

Event Storming and DDD for the Gotthard Base Tunnel

JUG Zurich – April 25, 2023
Martin Häufel, Accenture
Stefan Heinzer, ELCA



RBC

Zug ist in Tunnel eingetroffen

RBC Zugdaten
- Verbindungsanforderung
- Betriebsart

Konflikt
- Einschüderung mit Zustand
- Reihenfolge - ist plan disponibel

Konflikte wurden an Disponenten gemeldet

Plan-... wurde...
- cache
- map
- push
- TA (M.1)

TA ADAPTER

Einschükung

TA hat Konflikt zu Zufahrt und Einschüfung gemeldet

Zurückgehaltener Stand

TA 13 RBC Daten

NEUE VMA von RBC GEFELDET

Freihaltungen

TAG-11:1

TAG-11:1 Betriebsart (Ereignis)

Konflikt-Validierung?

Schluss: Daten publiziert

Update

Update

TAG-14.3:1 Ist-Zone

TAG-14.1 Planzone (aktiv)

TAG-11:1 Betriebsart (Ereignis)

Erhaltungsbereich aktiviert

TA hat Ereignis in Betrieb gemeldet

Störungsbehebung für Zug erhalten

Freihaltungen für Zug werden aufgegeben

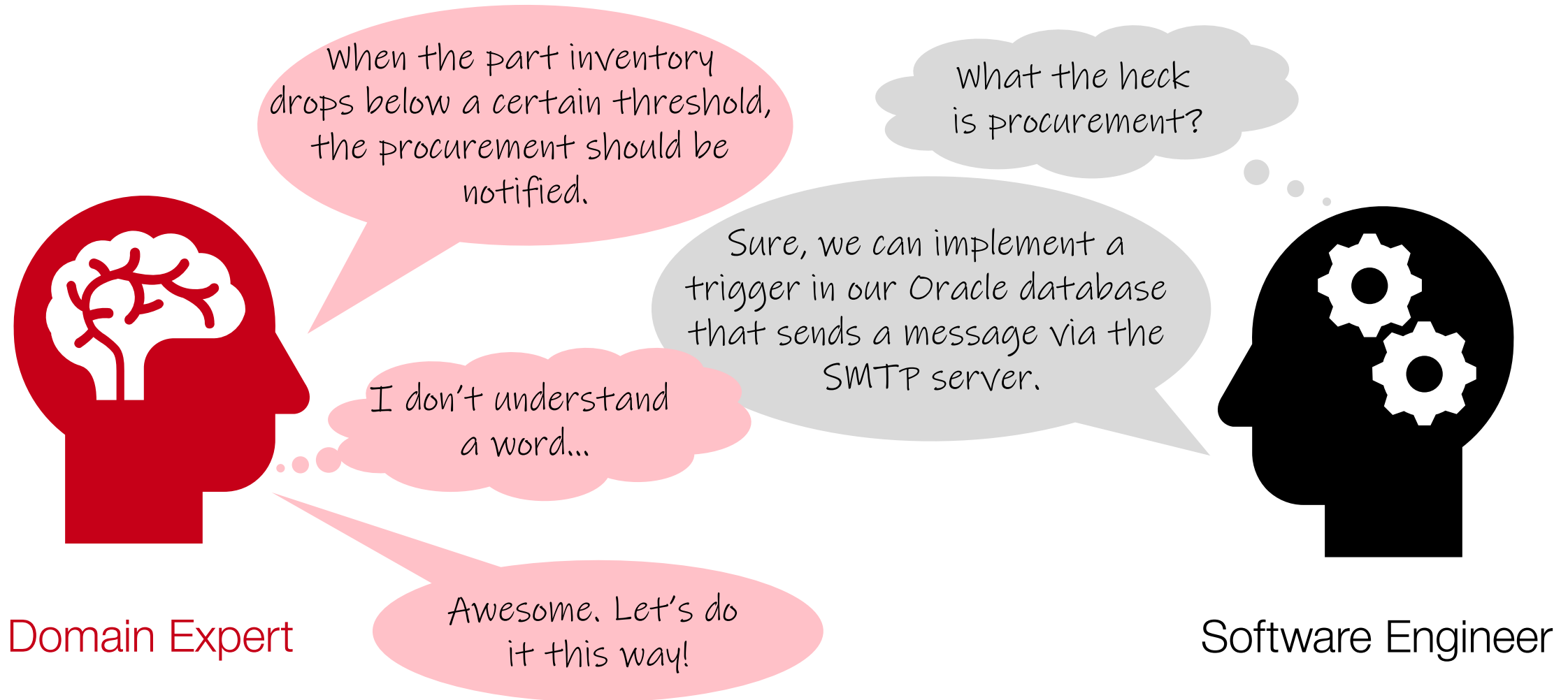
Einschränkung (M.1) TA aktiviert

TA kehrt von Ereignis - in Normalbetrieb zurück

Ist-Zone Typ Hindernis ist still das wurde gemeldet

Eine Weiche hat eine Störung gemeldet

A reason why projects struggle



About Martin



**accenture**

Martin Häufel

Technology Architect at Accenture, Munich.

Focus Topics

Cloud-Native software development and Event-Driven Architecture

Education

Physics Diploma at Technical University of Munich

Varia

39 years old, married

Hobbies: Listening to Jazz, skiing, gardening (since recently)

Supports iCow, a Kenyan startup company that provides small-holder farmers access to agricultural know-how through a mobile solution that does not require smartphones.

About Stefan



Stefan Heinzer

Lives with his family in Granada (ES) and works as senior architect and project coach at ELCA.

Focus Topics

Software Craftsmanship, Domain Driven Design, Event-Driven Architecture, Web & Cloud, Coaching, UX

Education

Ms. Sc. ETH in Computer Science
PhD in Biomedical Engineering Uni/ETH

Varia

46 years old, married, two kids in primary school
Hobbies: Music, travelling, hiking / bicycling, cooking
Working at ELCA Spain office until summer 2024

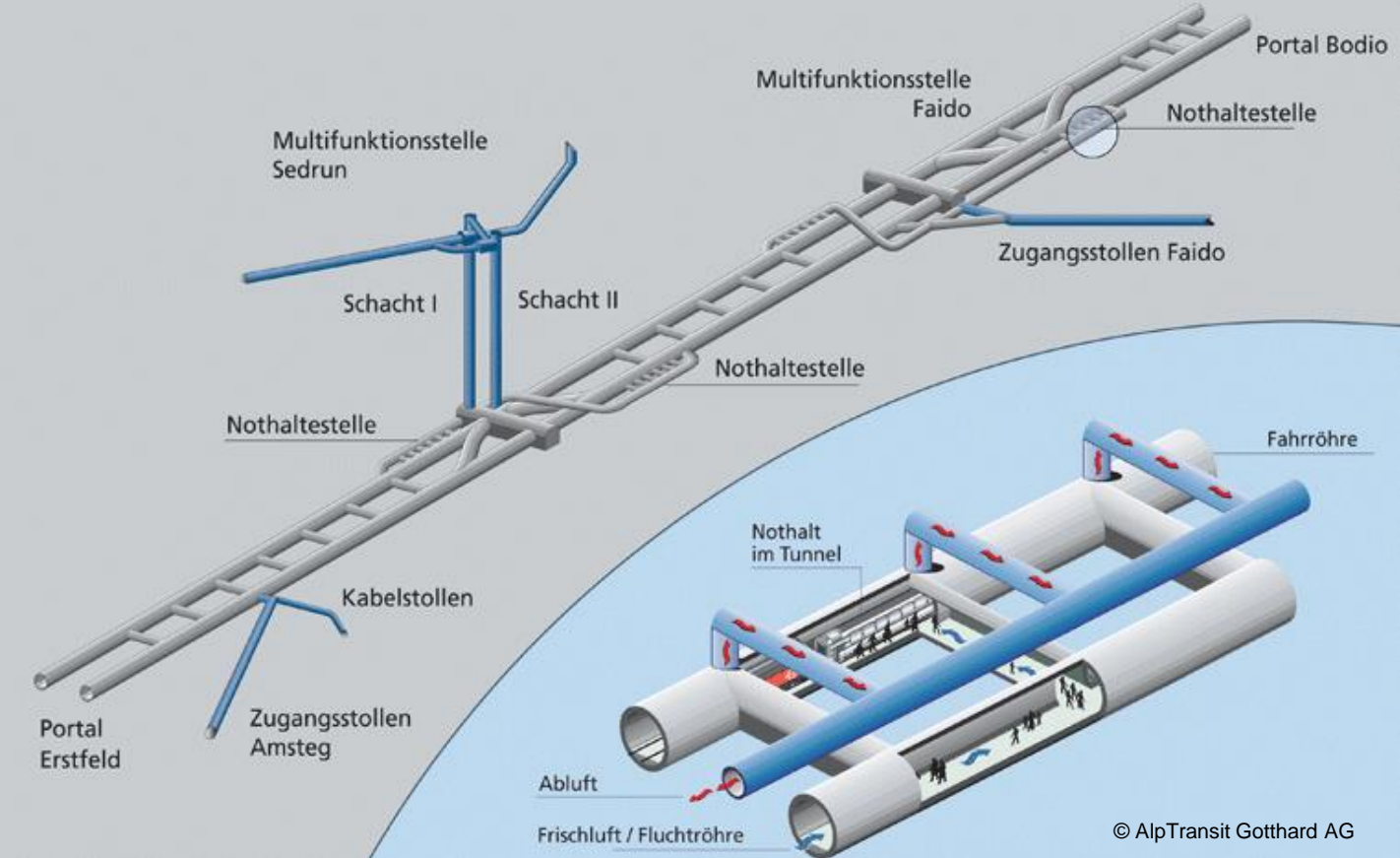
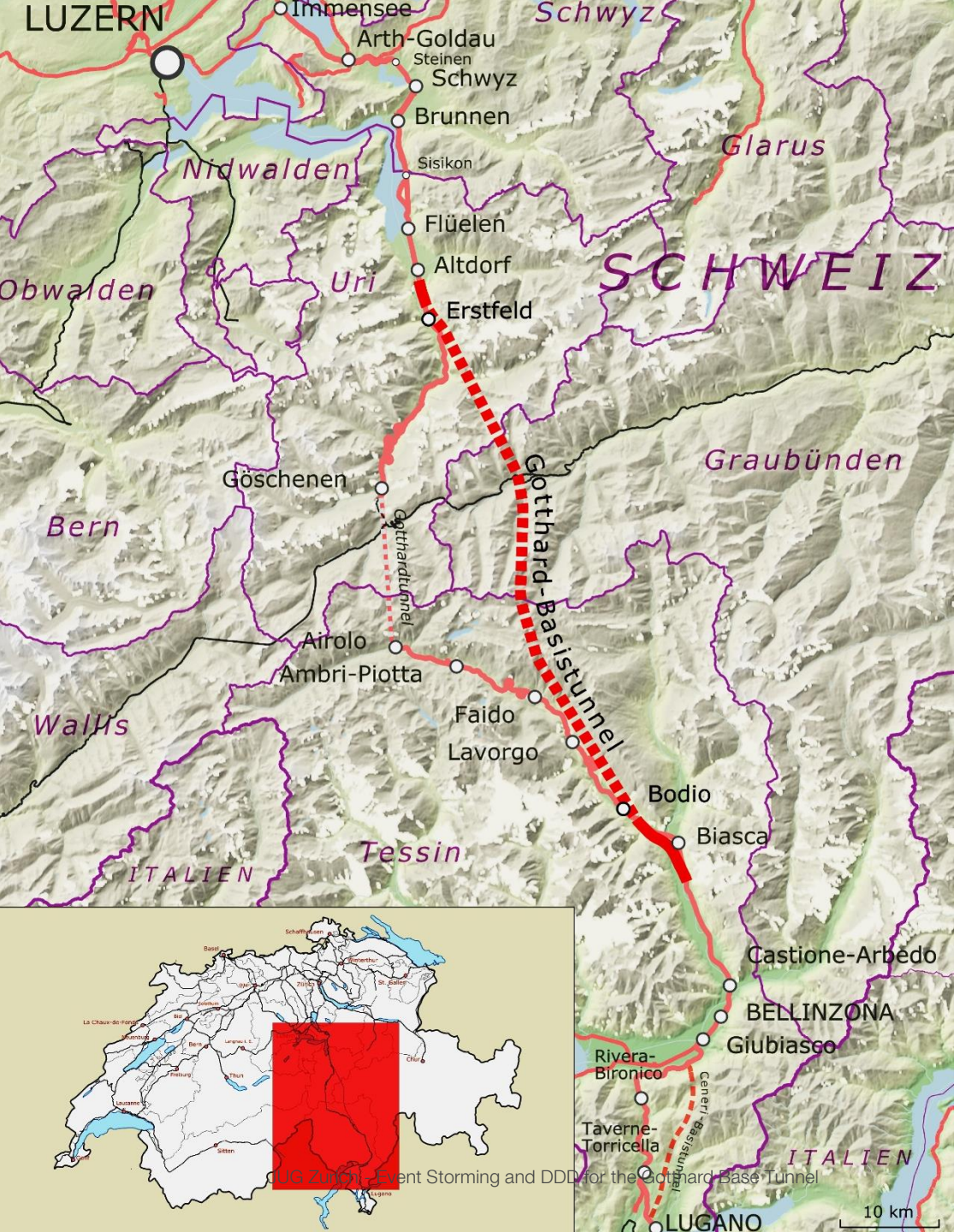


Agenda

1. Business context «tunnel automation adapter»
2. Modelling with Event Storming and DDD
3. Implementation with Java technology
4. Proven Coding Practices
5. DDD and CENELEC
6. Wrap-up



1 - Business context



© AlpTransit Gotthard AG

Gotthard base tunnel

Construction: 1999 - 2016
 Length: 57.1 km
 Cost: CHF 9.560 Billion
 Capacity: 200 – 260 freight and 65 passenger trains/day
 Max speed: 230 km/h (passenger), 100 – 160 km/h (freight)

UIC Zuber, Event Storming and DDD for the Gotthard Base Tunnel

10 km

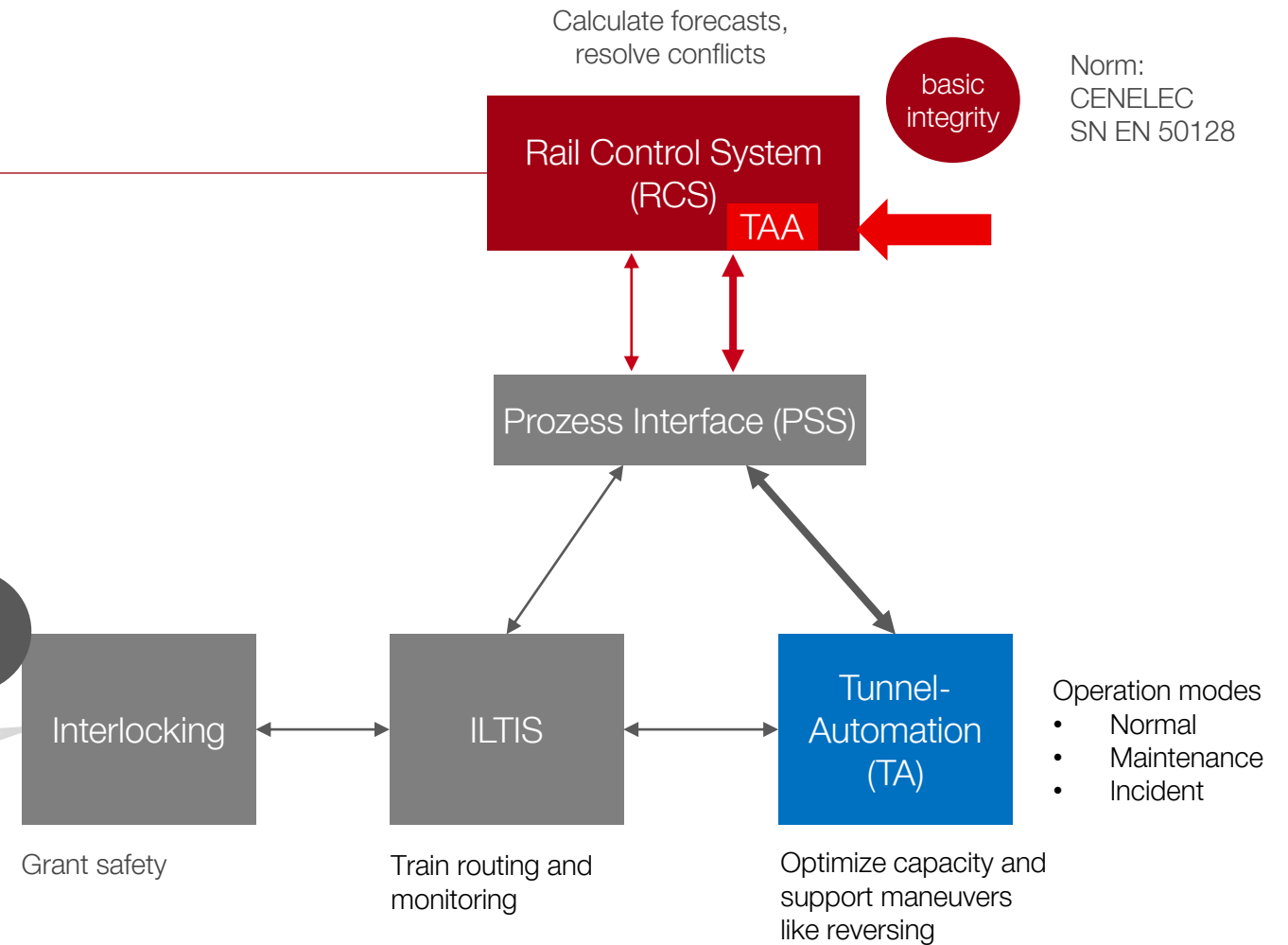
Context of the Tunnel Automation Adapter



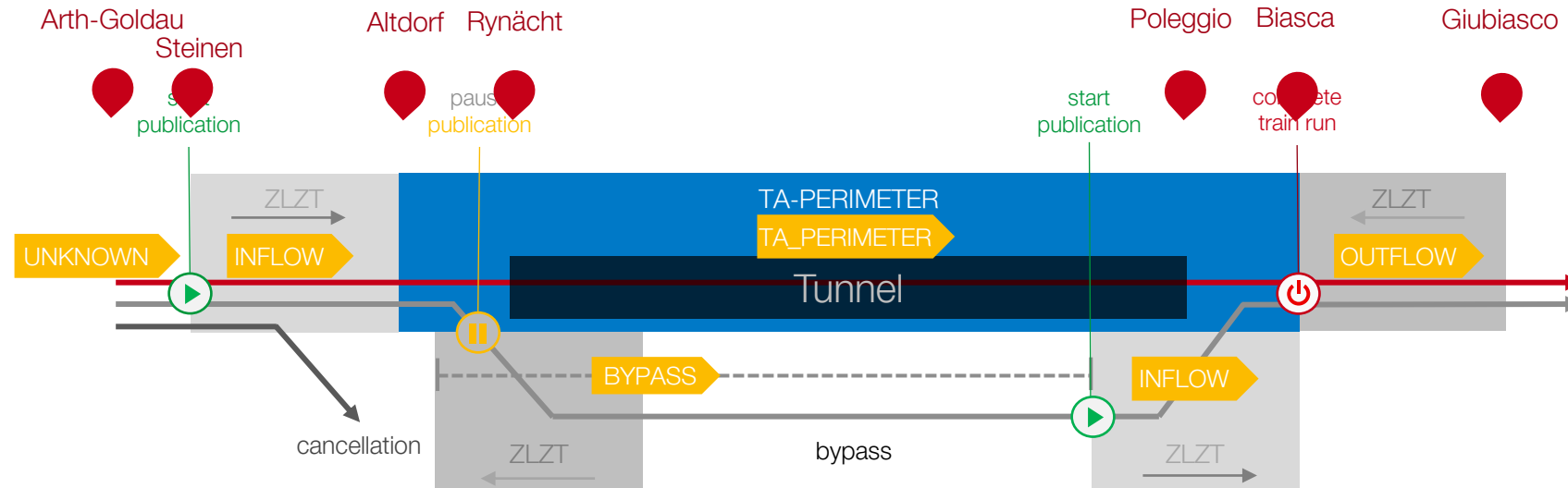
Operation center (disposition)



Points, Signals, On-Board Unit



Some Business Rules

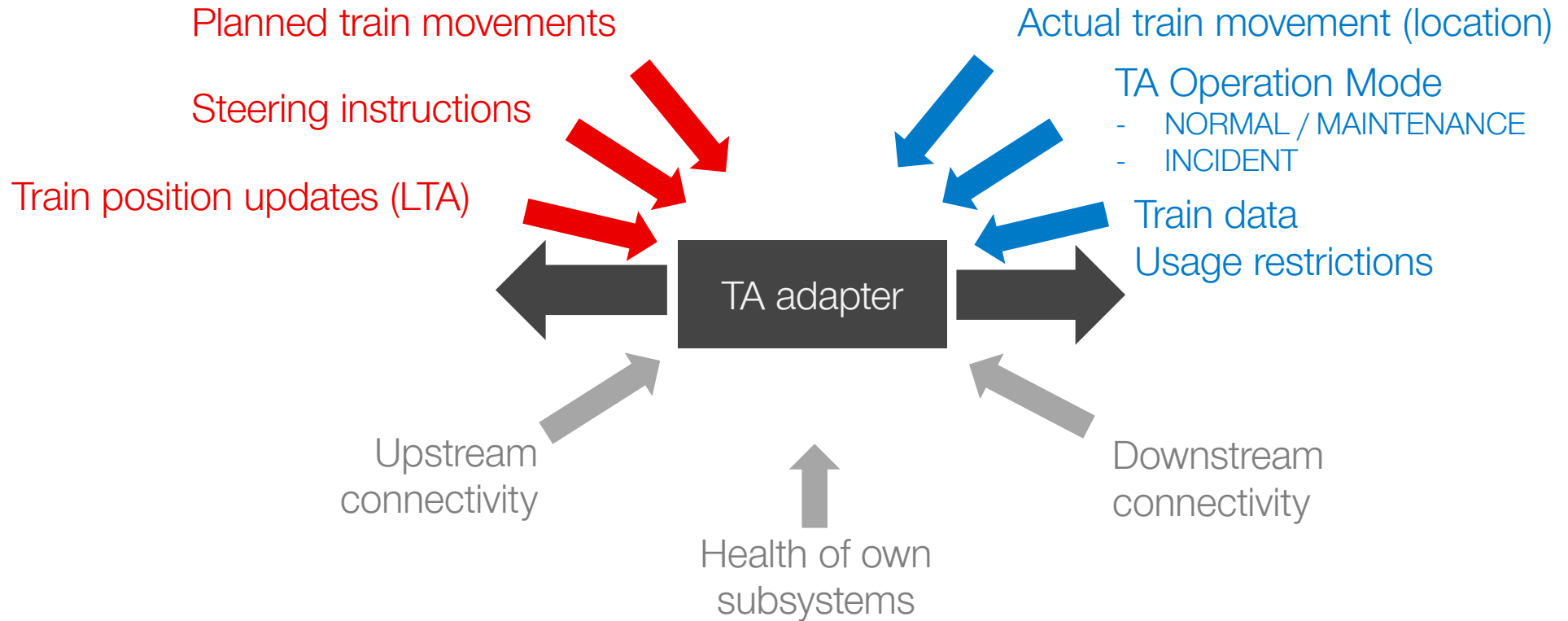


- Train attributes train length, carriage type (freight, passengers), etc. must be published when the train enters the TA inflow zone («ZLZT»)
- Train attributes must be republished when they change
- Publication must stop when the train leaves the TA perimeter
- Forecast Publication must be paused when TA operates in incident mode
- Publication must be paused when the train is in the bypass
- The TA can request the current state at any time («general transmission»)
- etc.

Challenges

Planning System (RCS)

TA / Control System (ILTIS)



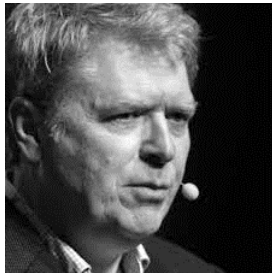
Events occur at any time → State changes at any time in any order



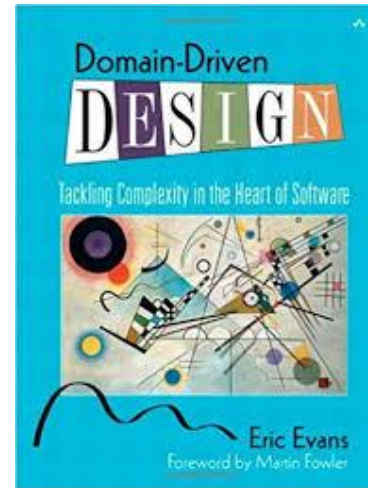
2 - Modelling with DDD and Event Storming

Domain Driven Design :: An Overview

«DDD is a Set of Guiding Principles»



Eric Evans



2003

Focus on the **core domain**

Explore models in a creative **collaboration** of domain practitioners and software practitioners

Speak a **ubiquitous language** within an explicitly bounded context

Let's align with the business!



Domain Expert



i.e. the same well-defined terms are used *everywhere*:

- conversations
- requirements specification
- test specification
- software design
- source code



Software Engineer

How can we model a system
and establish a common language?

Event Storming

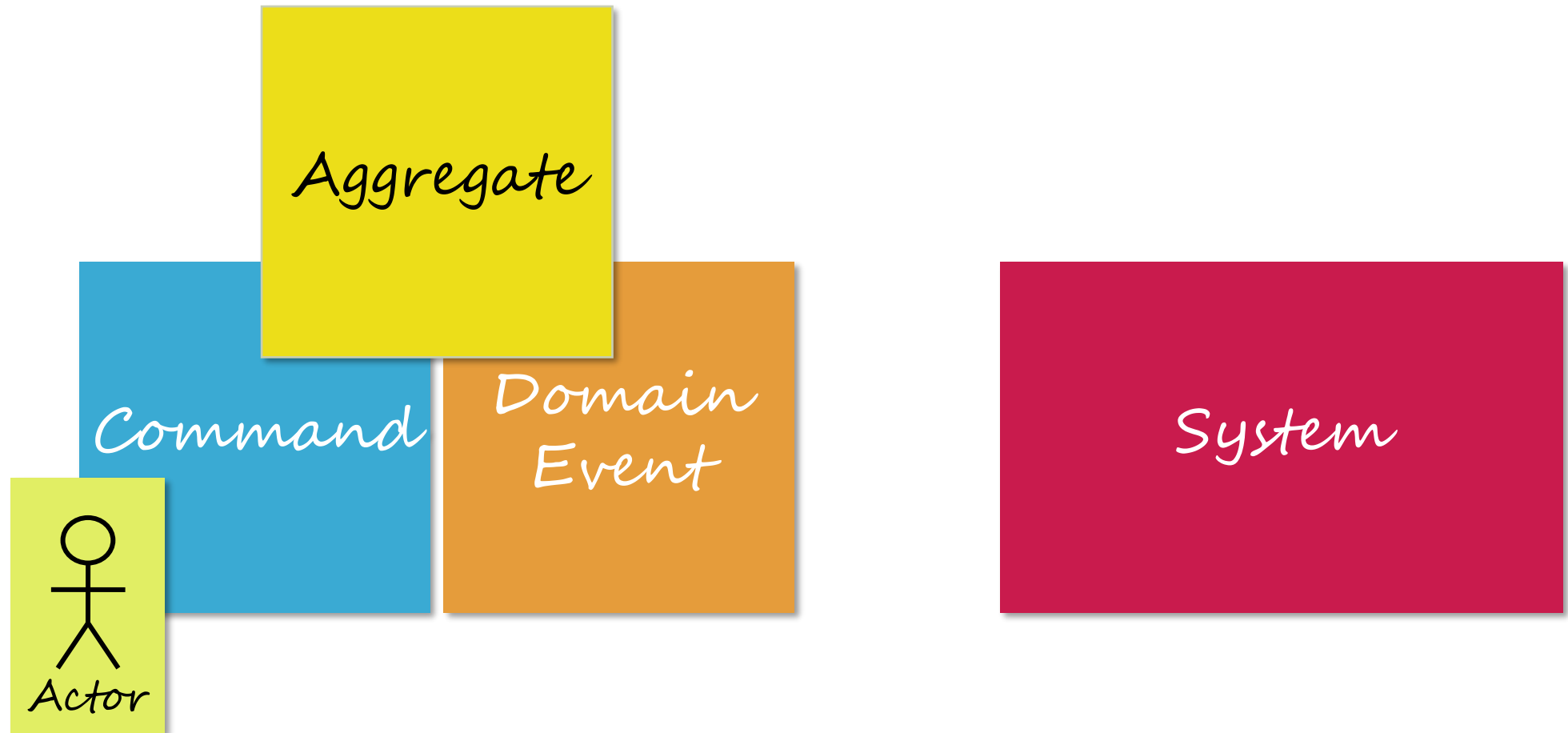
- Workshop Business and Dev People (1 – 2 days)
- Low tech to involve everyone
 - Lots of post-its of different colors
 - An “unlimited” modelling surface (e.g. a long wall)
- Start identifying Events: things that happen in the system
- Later identify Commands, Actors, Aggregates and Read Models
- Derive Bounded Contexts



Book by
Alberto Brandolini



Event Storming - Basic Modelling Elements



TA Adapter Event Storming – Release 1



Bounded context & Aggregates

Restriction
Train Run
Steering Instruction

Plan Change Detection

Update Publication to TA

Operating State

Plan Update



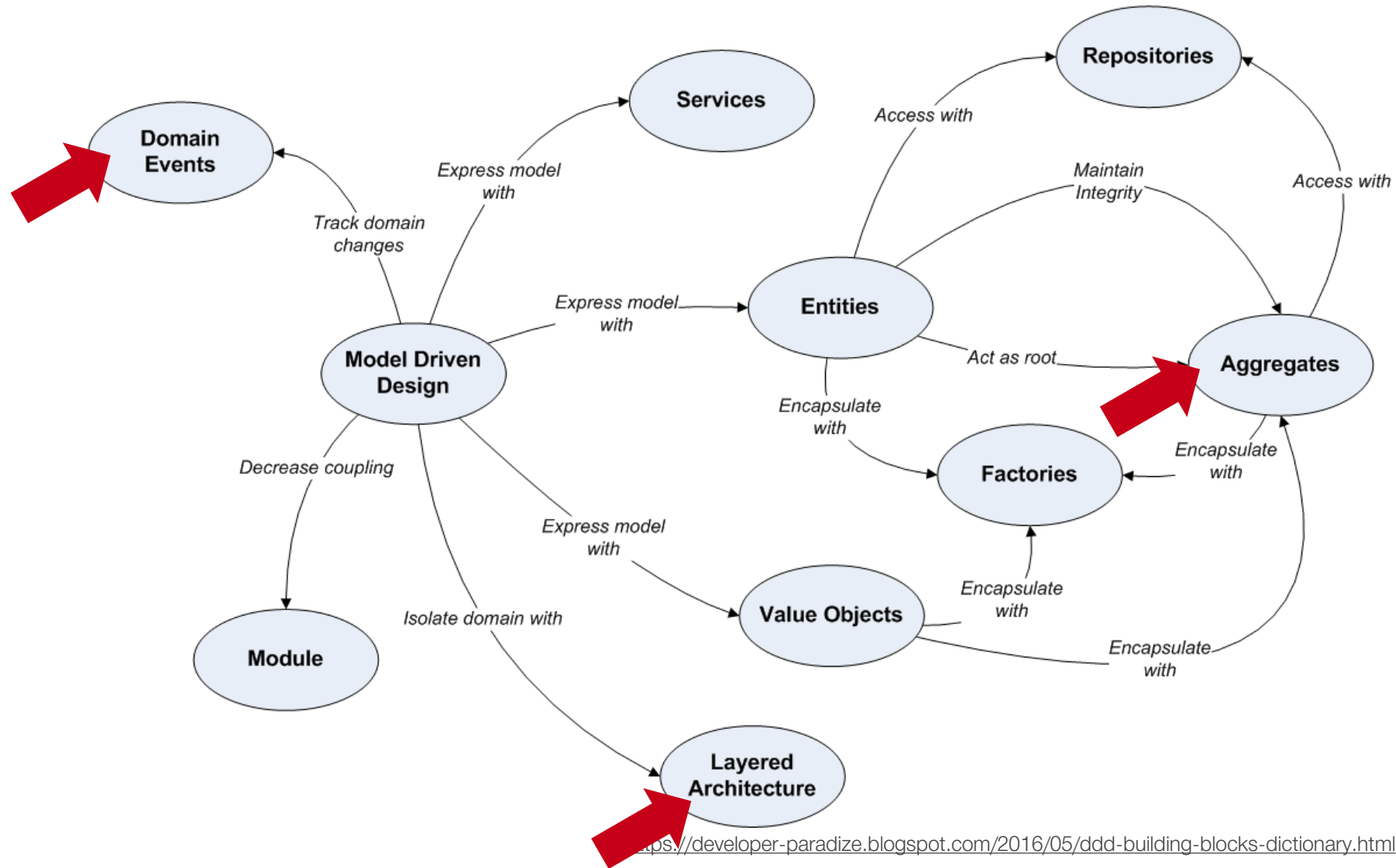


Event Storming Benefits

- Business and Tech People talk to each other (!)
- Massive knowledge gain for all participants
- Very effective / quick results
- Ubiquitous language emerges naturally
- Solid backbone for overall process
- Details can be refined during sprints

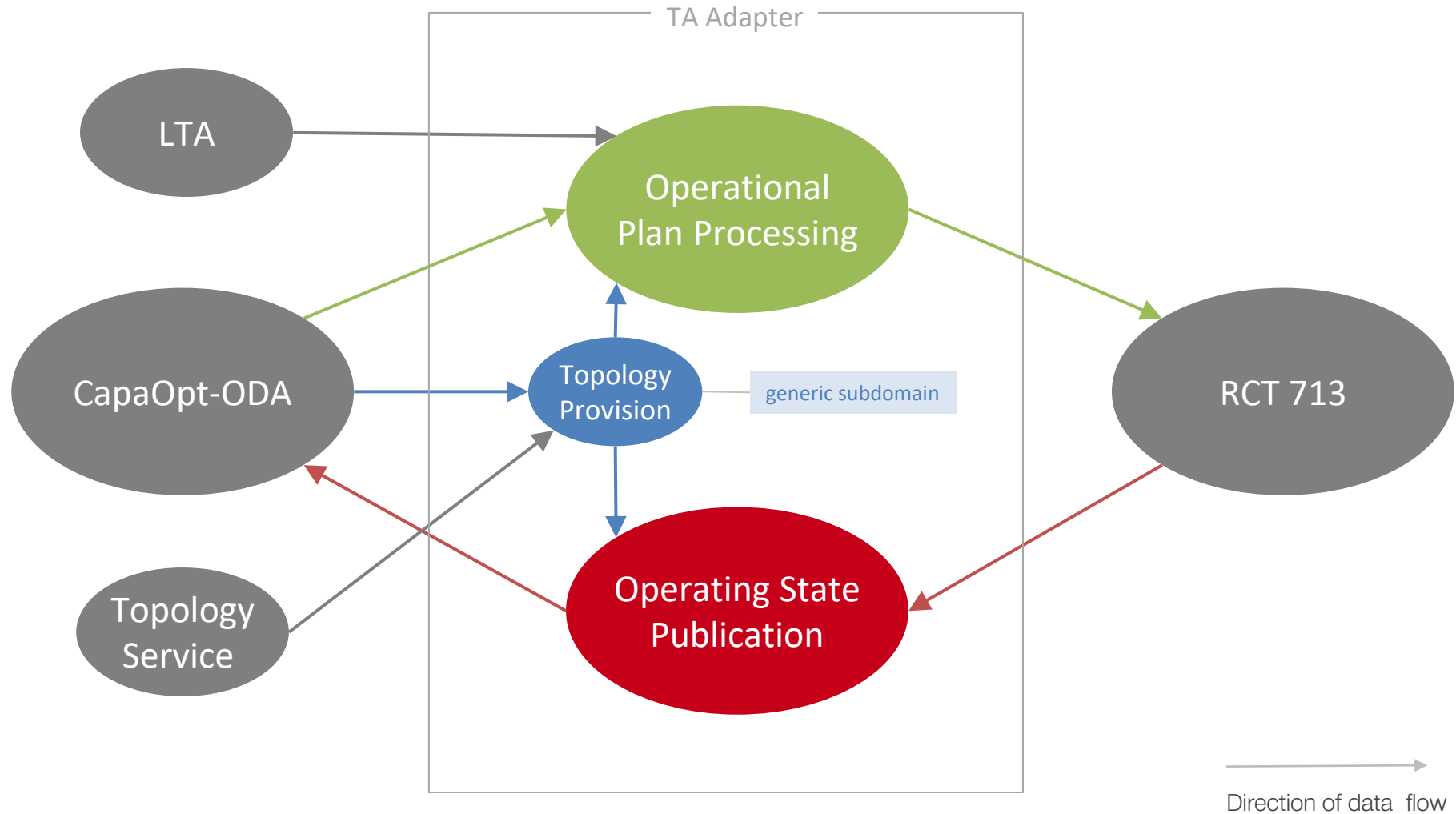
How do we get the
detailed design right?

DDD building blocks



<https://developer-paradize.blogspot.com/2016/05/ddd-building-blocks-dictionary.html>

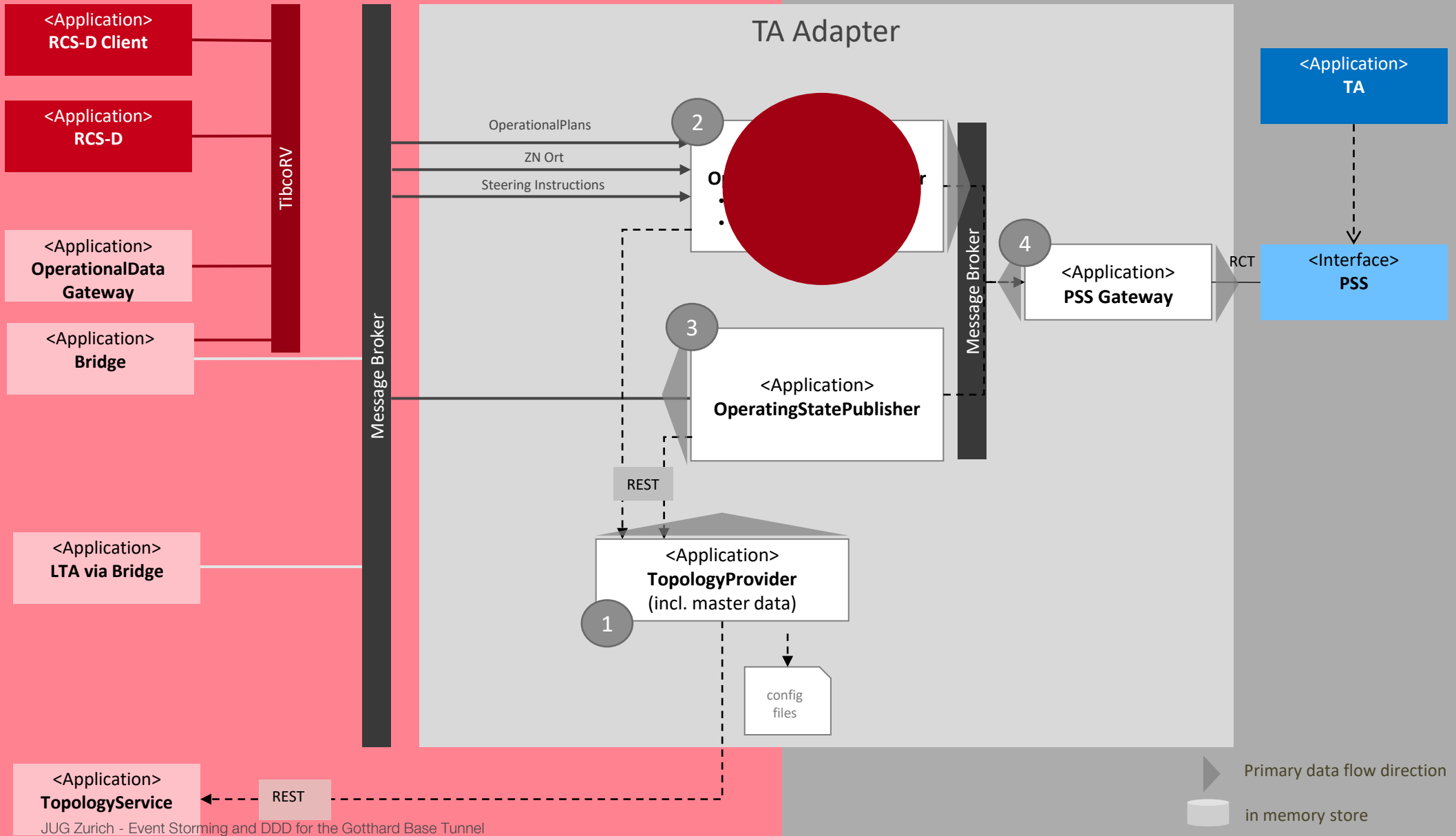
Bounded Contexts



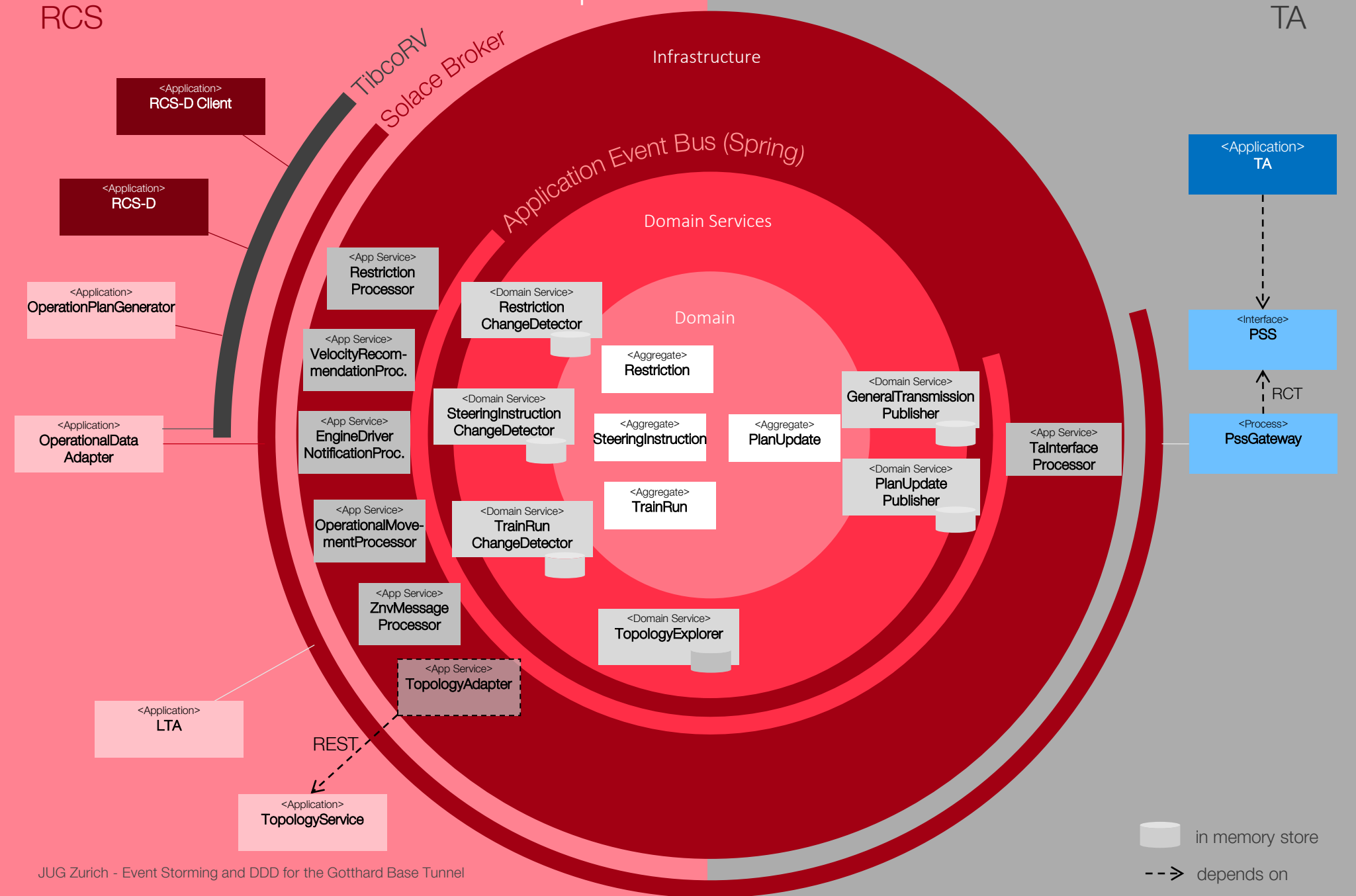
TA Adapter :: Top Level Architecture

RCS

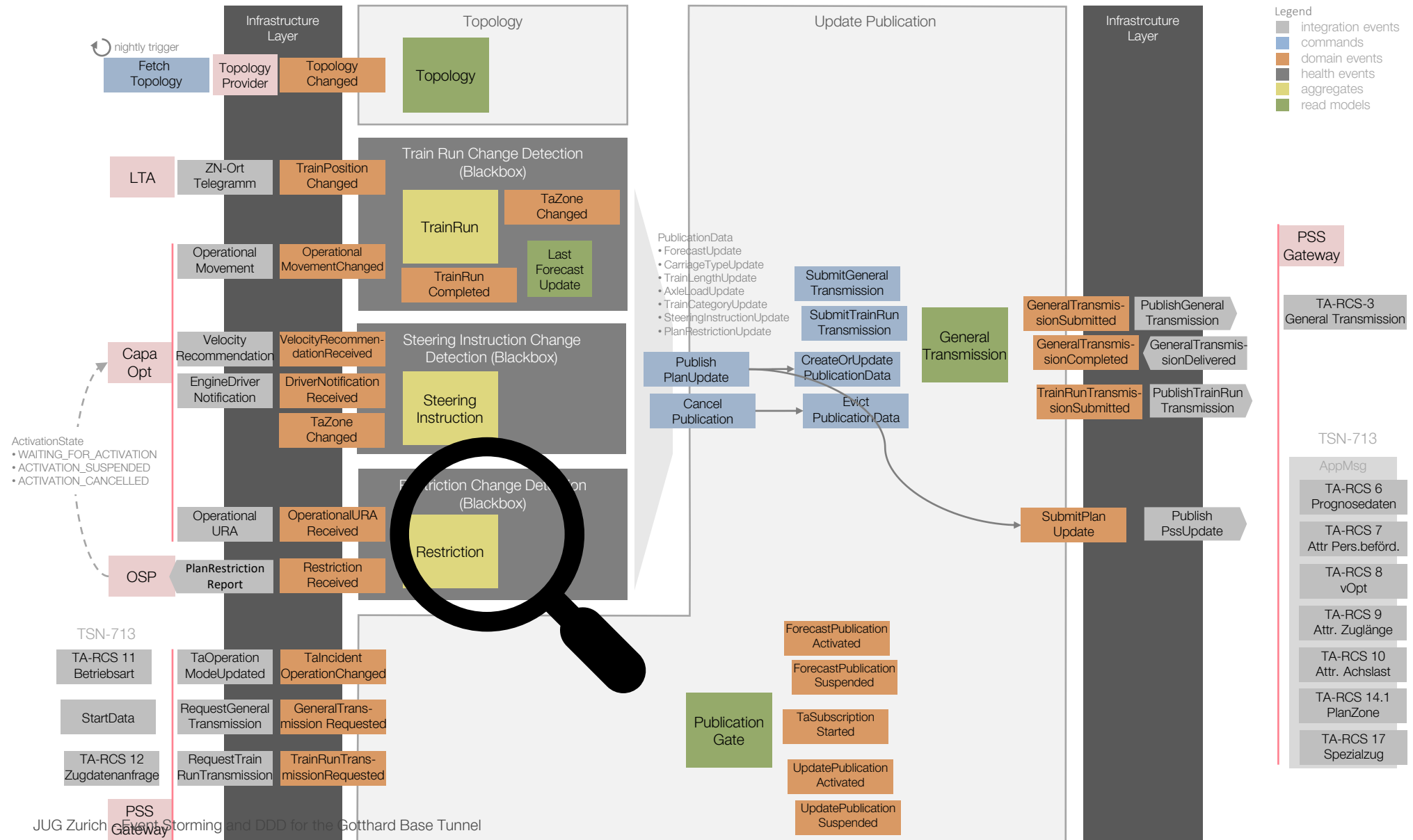
TA



TAA Operational Plan Processor



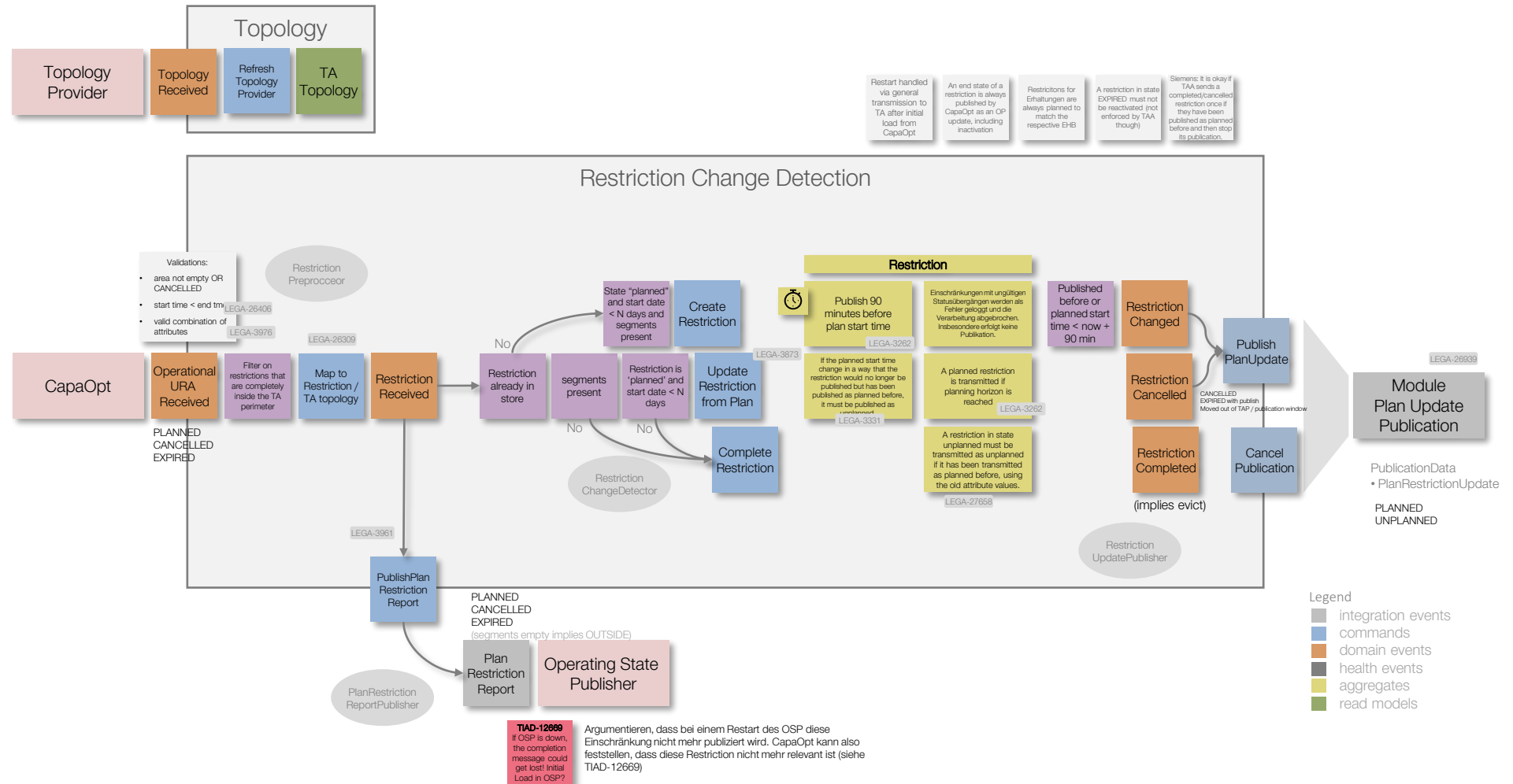
Event Model (TAA Operational Plan Processor)



Design Level Event-Storming



Module «Restriction Change Detection»





3 - Implementation with Java technology



Used Technology Stack

OpenJDK 17



- Spring Cloud Stream
- Spring Integration (TCP)



Solace PubSub+
Message Broker



Testing



JUnit 5

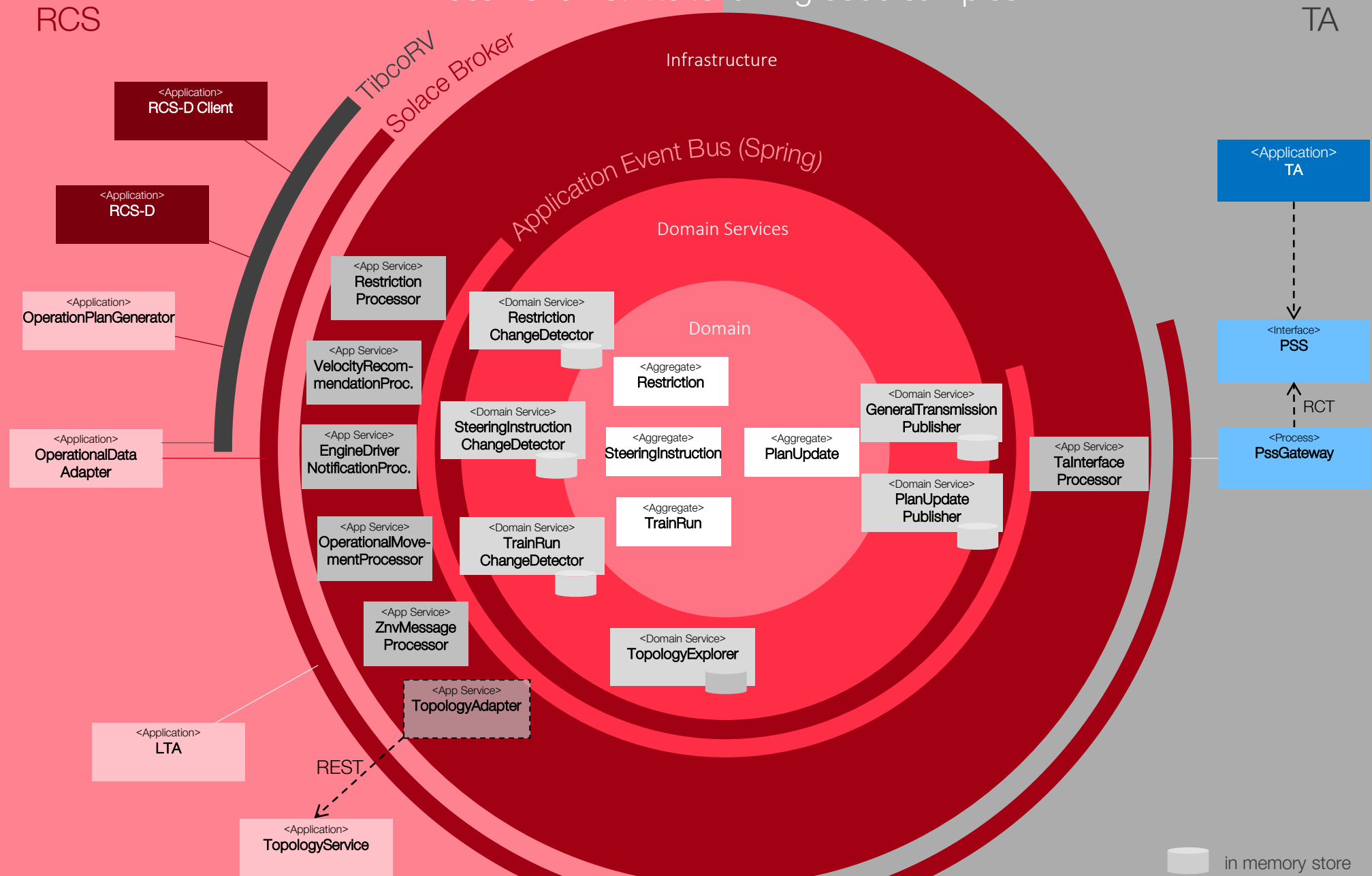


cucumber



Xray
for JIRA

Localization of the following code samples





Integration Event to Domain Event

```
23  @Slf4j
24  @Component
25  @RequiredArgsConstructor
26  public class OperationalUsageRestrictionAreaProcessor {
27
28      private final OperationalUsageRestrictionAreaValidator validator;
29      private final ApplicationEventPublisher applicationEventPublisher;
30
31
32      public Flux<OperationalUsageRestrictionArea> process(Flux<OperationalUsageRestrictionArea> operationalUsageRestrictionAreaFlux) {
33          return operationalUsageRestrictionAreaFlux
34              .filter(this::filter)
35              .filter(this::isValid)
36              .doOnNext(this::publish)
37              .onErrorContinue((throwable, obj) -> log.error("{} Error occurred upon processing operational usage restriction area",
38                  LogPrefix.onMessageFromUpstream(),
39                  throwable));
40      }
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67      private void publish(OperationalUsageRestrictionArea operationalUsageRestrictionArea) {
68          log.debug("{} {}", logPrefix(operationalUsageRestrictionArea), operationalUsageRestrictionArea);
69          applicationEventPublisher.publishEvent(OperationalUsageRestrictionAreaReceived.of(operationalUsageRestrictionArea));
70      }
```

Receive integration event from messaging middleware (binding via Spring Cloud Stream) <https://spring.io/projects/spring-cloud-stream>

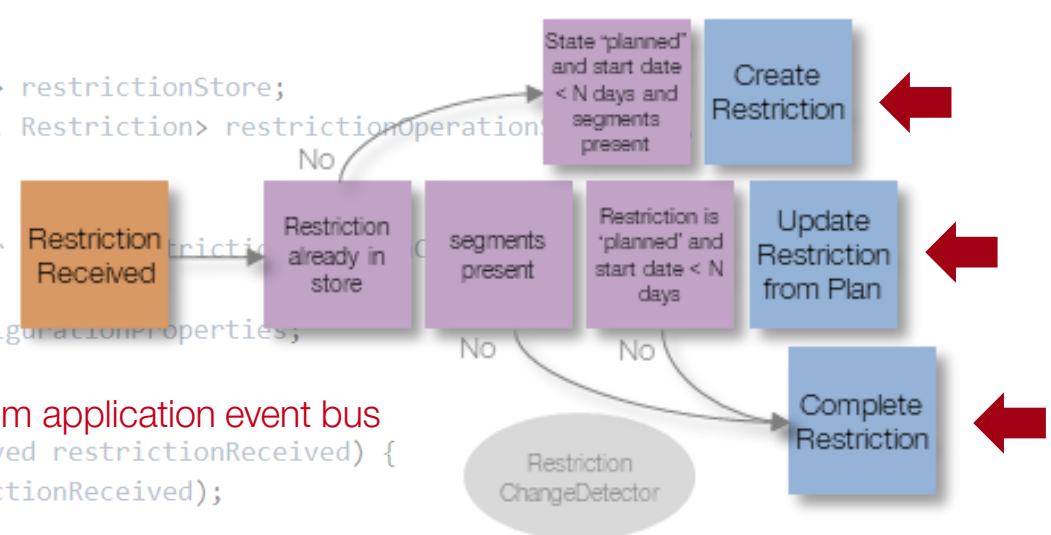
Publish domain event to internal application event bus

Domain Service :: Event to Command

```

75  @Slf4j
76  @DomainService
77  @RequiredArgsConstructor      ← Immutable, constructor injection
78  public class RestrictionChangeDetector {
79
80  private final AggregateStore<RestrictionId, Restriction> restrictionStore;
81  private final AggregateOperationScheduler<RestrictionId, Restriction> restrictionOperationScheduler;
82  private final NowProvider nowProvider;
83
84  private final RestrictionMutationChecker mutationChecker;
85
86  private final RestrictionChangeDetectionProperties configurationProperties;
87
88  @EventListener      ← Receive domain events from application event bus
89  public void processRestrictionReceived(RestrictionReceived restrictionReceived) {
90      log.debug("{} {}", LogPrefix.onProcessing(), restrictionReceived);
91      val now = nowProvider.now().toLocalDateTime();
92
93      val plannedRestriction = restrictionReceived.getPlannedRestriction();
94      val evictionCriteria = computeEvictionCriteria(now, plannedRestriction);
95      if (evictionCriteria.isEvictionRequired()) {
96          completeIfPresent(plannedRestriction.getRestrictionId(), evictionCriteria);
97          return;
98      }
99      restrictionStore.createOrUpdate(plannedRestriction.getRestrictionId())
100         .onCreate(id -> Optional.of(Restriction.create(plannedRestriction, getTiming(now))))
101         .onUpdate(restriction -> isChangeSupported(restriction, plannedRestriction)
102             ? restriction.updateFromPlan(plannedRestriction, getTiming(now))
103             : restriction)
104         .execute();      ← Commit aggregate and publish domain events

```





Aggregate «Restriction»

```

49     @Slf4j
50     @Getter
51     @Builder(toBuilder = true)
52     @EqualsAndHashCode(of = "id", callSuper = false)
53     public class Restriction extends AbstractAggregate<RestrictionId> {
54
55         @NonNull
56         private PlannedRestriction restrictionData;
57
58         @NonNull
59         @Getter(AccessLevel.NONE)
60         private Restriction.PublicationState publicationState;
61
62         @NonNull
63         @Builder.Default
64         private Optional<ScheduledFuture<?>> scheduledPublication = Optional.empty();

```

```

71     public static Restriction create(PlannedRestriction plannedRestriction, Timing timing) {
72         val result = Restriction.builder()
73             .restrictionData(plannedRestriction)
74             .publicationState(PublicationState.UNPUBLISHED)
75             .build();
76         result.registerEvent(RestrictionRegistered.of(result.getId()));
77         result.updateRestrictionData(plannedRestriction, timing);
78         return result;
79     }

```

Static factory with expressive name

← Domain event registration

← Return self-reference to enable chaining

Domain methods instead of setters

updateFromPlan()

```
81     public Restriction updateFromPlan(PlannedRestriction plannedRestriction, Timing timing) {
82         if (publicationState == PublicationState.COMPLETED) {
83             throw new IllegalStateException("Restriction " + getId() + " has been completed");
84         }
85         if (!plannedRestriction.getRestrictionId().equals(getId())) {
86             throw new IllegalArgumentException("Restriction ID must match: " + plannedRestriction.getRestrictionId() + " != " + restricti
87         }
88         return updateRestrictionData(plannedRestriction, timing);
89     }
```

Enforce business rules

← Update state and register domain events (if any)

complete()

```
140     public Restriction complete() {
141         if (publicationState == PublicationState.COMPLETED) {
142             throw new IllegalStateException("Restriction " + getId() + " has already been completed");
143         }
144         cancelScheduledPublication();
145         notifyCancelledIfPublishedBefore();
146         registerEvent(RestrictionCompleted.of(restrictionData));
147         publicationState = PublicationState.COMPLETED;
148         return this;
149     }
```

← Register domain event



Demo: Domain Events in Splunk

New Search Save As ▾ Create Table View Close

index=sbb_tms_internal_prod_events openshift_namespace=tms-iaad-rti-prod openshift_container_name=taa-operational-plan-processor 41027 NOT ForecastChanged Last 60 minutes ▾ 🔍

✓ 99 events (6/21/22 9:10:00.000 AM to 6/21/22 10:10:43.000 AM) No Event Sampling ▾ Job ▾ ⏸ ⏪ ⏩ 📄 ⬇ 🔦 Smart Mode ▾

Events (99) Patterns Statistics Visualization

Format Timeline ▾ — Zoom Out + Zoom to Selection × Deselect 1 minute per column

Raw ▾ ✍ Format 20 Per Page ▾ < Prev 1 2 3 4 5 Next >

< Hide Fields ☰ All Fields	i Event
SELECTED FIELDS a host 1 a openshift_container_name 1 a source 1 a sourcetype 1	> 2022-06-21 09:54:30.777 INFO 1 --- [ool-16-thread-1] c.s.t.i.t.o.c.a.AggregateStoreImpl : <e> TRAIN [41027 / 2022-06-21]: TrainLengthChanged(trainRunId=41027(2022-06-21), oldValue=Optional[Length(valueInMeters=578)], newValue=Length(valueInMeters=578))
INTERESTING FIELDS a app 1 a application_id 1 a cluster 1 a dest 1 a dest_ip 1 a dns 1 a dvc 1	> 2022-06-21 09:54:30.777 INFO 1 --- [ool-16-thread-1] c.s.t.i.t.o.c.a.AggregateStoreImpl : <e> TRAIN [41027 / 2022-06-21]: PlanUpdateCreated(publicationData=ForecastUpdate(trainRunId=41027(2022-06-21), earliestForecast=2022-06-21T10:14:38, latestForecast=2022-06-21T10:54:40.440))
	> 2022-06-21 09:54:30.777 INFO 1 --- [ool-16-thread-1] c.s.t.i.t.o.c.a.AggregateStoreImpl : <e> TRAIN [41027 / 2022-06-21]: TaZoneChanged(trainRunId=41027(2022-06-21), UNKNOWN --> INFLOW)
	> 2022-06-21 09:54:30.776 INFO 1 --- [ool-16-thread-1] c.s.t.i.t.o.t.ZnvMessageProcessor : >> TRAIN [41027 - Standard]: ZnvMessage train position on route field STN@213 with timestamp 2022-06-21T09:54:30.730+02:00[Europe/Berlin] for cell ZGD
	> 2022-06-21 09:54:27.249 INFO 1 --- [pool-2-thread-1] s.t.i.t.o.u.GeneralTransmissionPublisher : >> TRAIN [41027 / 2022-06-21]: Publishing 1 train run data for train run
	> 2022-06-21 09:54:27.249 INFO 1 --- [ool-19-thread-1] c.s.t.i.t.o.t.TaInterfaceProcessor : << TRAIN [41027 / 2022-06-21]: RequestTrainRunTransmission(trainNumber=41027, prodDatum=2022-06-21)
	> 2022-06-21 09:53:54.687 INFO 1 --- [ool-16-thread-1] c.s.t.i.t.o.t.ZnvMessageProcessor : >> TRAIN [41027 - Standard]: ZnvMessage train position on route field STN@212 with timestamp 2022-06-21T09:53:54.649+02:00[Europe/Berlin] for cell ZGD





4 - Proven coding practices



Coding Practices :: Overview


1. Null-free programming policy using Optional
2. Immutable objects with Lombok
3. Event-driven integration tests, Cucumber

Null-Free Programming using Optional

Rule

If you cannot guarantee that a variable is never null, model it as an `Optional<T>`

```
53 |     public class Restriction extends AbstractAggregate<RestrictionId> {  
54 |  
55 |         @NonNull  
56 |         private PlannedRestriction restrictionData;  
  
62 |         @NonNull  
63 |         @Builder.Default  
64 |         private Optional<ScheduledFuture<?>> scheduledPublication = Optional.empty();
```



Benefits

- No need for null checks, cleaner code
- Detect design flaws more easily
- Less `NullPointerException`s

Drawbacks

- Not in line with JDK, which uses `null` in various APIs
- Marked as code smell by Sonar when used on fields
- Team has to get used to it

Immutable Objects with Lombok

Declaration of a value object (immutable)

```

8 @Value
9 @Builder
10 public class PlanUpdateRenewed implements PlanUpdateEvent {
11
12     @NonNull
13     PublicationData previousPublicationData;
14
15     @NonNull
16     PublicationData publicationData;
17
18 }

```

Instantiation using the builder

```

registerEvent(PlanUpdateRenewed.builder()
    .previousPublicationData(this.publicationData)
    .publicationData(publicationData)
    .build());

```



Effective class

```

ch.sbb.tms.iad.rti.taadapter.updatepublication.events
└─ PlanUpdateRenewed
    └─ PlanUpdateRenewedBuilder
        ├── previousPublicationData : PublicationData
        ├── publicationData : PublicationData
        ├── PlanUpdateRenewedBuilder()
        ├── build() : PlanUpdateRenewed
        ├── previousPublicationData(@NonNull PublicationData) : PlanUpdateRenewedBuilder
        ├── publicationData(@NonNull PublicationData) : PlanUpdateRenewedBuilder
        ├── toString() : String
        └─ builder() : PlanUpdateRenewedBuilder
            ├── previousPublicationData : PublicationData
            └─ publicationData : PublicationData
    └─ PlanUpdateRenewed(@NonNull PublicationData, @NonNull PublicationData)
        ├── equals(Object) : boolean
        ├── getPreviousPublicationData() : PublicationData
        ├── getPublicationData() : PublicationData
        ├── hashCode() : int
        └─ toString() : String

```

Event-Driven Integration Testing

1 Arrange

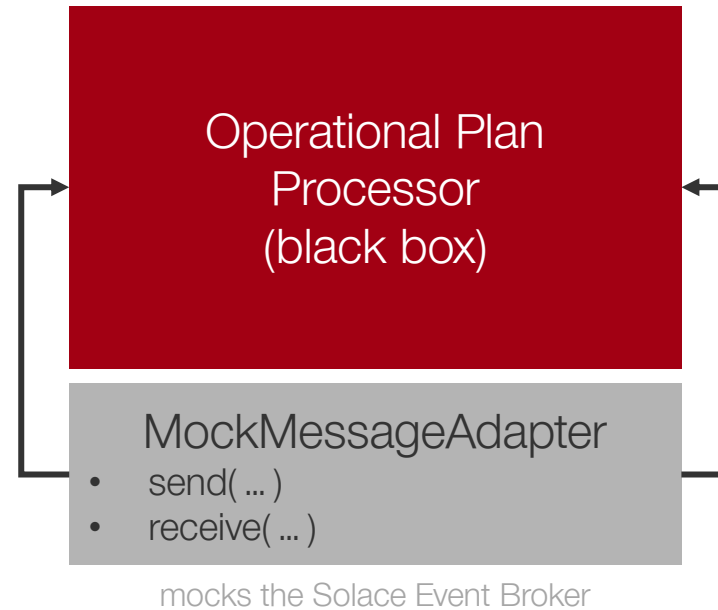
start application

Initialize desired state

2 Act

`send(operationalMovement)`

`send(trainPosition)`



3 Assert

`result = receive(PlanUpdate.class)`

`assertThat(result)...`

Use integration events only!

Feature: Forecast data

Scenario: Forecast data: Train begins before the inflow zone and then enters the inflow zone

- Given Advanced Topology is loaded
And TAA Operational Plan Processor is ready

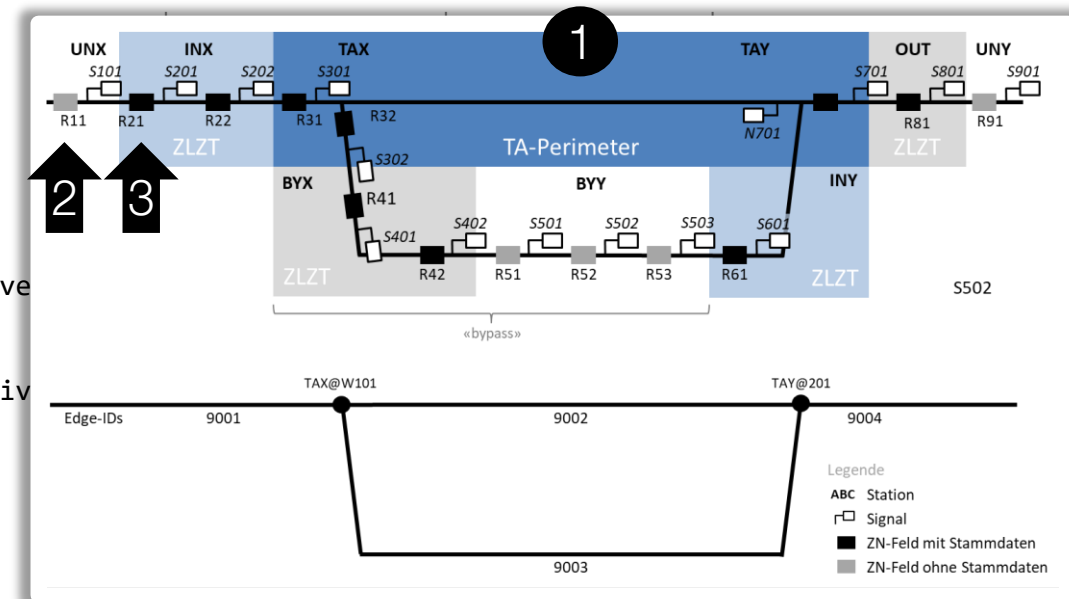
When OperationalMovement for train "300" with 1 segments is received
And Segment 1 with trainlength=200 has the following events

position	event	signalId	OP	time	trainCategory	passengerTransport	edgeId	offset
1	OperationPointEvent TrainCategoryEvent EntryPermissionEvent DepartureEvent		UNX	18:00:00	LRZ ENTRY_PERMISSION_EXPIRED	true	9001	100
2	ReservationStartEvent SignalEvent PassthroughEvent	101		18:00:30				
3	SignalEvent PassthroughEvent	201		18:00:52				
...								
12	SignalEvent ReservationEndEvent	801		18:08:00				

- And ZnvMessage with route field "UNX@R11" for train "300" is received
Then no ForecastUpdate for train "300" is published

- When ZnvMessage with route field "INX@R21" for train "300" is received
Then ForecastUpdate for train "300" is published as follows

referencePointId	arrivalTime	departureTime
TAX@S301	18:03:55	18:03:55
TAY@S701	18:07:00	18:07:00

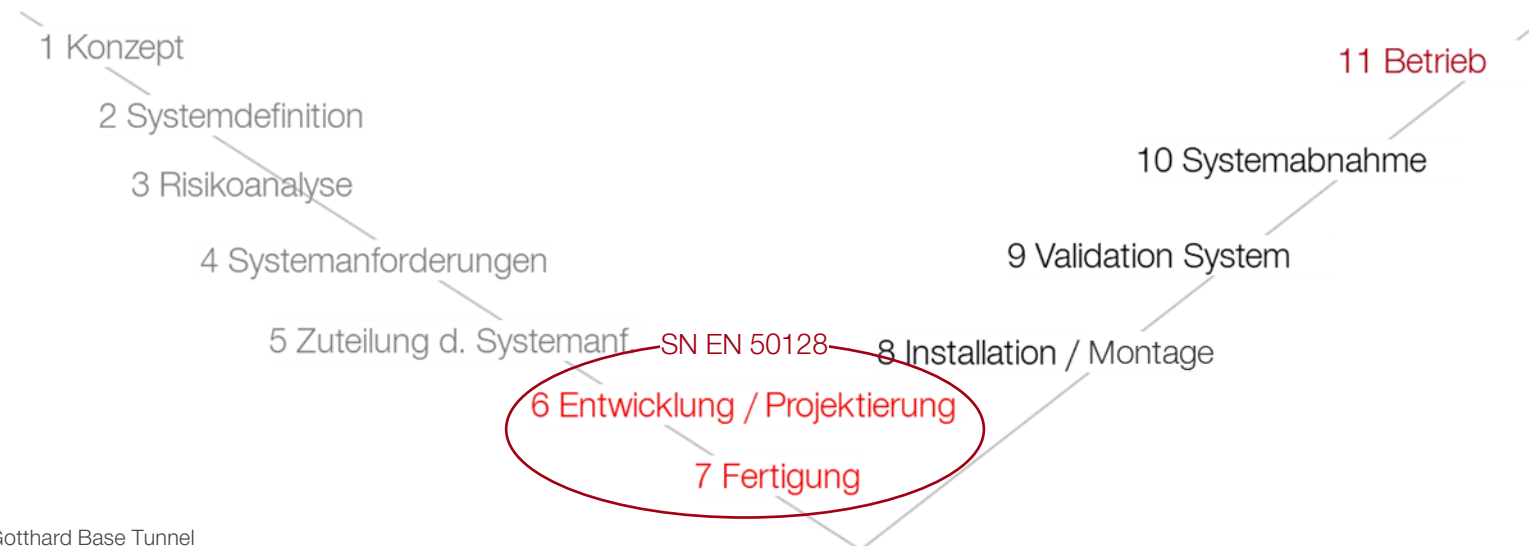




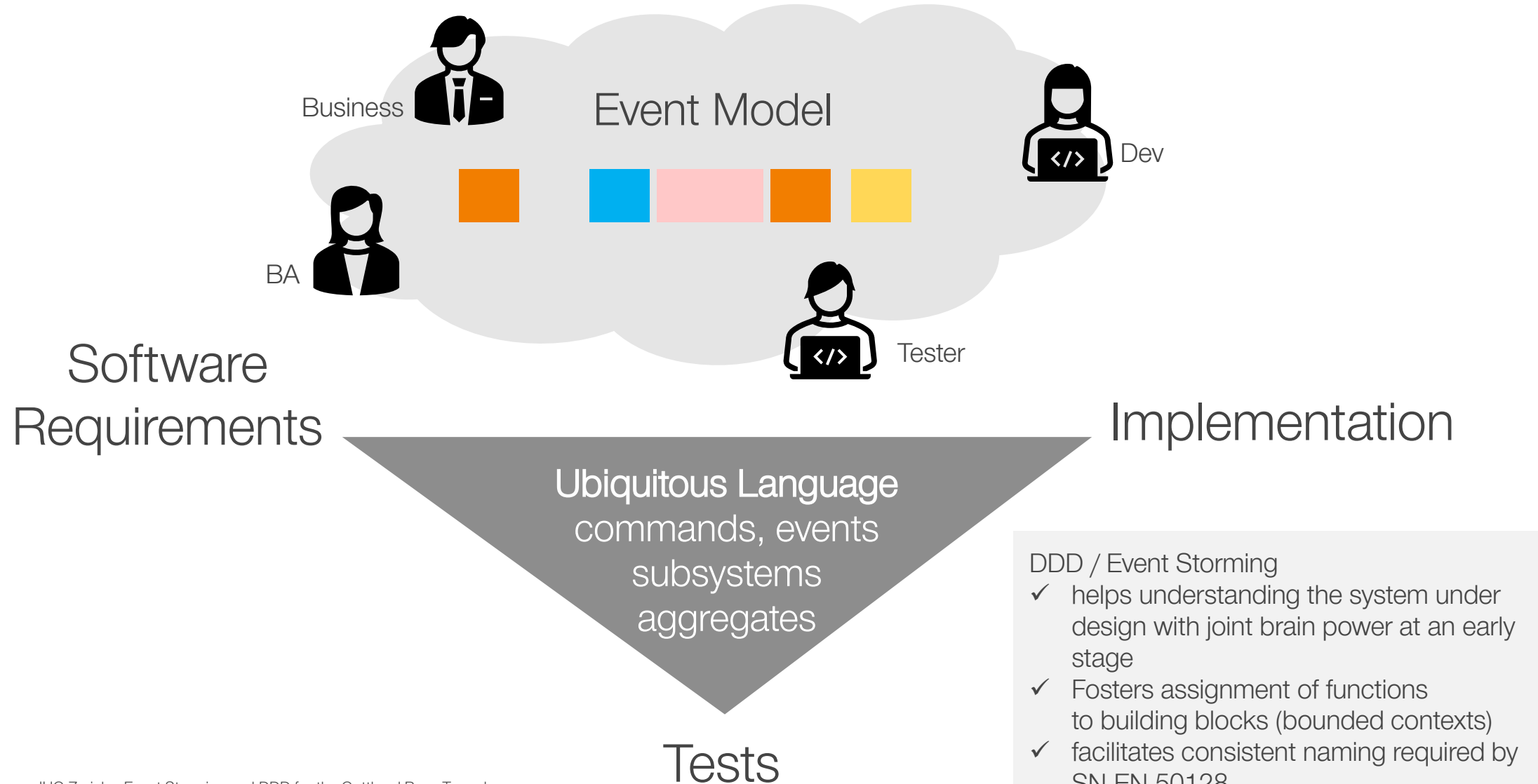
5 – DDD & CENELEC

What is CENELEC SN EN 50126?

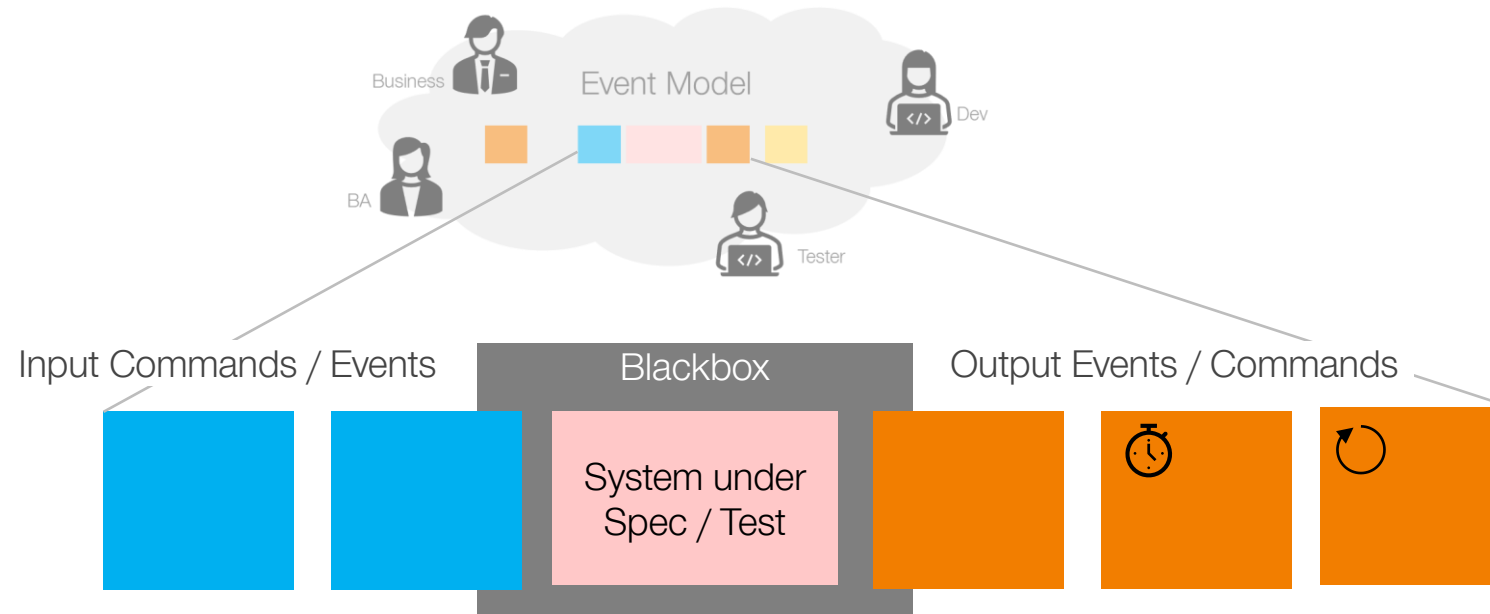
- Norm how to design, build, and deploy safety relevant installations in the railway domain
- Required by Swiss Federal Office of Transport (BAV) due to increasing degree of automation in train steering related systems
- Process follows a strict waterfall model



Synergies of CENELEC and DDD



Requirements and Tests based on Events



Software Requirement
The <system> must <action> [after X seconds] [N times | every X seconds] whenever a <command | event> has been received.

Test Case
When <command | event> has been received
And <command | event> has been received
Then a <event | command> must have been sent [N times | every X seconds]

Waterfall and Agility?



Business Vision

Event Model

Suitable «flight altitude»



Code



Business



BA



Tester



Dev

Example 1: Refactoring of internal APIs

Example 2: Elimination of plan update store



6 - Wrap-up

Benefits of DDD for the TA Adapter

- ✓ Strict separation of application logic (integration / mapping) and business logic reduces complexity
- ✓ Very concise business logic in aggregates, hardly any complicated code
- ✓ The joint work on the model helped us identify and realize important simplifications
- ✓ At runtime, event log gives precise insight into the application
- ✓ Close to zero bugs, fun coding 😊

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code

Since 3.0.0-SNAPSHOT
Started 10 days ago

Overall Code

0 Bugs

0 Vulnerabilities

0 Security Hotspots

6h 2min Debt

31 Code Smells

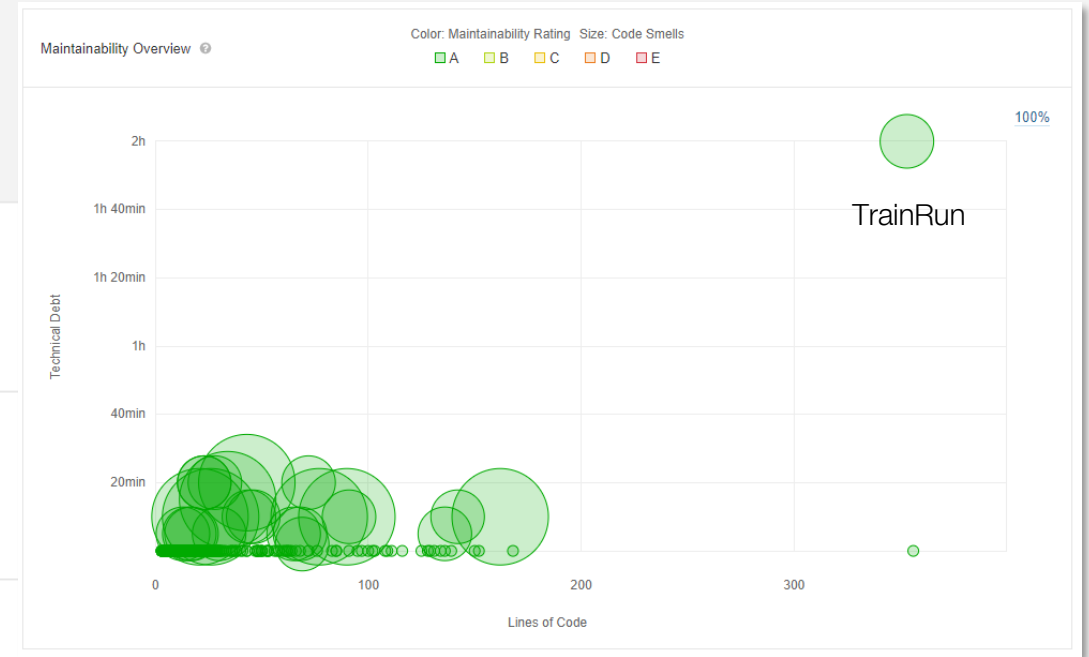
Maintainability A

97.0% Coverage on 2.7k Lines to cover

1k Unit Tests

0.4% Duplications on 9.6k Lines

4 Duplicated Blocks



Reviewed

Security Review A

Conclusion



Domain-driven design offers effective tools and pattern to build quality software

DDD is not a one time activity and requires constant efforts to align team members and business to be effective.

DDD helps to develop a crisp understanding of the software being built and is therefore a good match for norms such as CENELEC SN EN 50128.



Questions?

Thank you!