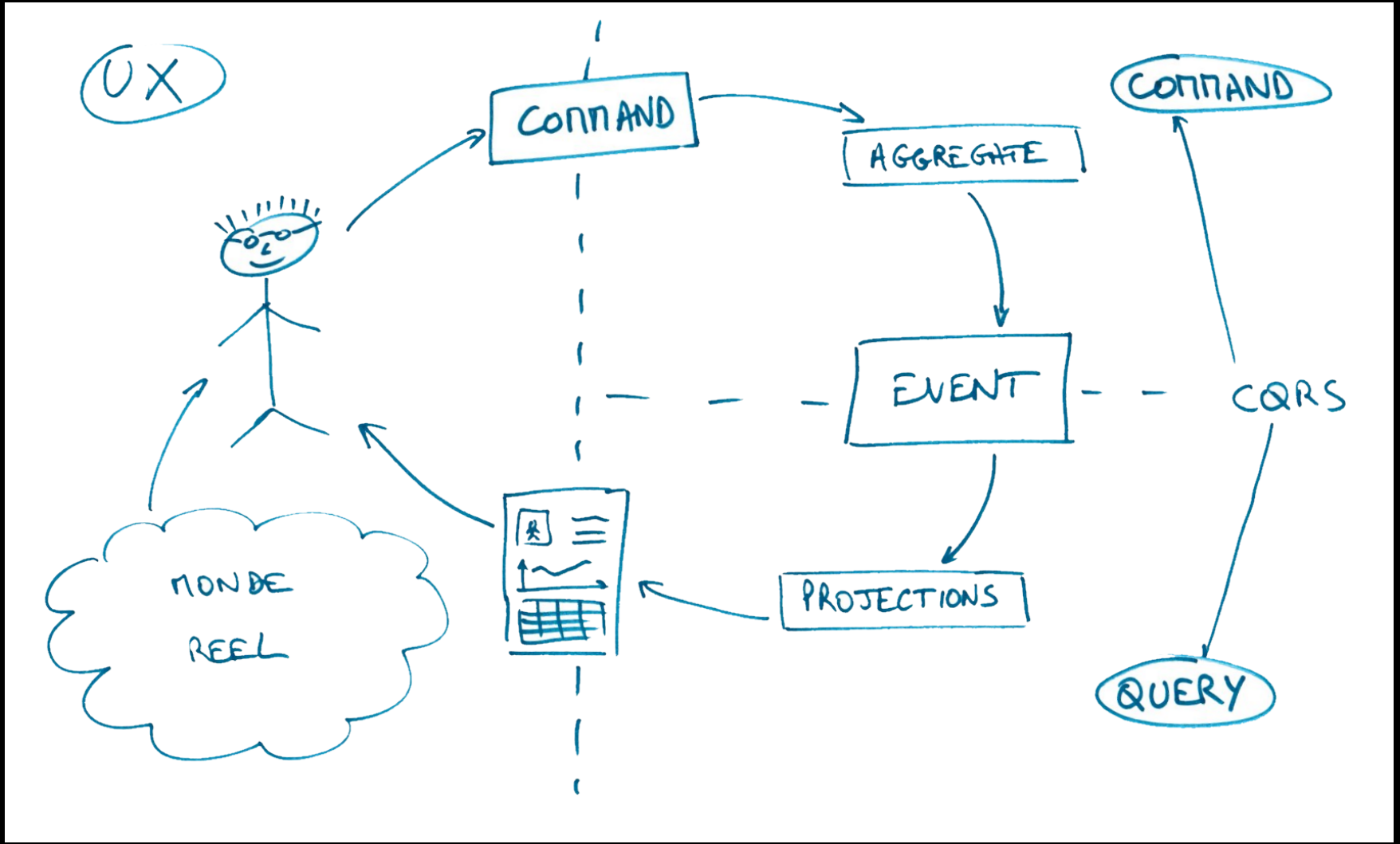
AGILITÉ PAR LE CODE GRÂCE À **CQRS** ET **EVENTSOURCING**

WORKSHOP – #MIXIT15

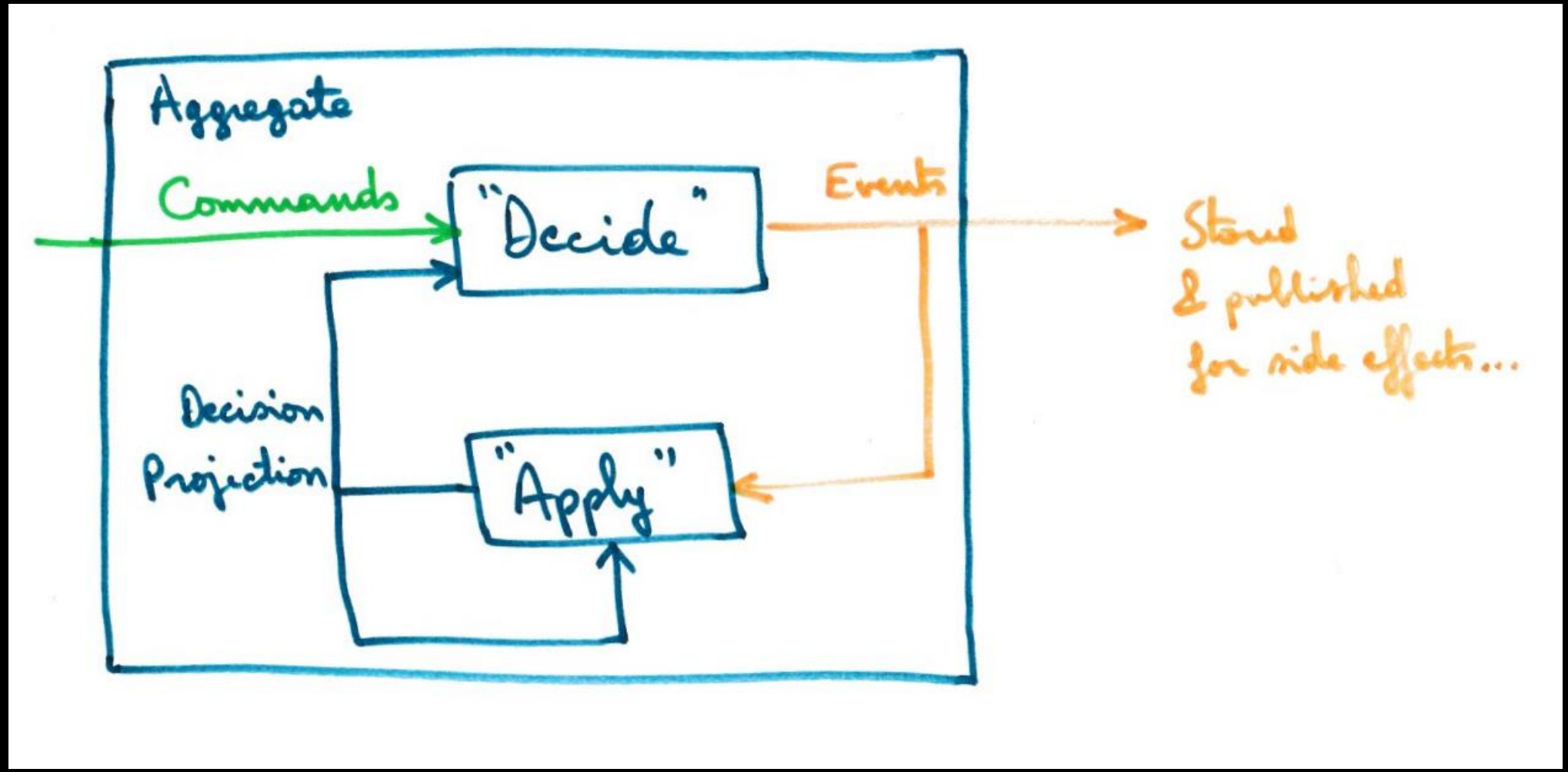
Forent @florentpellet
Clément @clem_bouillier
Jean @jeanhelou
Emilien @ouarzy

CQRS CONCEPT



Ref. "Conceptual CQRS" - Alberto Brandolini

EVENT SOURCING CONCEPT

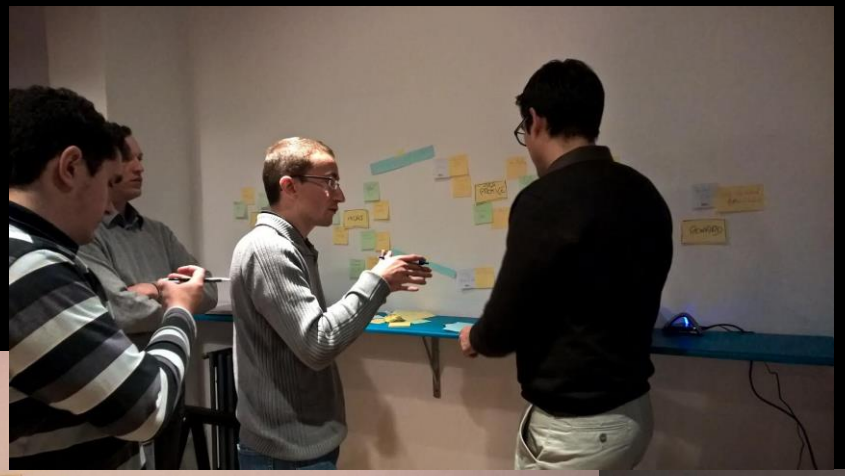


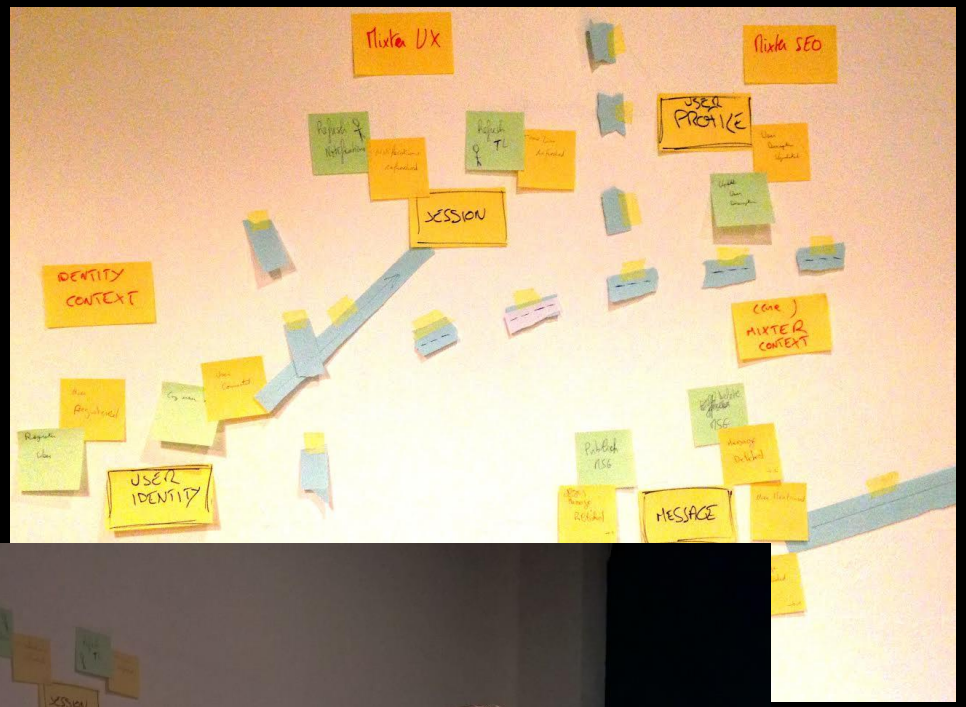
NB : DecisionProjection is also called State

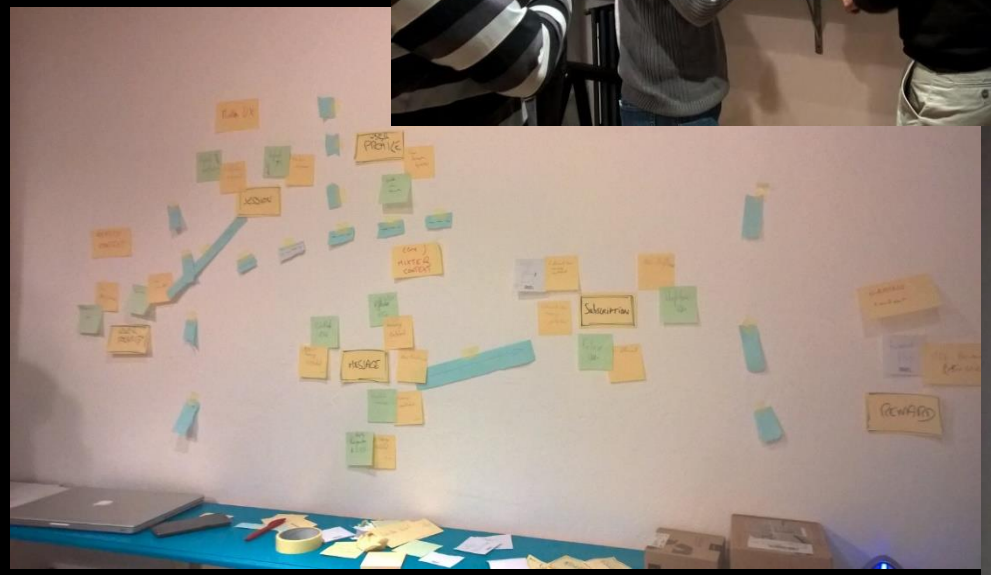
Ref. Jérémie Chassaing

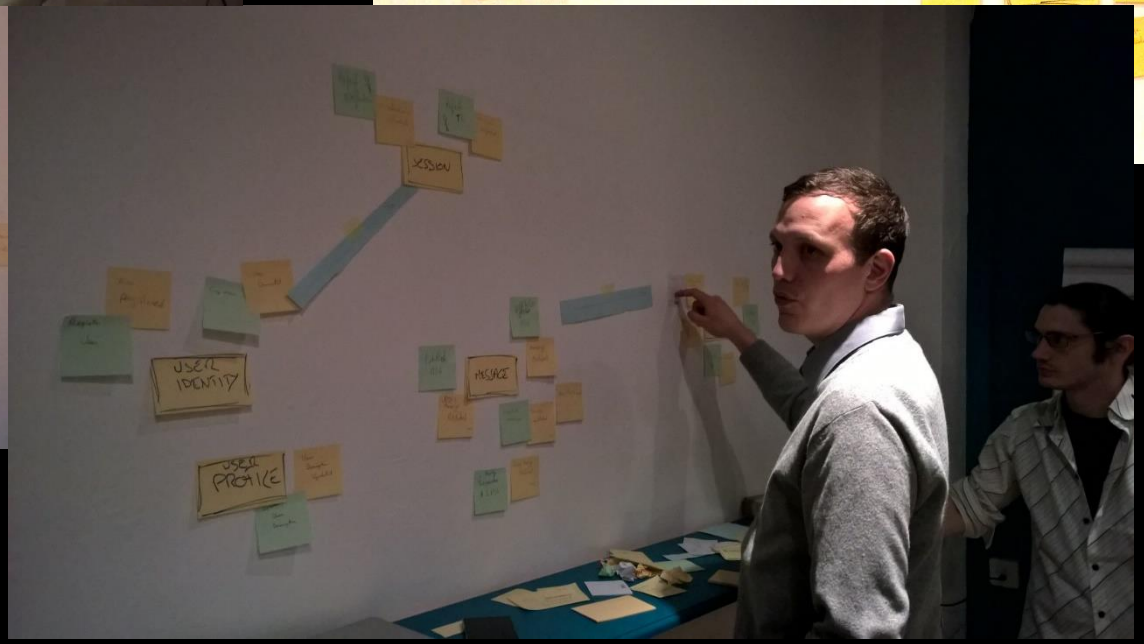
EVENT STORMING MIXTER

mix-IT

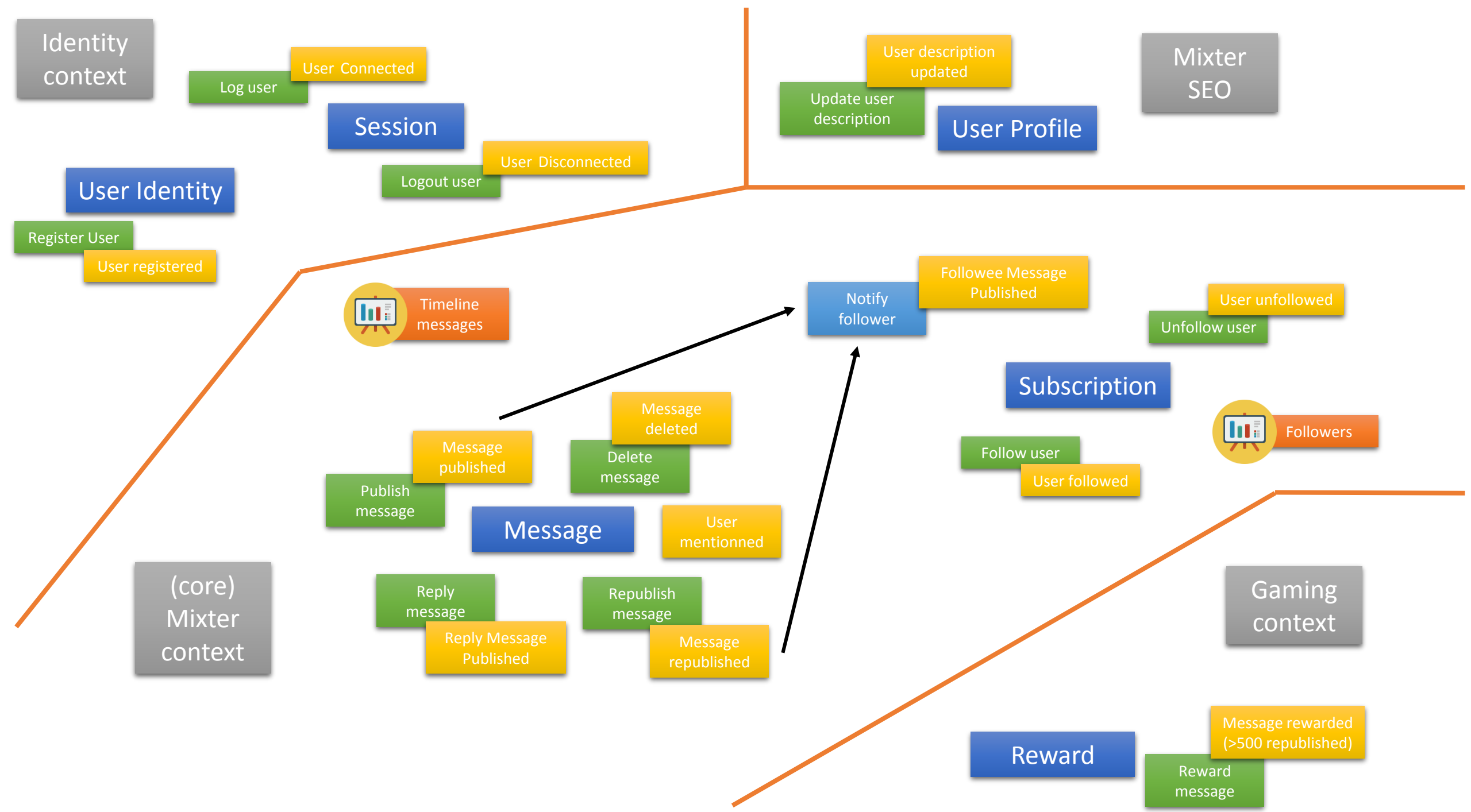


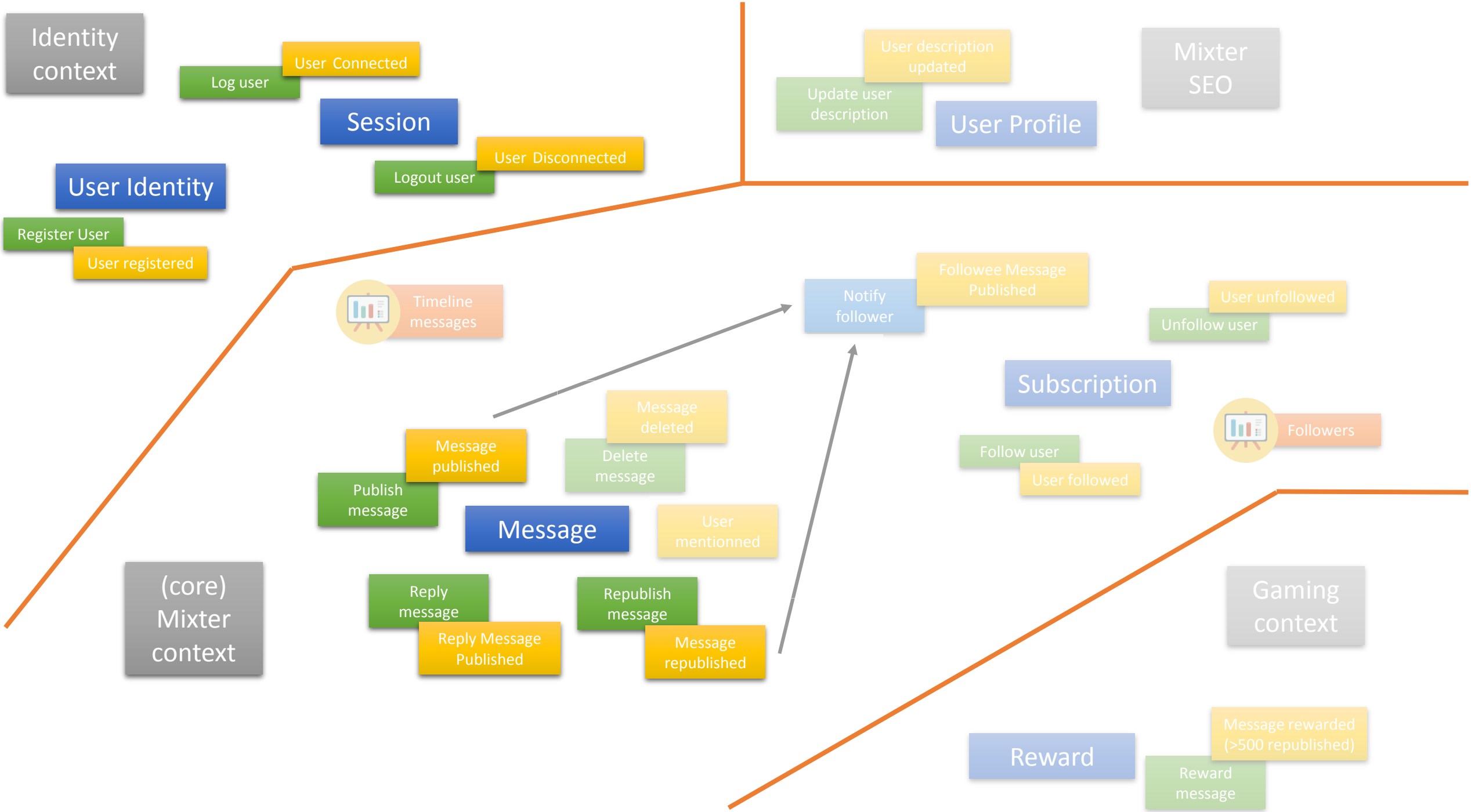






Ref. "Event Storming" - Alberto Brandolini


Identity context
Log user
User Connected
Session
User Disconnected
Logout user
User Identity
Register User
User registered
User description updated
Update user description
User Profile
Mixter SEO
Timeline messages
Notify follower
Followee Message Published
User unfollowed
Unfollow user
Subscription
Follow user
User followed
Followers
Message deleted
Delete message
Message published
Publish message
Message
User mentionned
(core) Mixter context
Reply message
Reply Message Published
Republish message
Message republished
Gaming context
Reward
Message rewarded (>500 republished)
Reward message


Identity context
Log user
User Connected
Session
User Disconnected
Logout user
User Identity
Register User
User registered
User description updated
Update user description
User Profile
Mixter SEO
Timeline messages
Notify follower
Followee Message Published
User unfollowed
Unfollow user
Subscription
Message deleted
Delete message
Message published
Publish message
Message
User mentionned
Follow user
User followed
Followers
(core) Mixter context
Reply message
Reply Message Published
Republish message
Message republished
Gaming context
Reward
Message rewarded (>500 republished)
Reward message

ONLY 1H30!

- IT'S SHORT !

- 4 FACILITATORS
  - ANY QUESTIONS => SIGNAL US

Clément

Emilien

Florent

Jean

TEST DRIVEN WORKSHOP

- Working in pairs

- Red test => Green test
  - 1 git TAG by test (commit if green)

    => git merge [language]-x.y where x and y are incremented to follow workshop (commit your previous solution before merge !)
  - NB : this is based on 2 branches :
    - one with test only (one by commit with tag)
    - one with our solution (only if you are late)
- 3 steps (+2 bonus)
  - Command DeleteMessage
  - Query Timeline Message
  - Events in aggregate

REPOSITORY GIT

- GIT CLONE **HTTPS://GITHUB.COM/JEANTIL/MIXTER.GIT**
  - BRANCHES:
    - CSHARP-WORKSHOP (.NET 4.5.1, VS 2013)
    - JAVA-WORKSHOP (SDK8, MAVEN)
    - JS-WORKSHOP (NODE.JS)
    - PHP-WORKSHOP (5.5, COMPOSER)

- SLIDE : HTTPS://GITHUB.COM/JEANTIL/MIXTER/RAW/SLIDE/SLIDE.PDF

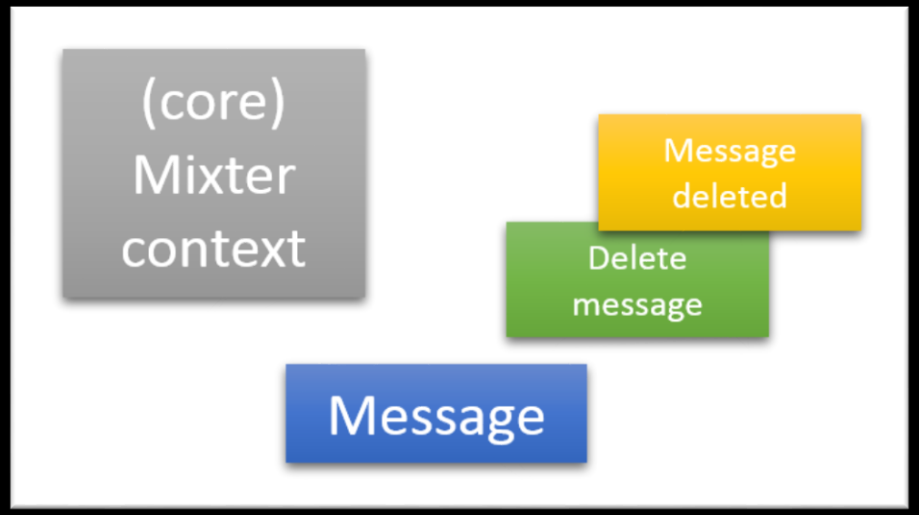
1. DELETE COMMAND

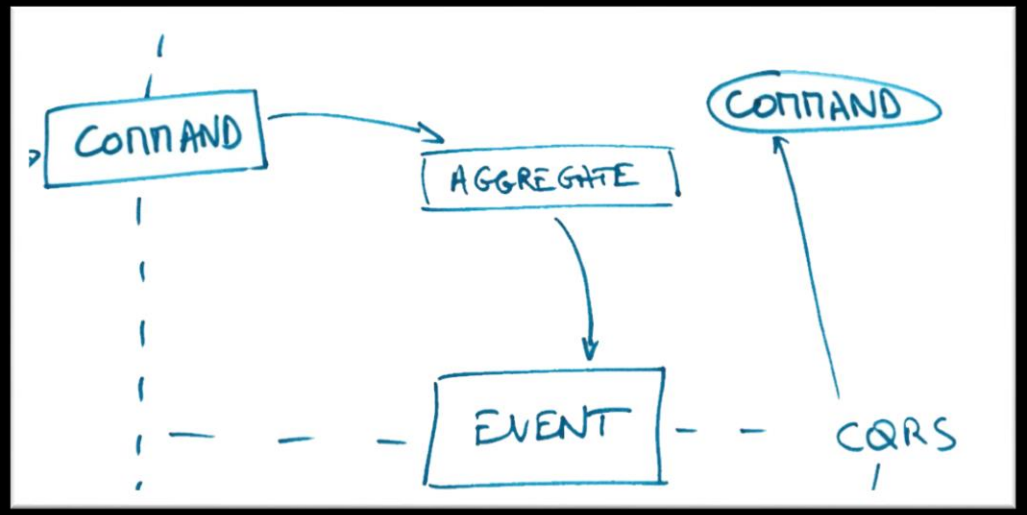
mix-IT

What we will learn

- publish events from aggregate,
- use projection for decision inside aggregate (contains only "state" for future decision, DO NOT keep all state like in an entity)
- Implement "business rules" that insure aggregate consistency (based on decision projection and command=method parameters)

In brief : the C of CQRS





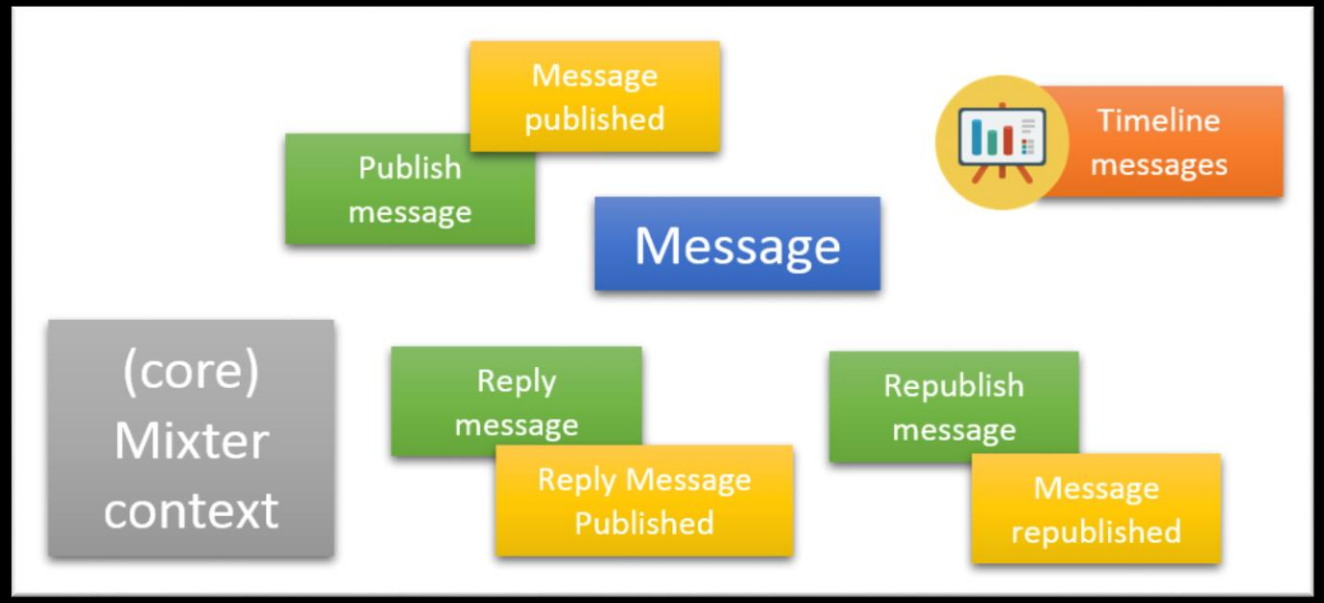
2. TIMELINE MESSAGES PROJECTION

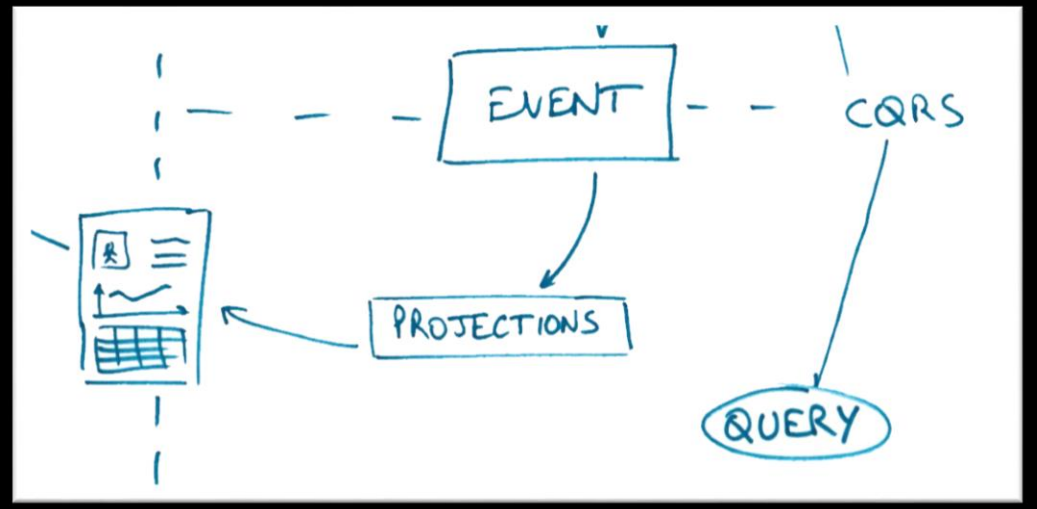
mix-IT

What we will learn

- Create another model for Query (Projection, TimelineMessageProjection)
- Transform events in a projection model through an EventHandler
- A projection repository (in-memory) with its interface is given

In brief : Q of CQRS





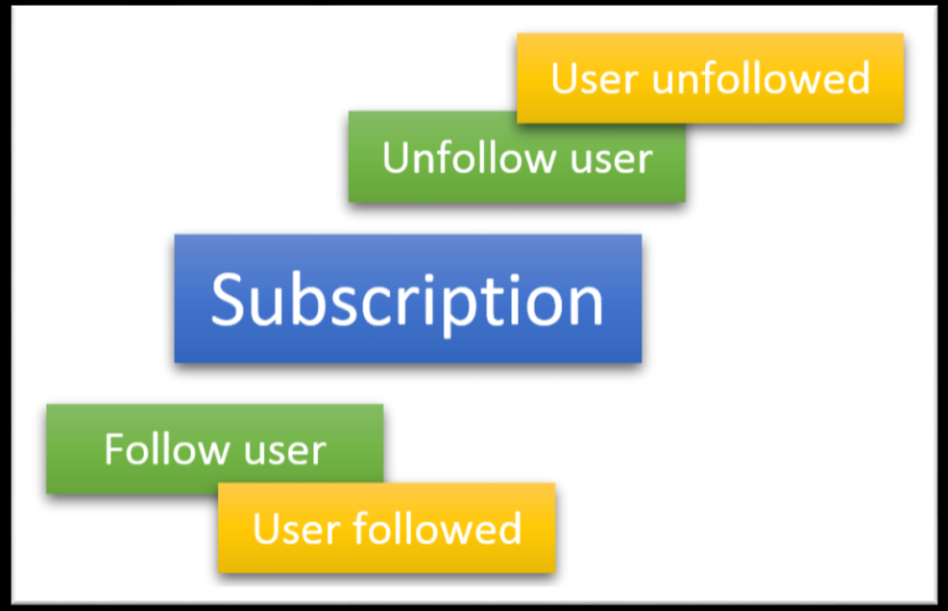
3. SUBSCRIPTION AGGREGATE

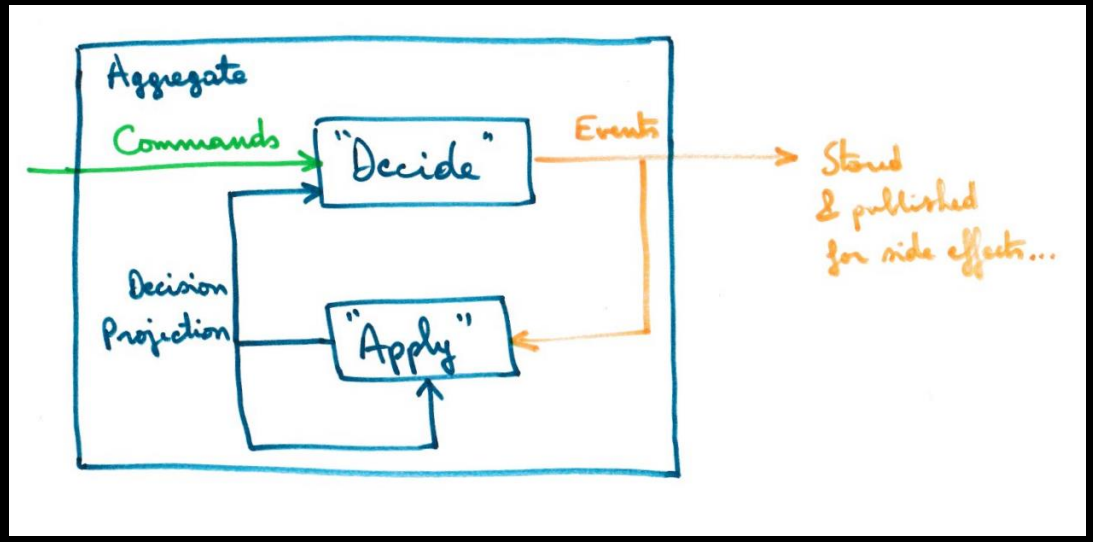
mix-IT

WHAT WE WILL LEARN

- CREATE A NEW AGGREGATE (SUBSCRIPTION)
- RAISE EVENTS FROM IT : USERFOLLOWED AND USERUNFOLLOWED
- CREATE A DECISION PROJECTION FOR IT
- IMPLEMENT REPLAY OF EVENTS (EVENT SOURCED AGGREGATE)

IN BRIEF : C OF CQRS + EVENT SOURCING



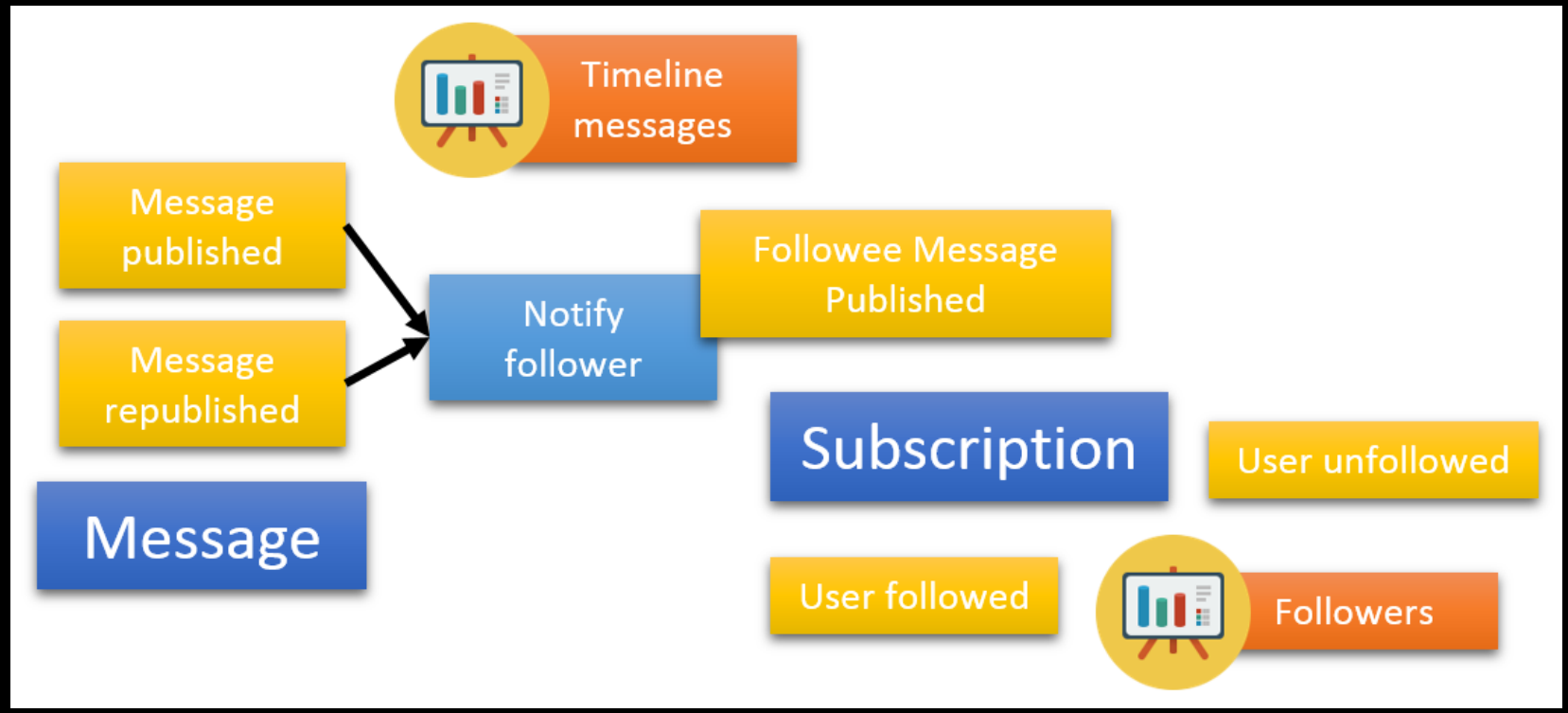


4. AGGREGATES INTERACTION

mix-IT

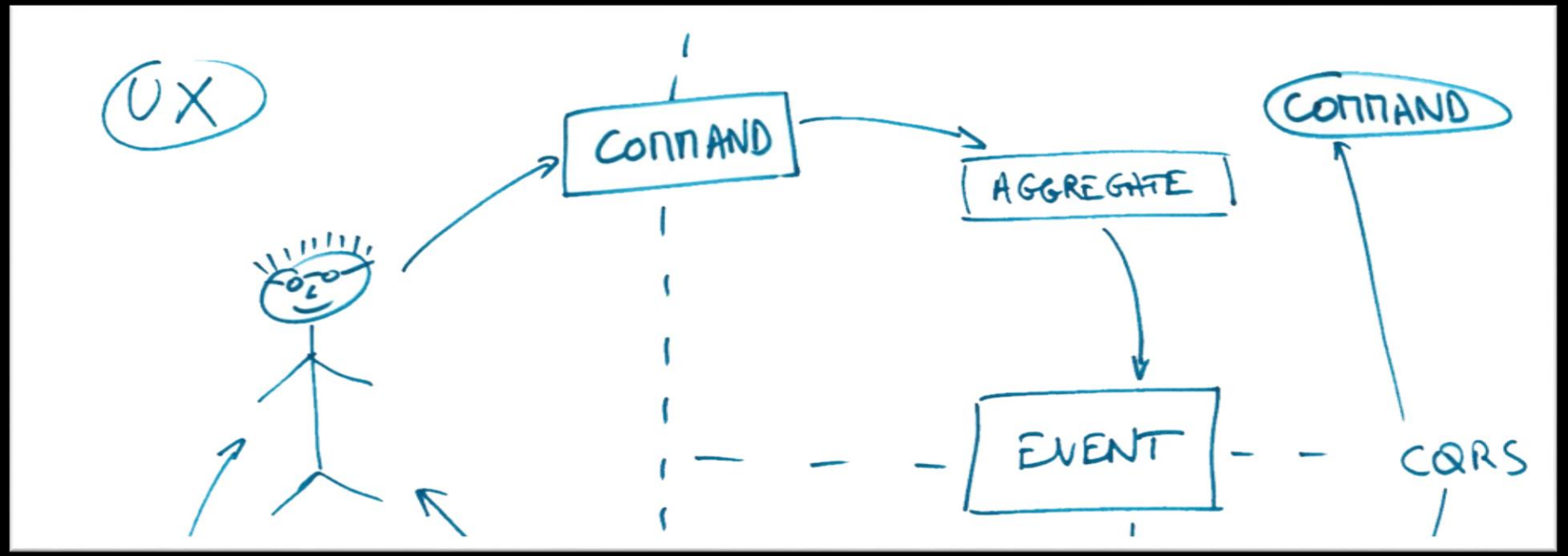
What we will learn

- Coordinate several aggregates to limit coupling
- Concept of " Eventual consistency"



5. COMMAND HANDLER

REQUEST REST TO EXECUTE DELETE MESSAGE COMMAND, WITH SESSION VALIDITY VERIFICATION



mix-IT

AGILITÉ PAR LE CODE GRÂCE À **CQRS** ET **EVENTSOURCING**

THANKS!

Forent @florentpellet
Clément @clem_bouiller
Jean @jeanhelou
Emilien @ouarzy