Connectable Observables, Resource Management, and Subjects

Russell Elledge
ENTERPRISE ARCHITECT

@MC2FTW mastercraftcoding.com

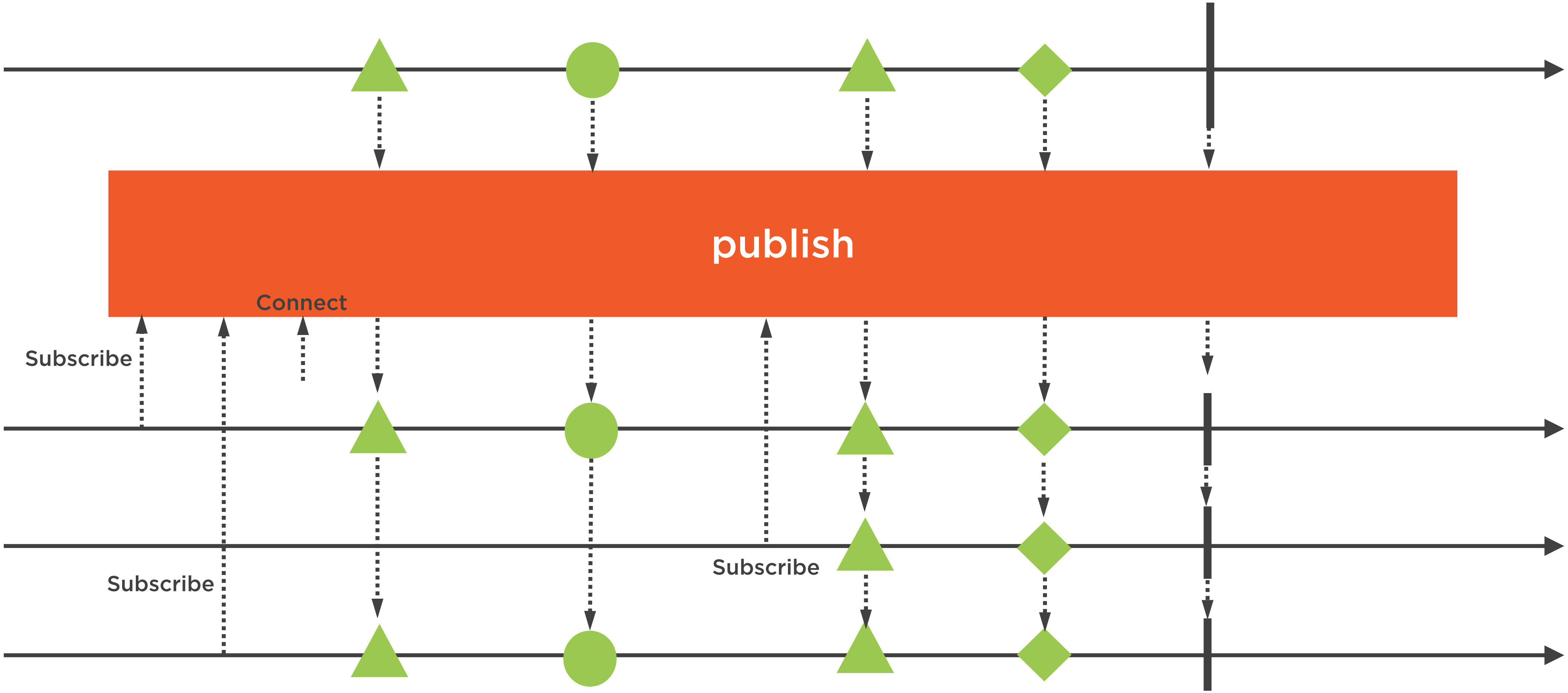
Connectable Observables

Life Cycle of a Connectable Observable

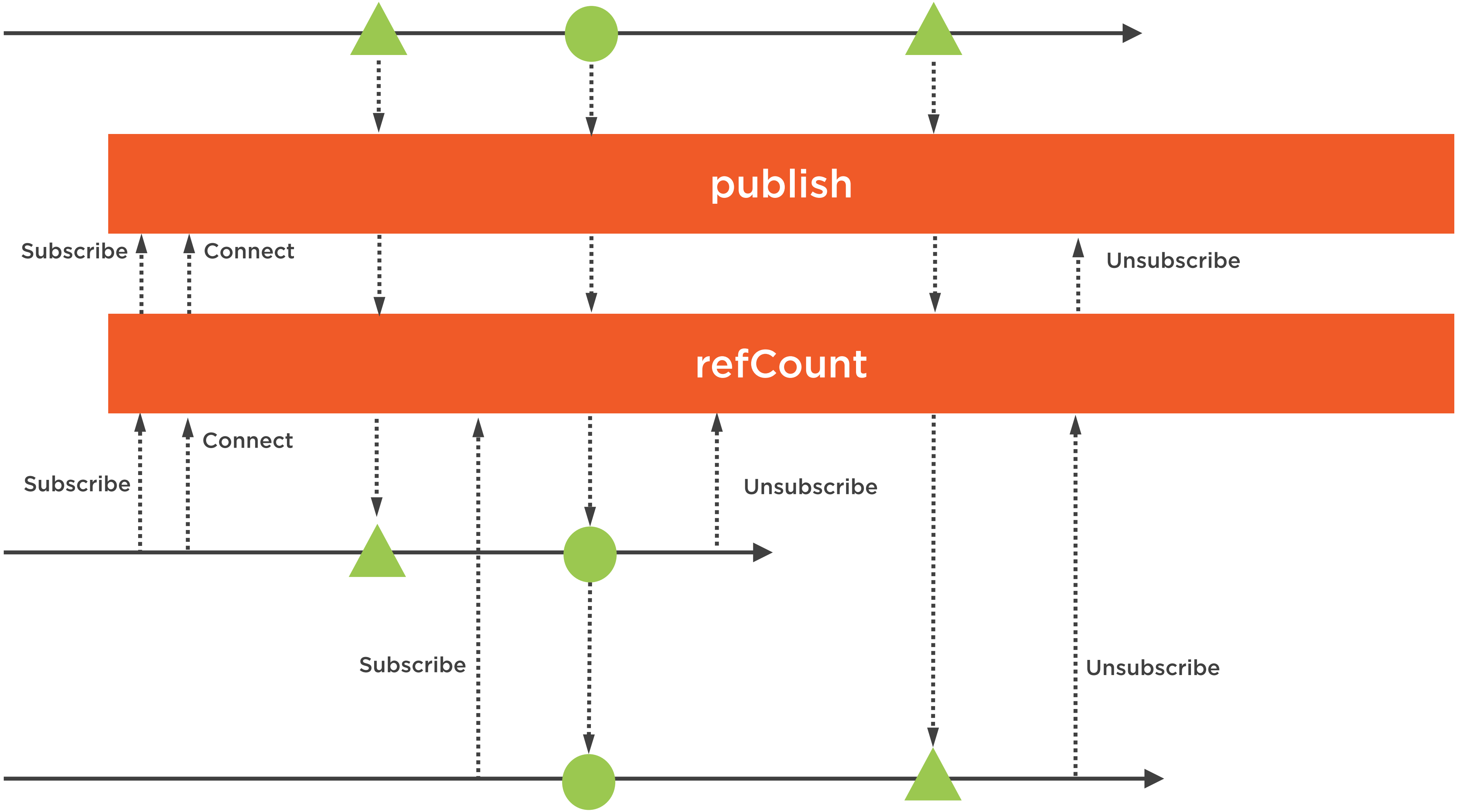
Creating Connectable Observables

Sharing Observables between Subscribers

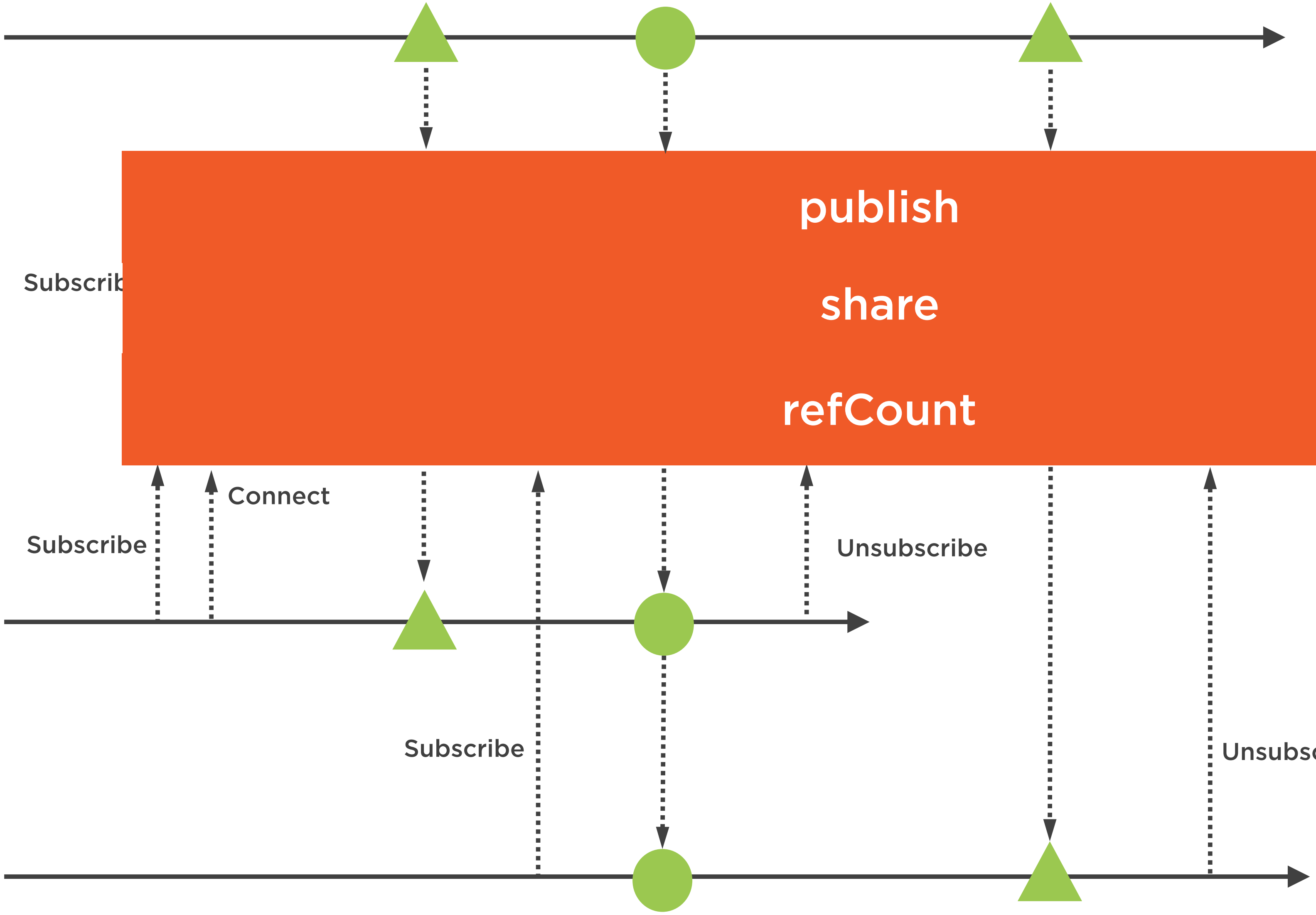
Connectable Observables - Lifecycle



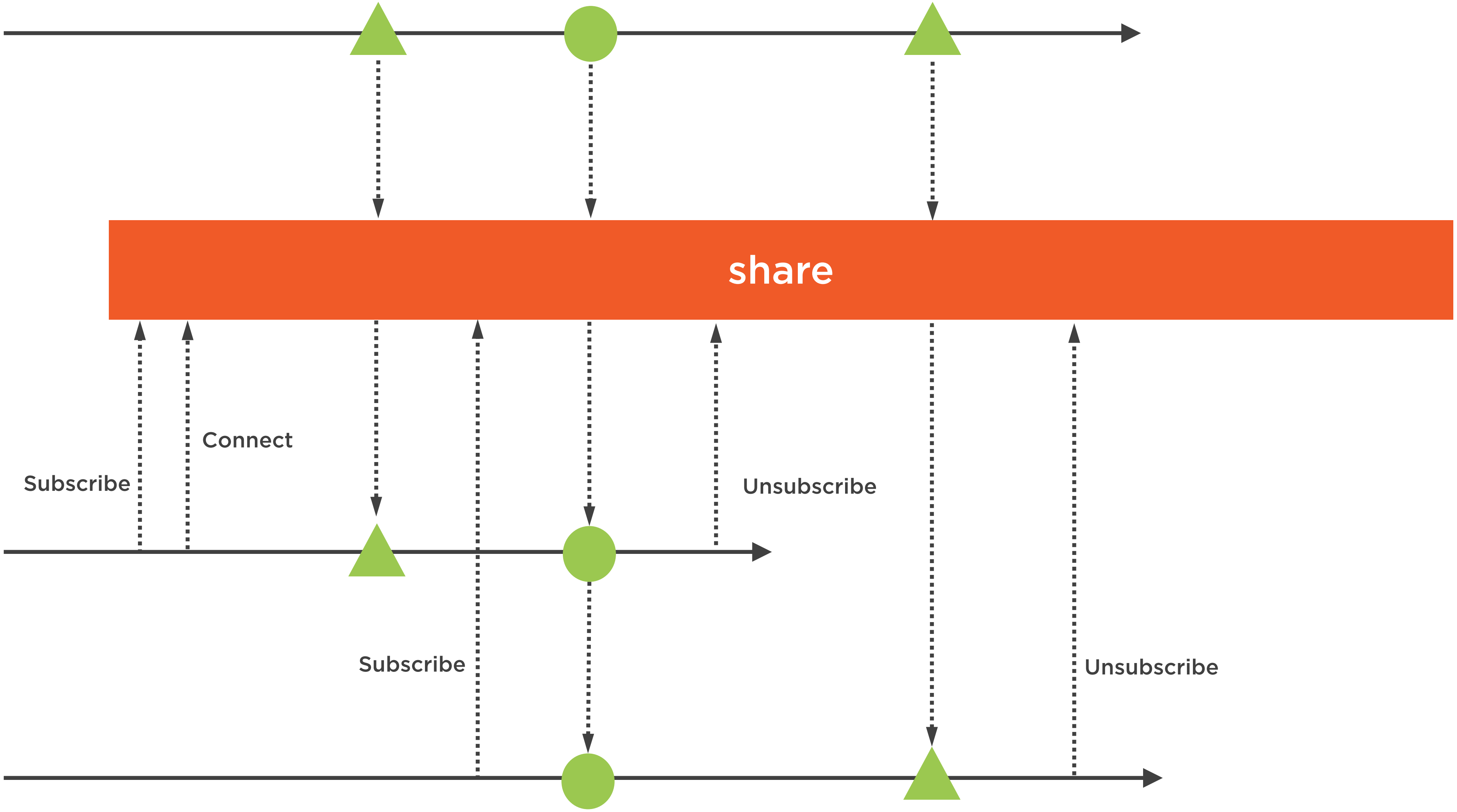
Sharing Observables



Sharing Observables



Sharing Observables



Demo

publish

refCount

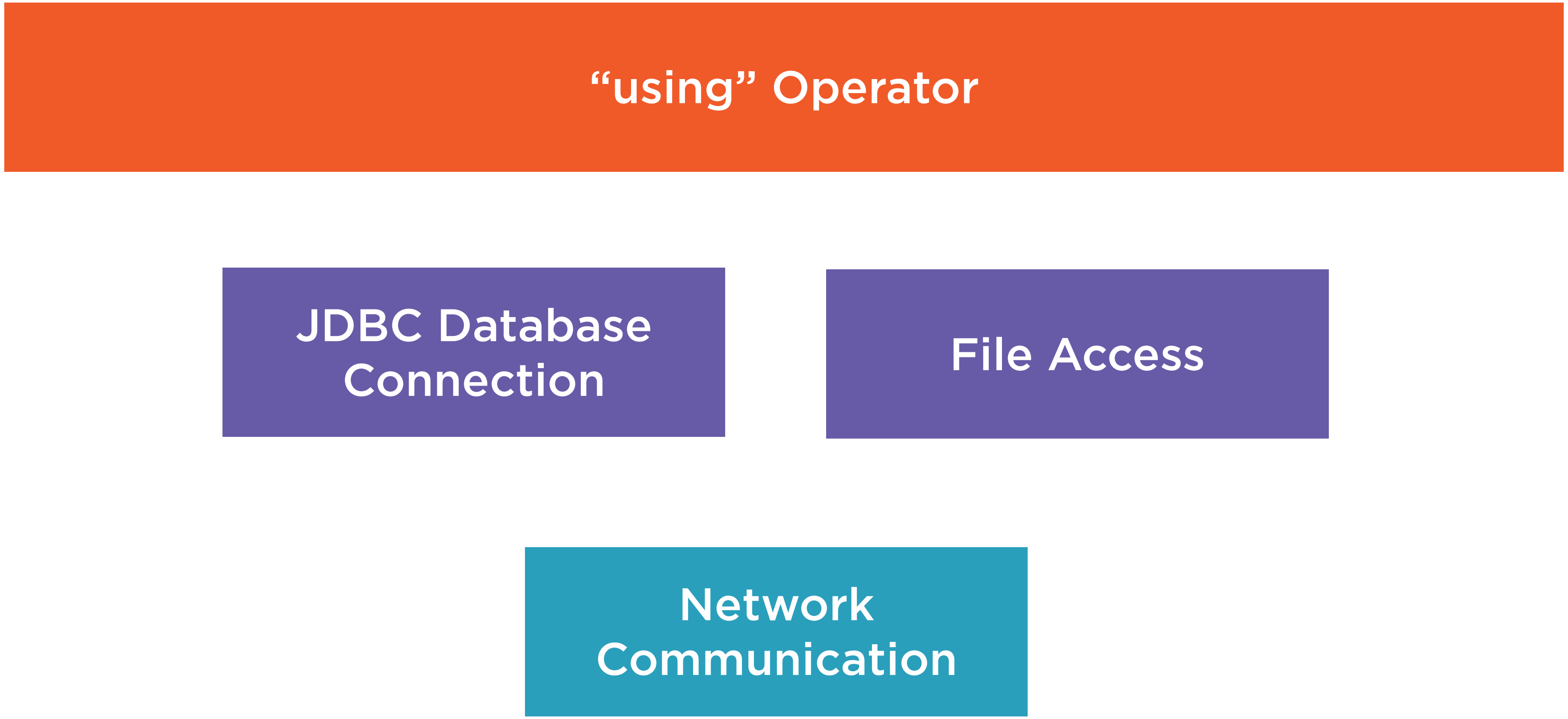
share

Resource Management

using Operator

- Resource Allocation
- Resource Consuming Observable
- Resource Disposal

Resource Management



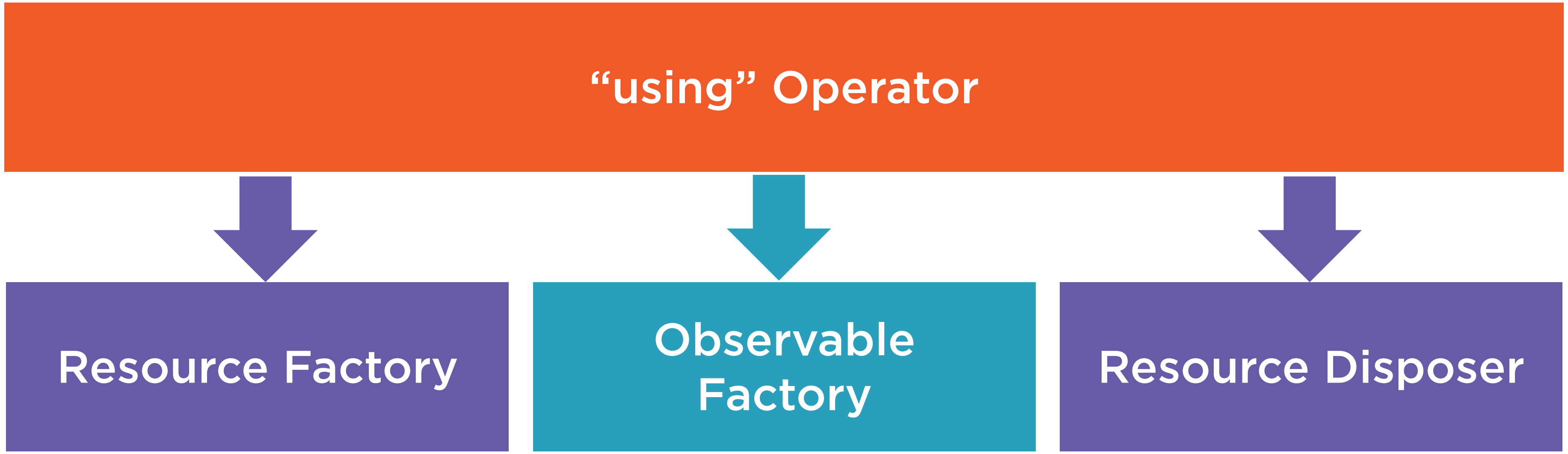
“using” Operator

JDBC Database Connection

File Access

Network Communication

Resource Management



“using” Operator

Resource Factory

Observable Factory

Resource Disposer

Demo

“using” Operator

- Resource Allocation
- Resource Consumption
- Resource Disposal

Event Driven Programming

Single Responsibility Principle

Isolating Knowledge of Service Use

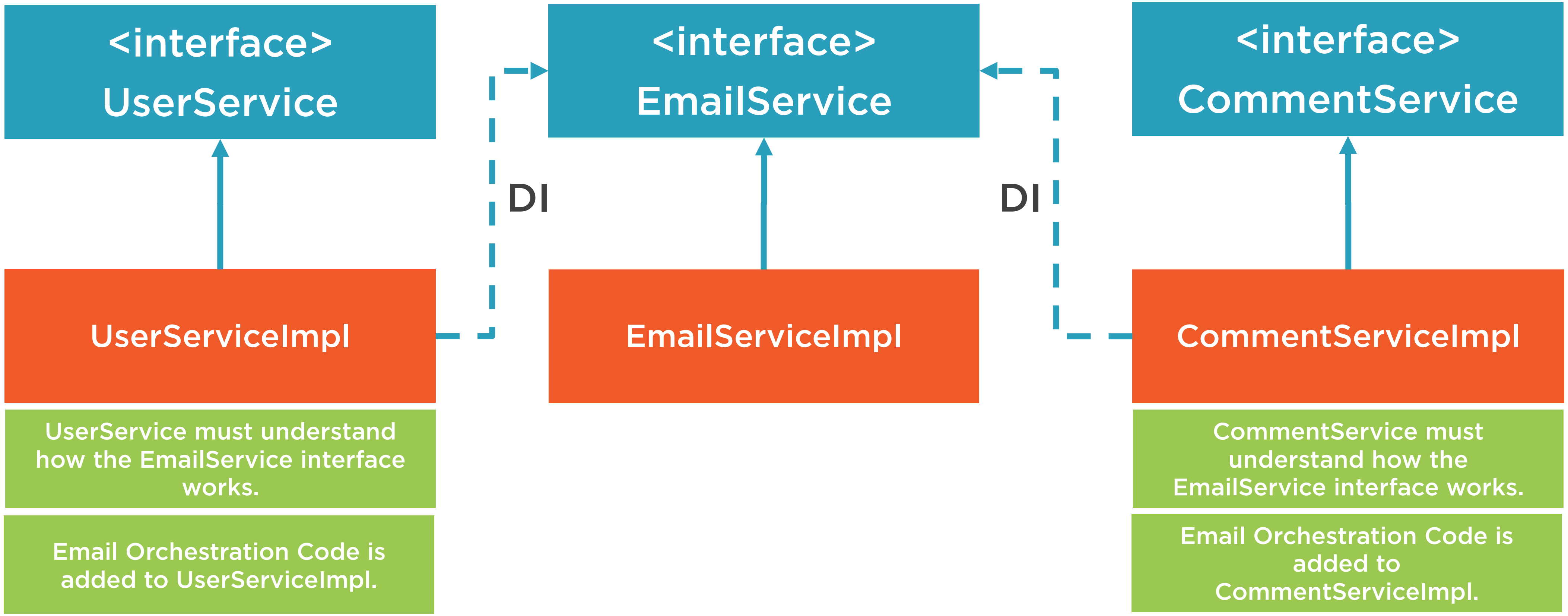
Avoiding Tight-Coupling of Services

Single Responsibility Principle

…is a computer programming principle that states that every [class] should have responsibility over a single part of the functionality provided by the software.

-- Wikipedia--

Event Driven Applications



<interface>
UserService

UserServiceImpl

UserService must understand how the EmailService interface works.

Email Orchestration Code is added to UserServiceImpl.

DI

<interface>
EmailService

EmailServiceImpl

DI

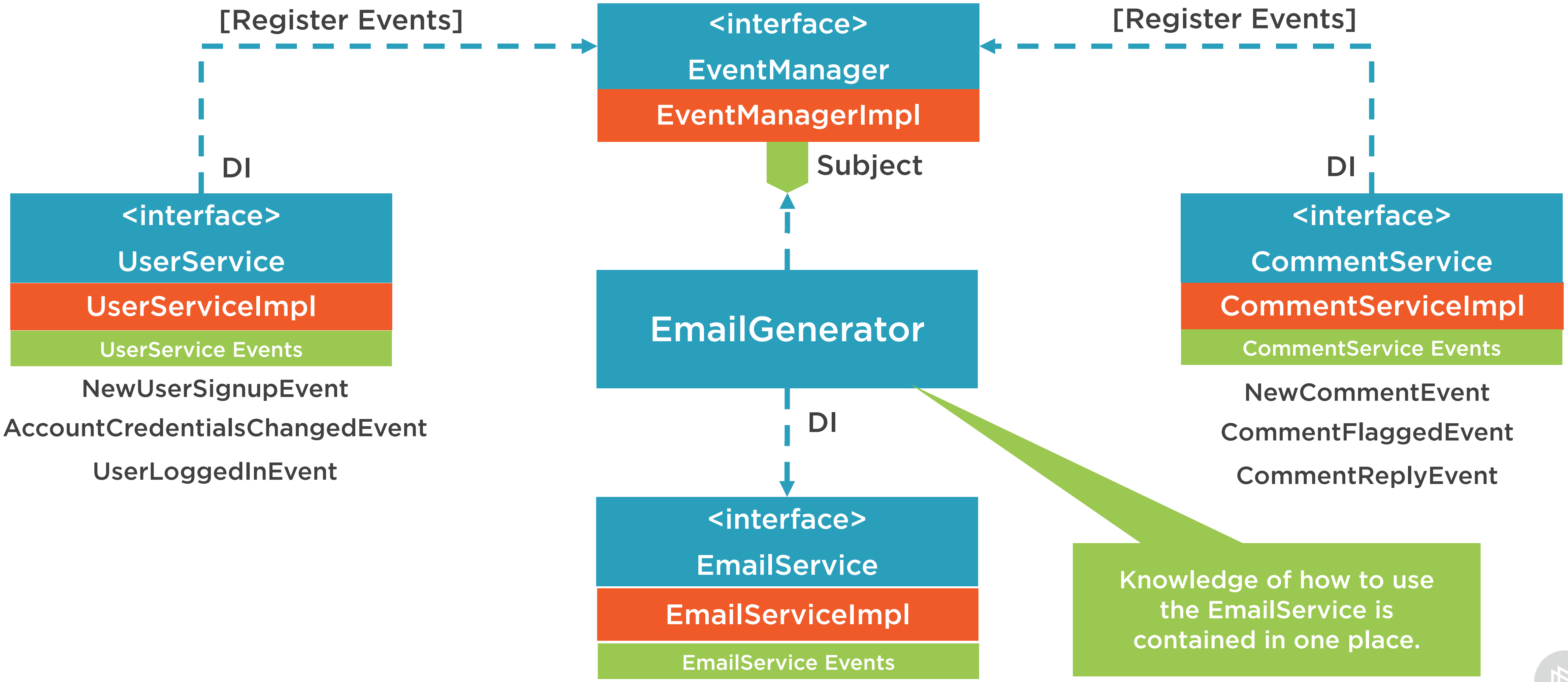
<interface>
CommentService

CommentServiceImpl

CommentService must understand how the EmailService interface works.

Email Orchestration Code is added to CommentServiceImpl.

Event Driven Applications



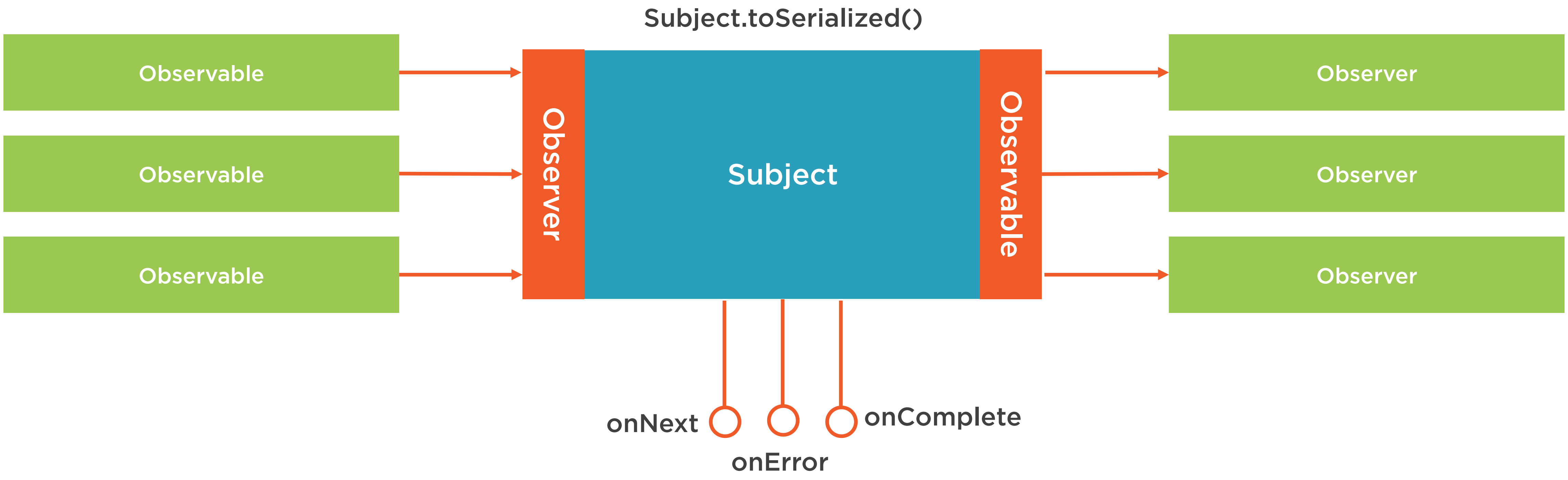
Knowledge of how to use the EmailService is contained in one place.

Subjects

Subjects

- PublishSubject
- BehaviorSubject
- AsyncSubject
- ReplaySubject

Subjects


Subject.toSerialized()
Observable
Observable
Observable
Observer
Subject
Observable
Observer
Observer
Observer
onNext
onError
onComplete

Subjects

PublishSubject

Does not replay past messages

BehaviorSubject

Replays the last message and all messages that come after

ReplaySubject

Replays all past messages to new subscribers

AsyncSubject

Emits only the last message

Demo

Event Driven Programming

Subjects

- PublishSubject
- BehaviorSubject
- ReplaySubject
- AsyncSubject