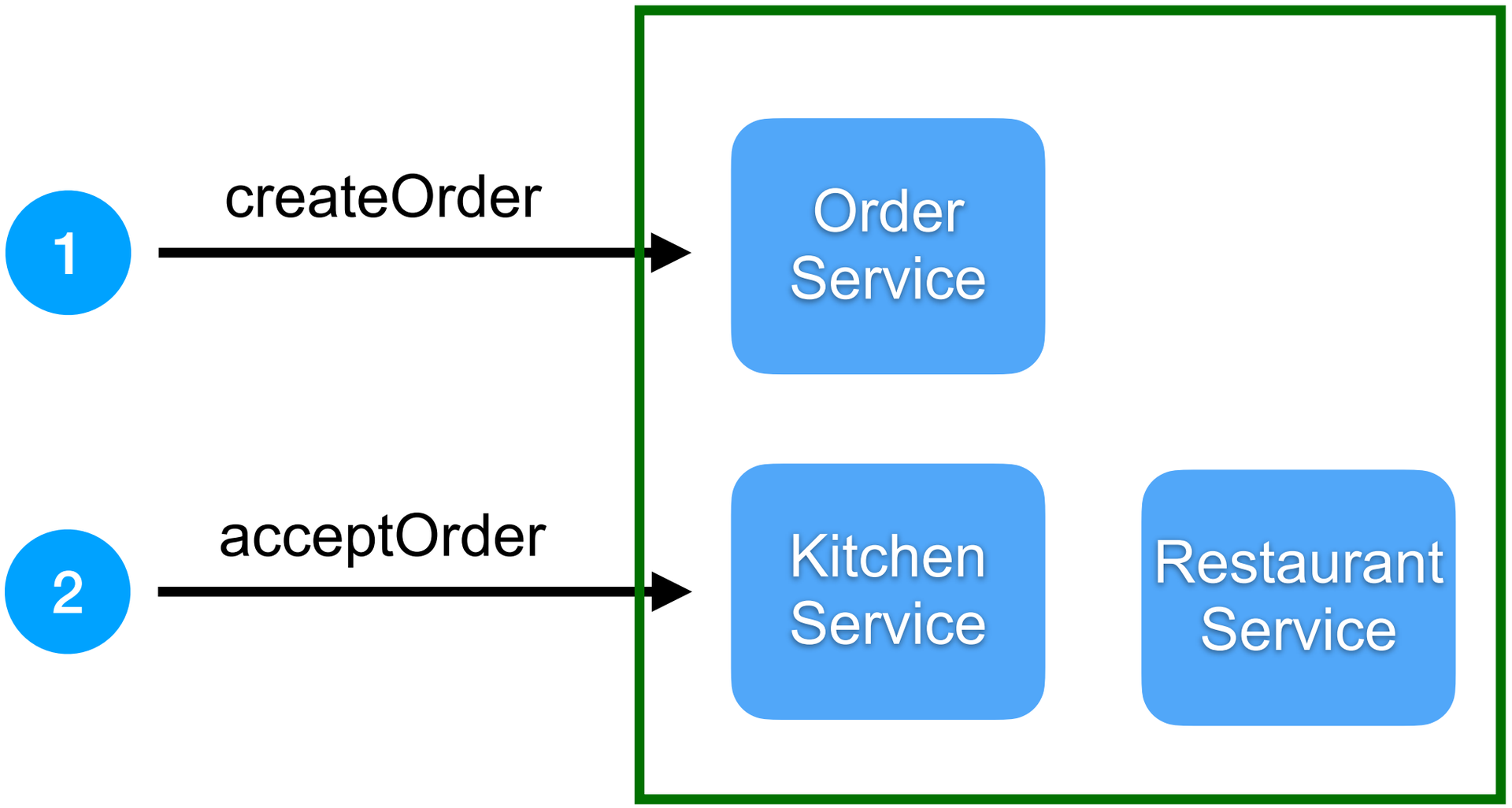
Try to decomposition into services

Business > Technical concept

What > How

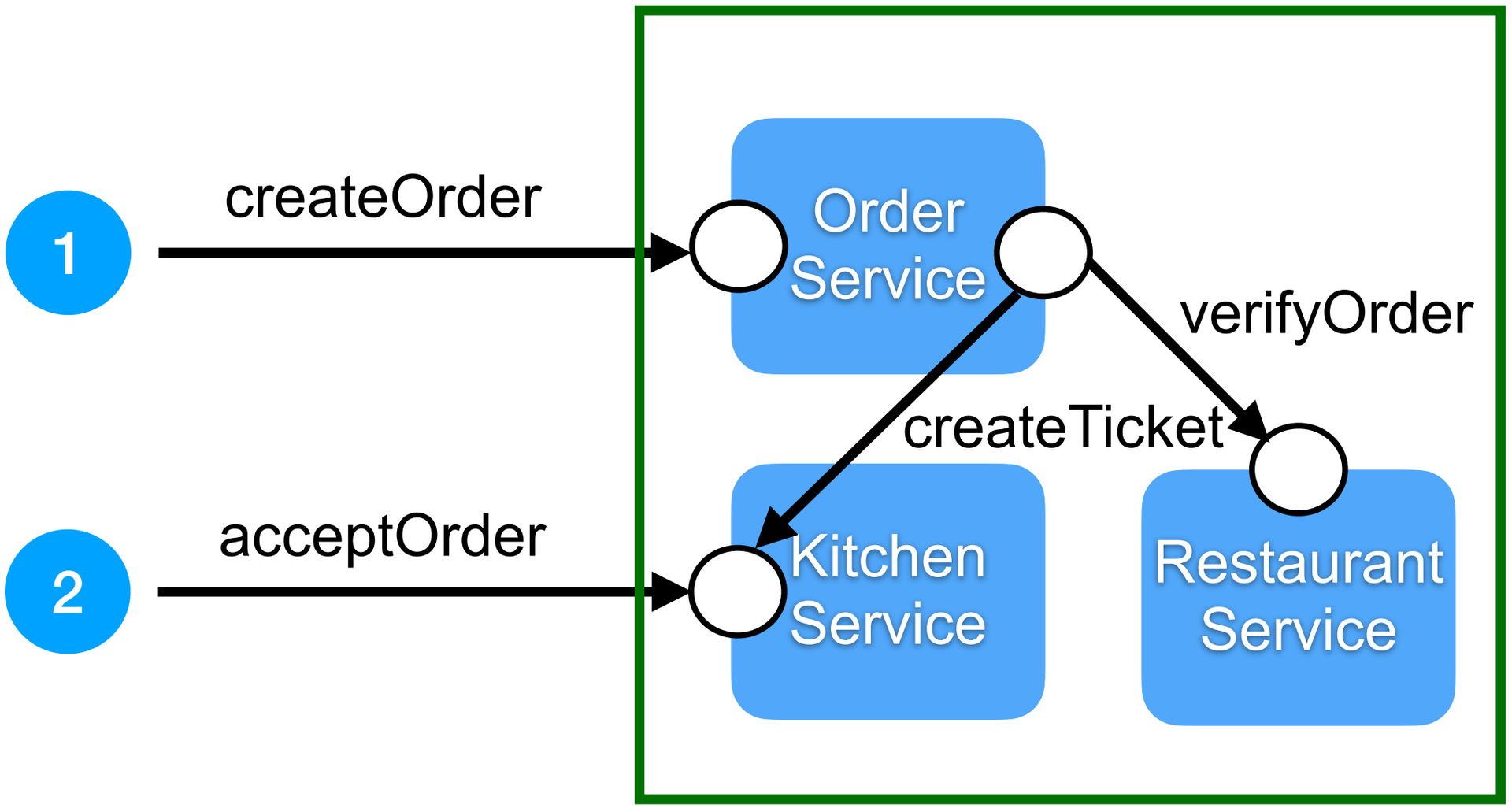
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

2. Identify services



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

3. Define service APIs and collaborations



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Problems when decompose services ?

Network latency
Reliability of communication (sync)
Maintain data consistency
God classes

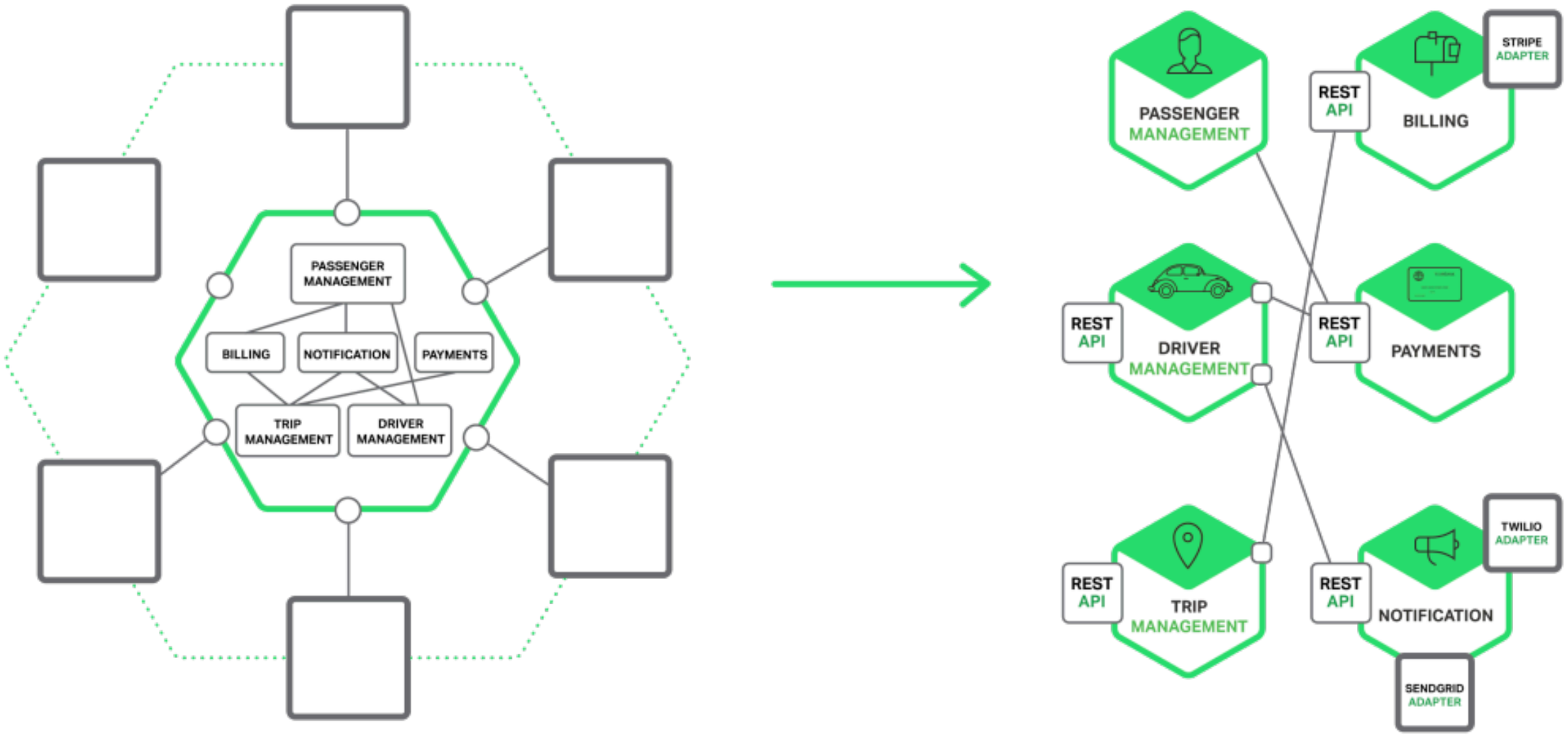
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Communication patterns

Communication style
Discovery service
Reliability of communication
Transactional messaging
External API

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Inter Process Communication (IPC)



https://www.nginx.com

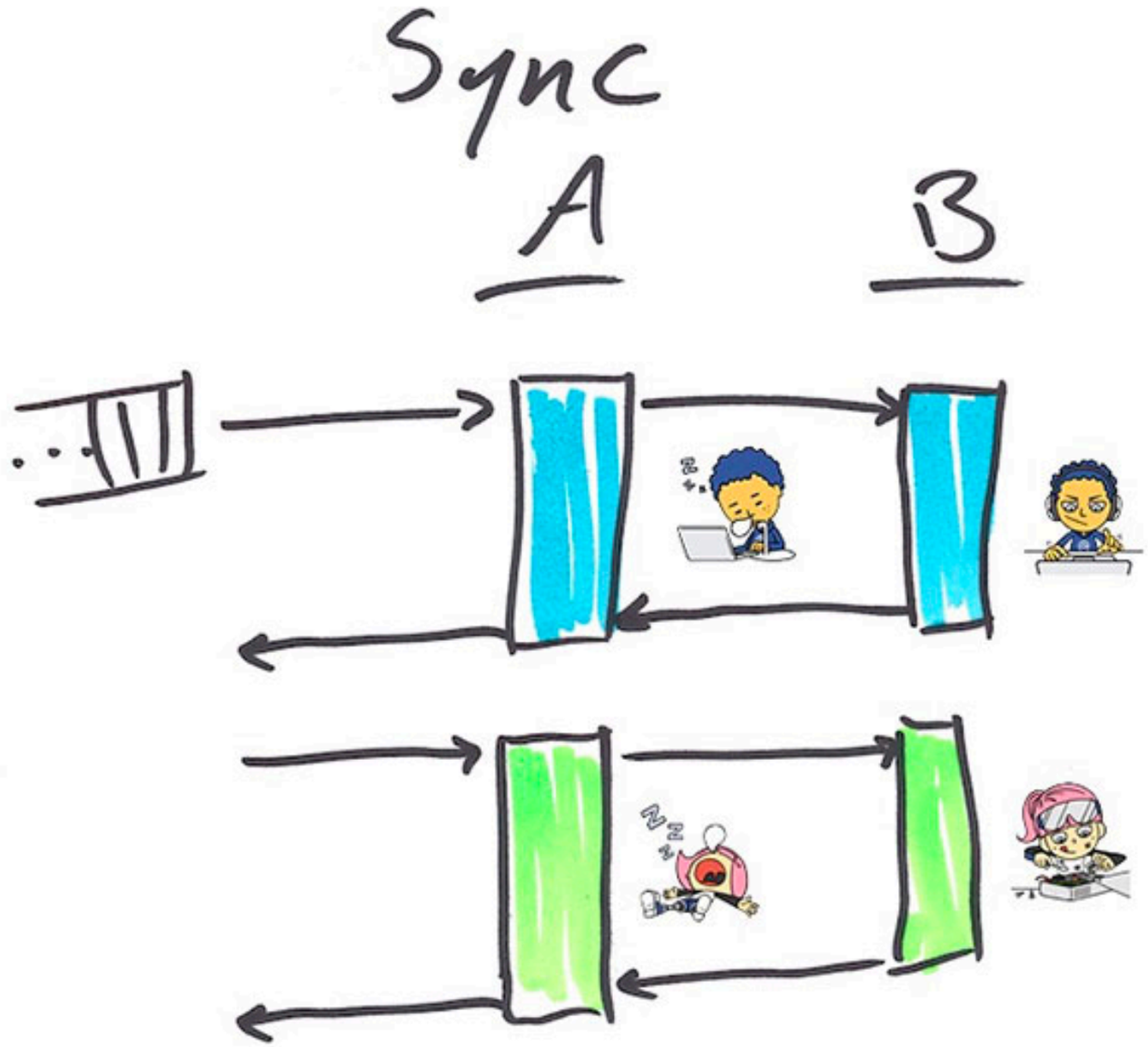
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Interaction Styles

	One-to-one	**One-to-many**
Synchronous	Request/response	-
Asynchronous	Async request/response	Publish/subscribe
	Notification	Publish/async response

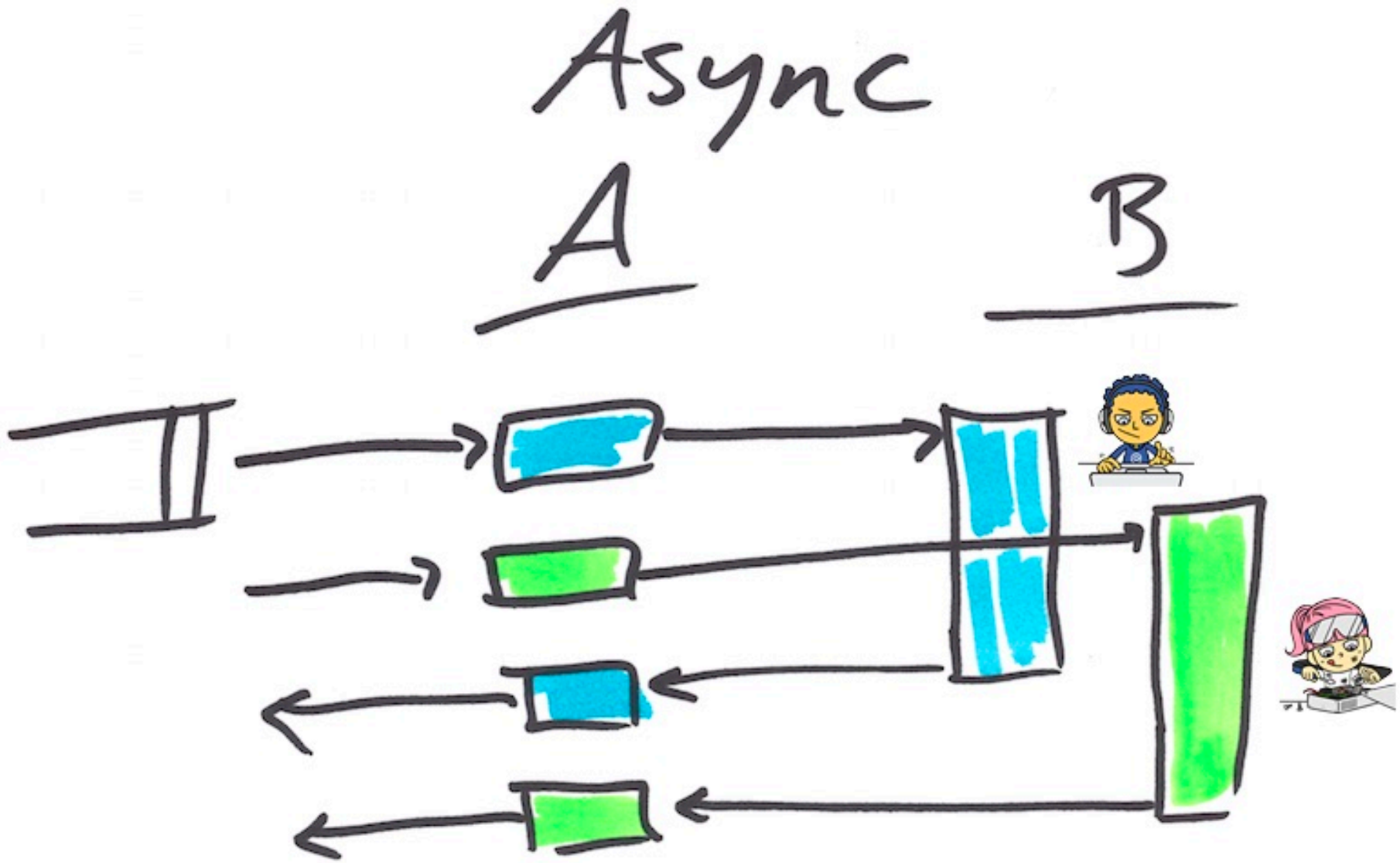
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Synchronous



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Asynchronous

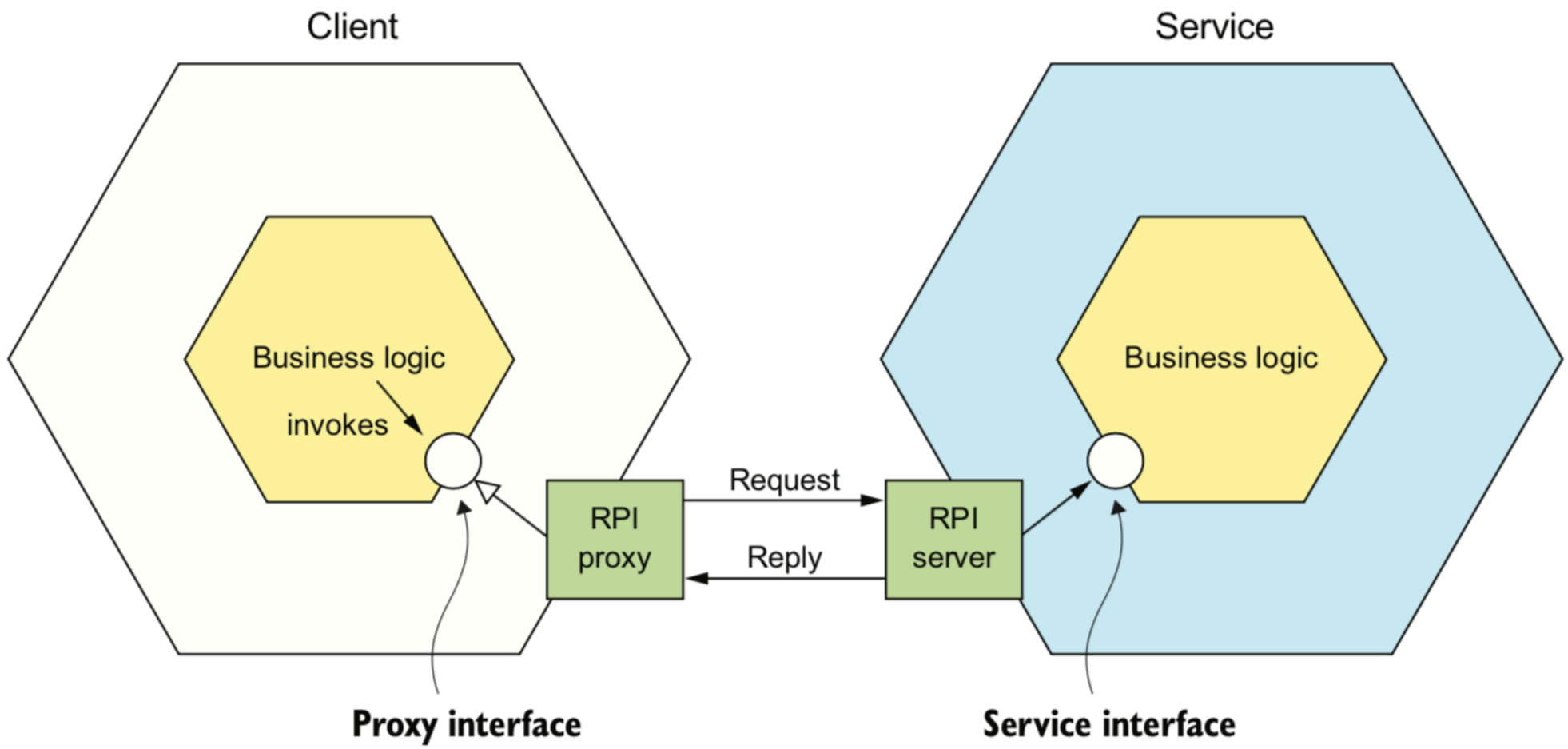


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Synchronous
Remote **P**rocedure **I**nvocation

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Remote Procedure Invocation



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Synchronous Remote Procedure Invocation

REST

gRPC

Handling failure with **Circuit breaker**

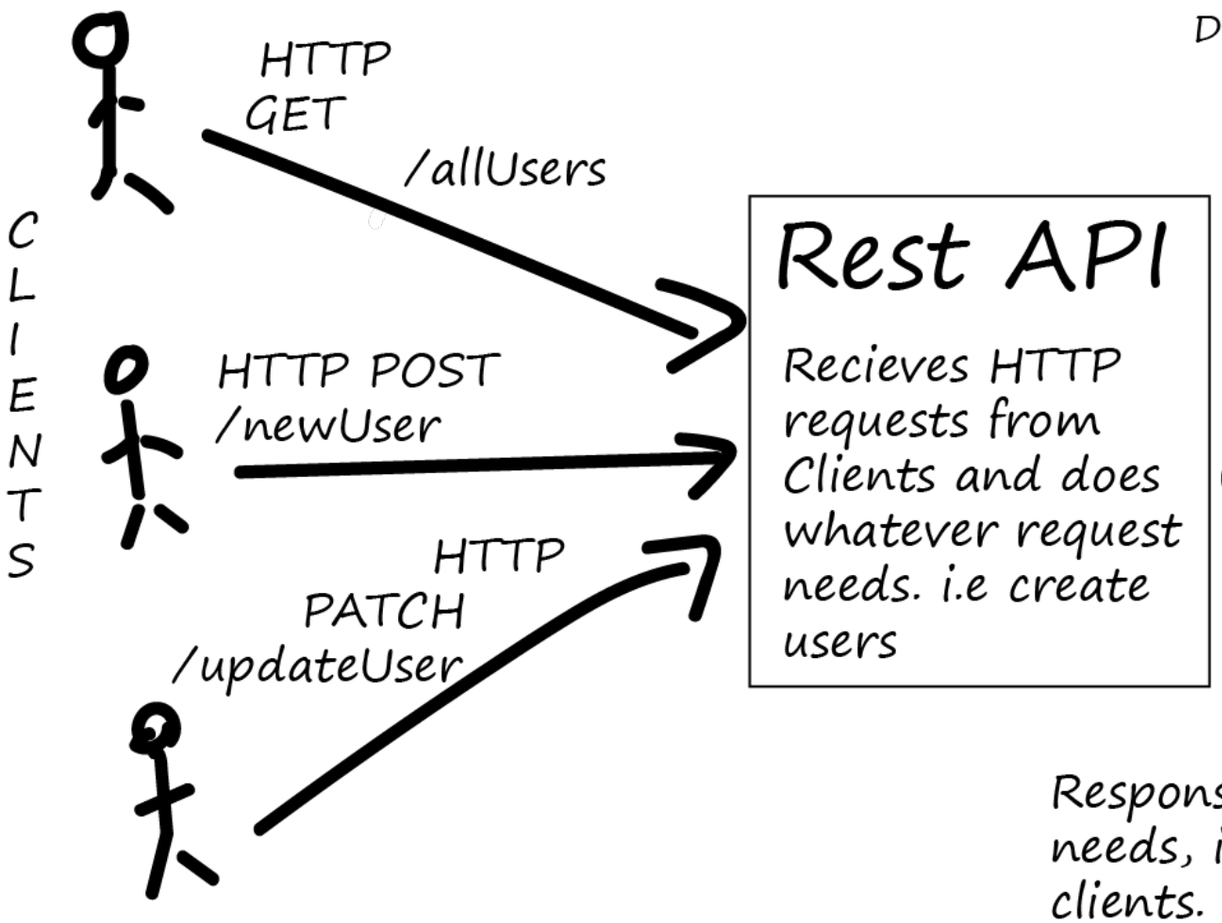
Service discovery

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

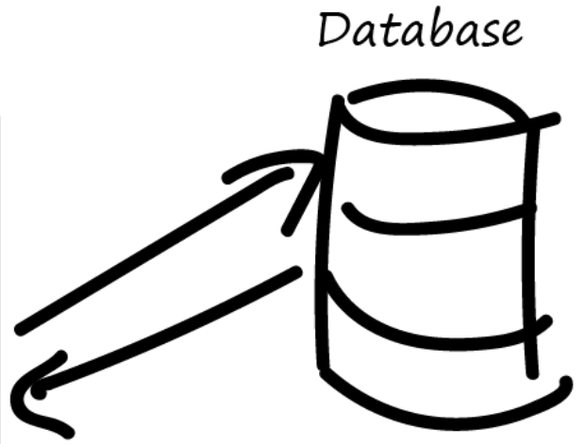
REpresentational **S**tate **T**ransfer

Rest API Basics

Typical HTTP Verbs:
GET -> Read from Database
PUT -> Update/Replace row in Database
PATCH -> Update/Modify row in Database
POST -> Create a new record in the database
DELETE -> Delete from the database



Our Clients, send HTTP Requests and wait for responses



Our Rest API queries the database for what it needs

Response: When the Rest API has what it needs, it sends back a response to the clients. This would typically be in JSON or XML format.

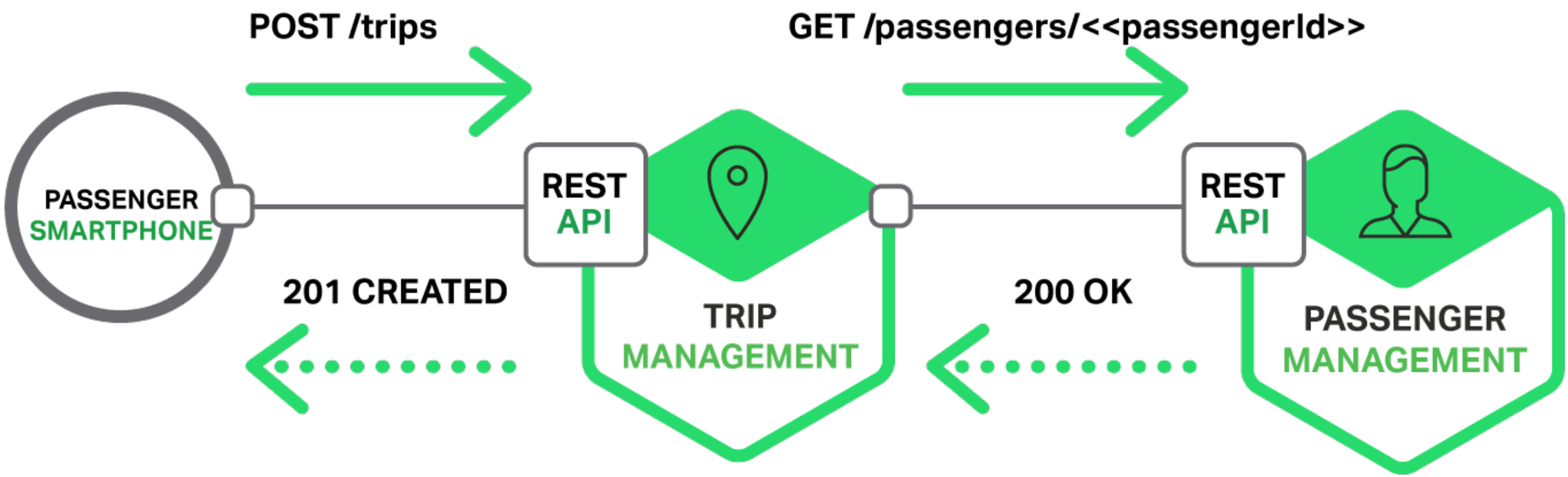
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

REST :: mapping with HTTP verbs

Activity	HTTP Method
Retrieve data	GET
Create new data	POST
Update data	PUT
Delete data	DELETE

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

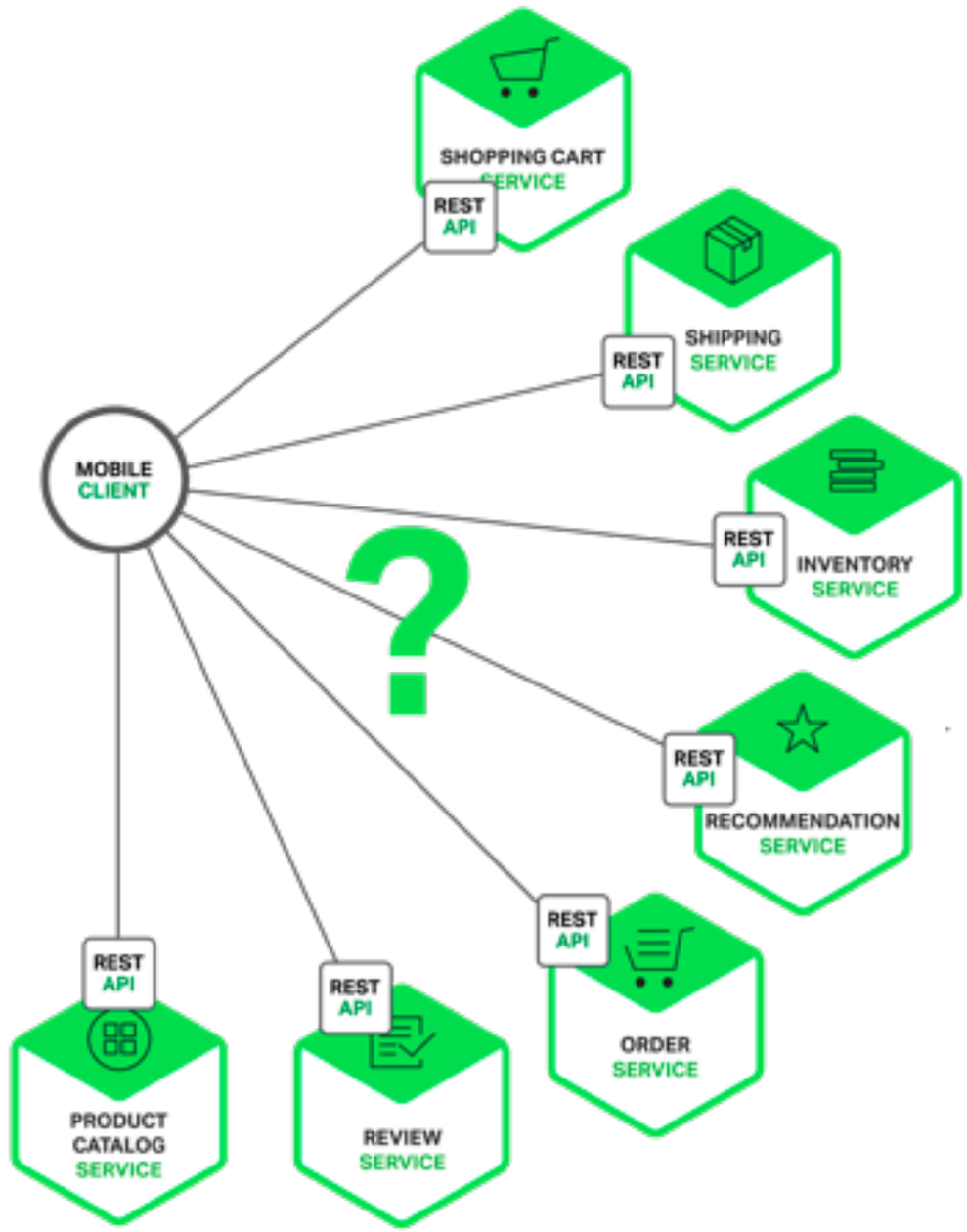
Request and response format



https://www.nginx.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

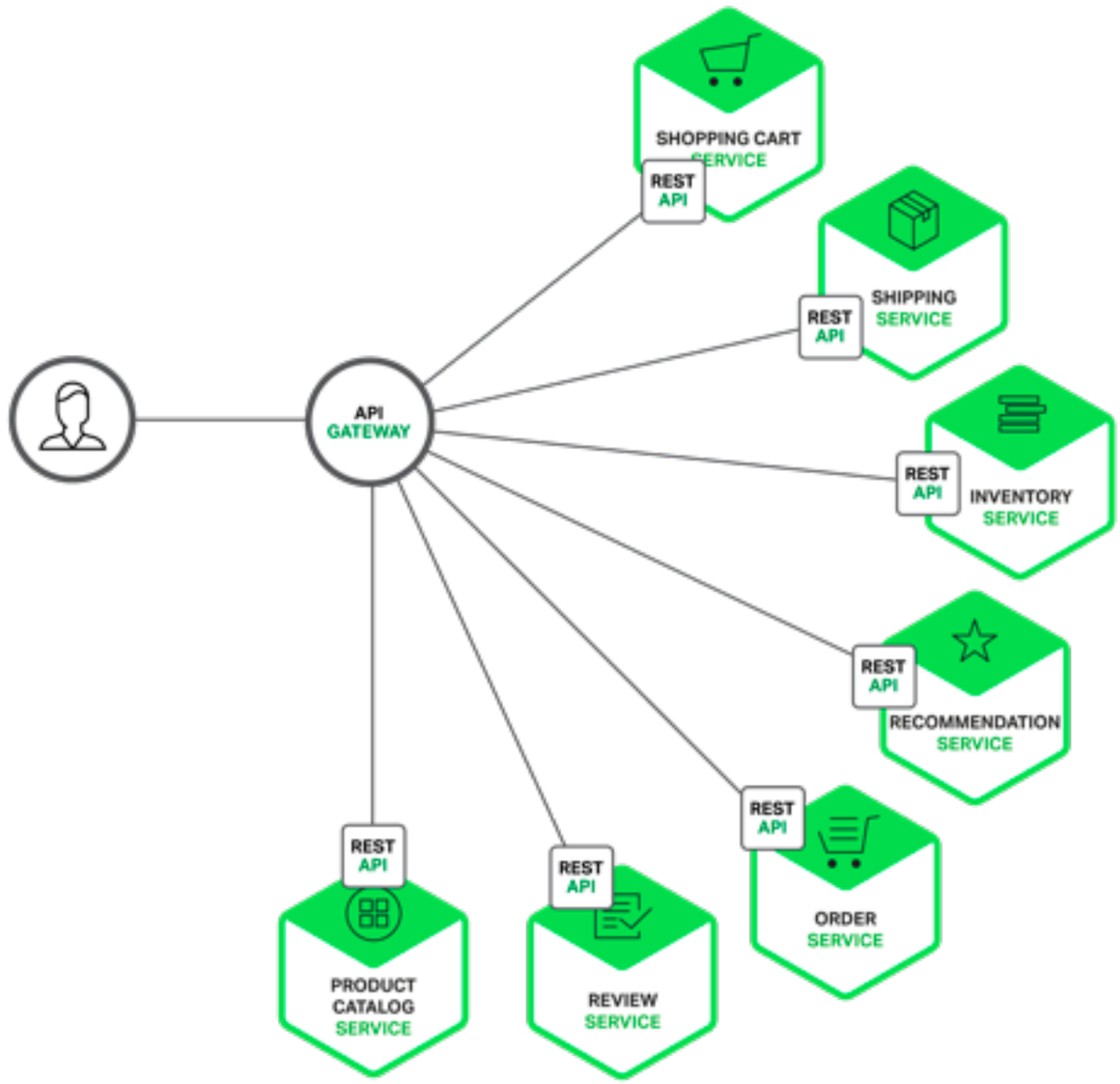
Direct communication !!



https://www.nginx.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

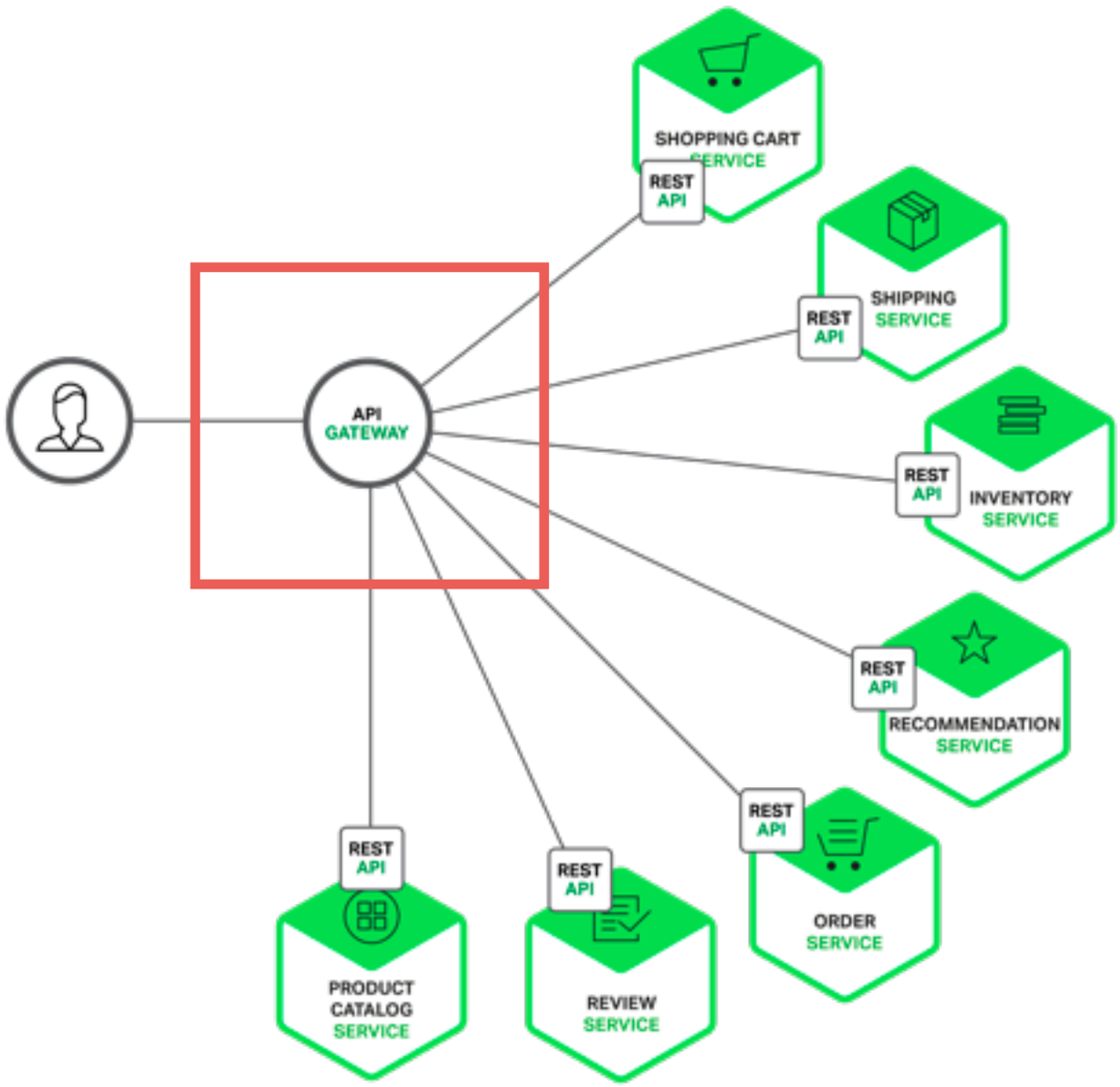
Working with API gateway



https://www.nginx.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

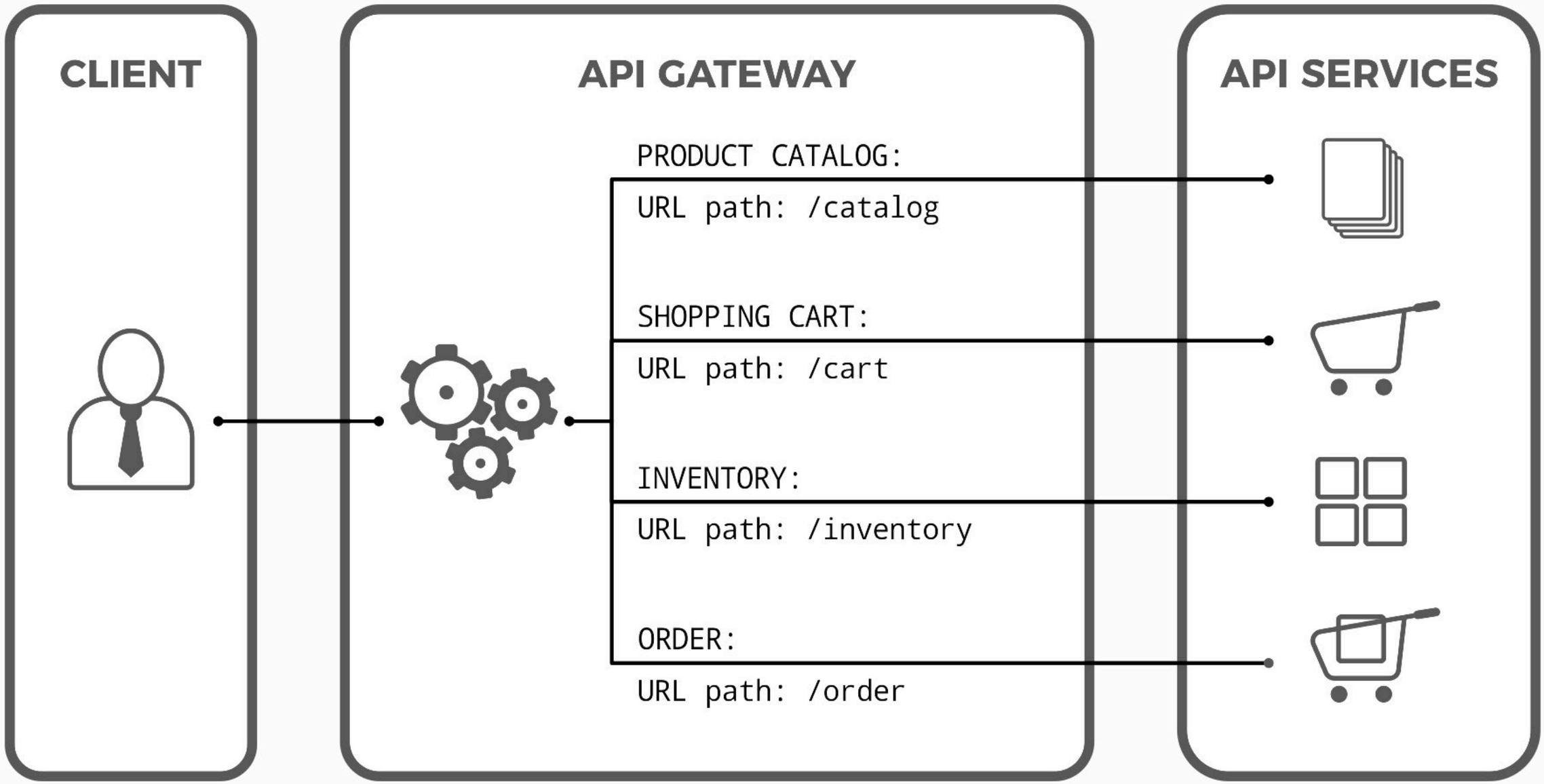
Avoid Single Point of Failure !!



https://www.nginx.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Working with API gateway



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Functions of API gateway

Authentication
Authorization
Rate limiting
Caching
Metrics collection
Request logging
Load balancing
Circuit breaking

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Benefits

Encapsulation interface structure
Client talk to service via gateway
Reduce round trip between service

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks

Development/Performance bottleneck

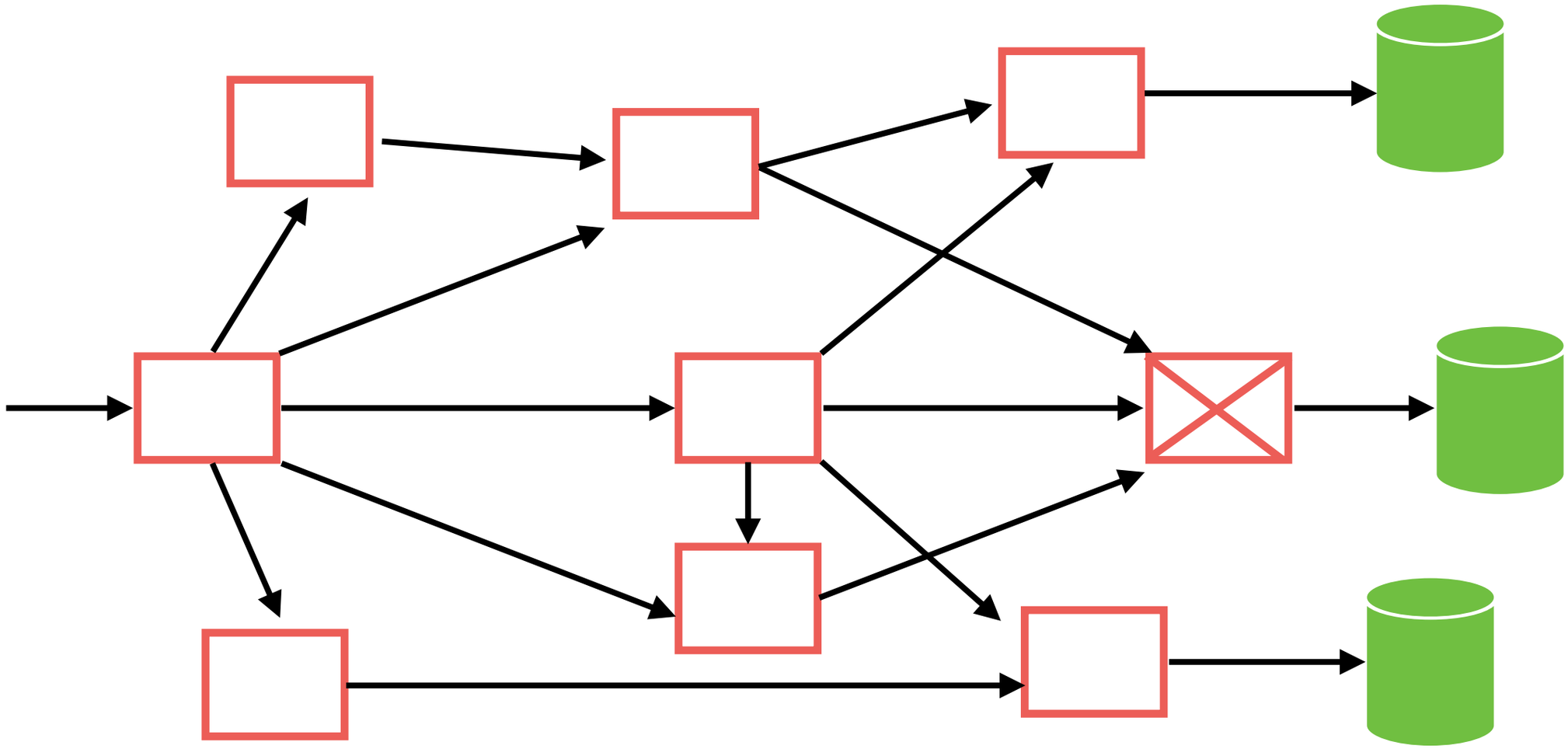
Single point of failure

Waiting for update data in gateway

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Circuit Breaker pattern

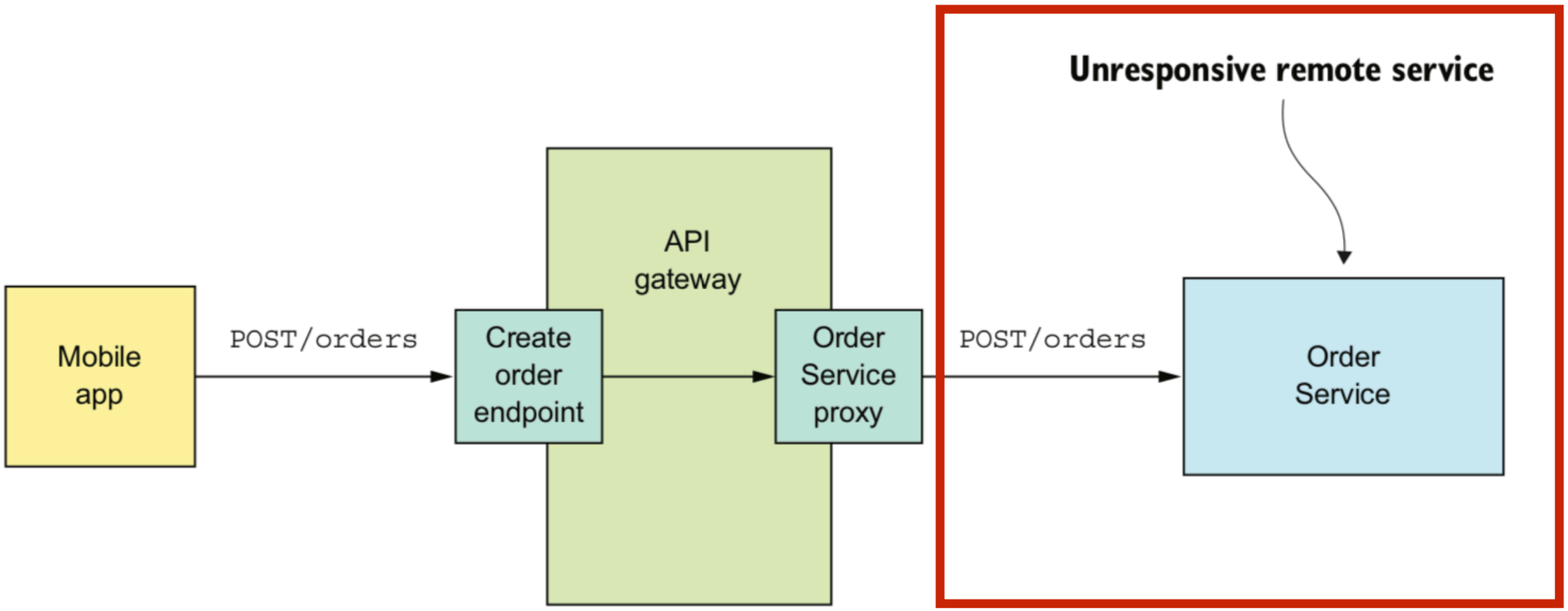
How to handling partial failure ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Circuit Breaker pattern

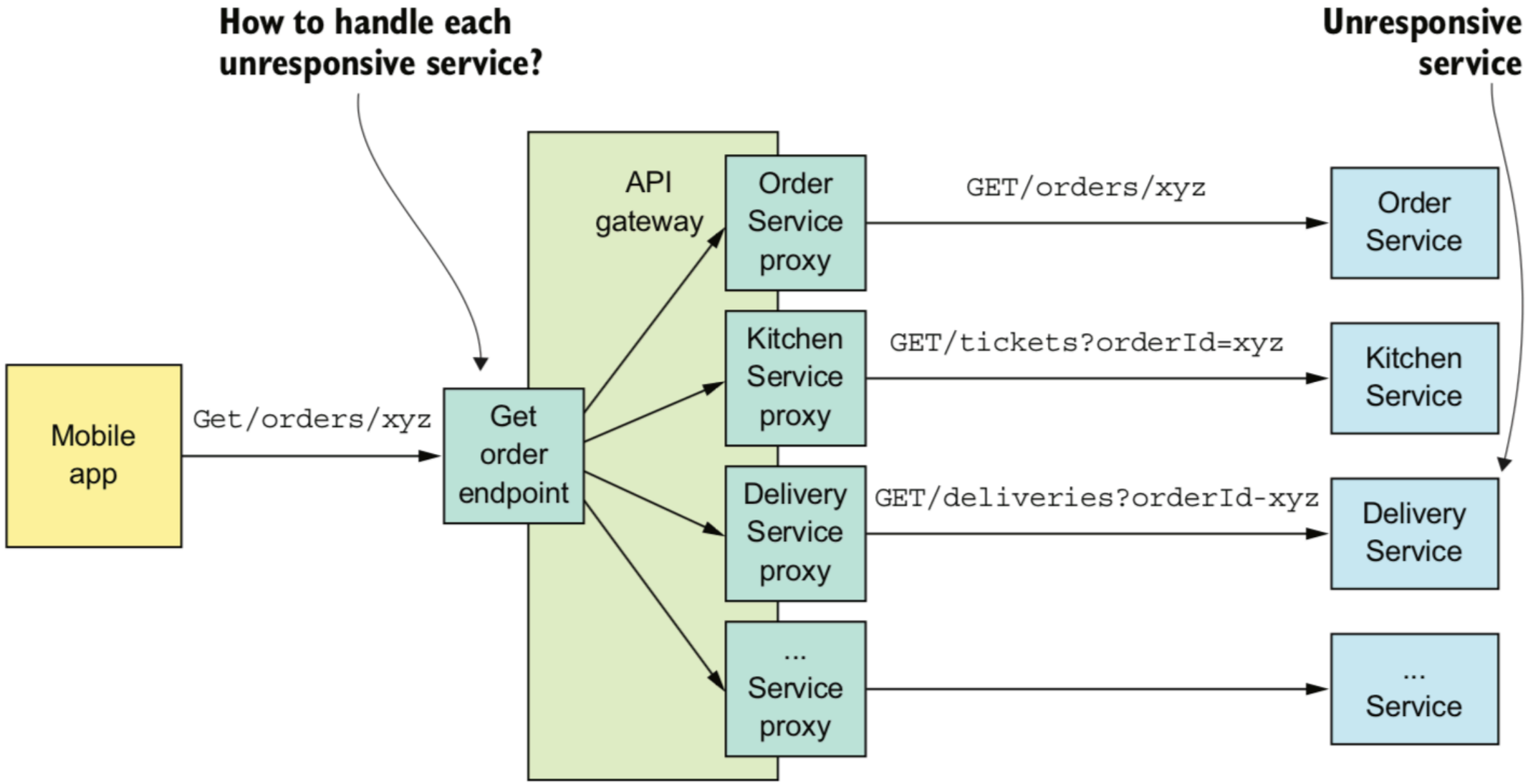
How to handling partial failure ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Circuit Breaker pattern

How to handling partial failure ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Develop robust patterns

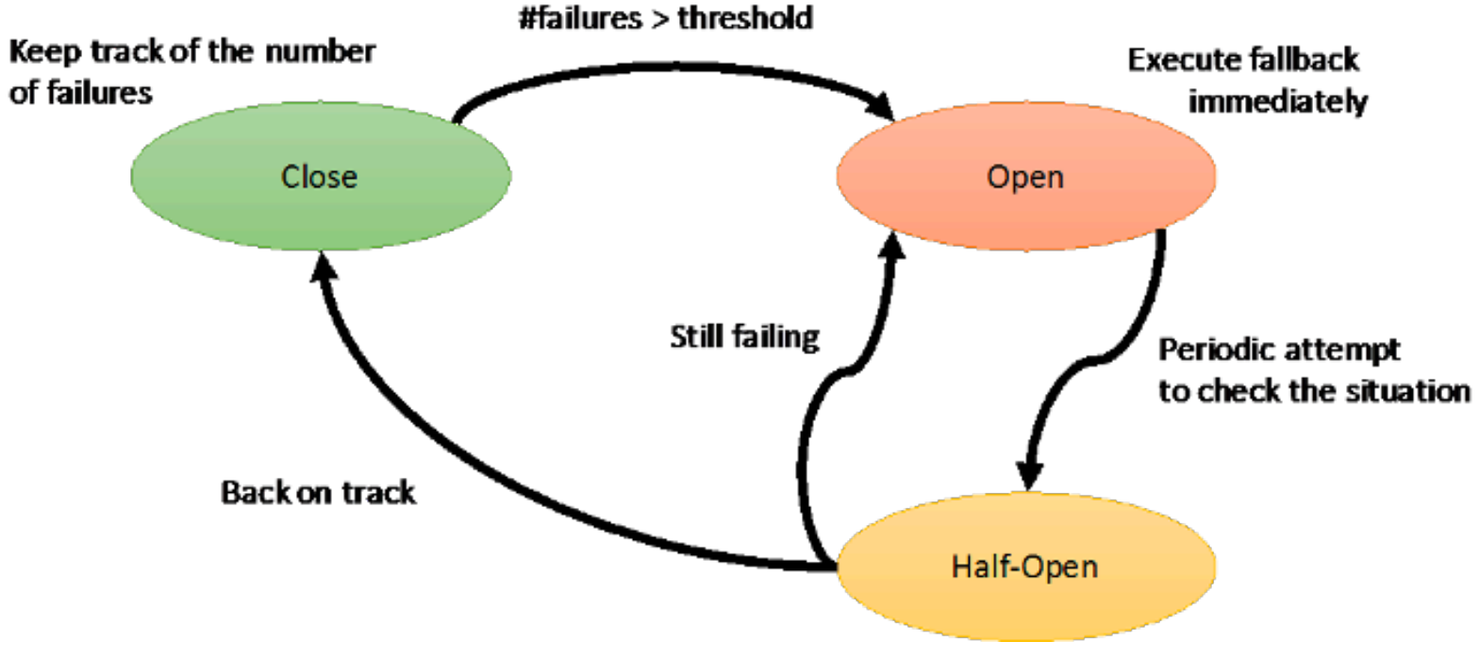
Network timeout
Limit request from client to service
Circuit breaker pattern

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Circuit Breaker pattern

Track the number of success and failure

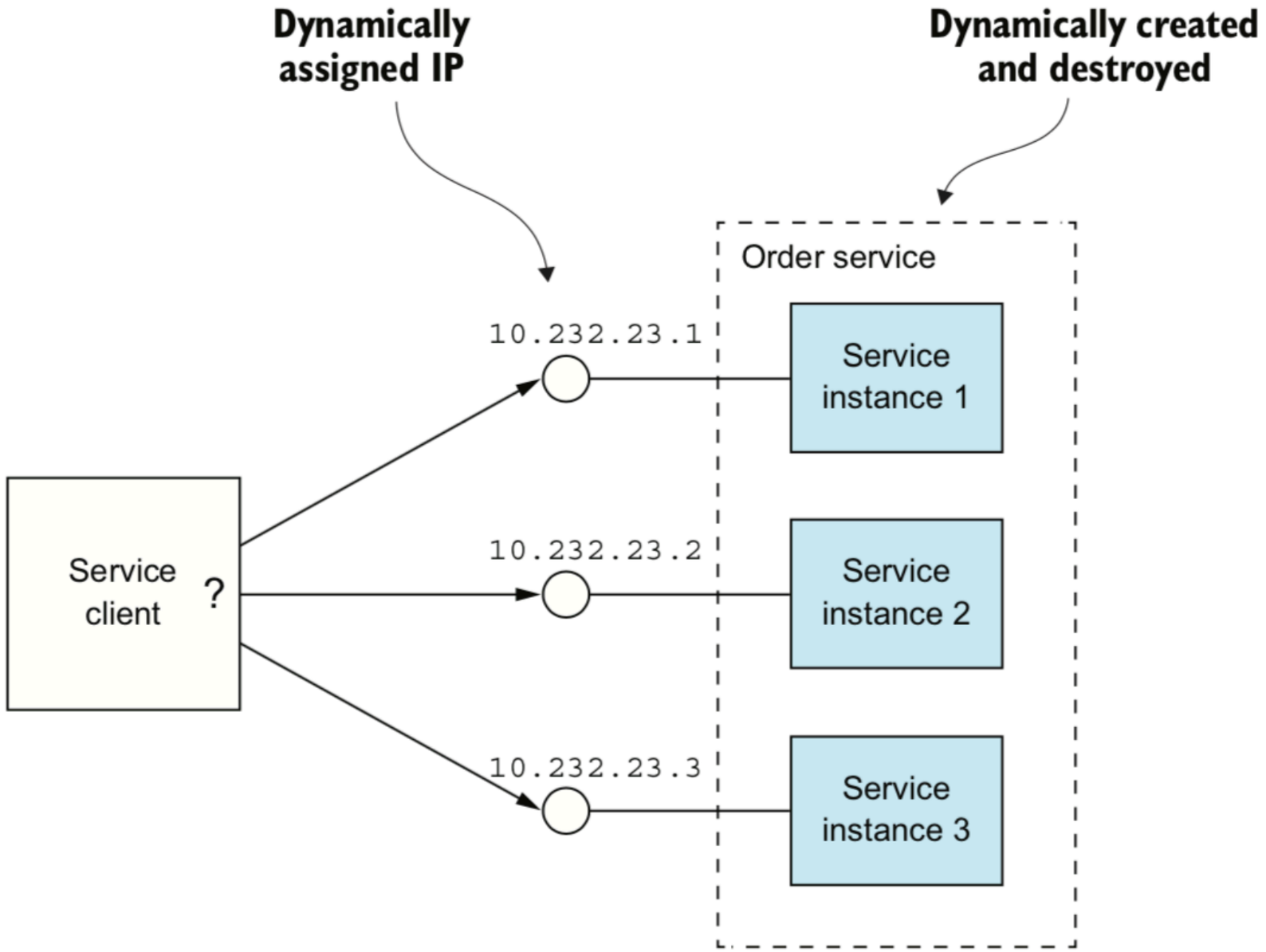
If error rate exceed some threshold then enable circuits breaker



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

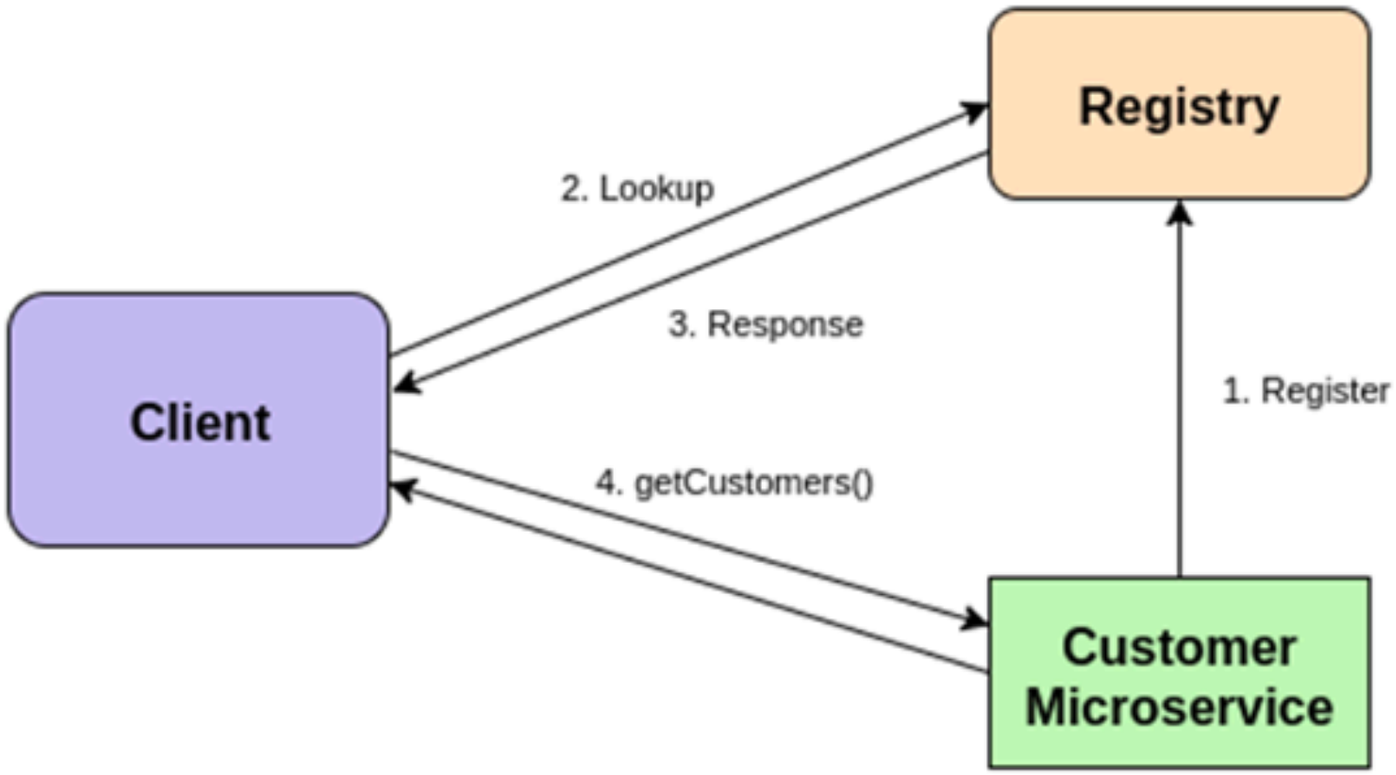
Service Discovery

How to call another services ?



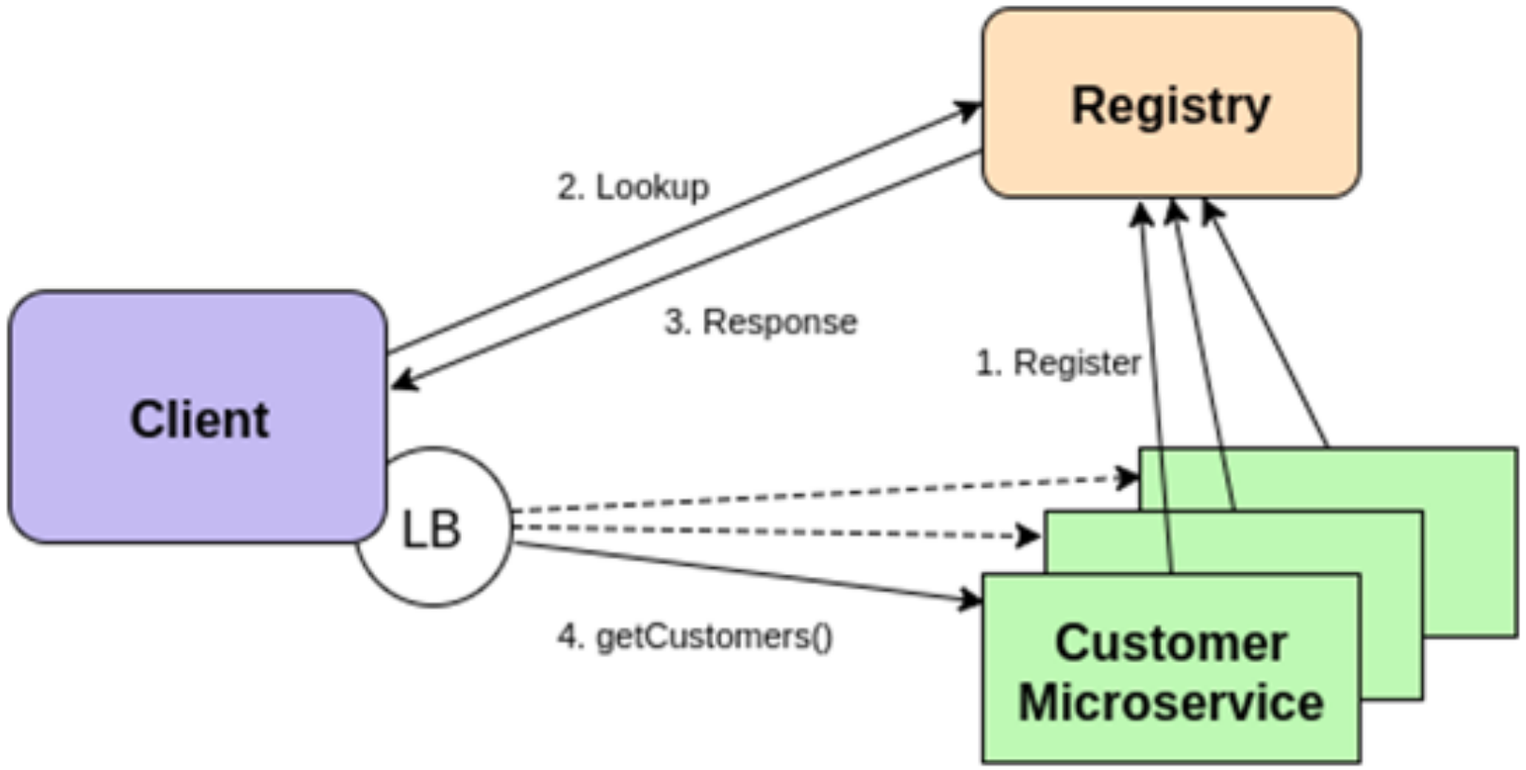
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service Discovery



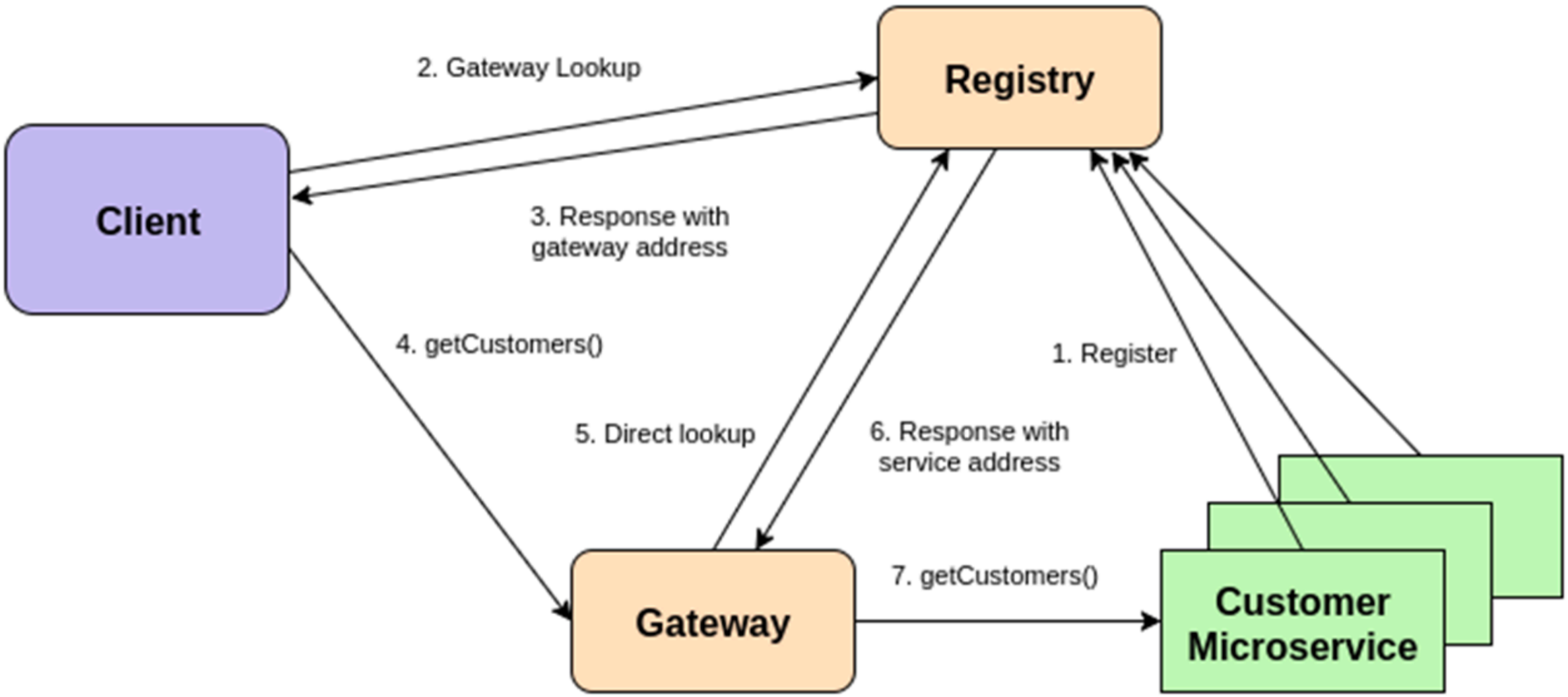
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service discovery with Load balance



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service discovery with API gateway

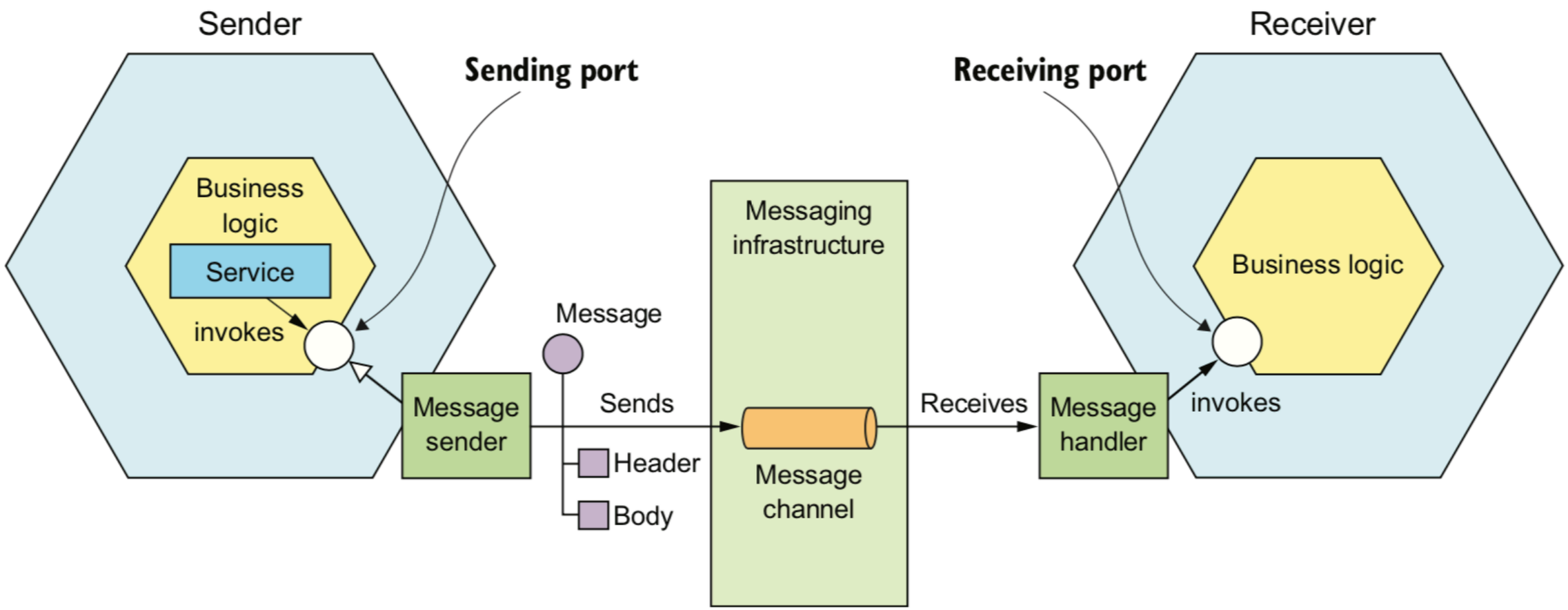


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Asynchronous messaging pattern

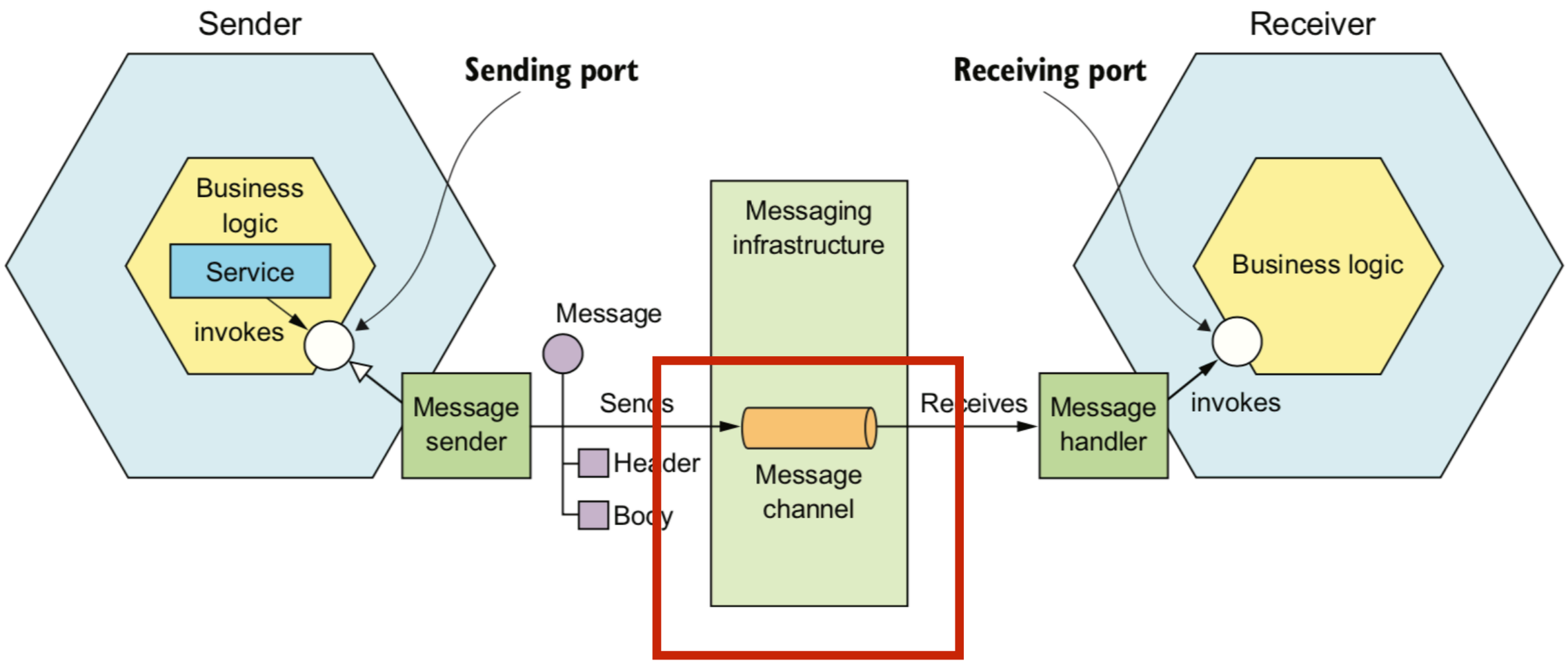
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Asynchronous messaging pattern



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

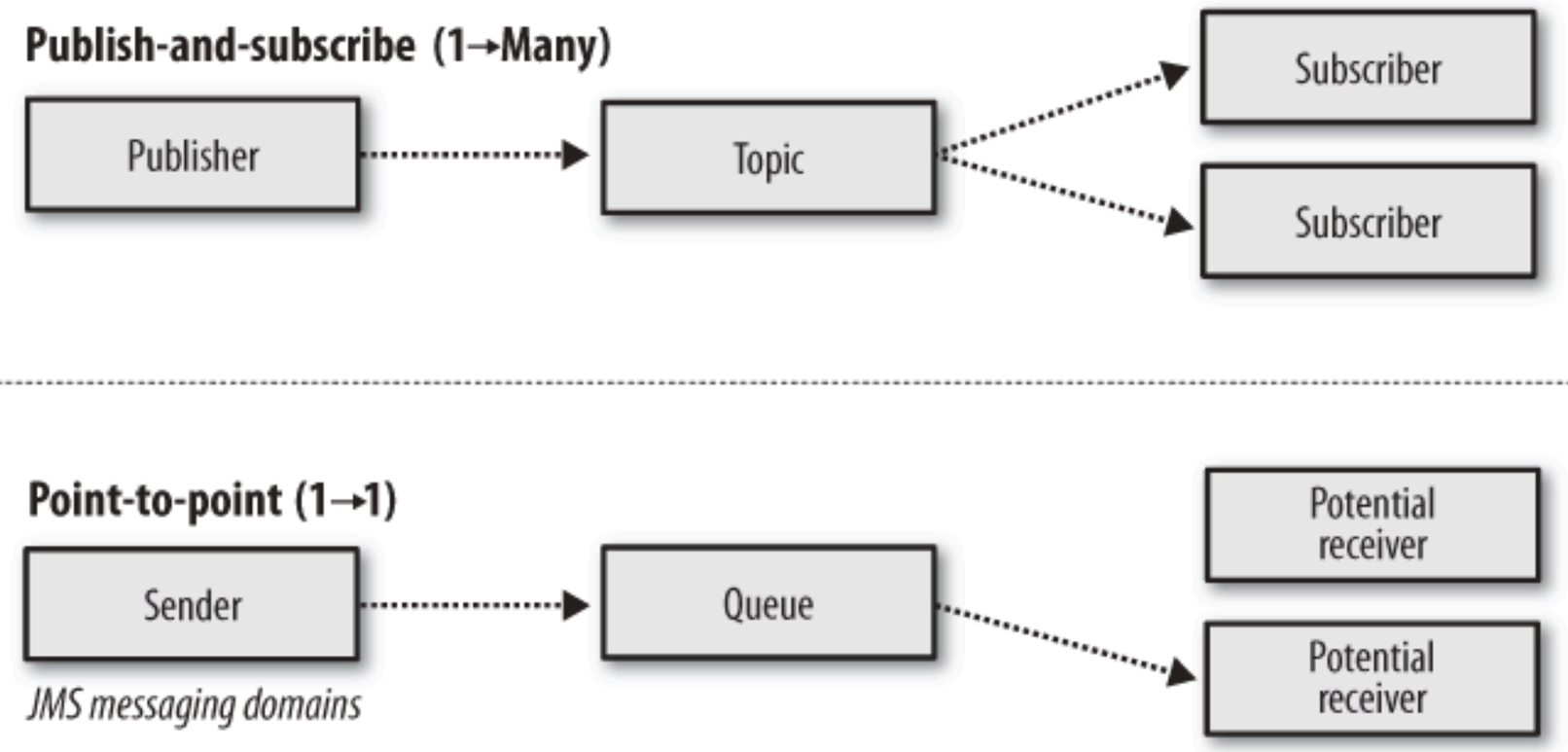
Asynchronous messaging pattern



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

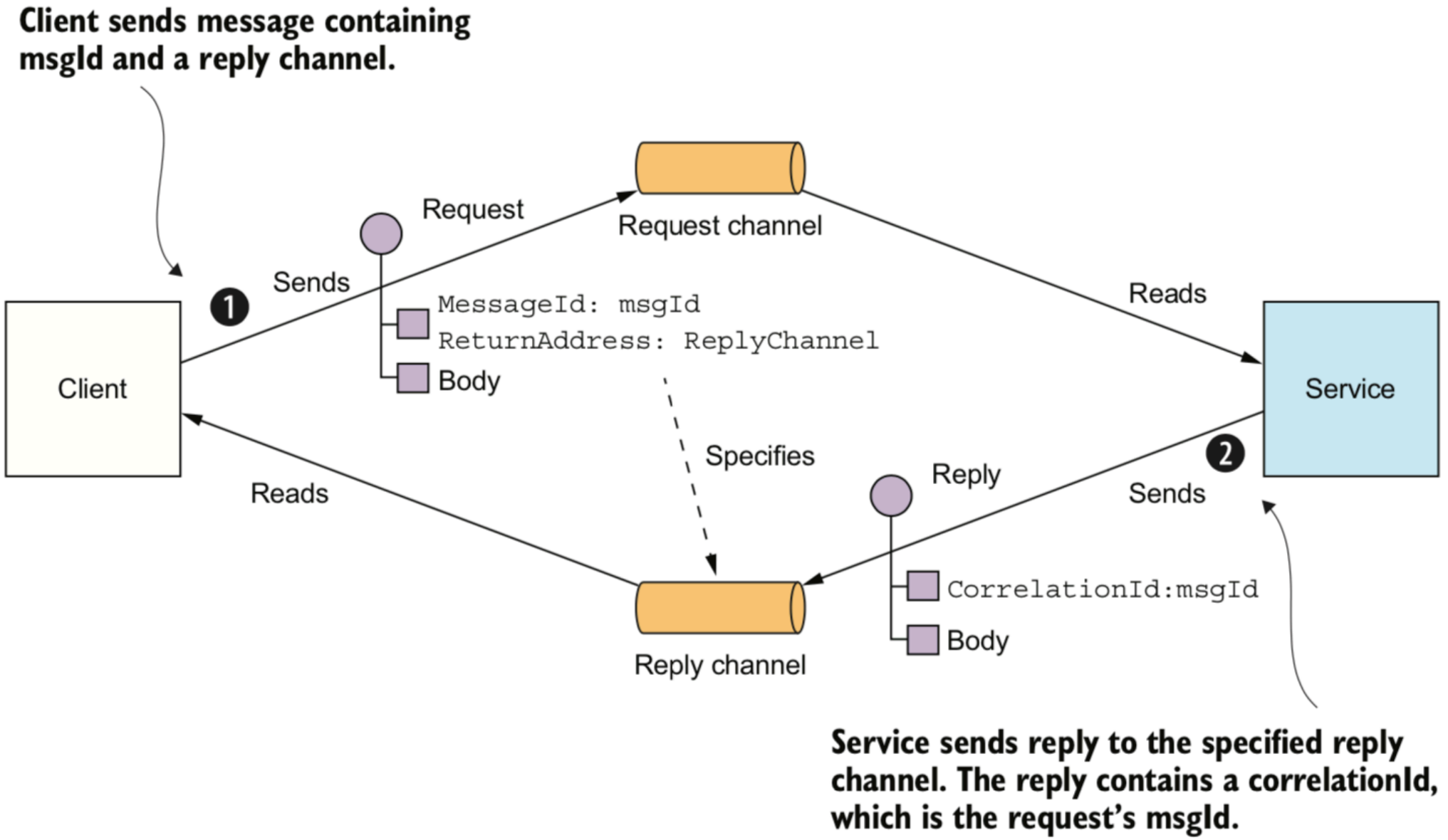
Messaging channels

Point-to-point
Publish-subscribe



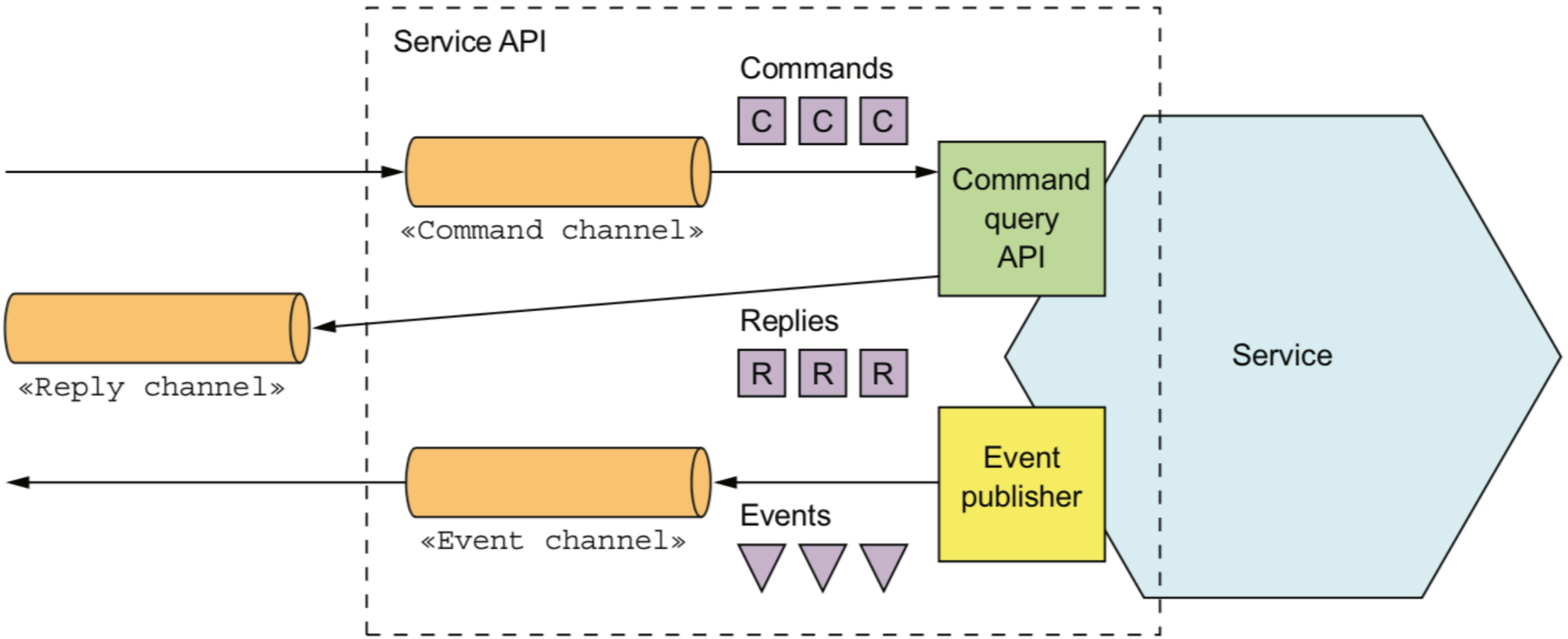
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Async request/response (1)



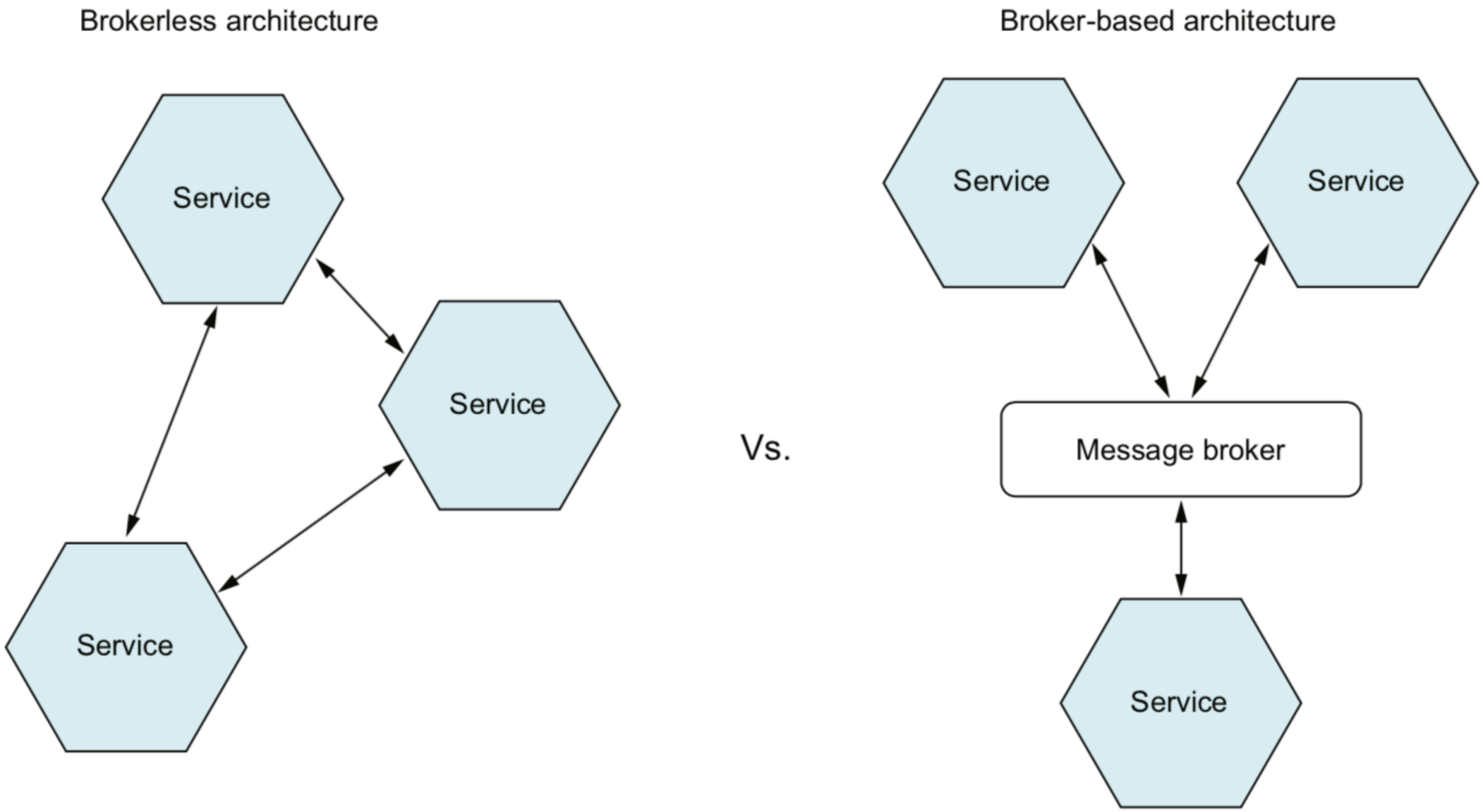
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Async request/response (2)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Messaging-based use message broker



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Message brokers

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Benefits

Loose-coupling
Message buffer
Flexible communication

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks

Performance bottleneck
Single point of failure
Operational complexity

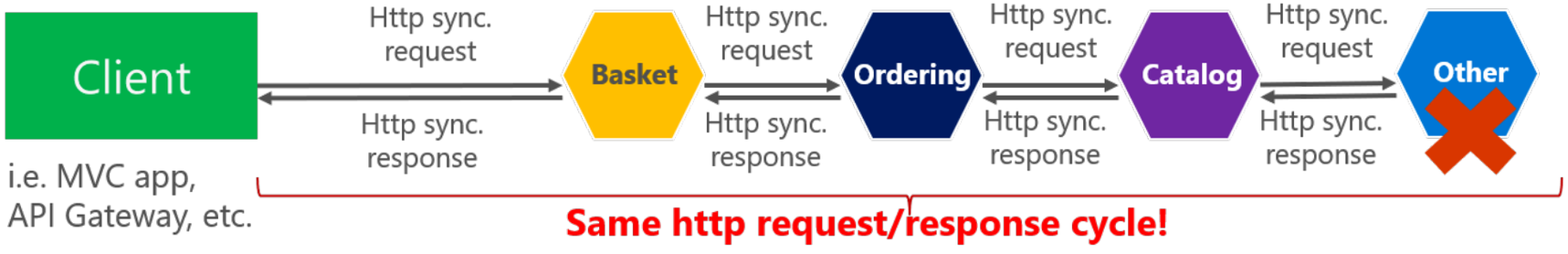
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Communication

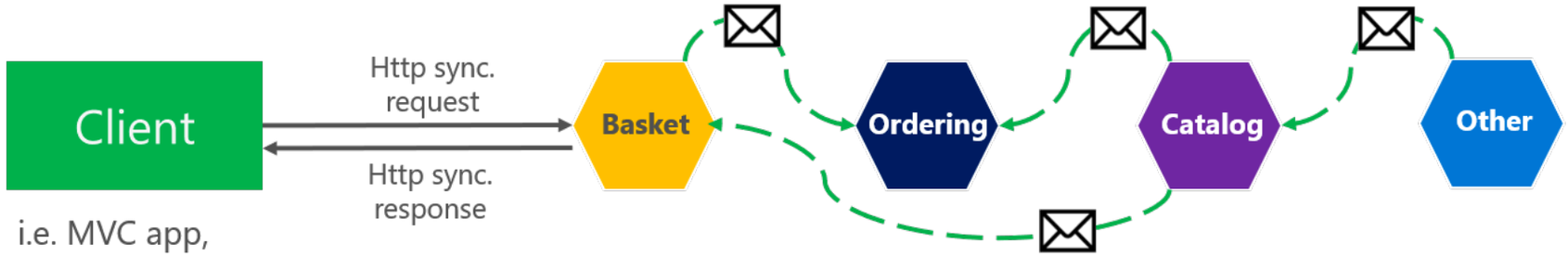
Synchronous vs. async communication across microservices

Anti-pattern

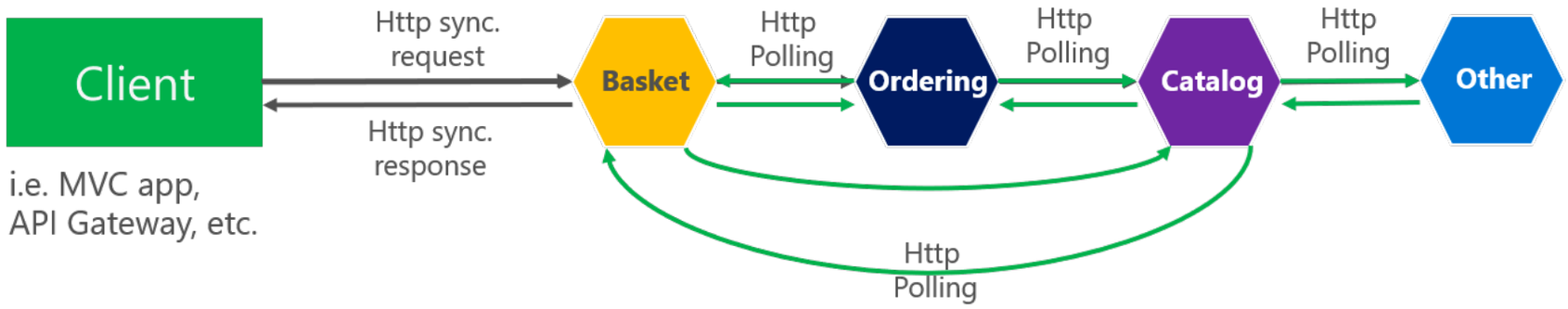
Synchronous
all req./resp. cycle



Asynchronous
Comm. across
internal microservices
(EventBus: i.e. **AMPQ**)

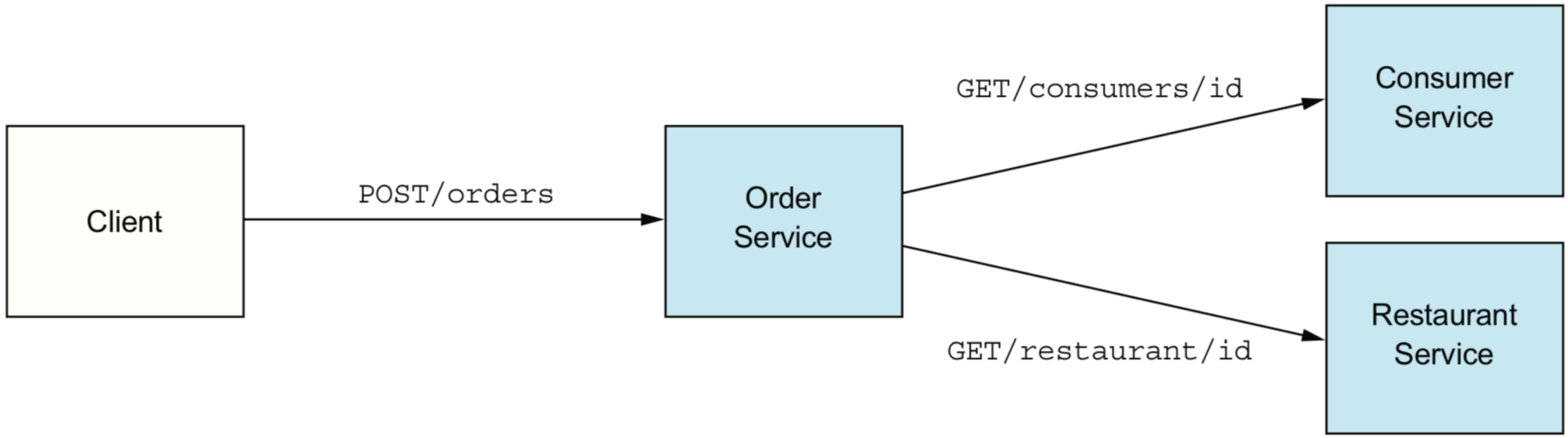


"Asynchronous"
Comm. across
internal microservices
(Polling: **Http**)



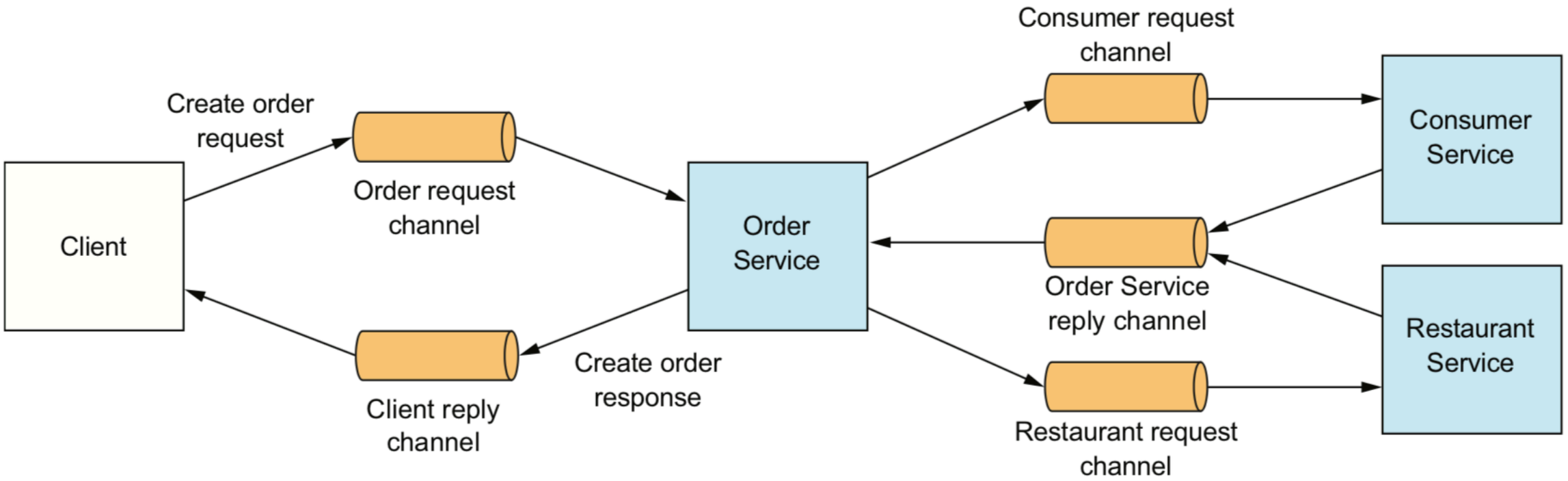
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Synchronous reduce availability



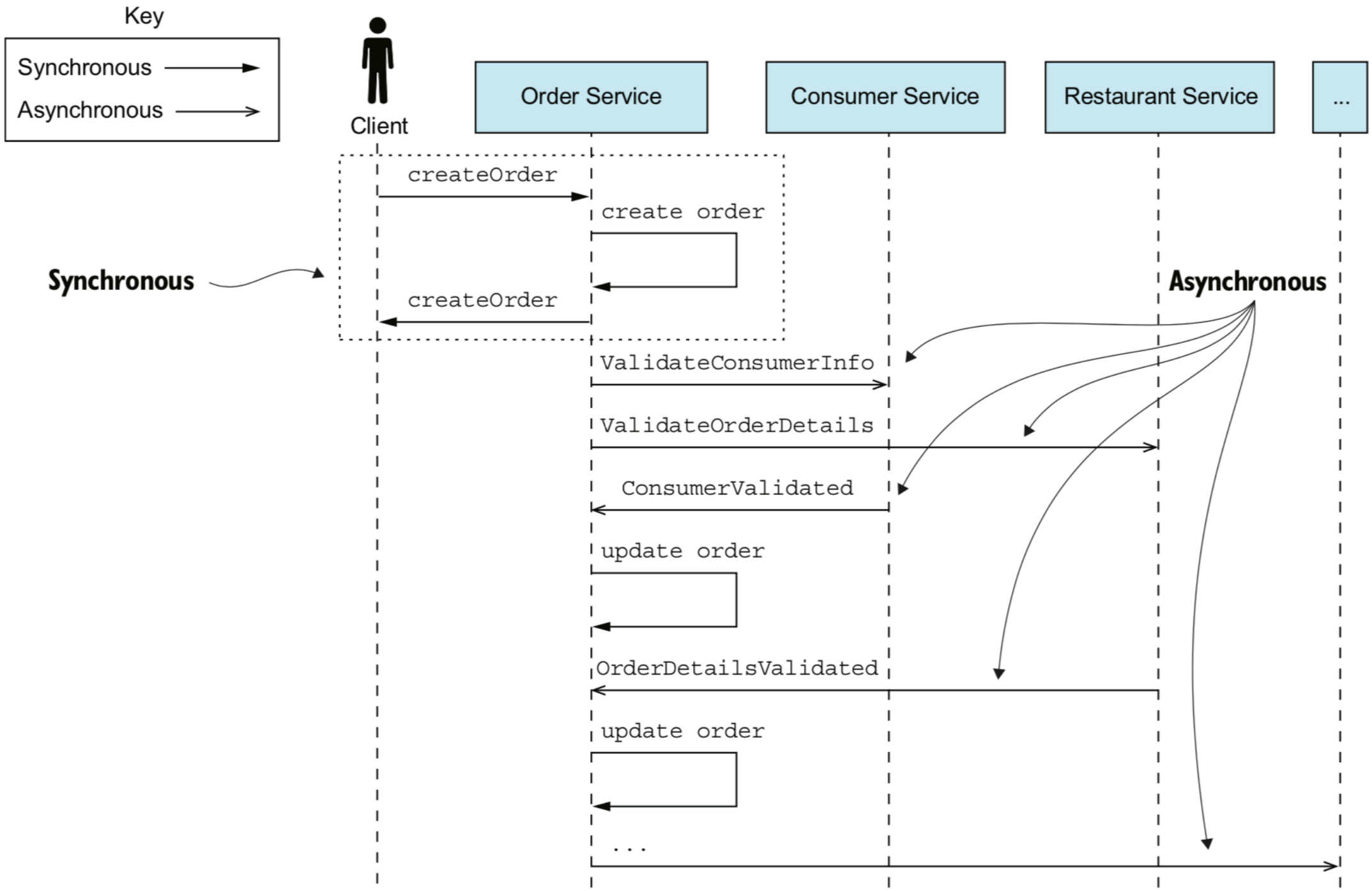
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Replace with asynchronous (1)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Replace with asynchronous (2)

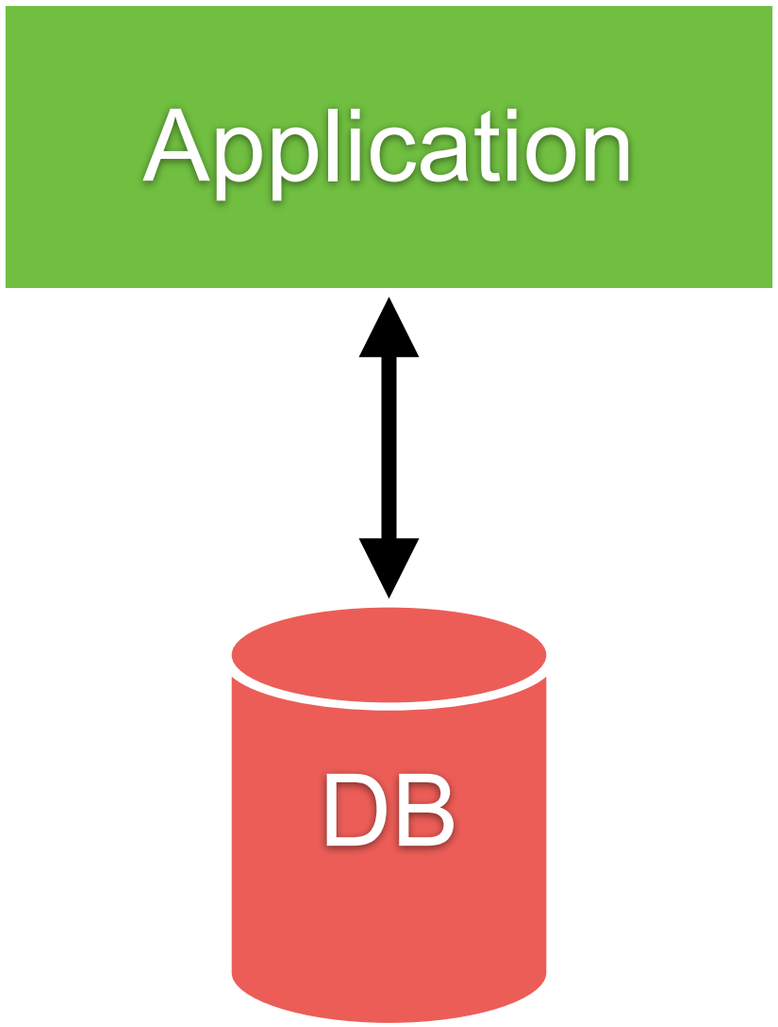


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Manage data consistency

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

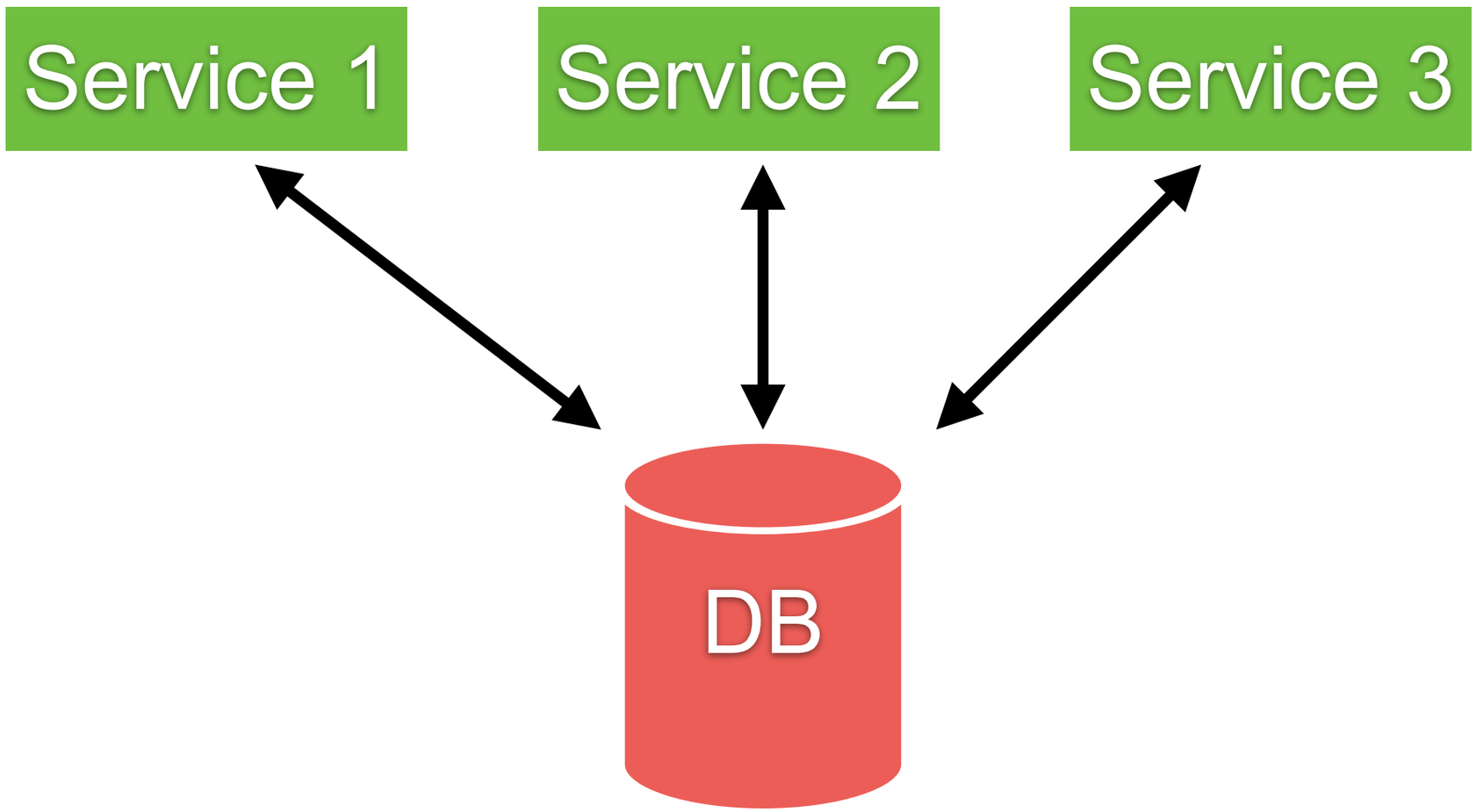
Monolith



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservices

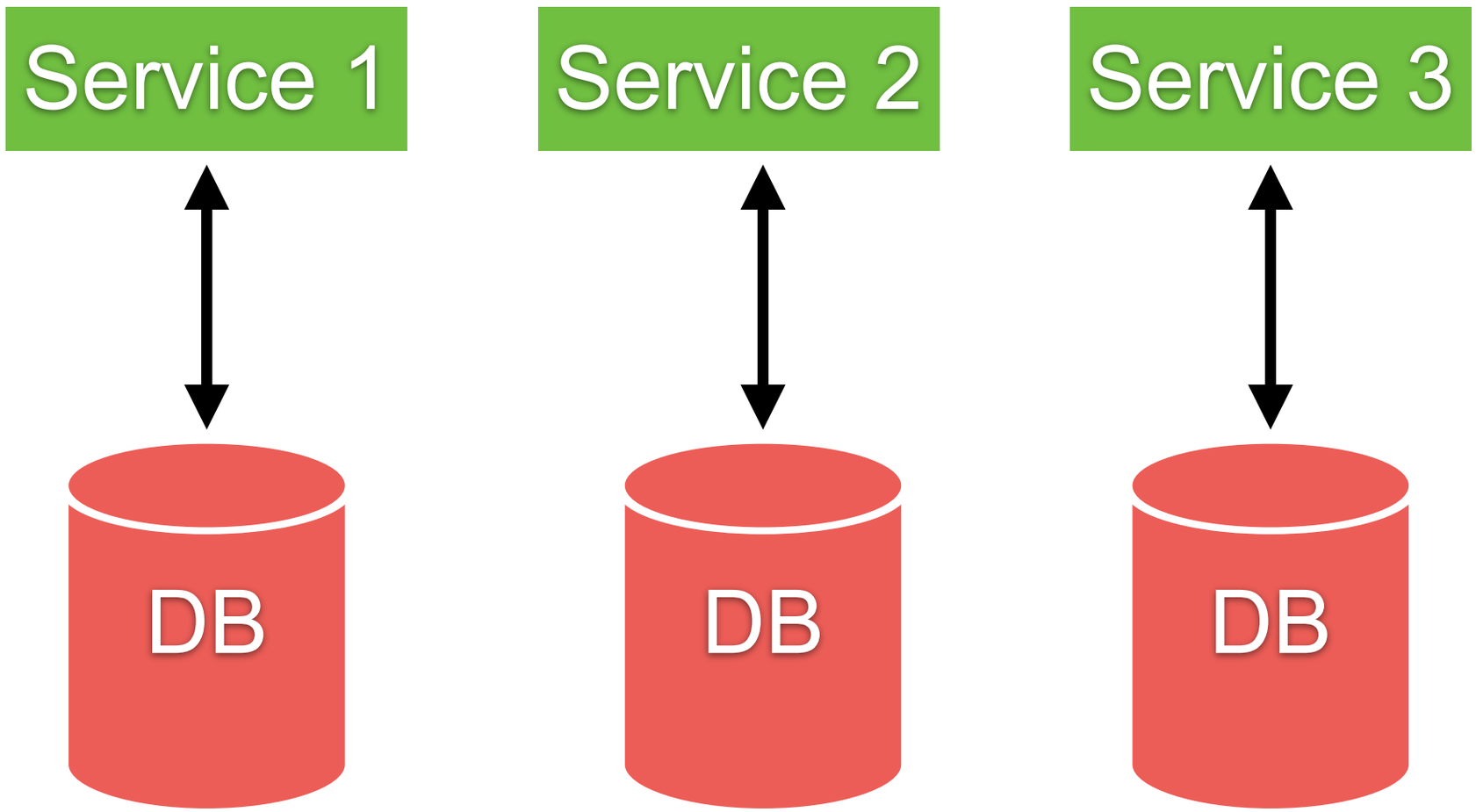
Shared database



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

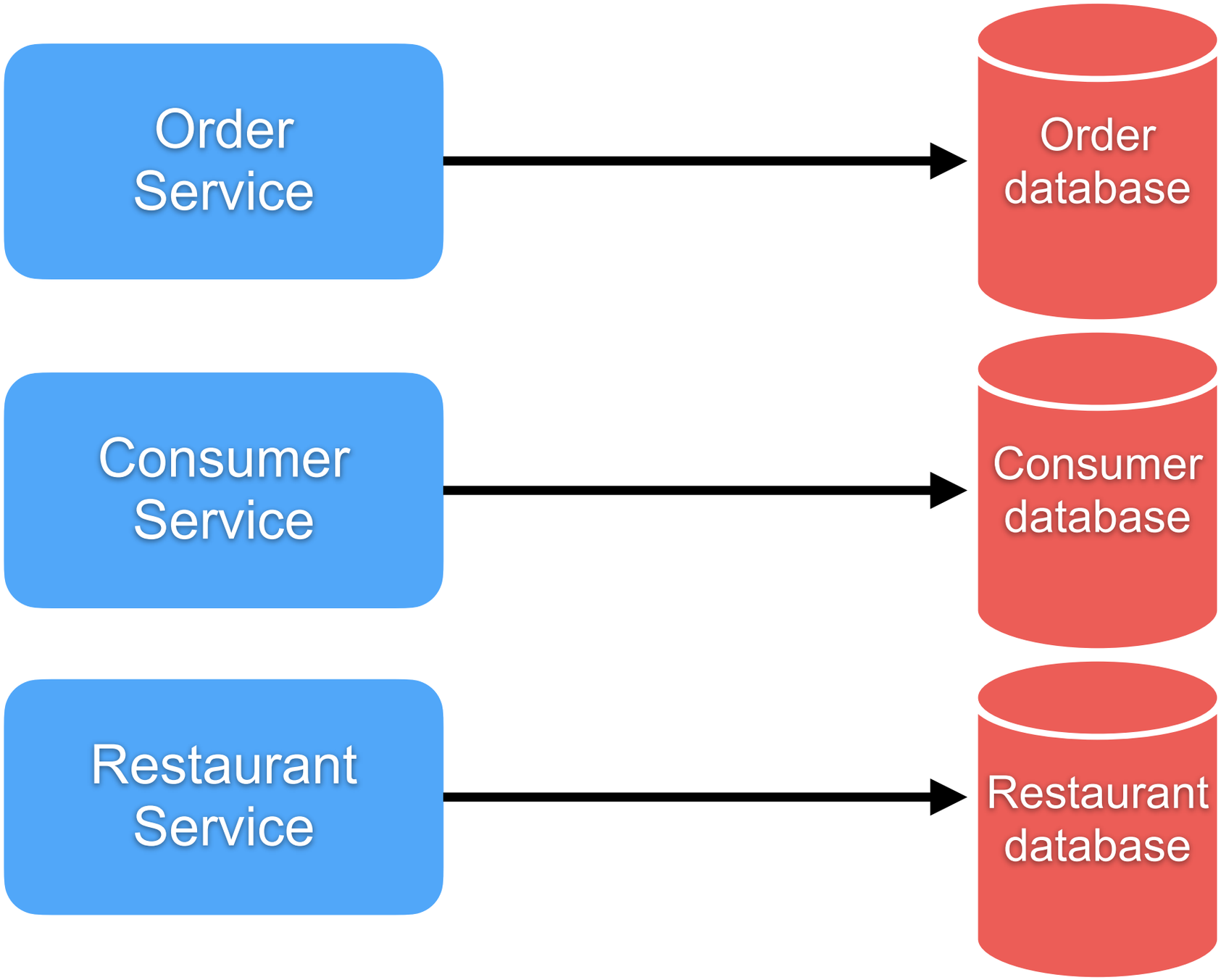
Microservices

Database per service



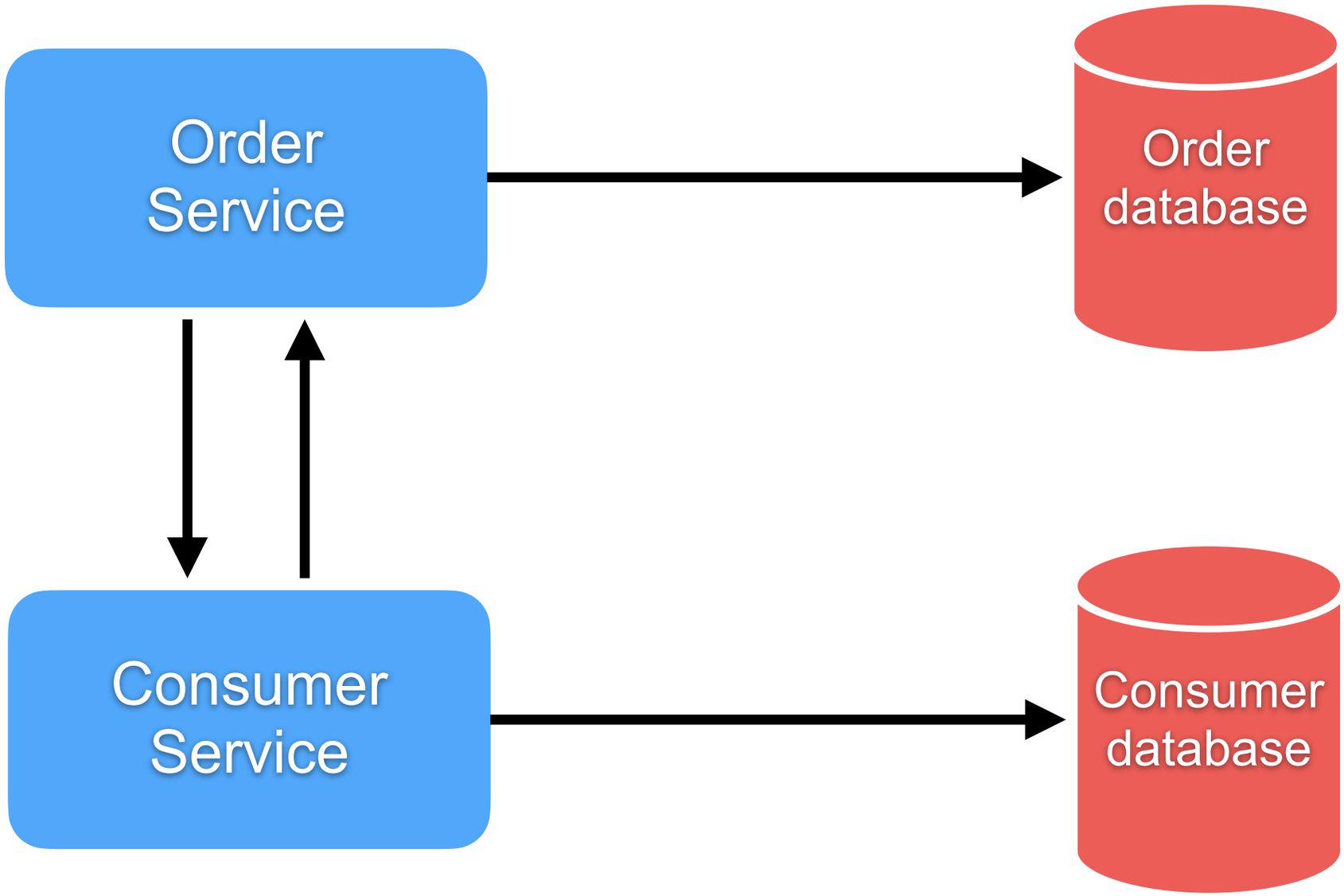
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Database per service

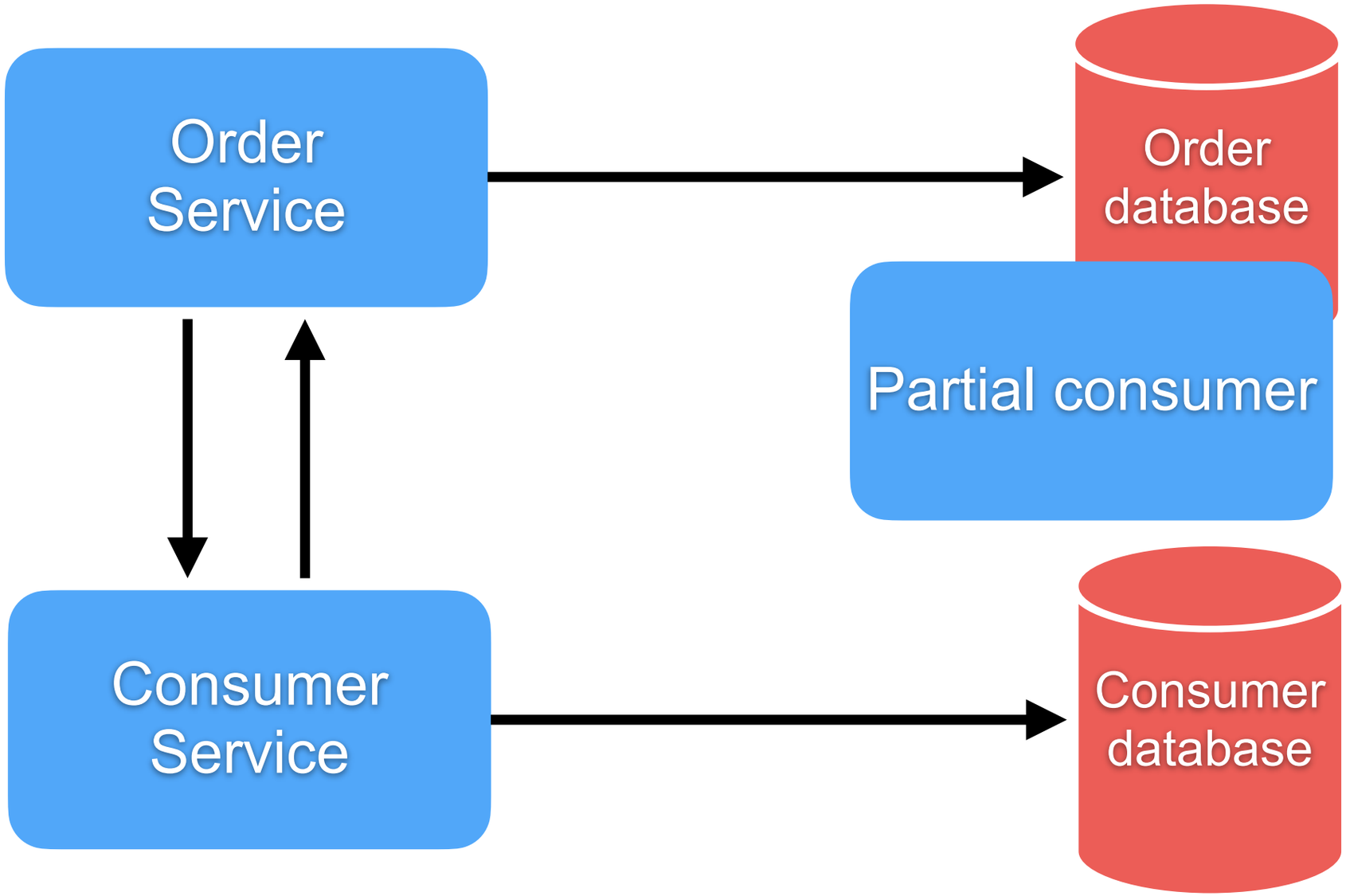


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

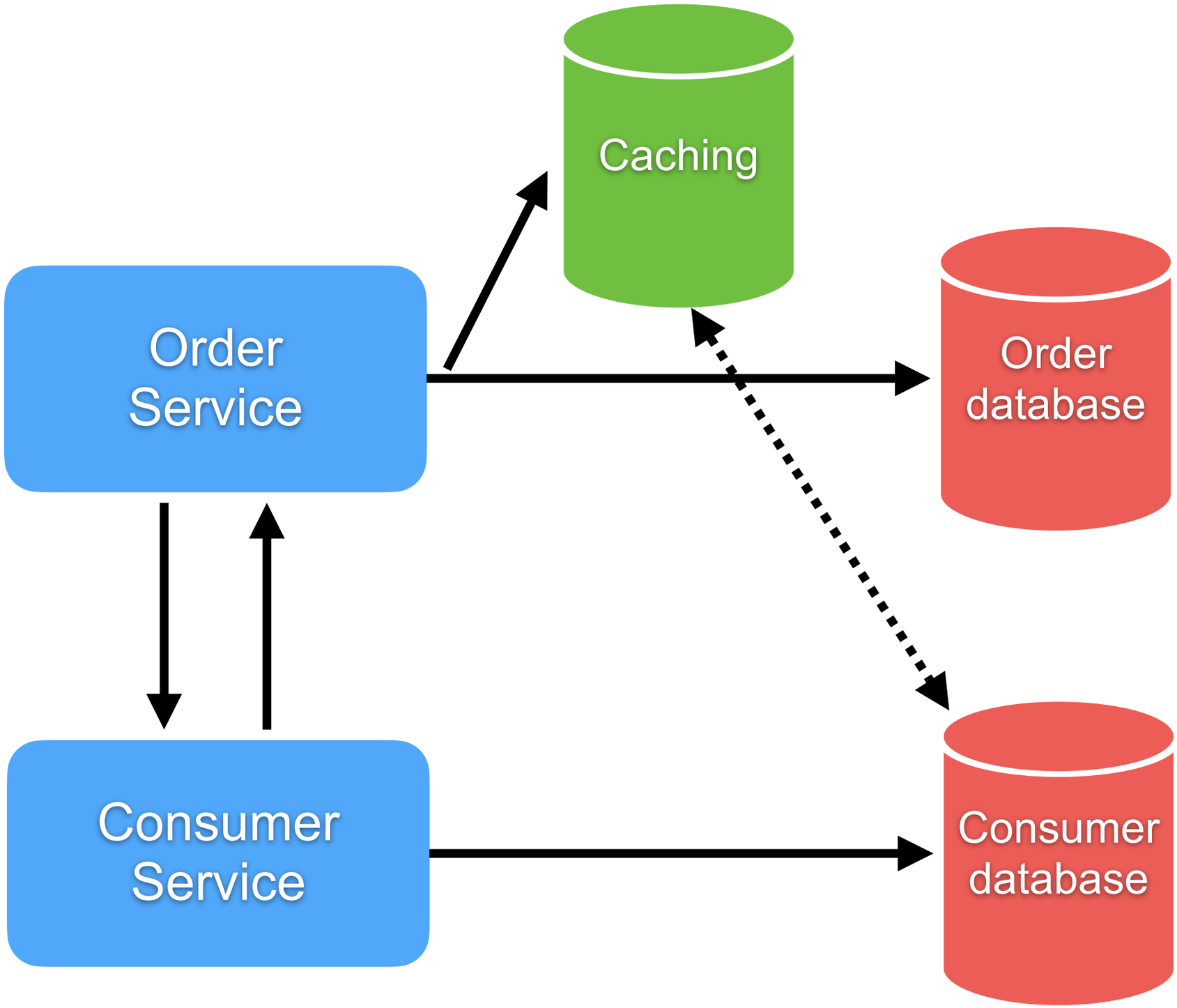
Loose coupling = encapsulation data



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

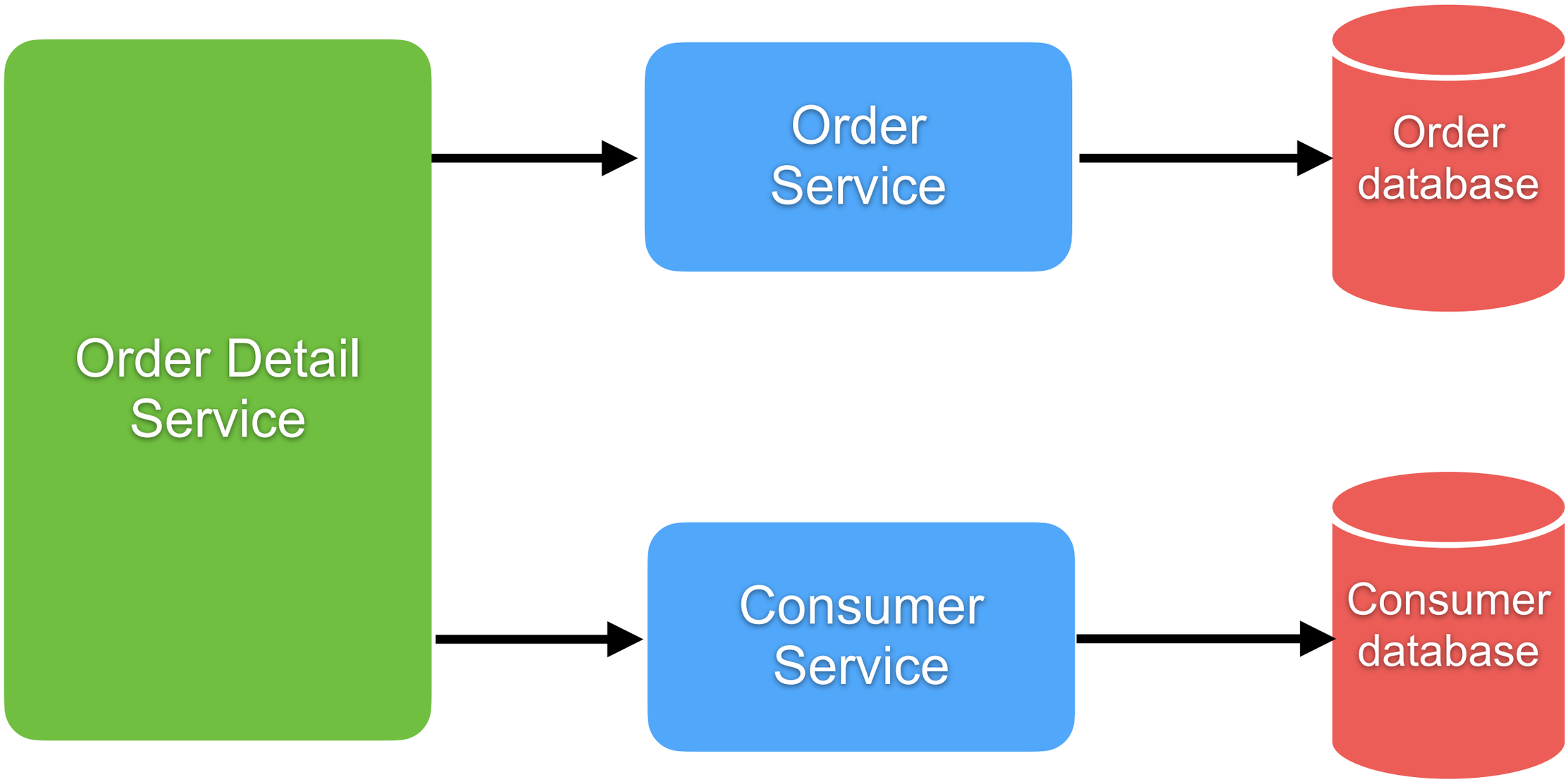


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



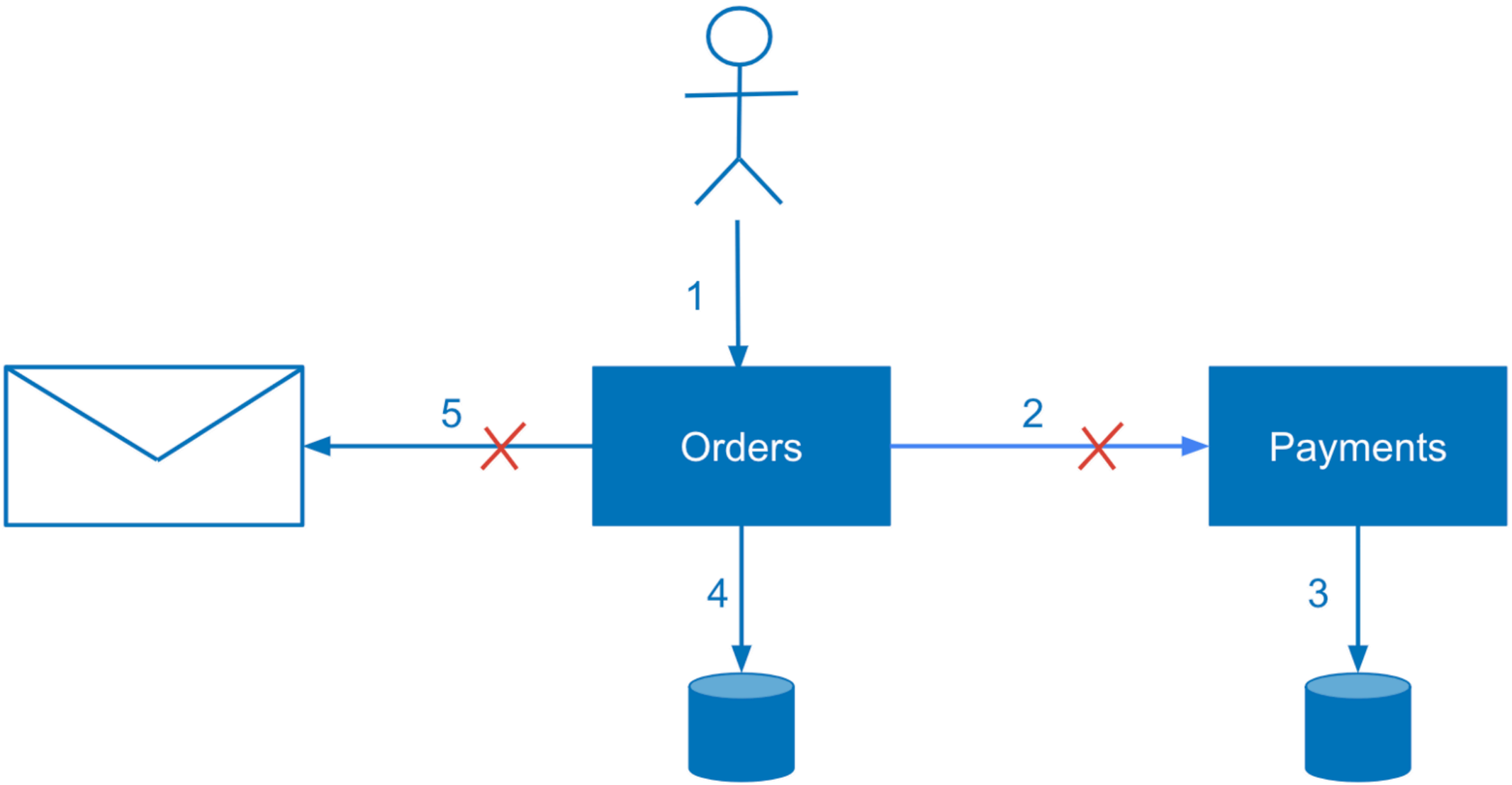
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Order detail orchestration service



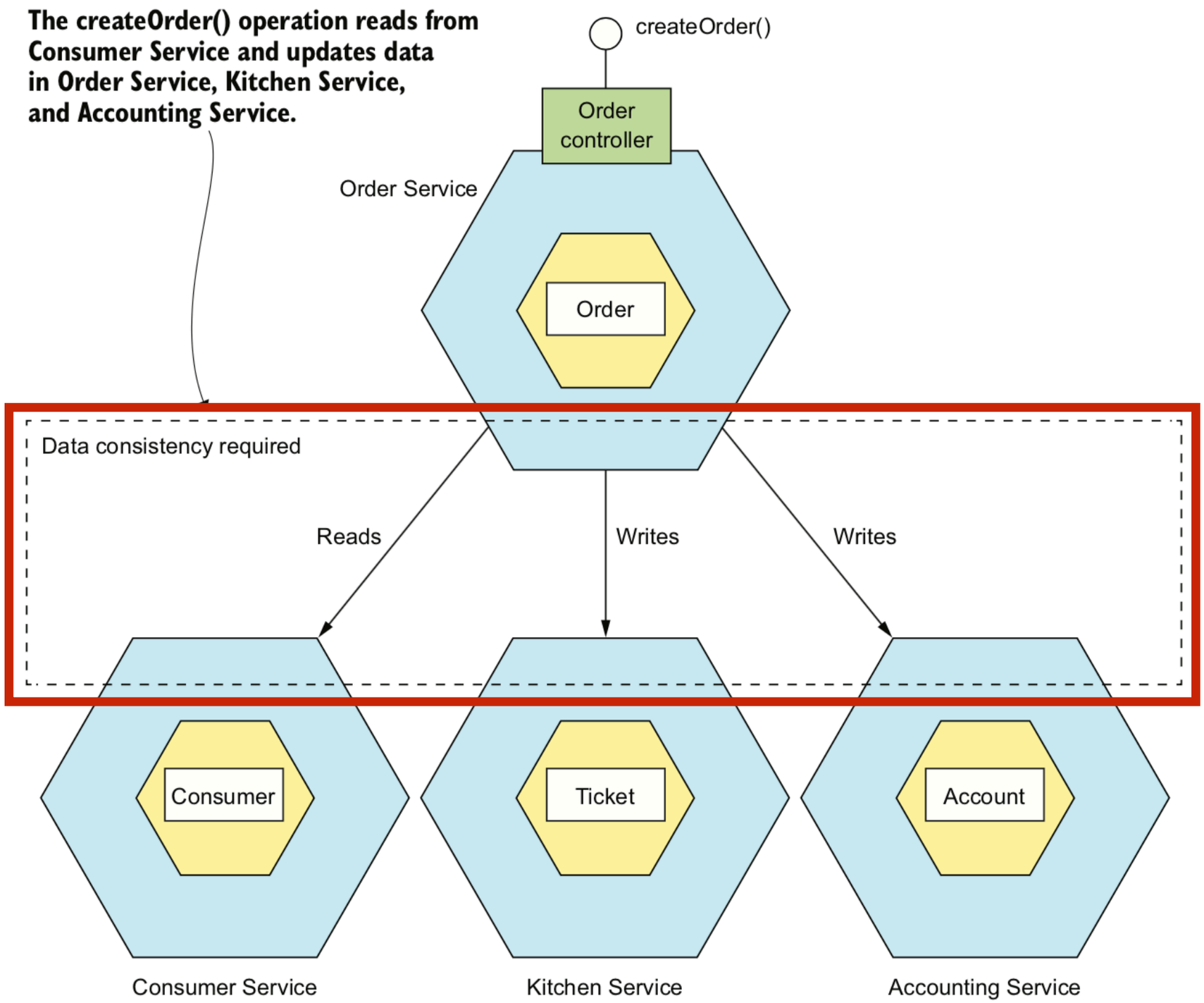
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Distributed process failure !!



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Problem ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Can’t use ACID transaction !!

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

A - Atomicity

All or Nothing Transactions

C - Consistency

Guarantees Committed Transaction State

I - Isolation

Transactions are Independent

D – Durability

Committed Data is Never Lost

(c) http://blog.sqlauthority.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

CAP Theorem

Consistency

Availability

Partition Tolerance

http://robertgreiner.com/2014/08/cap-theorem-revisited/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Database per service

High complexity

Query data across multiple databases

Challenge “How to join data ?”

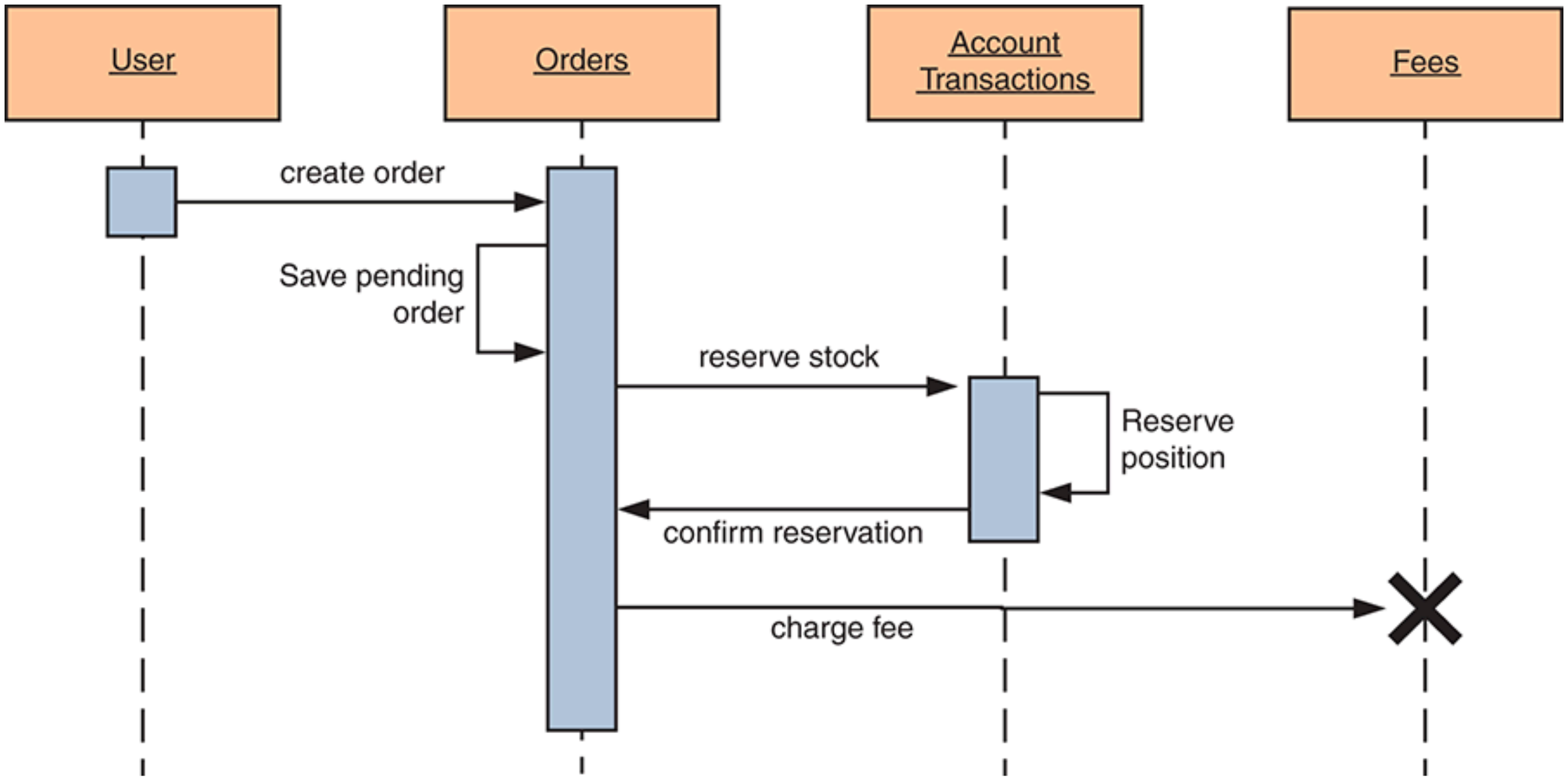
Maintain database consistency

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

How to maintain data consistency across services ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Problem



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Distributed transaction

Common approach is Two-phase commit(2PC)

Use transaction manager to split operations across multiple resources in 2 phases

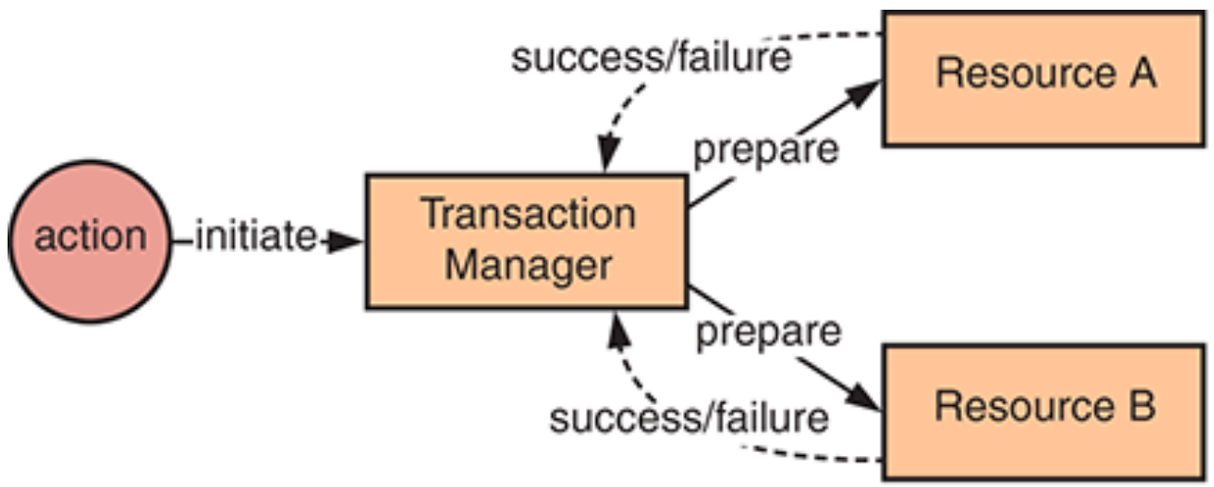
1. Prepare

2. Commit

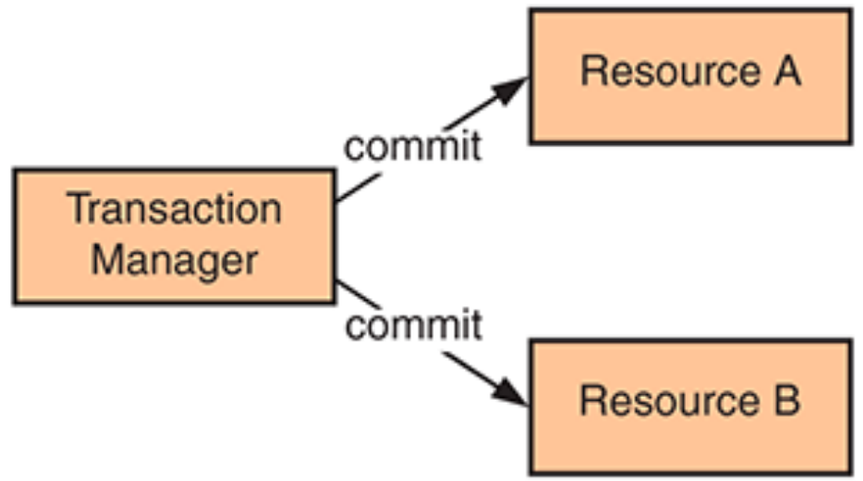
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Two-phase commit

Phase 1: prepare



In the **prepare** phase, the transaction manager instructs resources to prepare their relevant action.

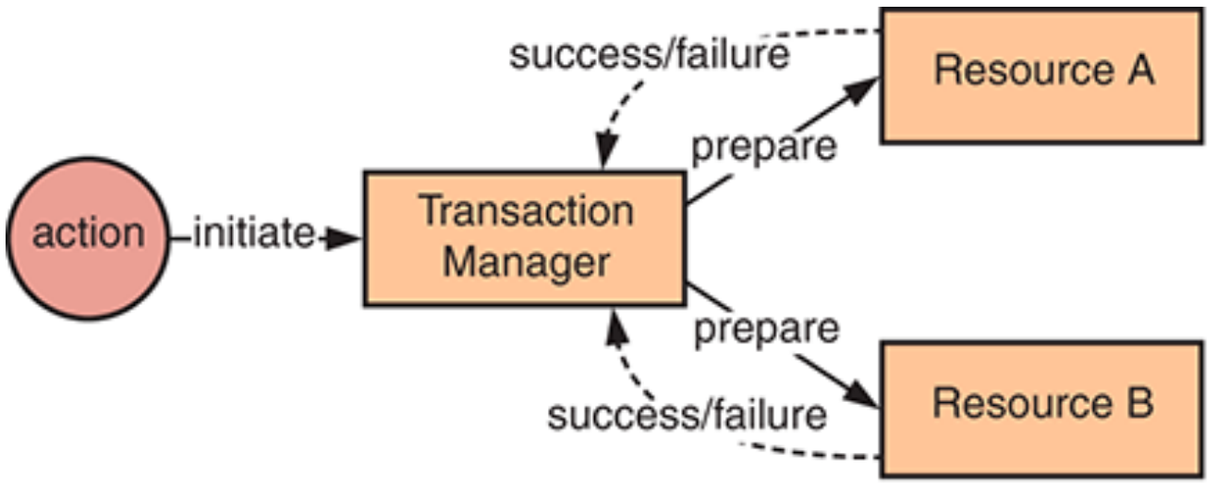


In the **commit** phase, the transaction manager instructs resources to commit or abort the prepared action.

https://en.wikipedia.org/wiki/Two-phase_commit_protocol

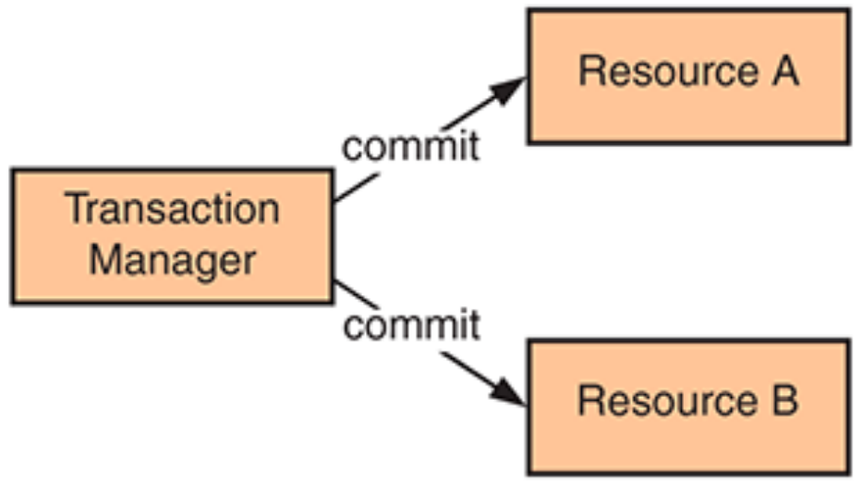
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Two-phase commit



In the **prepare** phase, the transaction manager instructs resources to prepare their relevant action.

Phase 2: commit



In the **commit** phase, the transaction manager instructs resources to commit or abort the prepared action.

https://en.wikipedia.org/wiki/Two-phase_commit_protocol

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Distributed transaction places a lock on resources under transaction to ensure isolation

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Inappropriate for long-running operations

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Increase risk of contention and deadlock

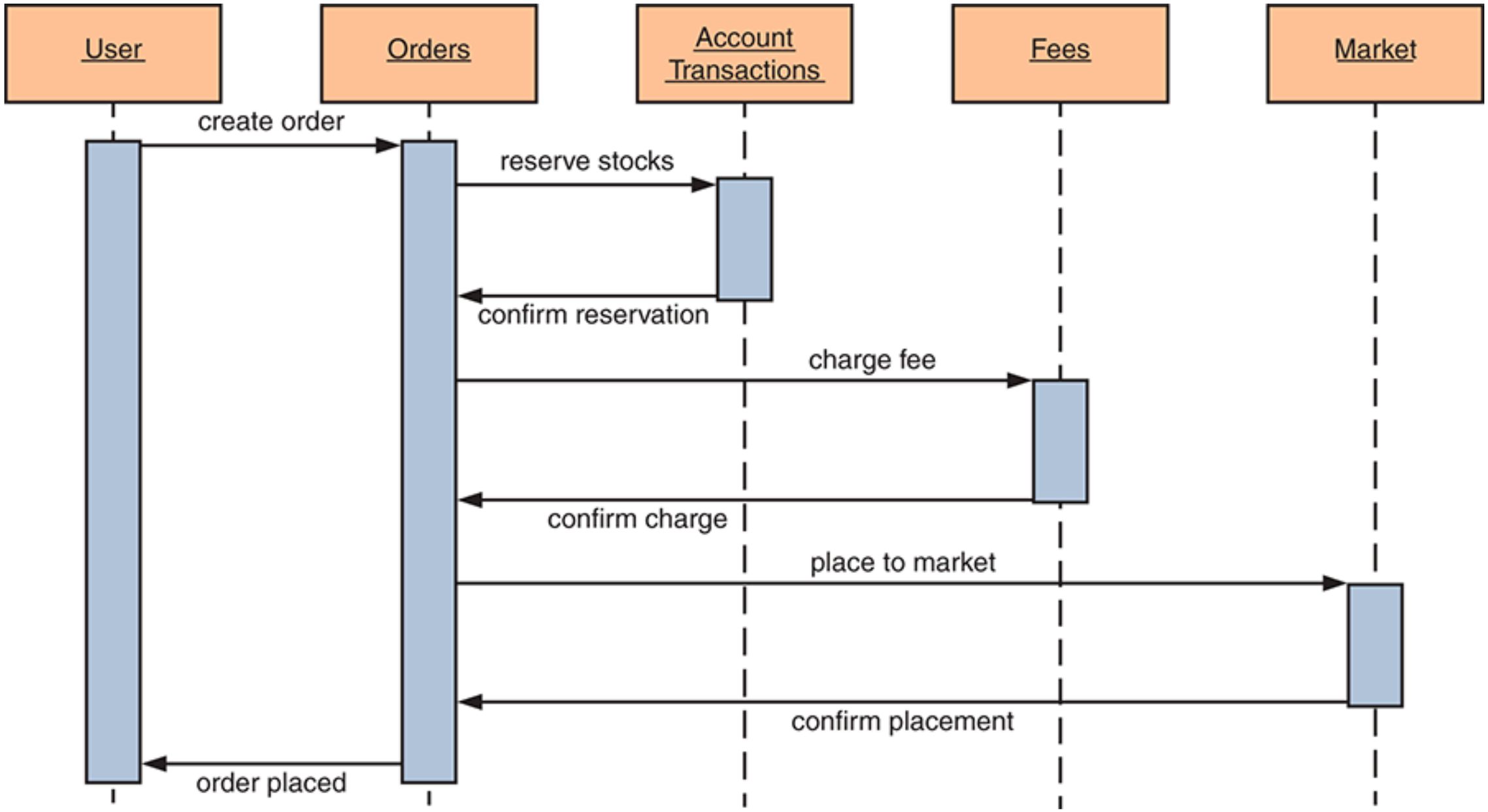
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

We need …

Don’t need distributed transaction
Reduce complexity of code
Reliable

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Synchronous process



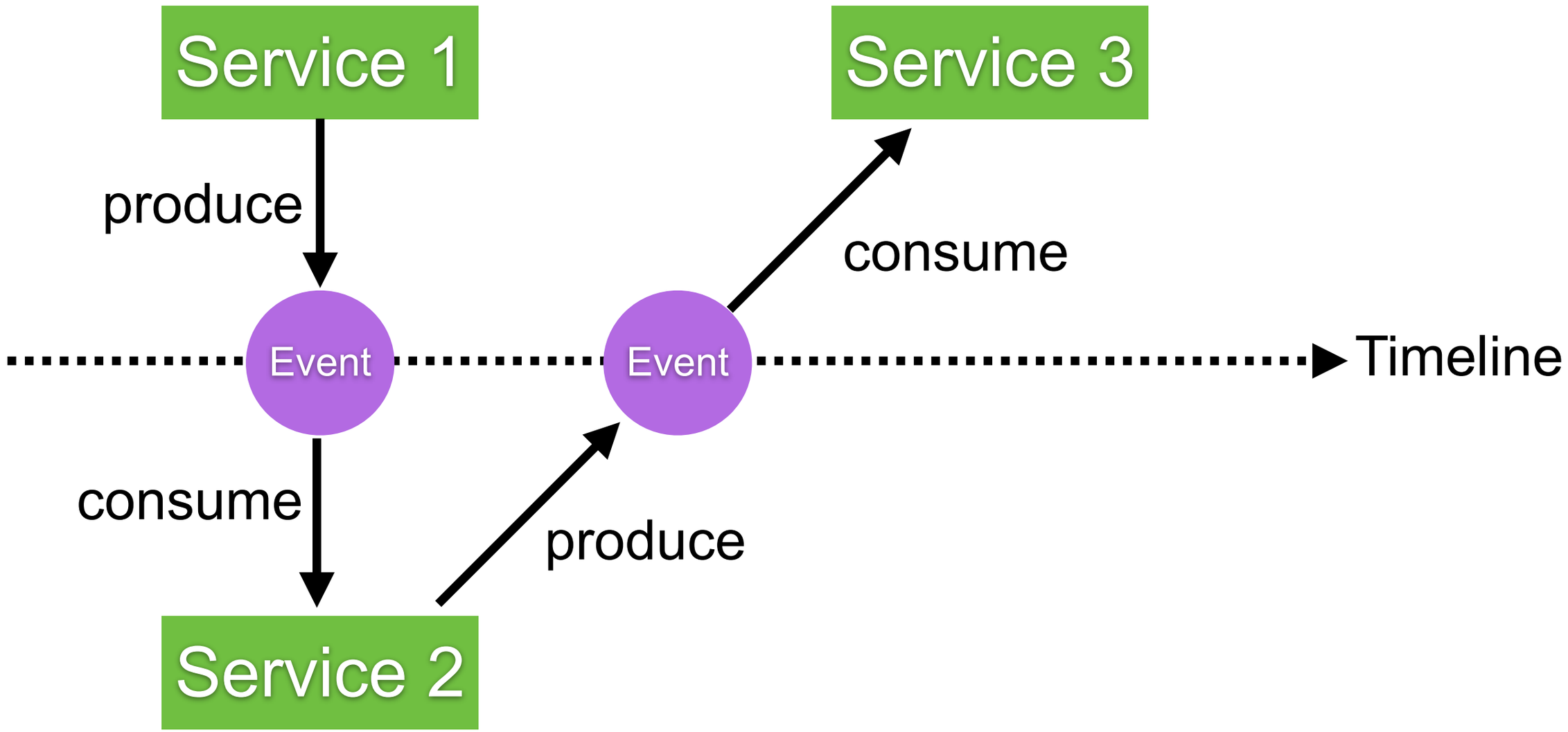
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event-based communication

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

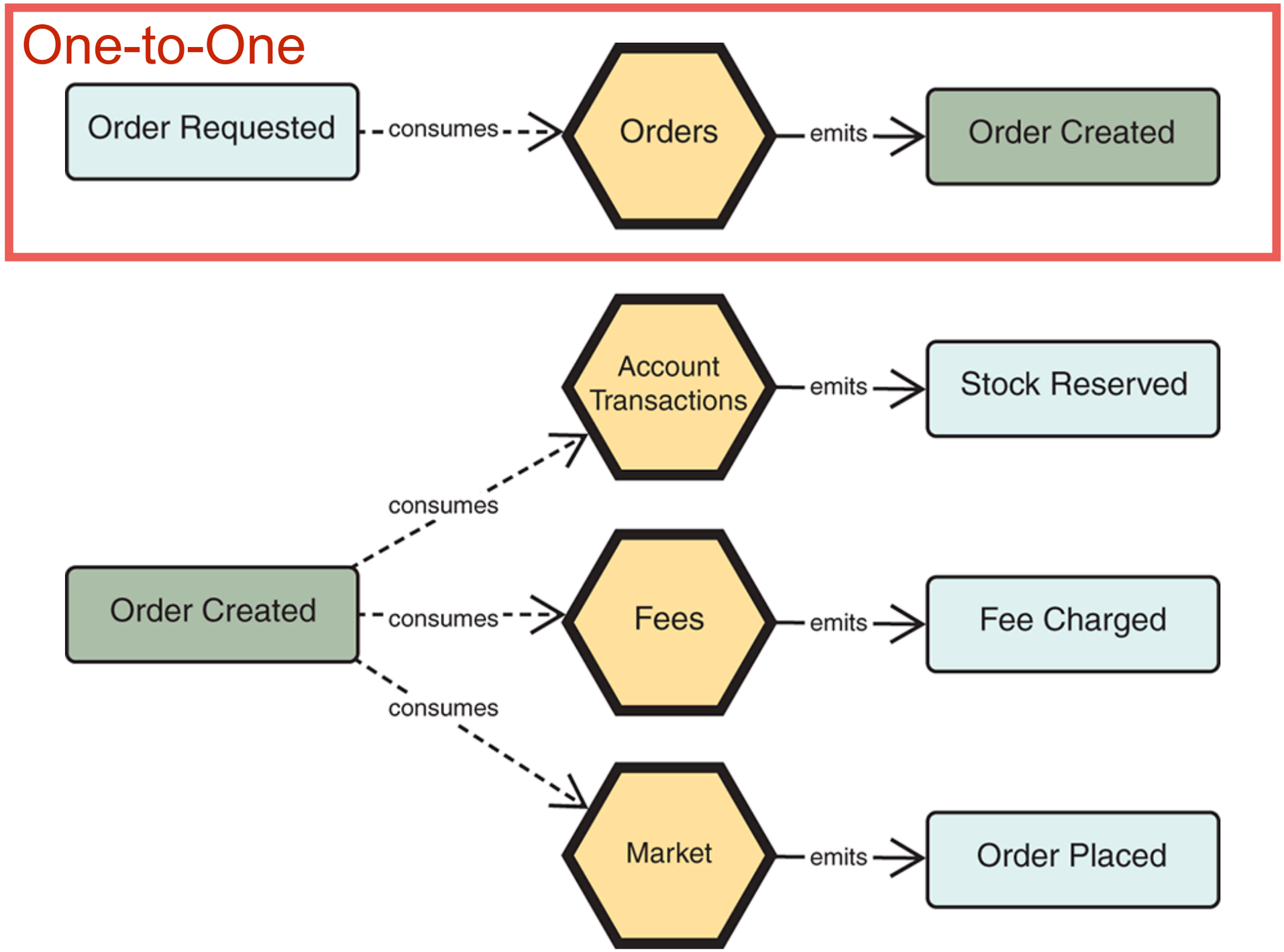
Choreography pattern

Sequence of Tx and emit **event** or **message** that trigger the next process in Tx



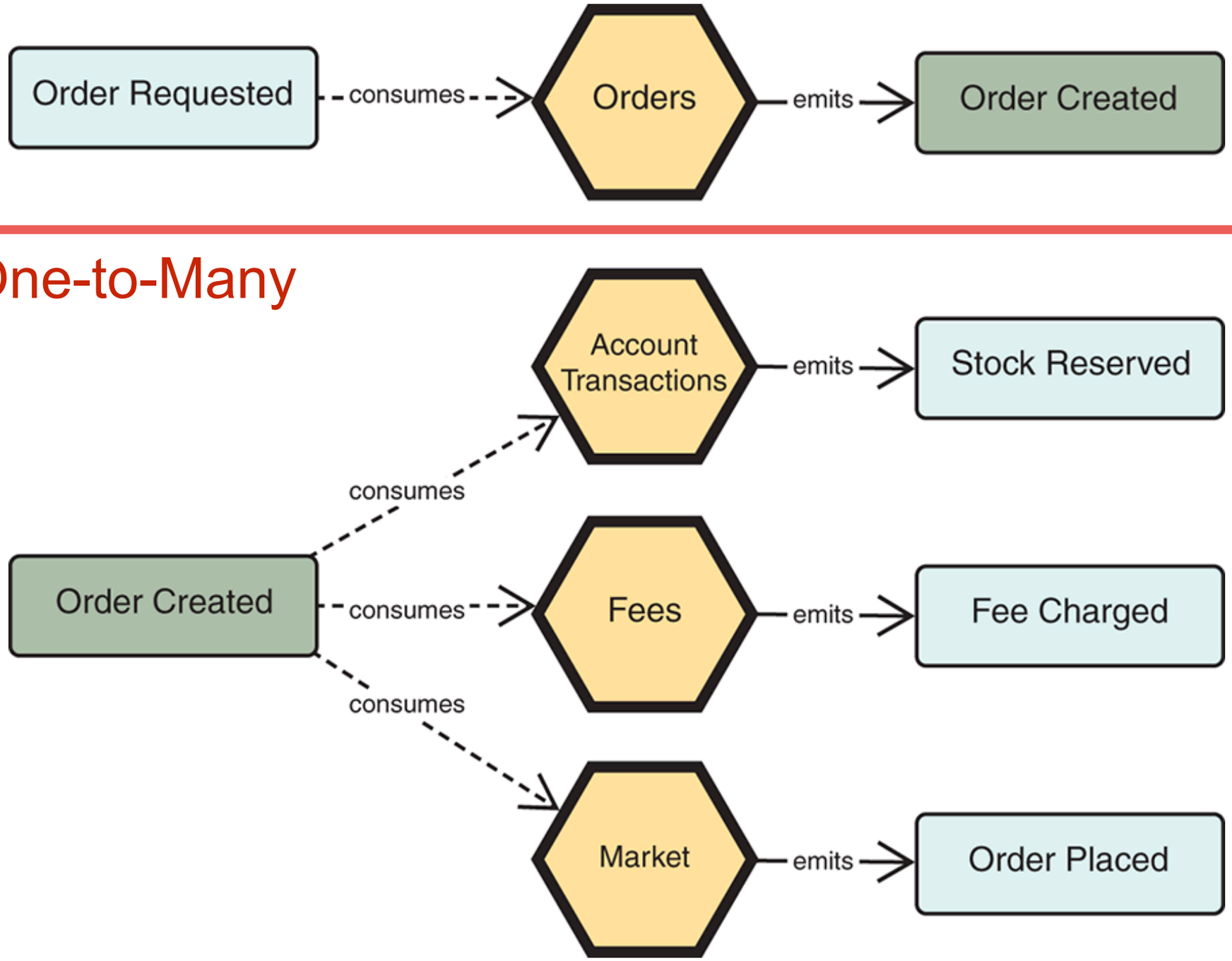
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service consume and emit event



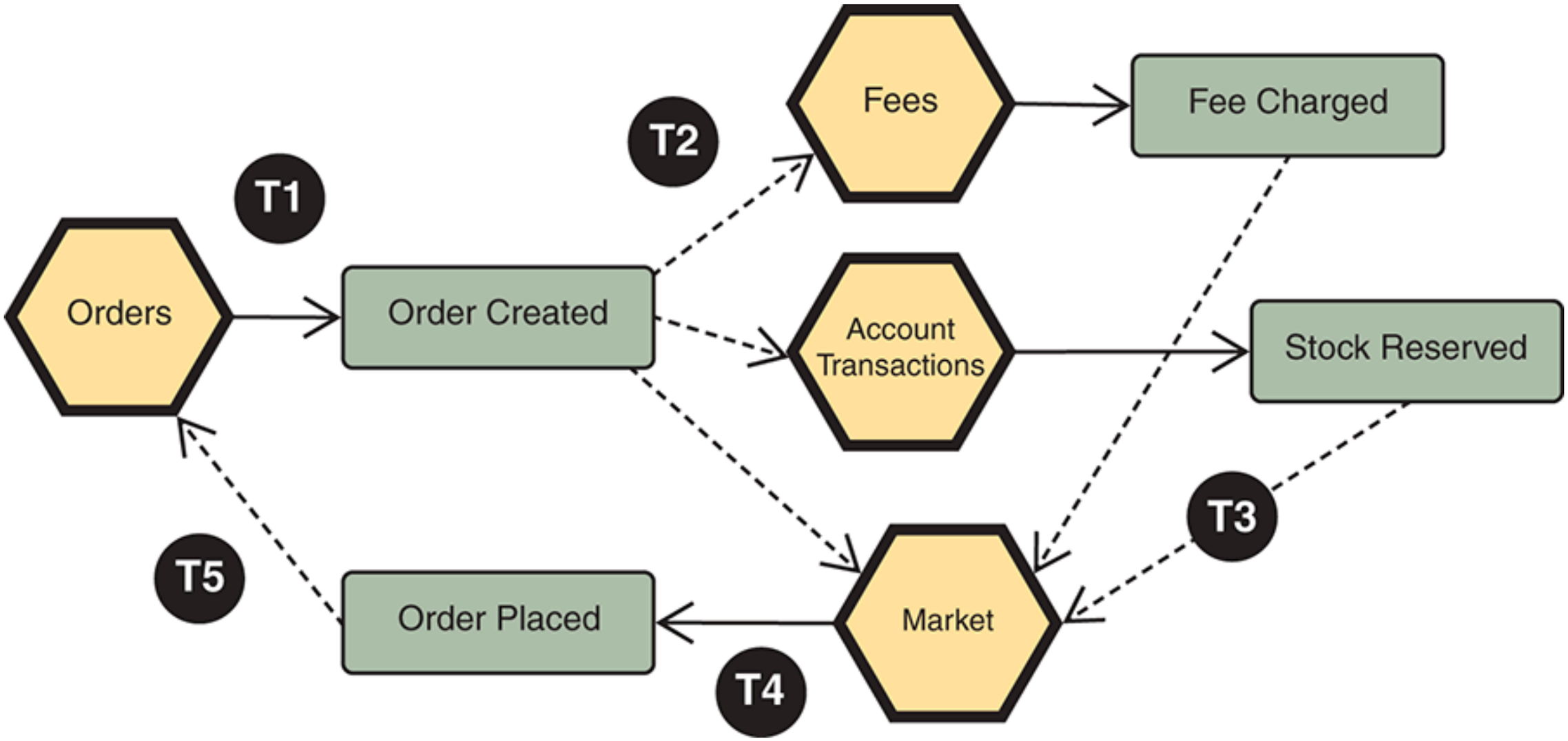
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Service consume and emit event



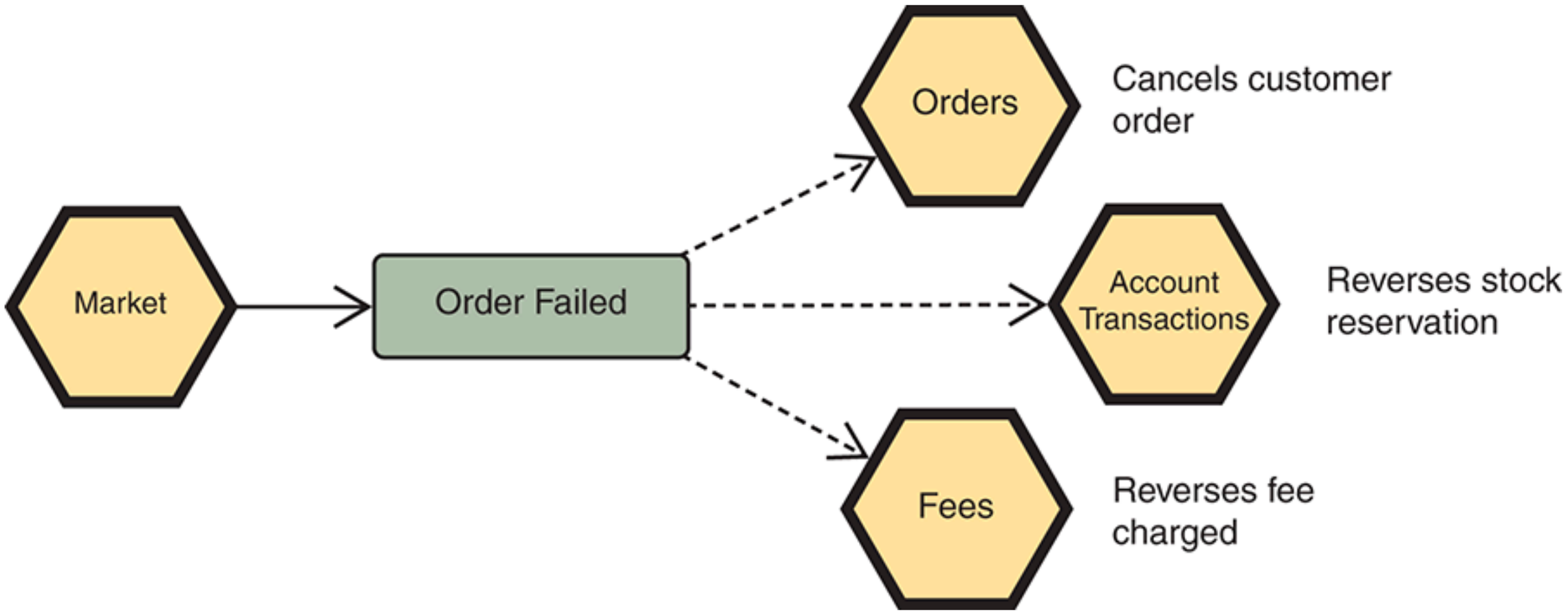
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example (Success)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

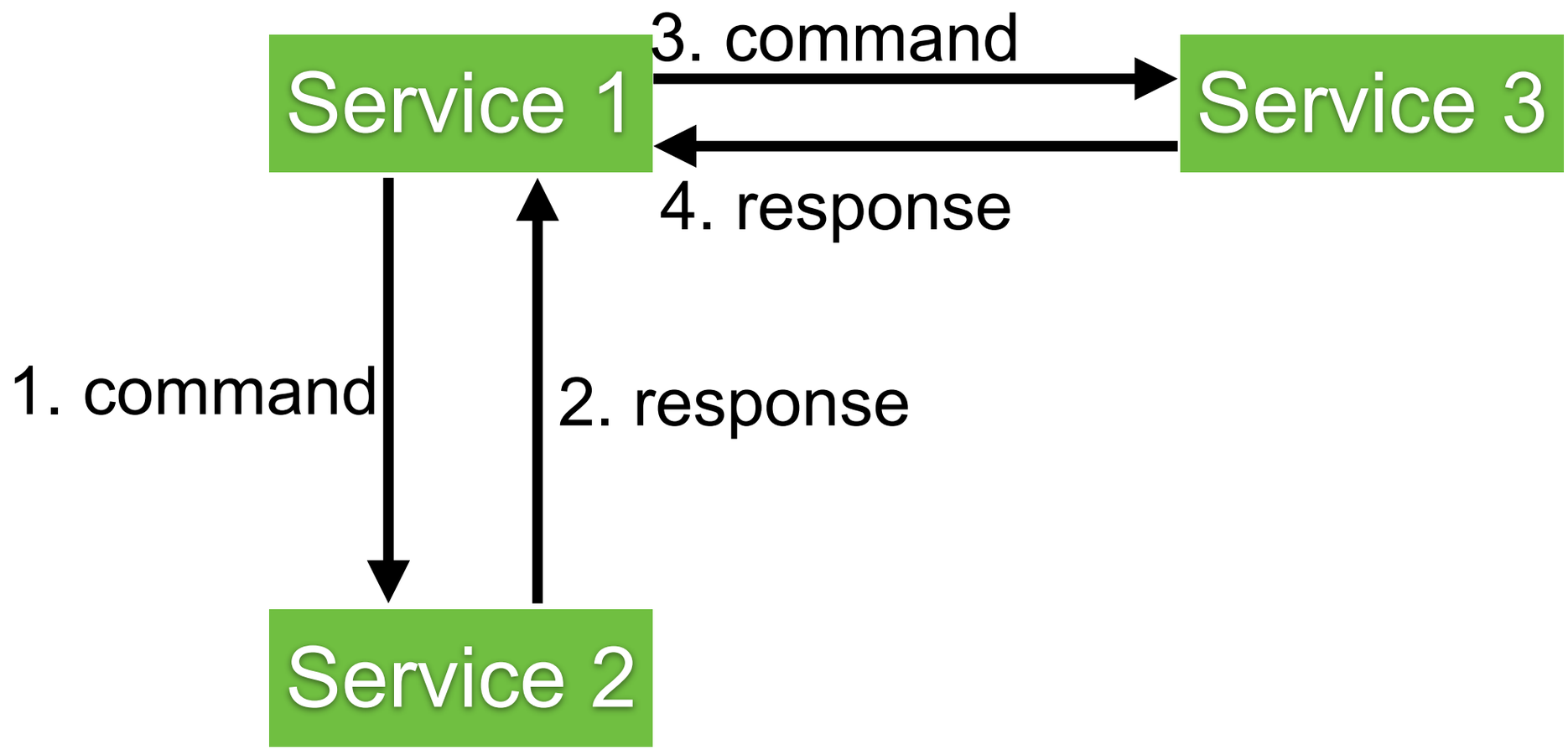
Example (Fail)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

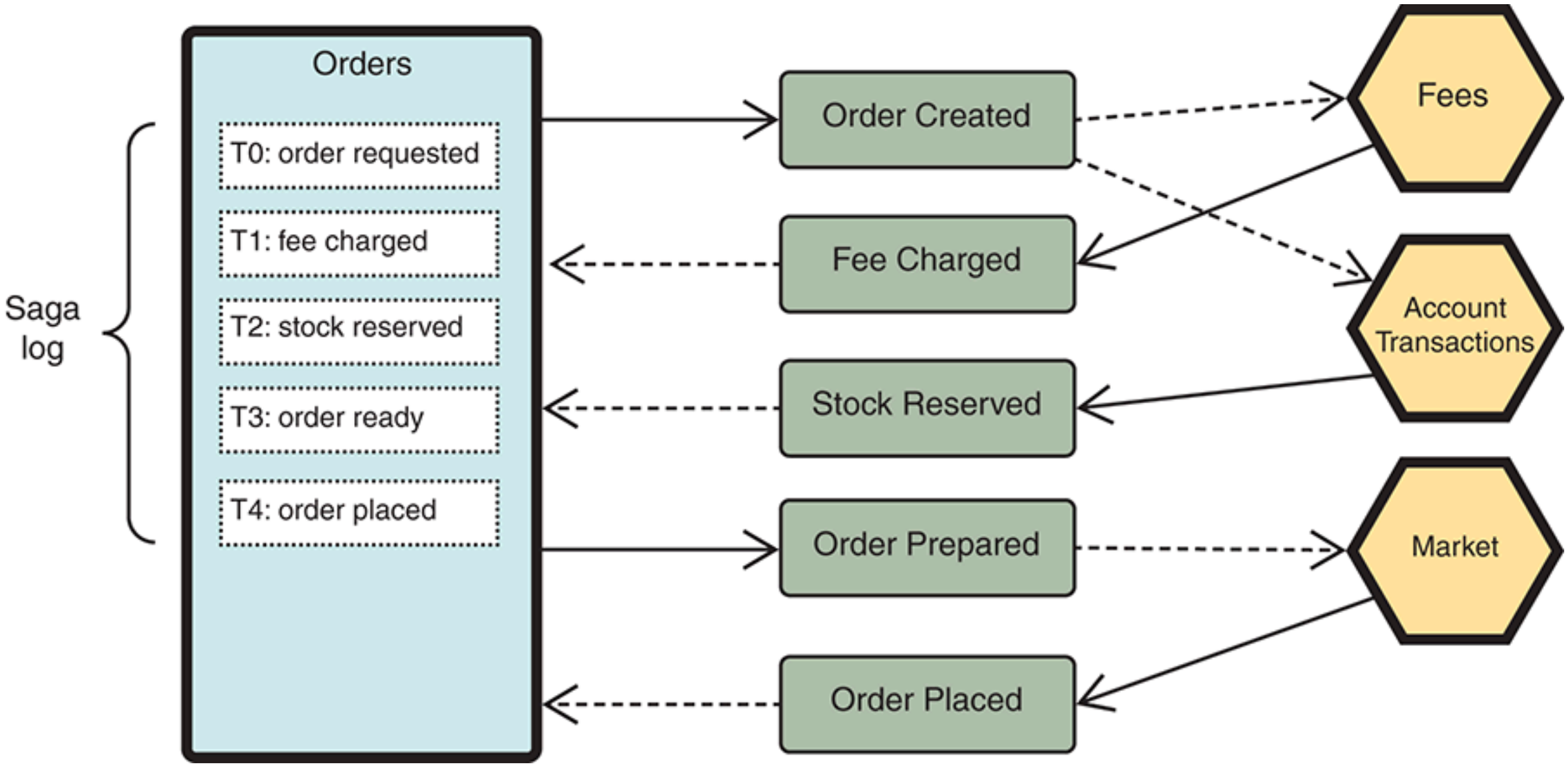
Orchestrate pattern

Sequence of Tx and emit **event** or **message** that trigger the next process in Tx



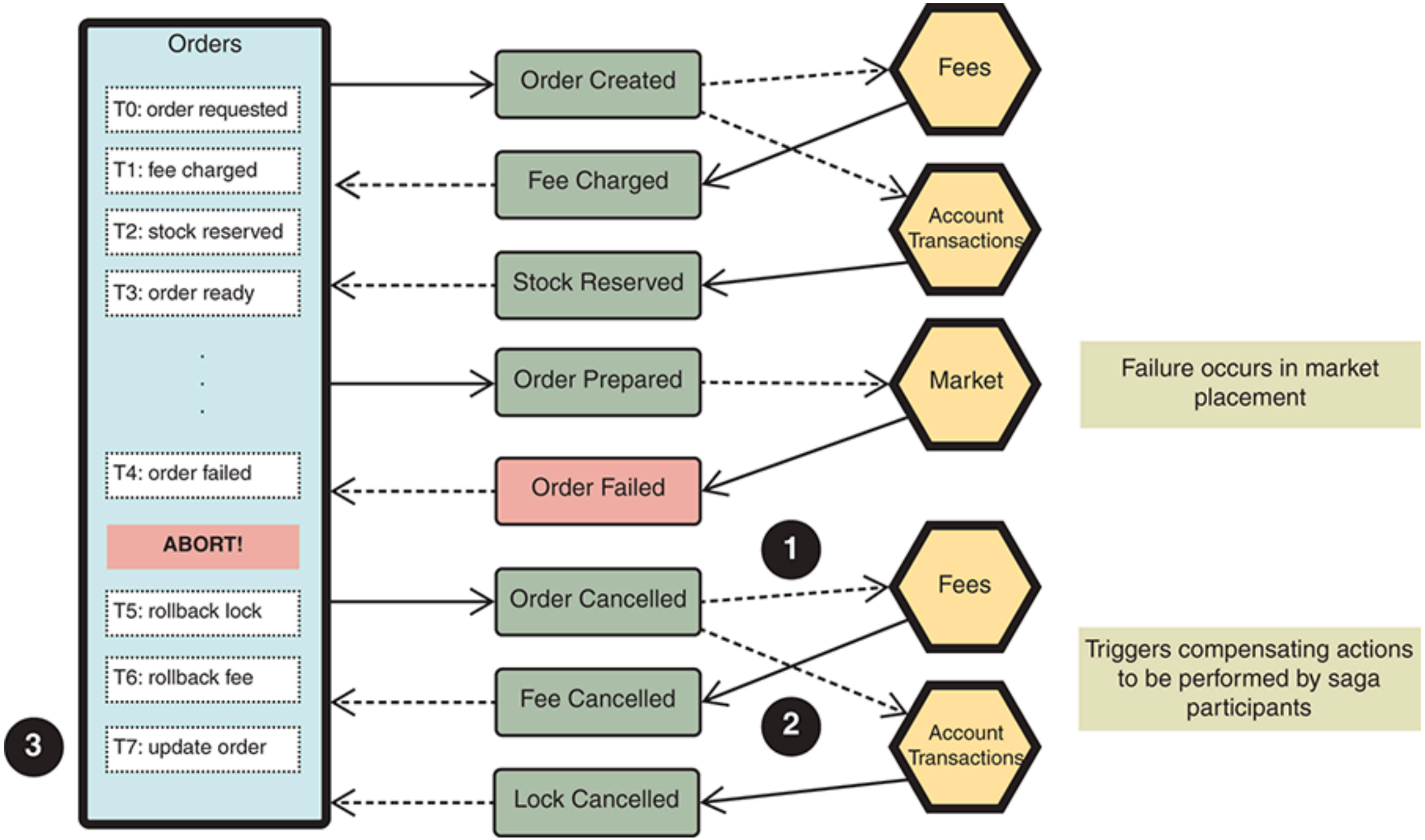
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example (Success)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Example (Fail)



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Consistency patterns

Name	Strategy
Compensating action	Perform action that undo prior action(s)
Retry	Retry until success of timeout
Ignore	Do nothing in the event of errors
Restart	Reset to the original state and start again
Tentative operation	Perform a tentative operation and confirm (or cancel) later

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

“Saga”

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Message vs Event

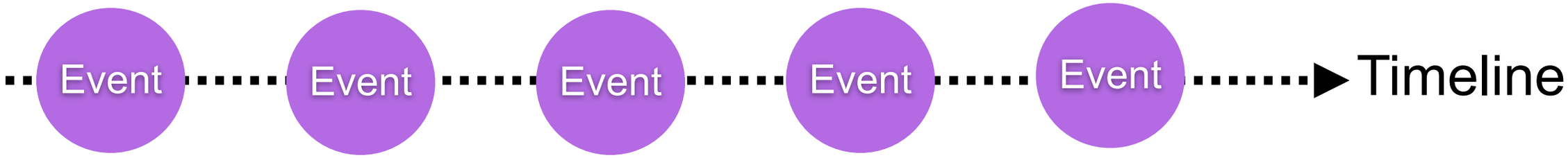
Message is addressed to someone

Event is something that happen and someone can react to that

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event sourcing

Maintain source of truth of business
Immutable sequence of events

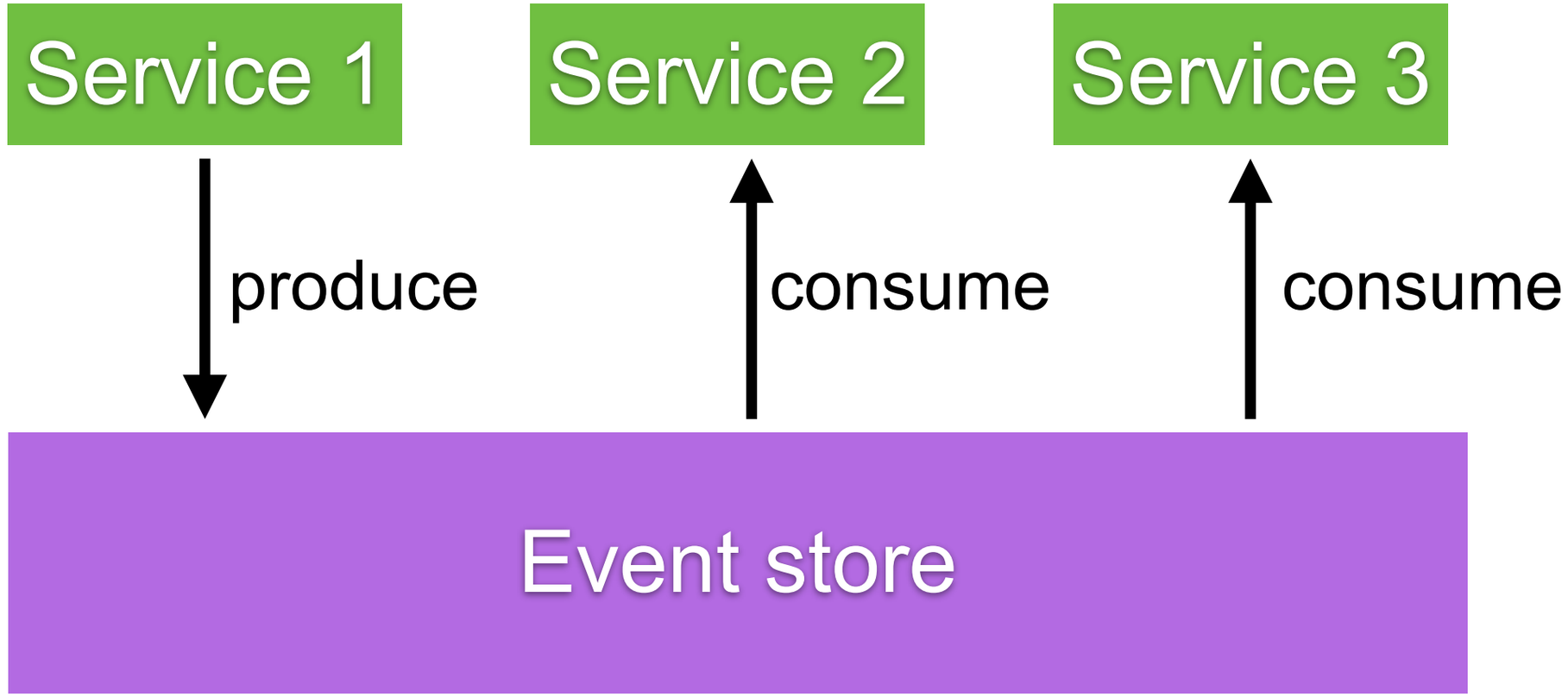


Sequence of event is keep in **Event store**

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event store

Keep events in order
Broadcast new event



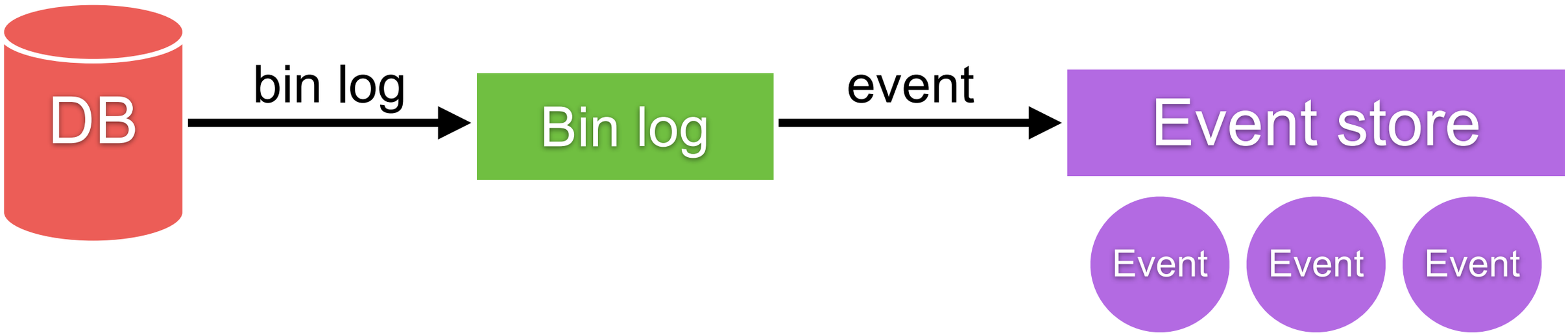
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

With Event sourcing, we can **decouple services** (query and command)

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event sourcing with database

Working with database



Binlog or Binary log event is information about data modification made to database

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event sourcing

Duplication data (denormalize database)
Complexity (separate query and command)
Difficult to maintain

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Event sourcing can’t solve all problems

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Start with **understand** your **purpose**

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Start with understand your
problem to resolve

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

How to implement queries ?

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Query data from multiple services

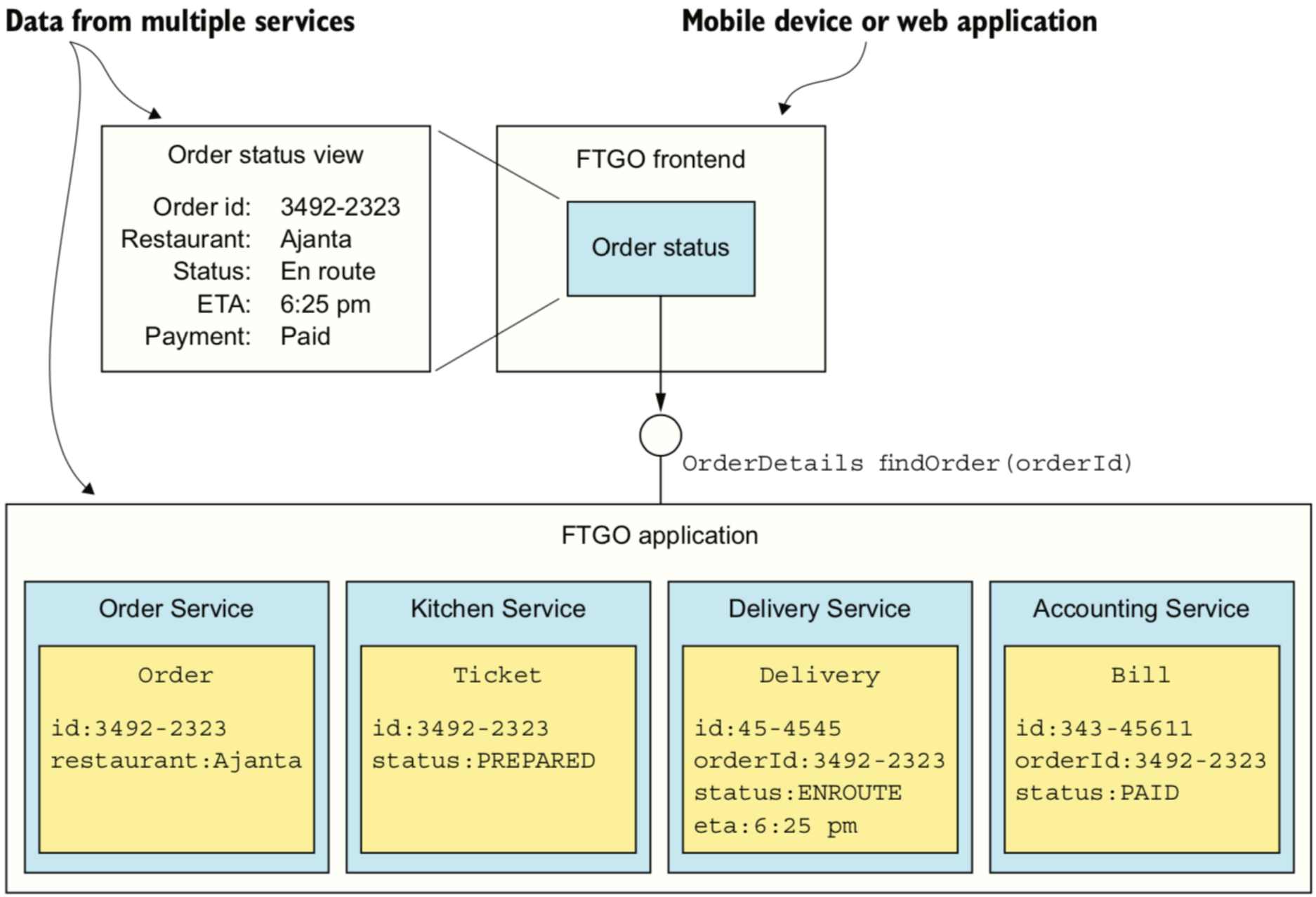
API composition

Cold data from services

CQRS (Command Query Responsibility Segregation)

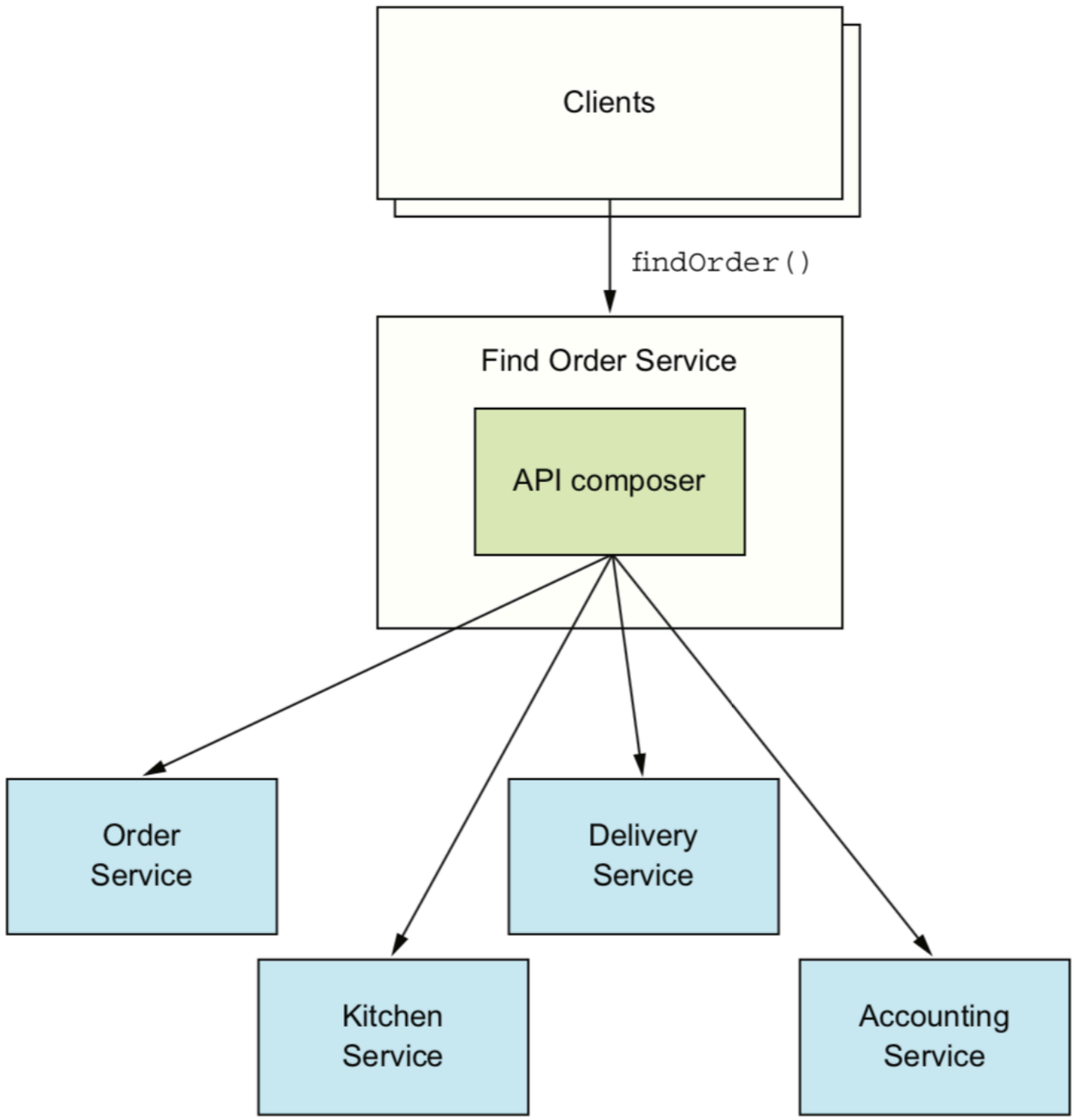
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Problem ?



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

API composition pattern



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks

Increase overhead

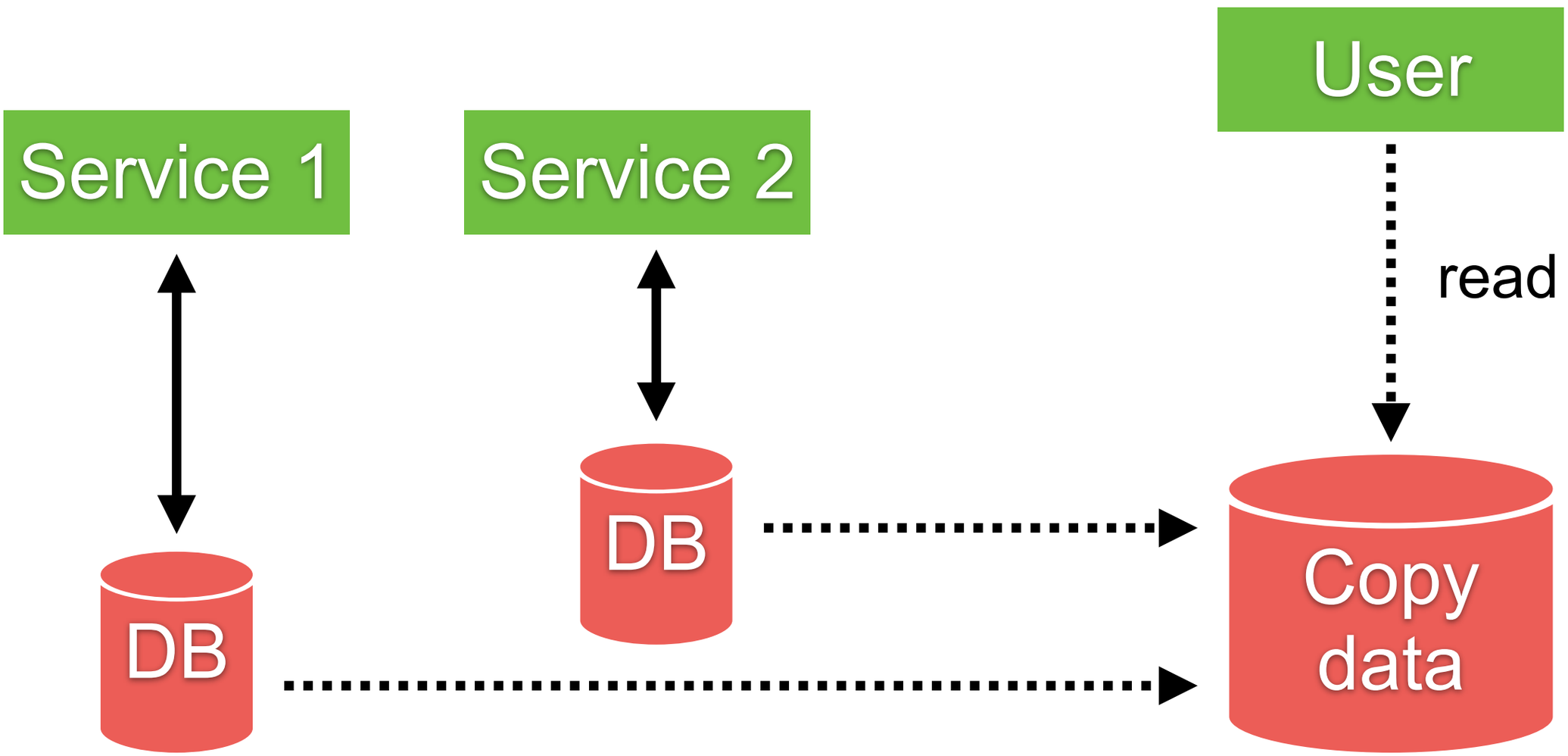
Lack of transactional data consistency

Reduce availability

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Storing copy data

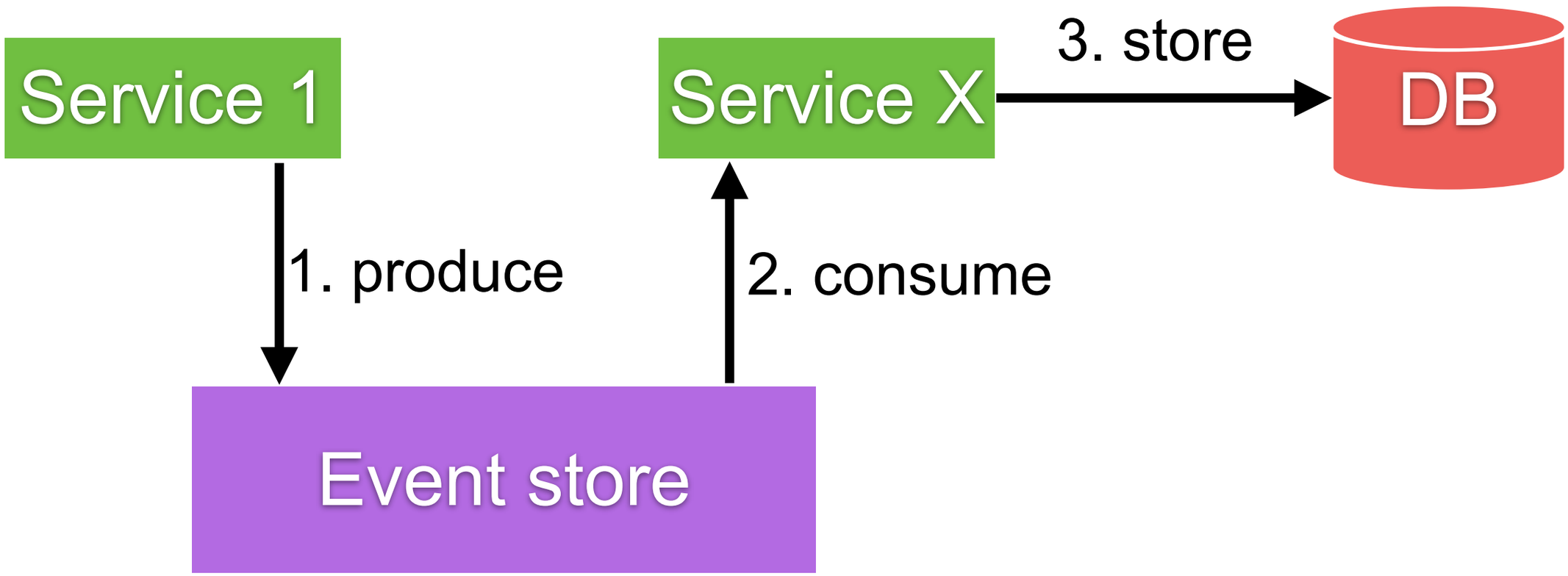
Copy or caching data to other database



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Working with Event-based

Publish event to store copy data



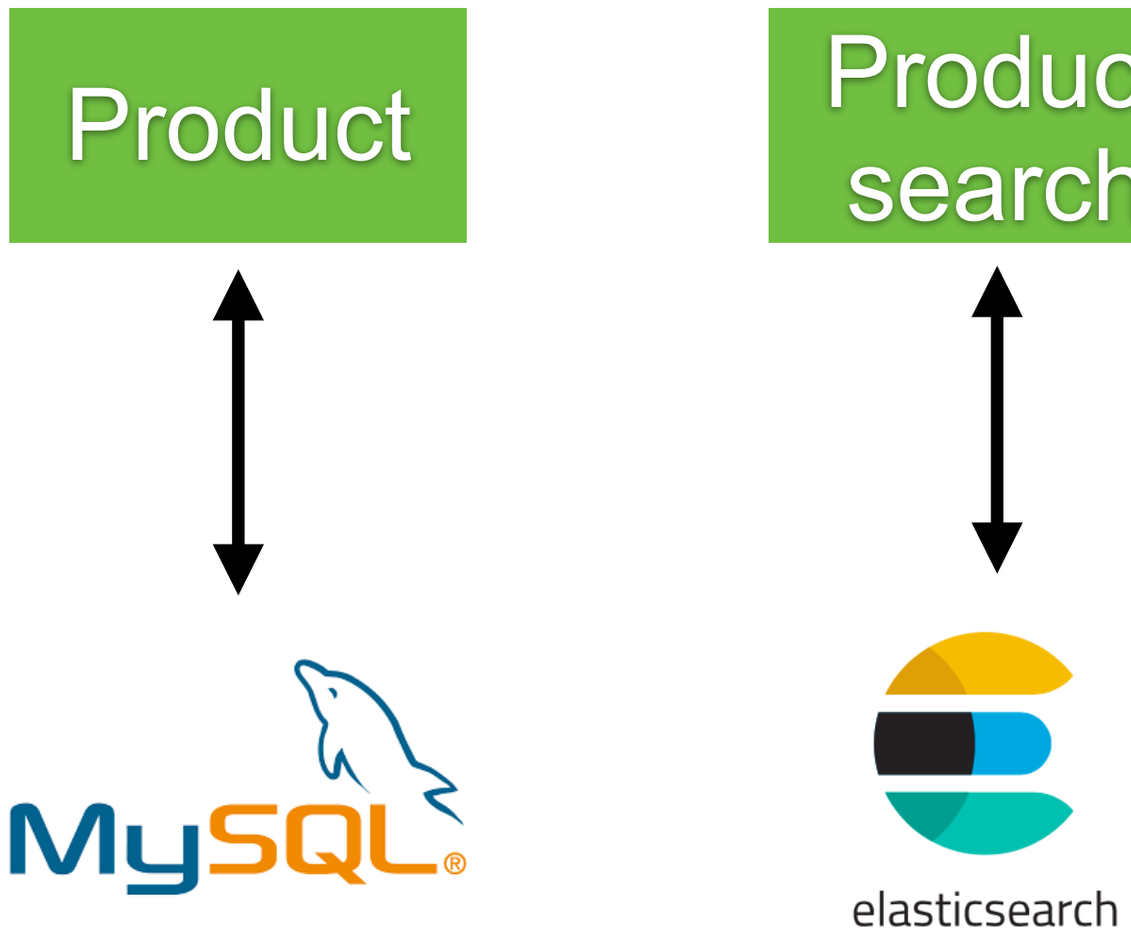
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Separate query and command

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Separate data for read and write

For example MySQL to write, Elasticsearch to search



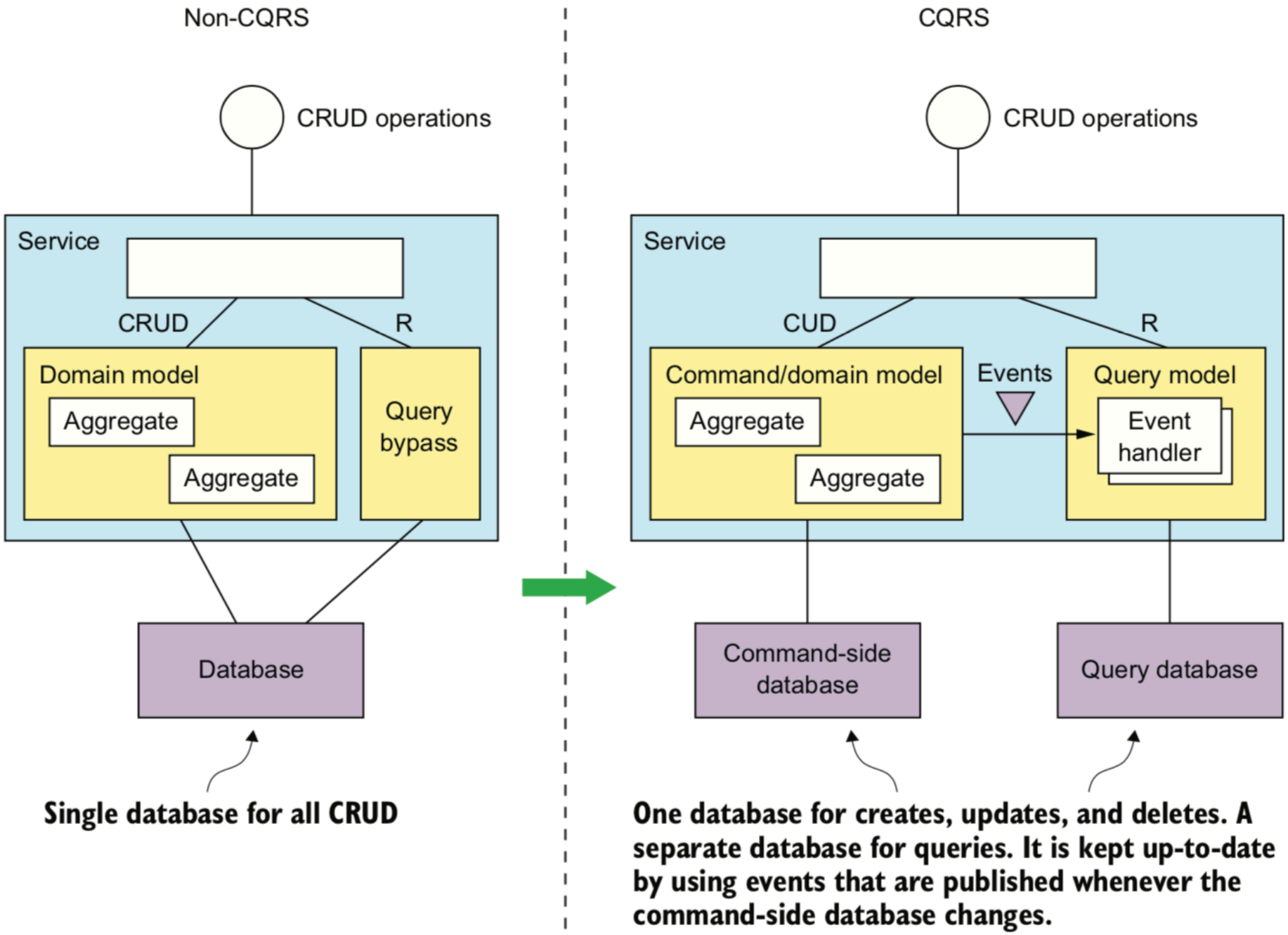
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

CQRS

Command **Q**uery **R**esponsibility **S**egregation

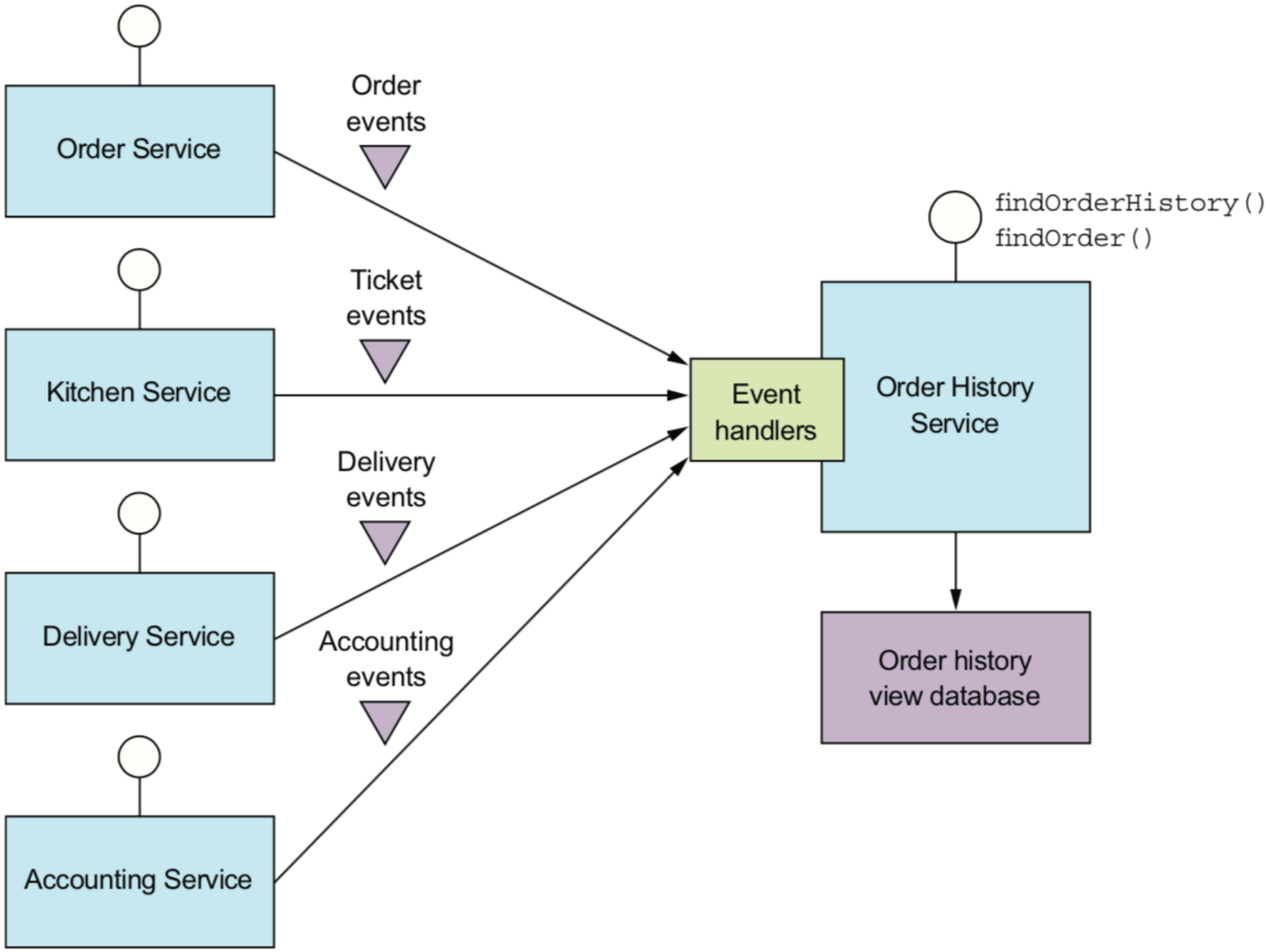
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

CQRS = Separate command from query



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

CQRS = Query-side service



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Benefits

Improve separation of concern
Efficiency query

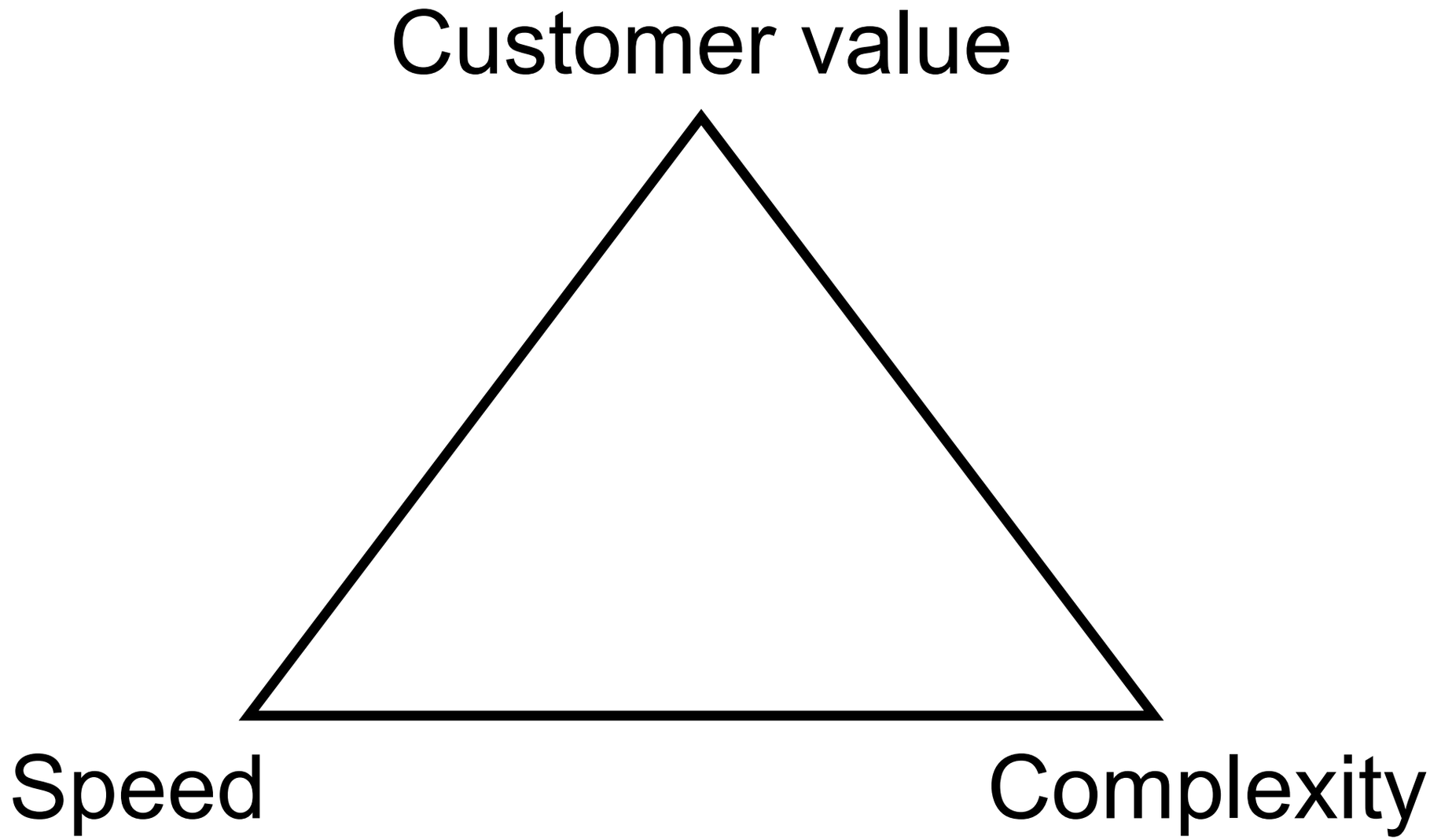
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Drawbacks

More complex

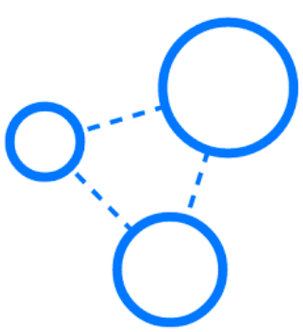
Lag between command and query side

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



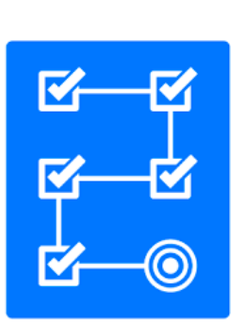
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Beyond Microservice
Process and **Organization**



Flexible organizational structure

With clear roles and accountabilities



Efficient meeting formats

Geared toward action and eliminating over-analysis

More autonomy to teams and individuals

Individuals solve issues directly without bureaucracy

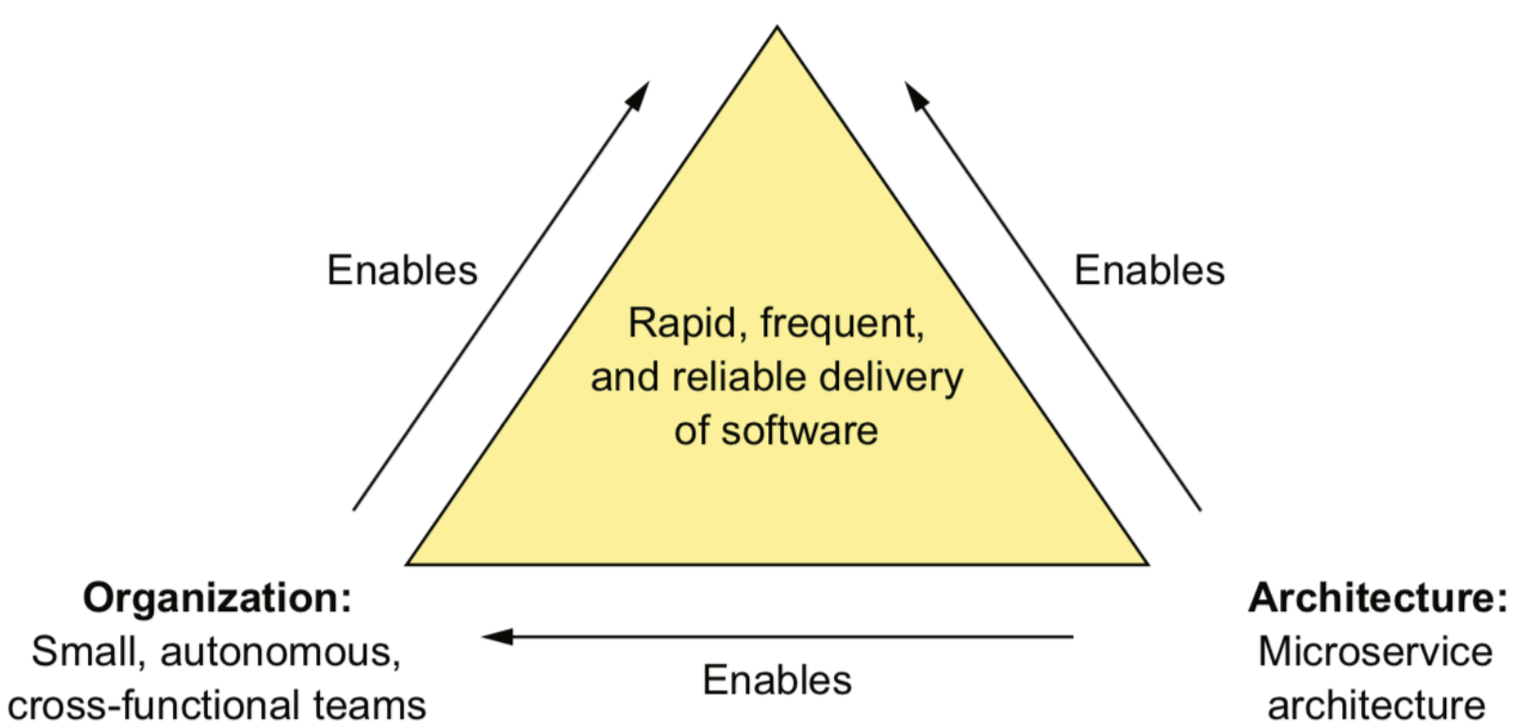
Unique decision-making process

To continuously evolve the organization's structure.

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Success project needs …

Organization process
Development process
Delivery process



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

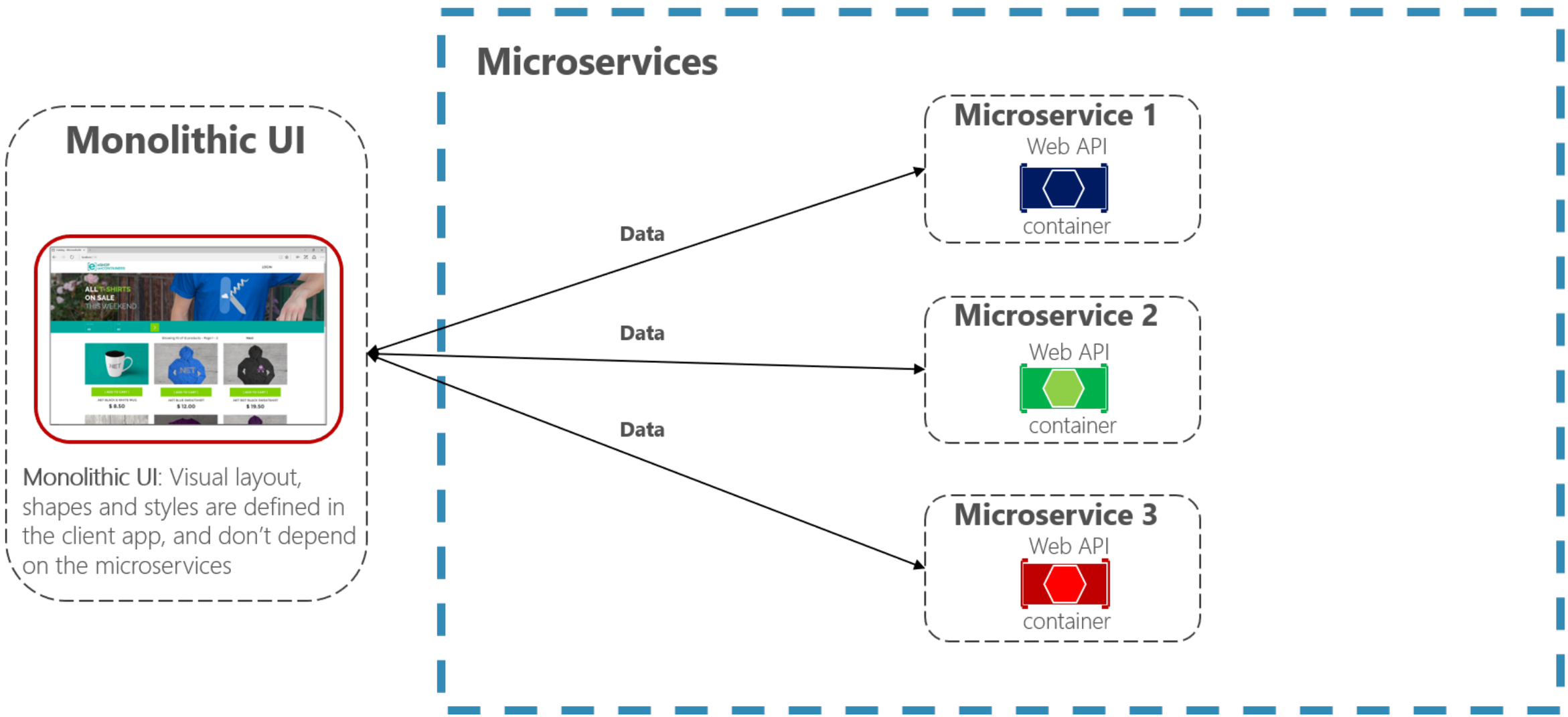
Don’t forget about the **human** side when adopt Microservice

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Integrate services with User Interface

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

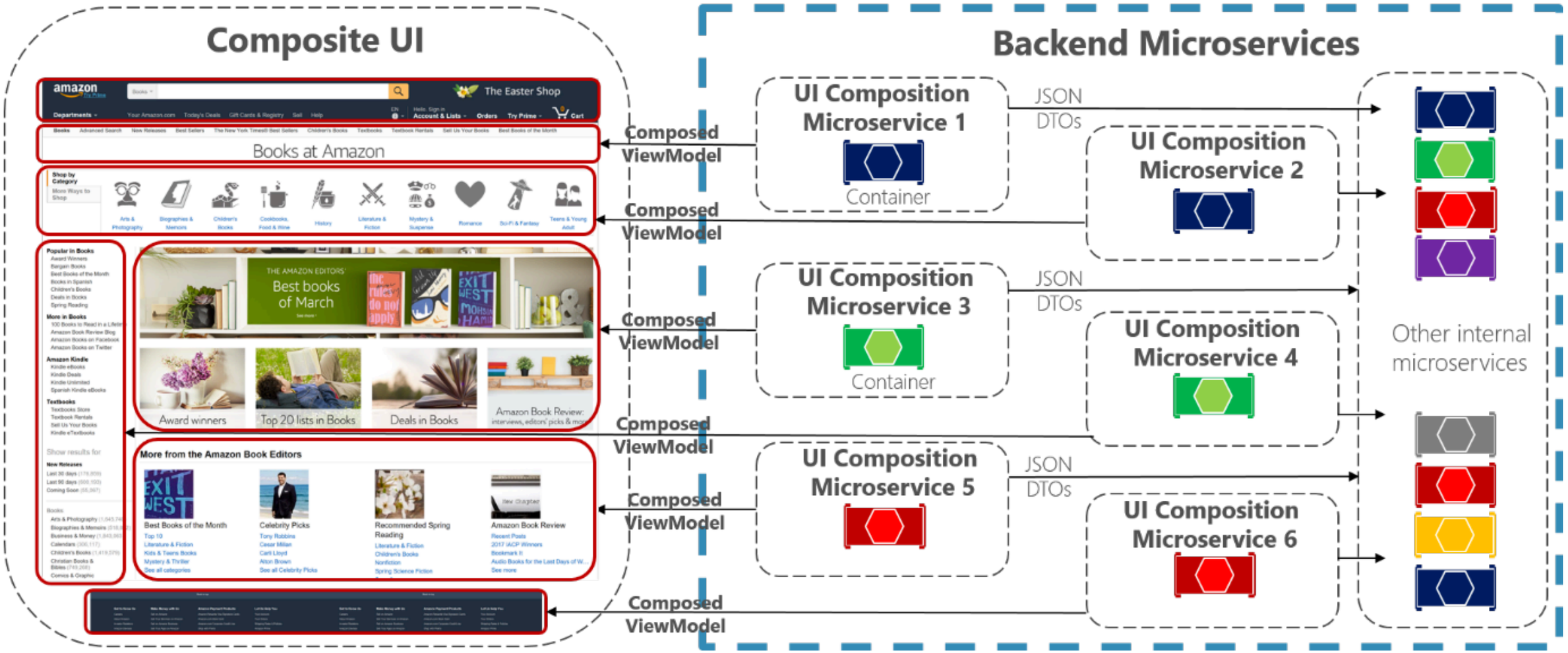
Monolithic UI consuming microservices



https://docs.microsoft.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

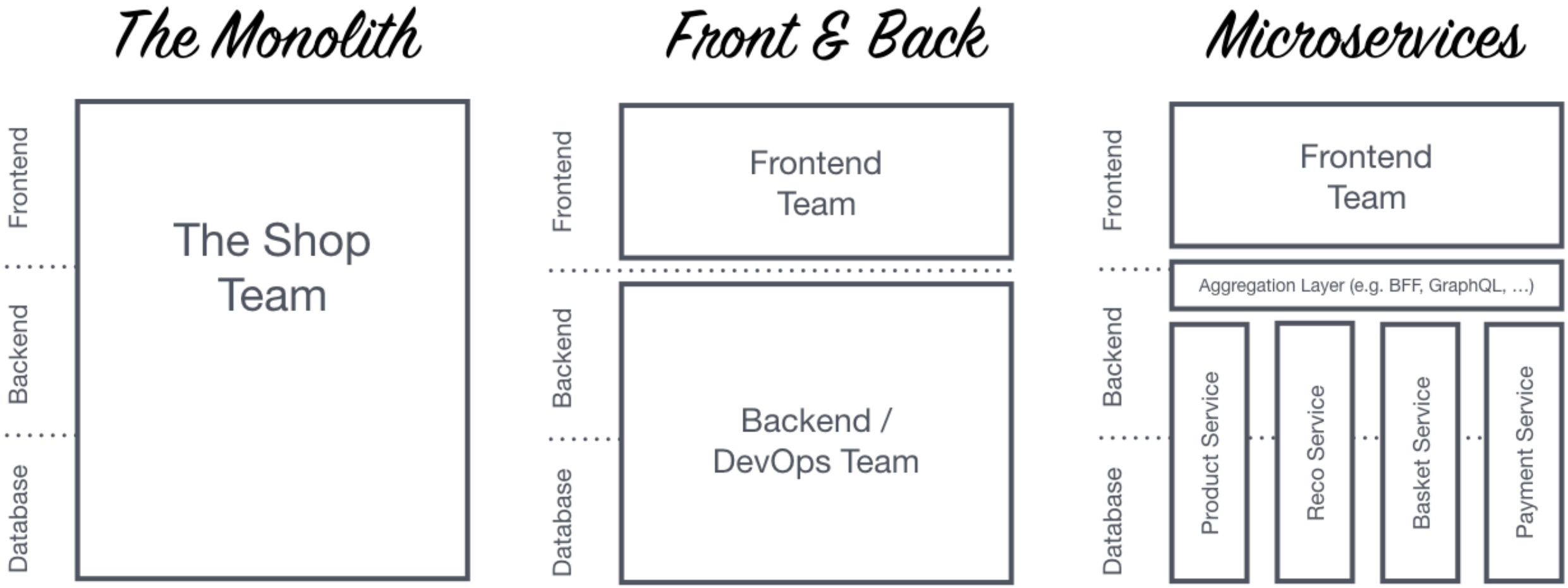
Composite UI *generated* by microservices



https://docs.microsoft.com

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

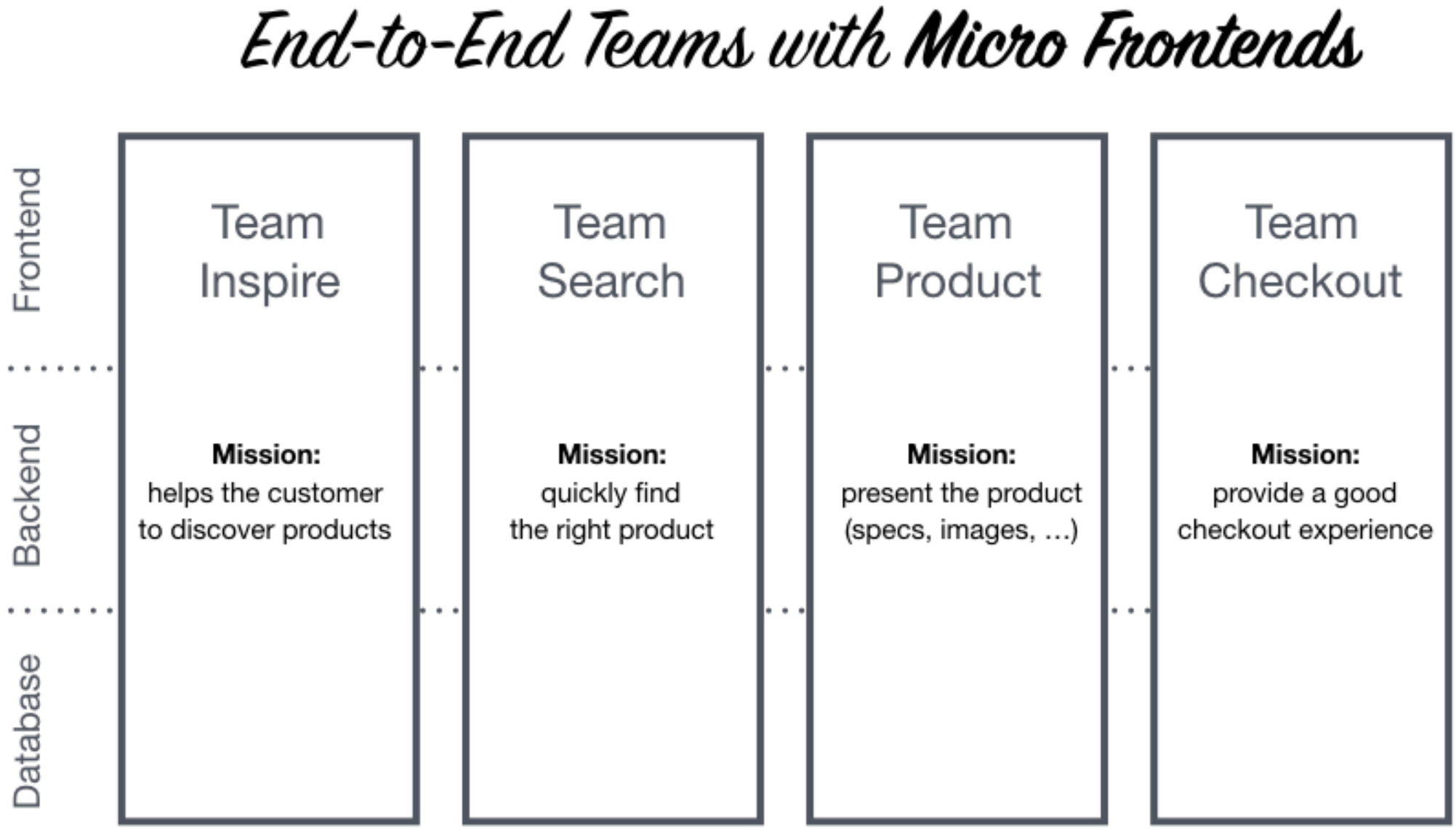
Monolith frontend



https://micro-frontends.org/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Micro frontend



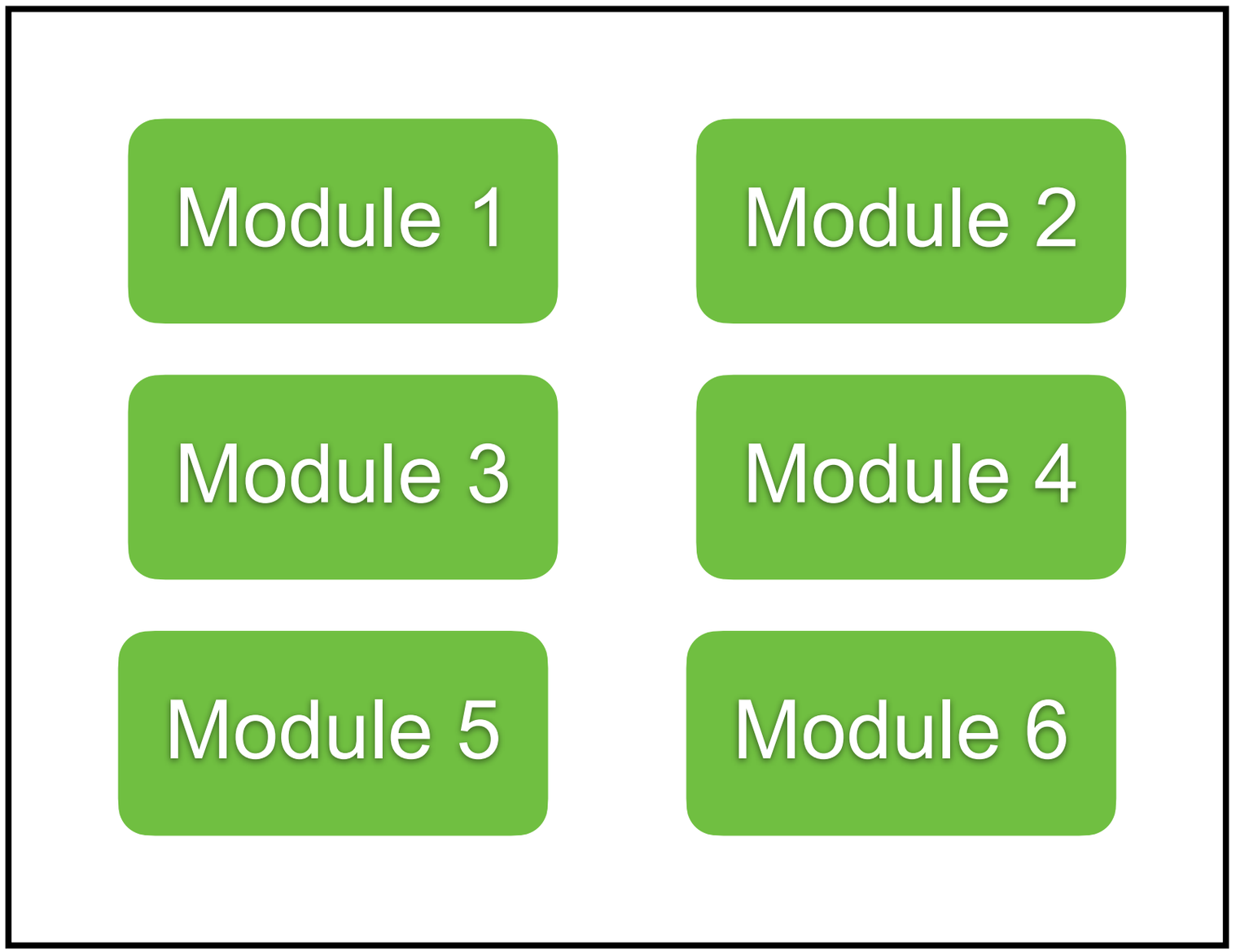
https://micro-frontends.org/

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Summary

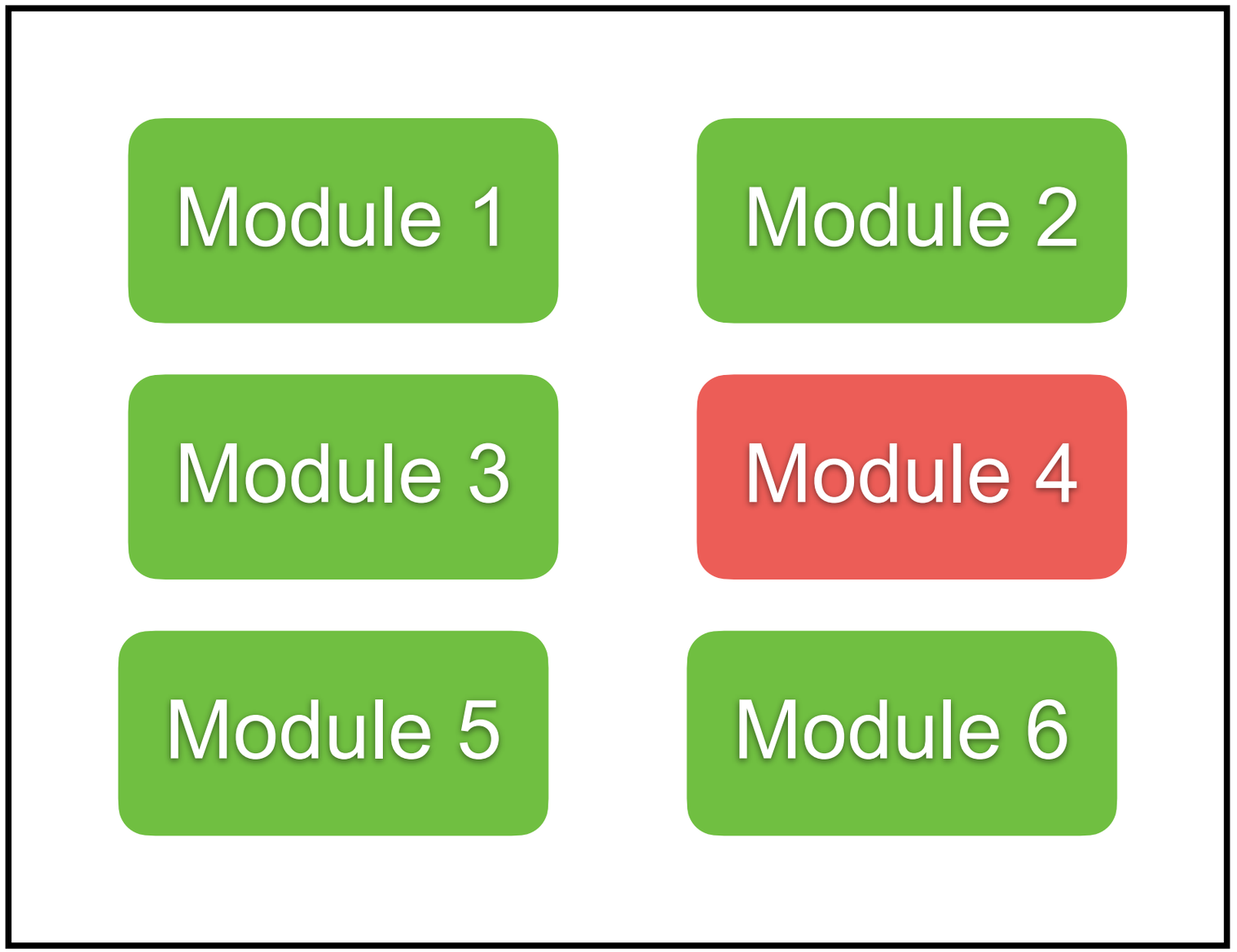
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Let’s start with good monolith

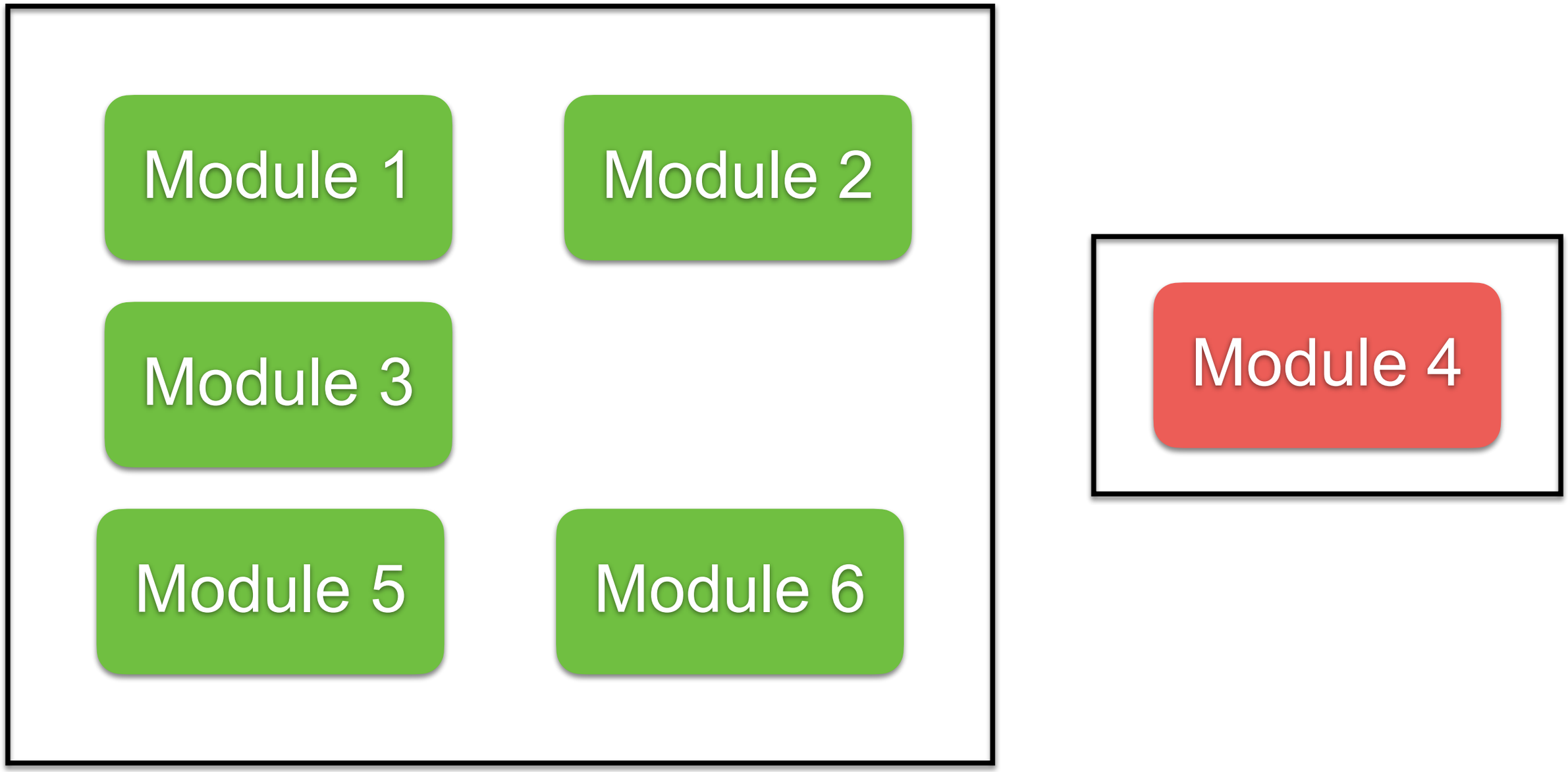


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Find your problem

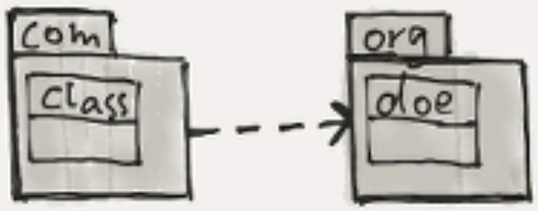


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

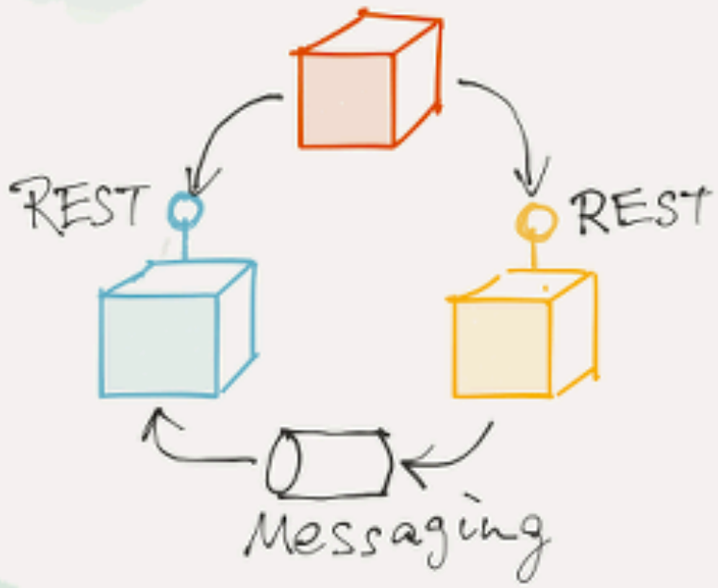


© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Architecture



Microservices

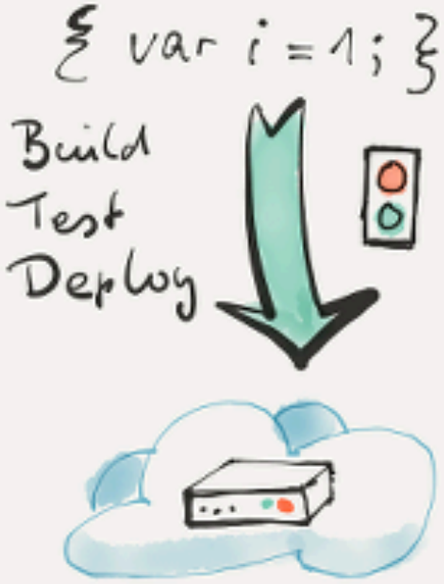


People & Teams

Communication
Collaboration

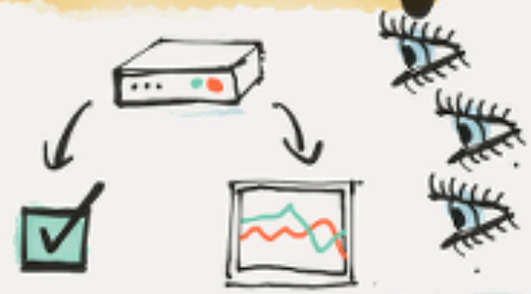
Deployment

Continous Delivery



Infrastructure

Monitoring

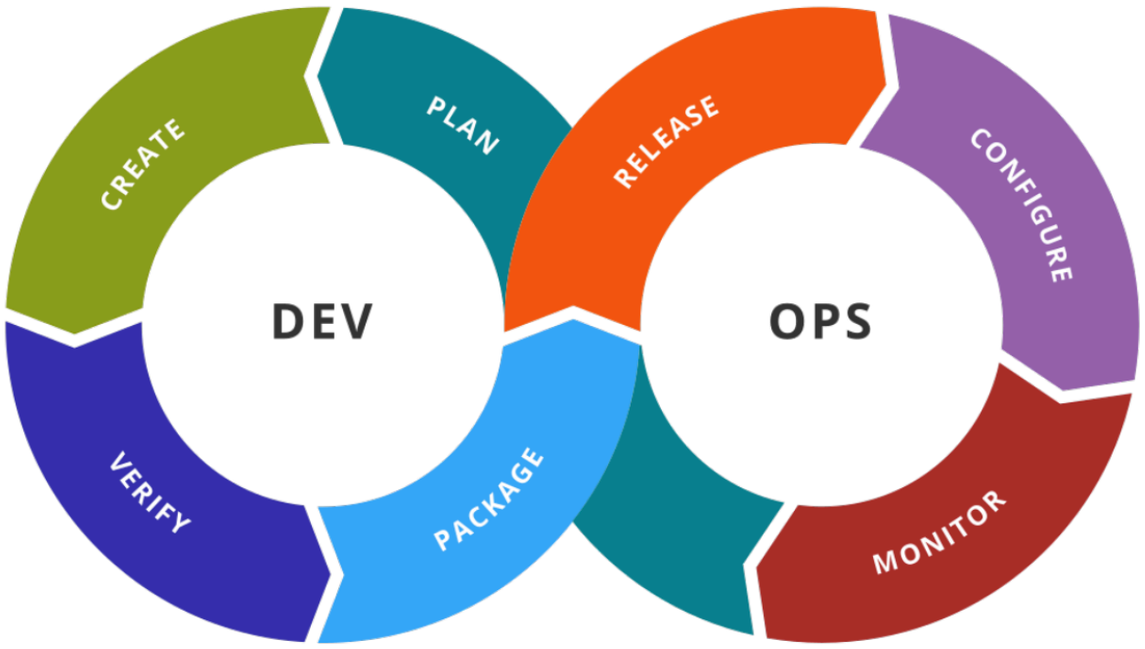


Features & Technology

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservice prerequisites

Rapid provisioning
Basic monitoring
Rapid Application Deployment
DevOps culture



© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Design Workshop

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

More topics of Microservice

Testing

Deployment

Monitoring

Security

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Let’s start your journey

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.