TECH WEEK

KAIZEN

Domain Driven Design

SUAREZ COLOMA Juan Pablo

DELPECH Jean-Baptiste

Sommaire

TECH WEEK

KAIZEN

1. Introduction au DDD

Qu’est-ce-que le Domain Driven Design ?

2. Kaizen Tech Beer

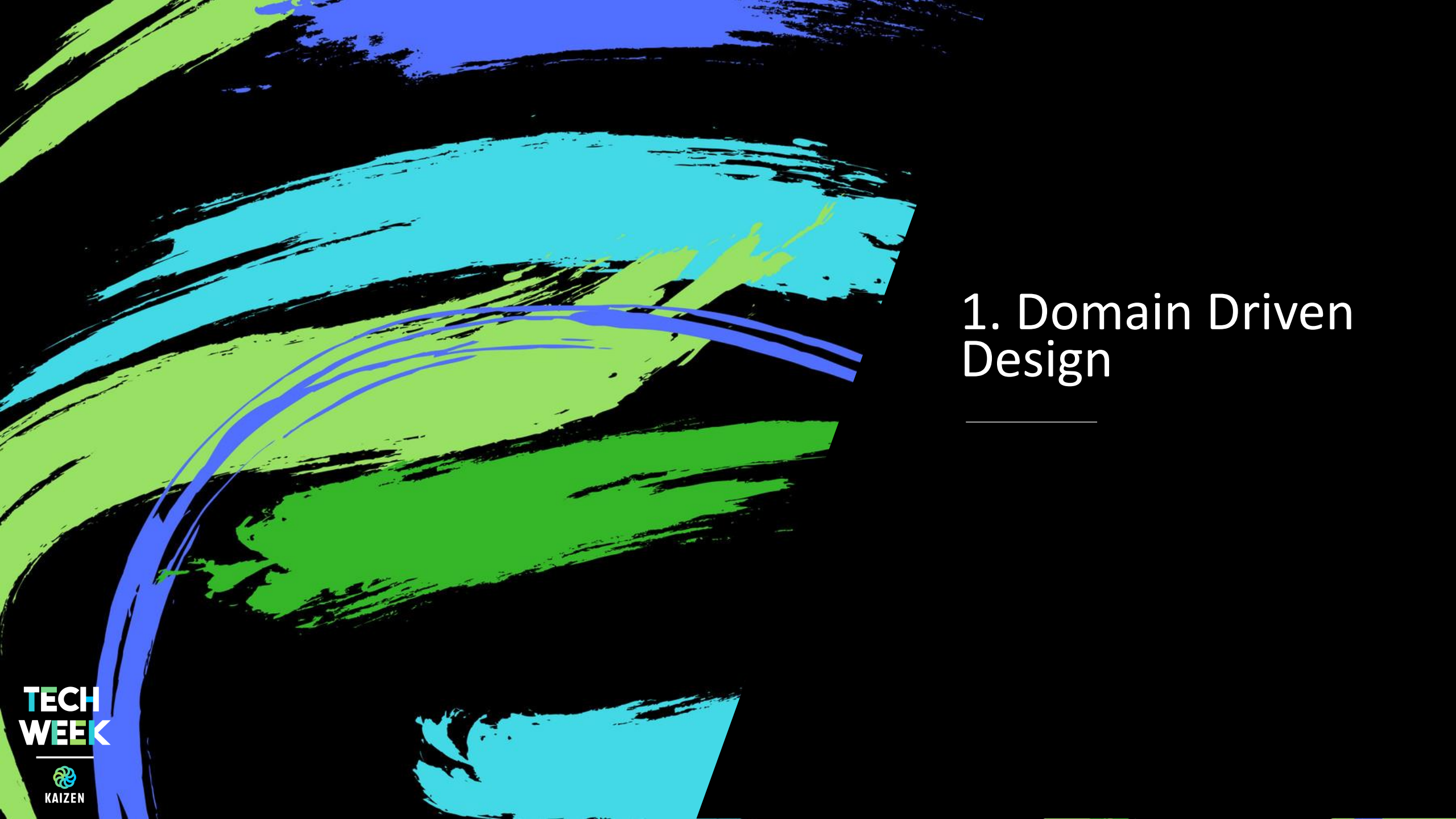
Développons cette application :

- Sans DDD
- Avec DDD

3. Conclusions – Retour d’expérience

Quoi de mieux pour conclure qu’un retour d’expérience ?

TECH WEEK
KAIZEN



1. Domain Driven Design

Qu’est-ce que le "DDD" ?

- "Conception Pilotée par le Domaine"
- Livre* écrit en 2003 par Eric Evans
- ***Approche*** pour développer des logiciels complexes
- Se focalise sur :
  - La connaissance du domaine
  - L'omniprésence du langage
  - La collaboration entre experts du domaine et développeurs

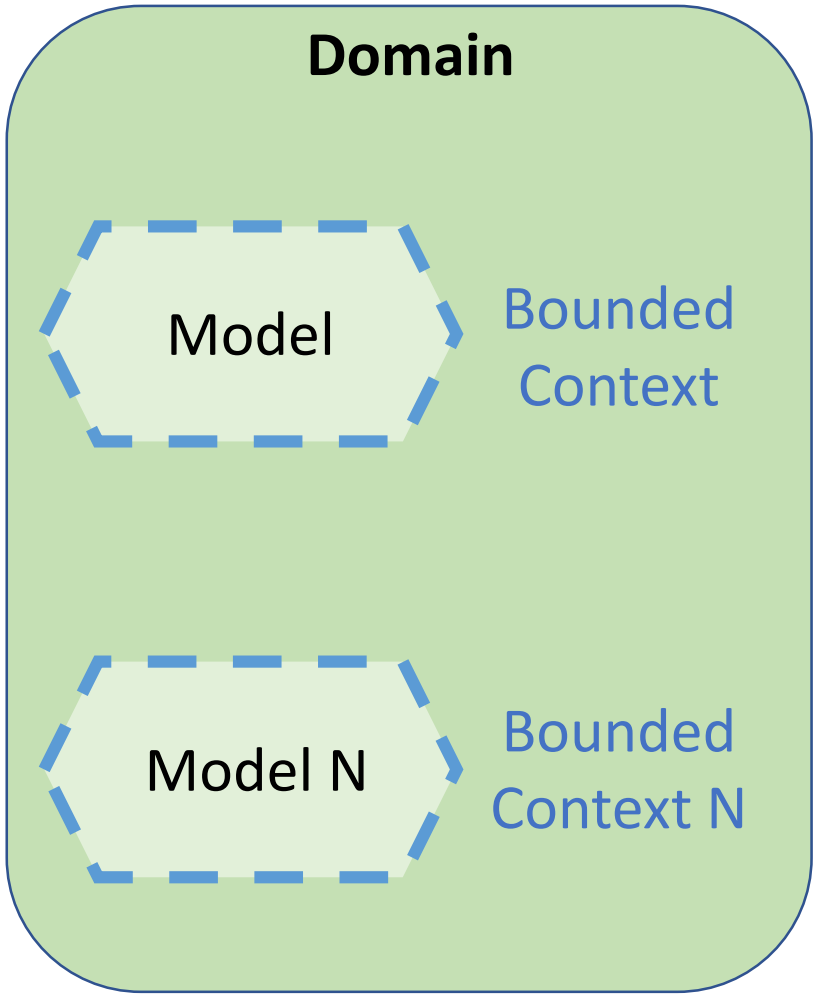
Domain Driven Design

TECH WEEK

KAIZEN

** Eric Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software. 2003*

Qu’est-ce que le "DDD" ?



Domain : ensemble de connaissances métier

Model : un aspect du *domain* métier

Bounded context (Contexte borné) :
définit le périmètre du *model*

Ubiquitous Language :
langage omniprésent, partagé au sein de l’équipe

Domain Driven Design

TECH WEEK
KAIZEN

Model Driven Design

- **Entity** : un objet avec une identité
- **Value Object** : un objet immutable
- **Domain event** : évènement qui notifie le changement
- **Aggregate** : un objet qui encapsule des *entities* et des *values objects* qui partagent une même responsabilité, il fixe les bornes.
- **Service** : toute logique domaine ou concept que l'on ne peut pas mettre dans un objet du domaine
- **Repository** : requête les *aggregates*
- **Factory** : encapsule la création des *aggregates*

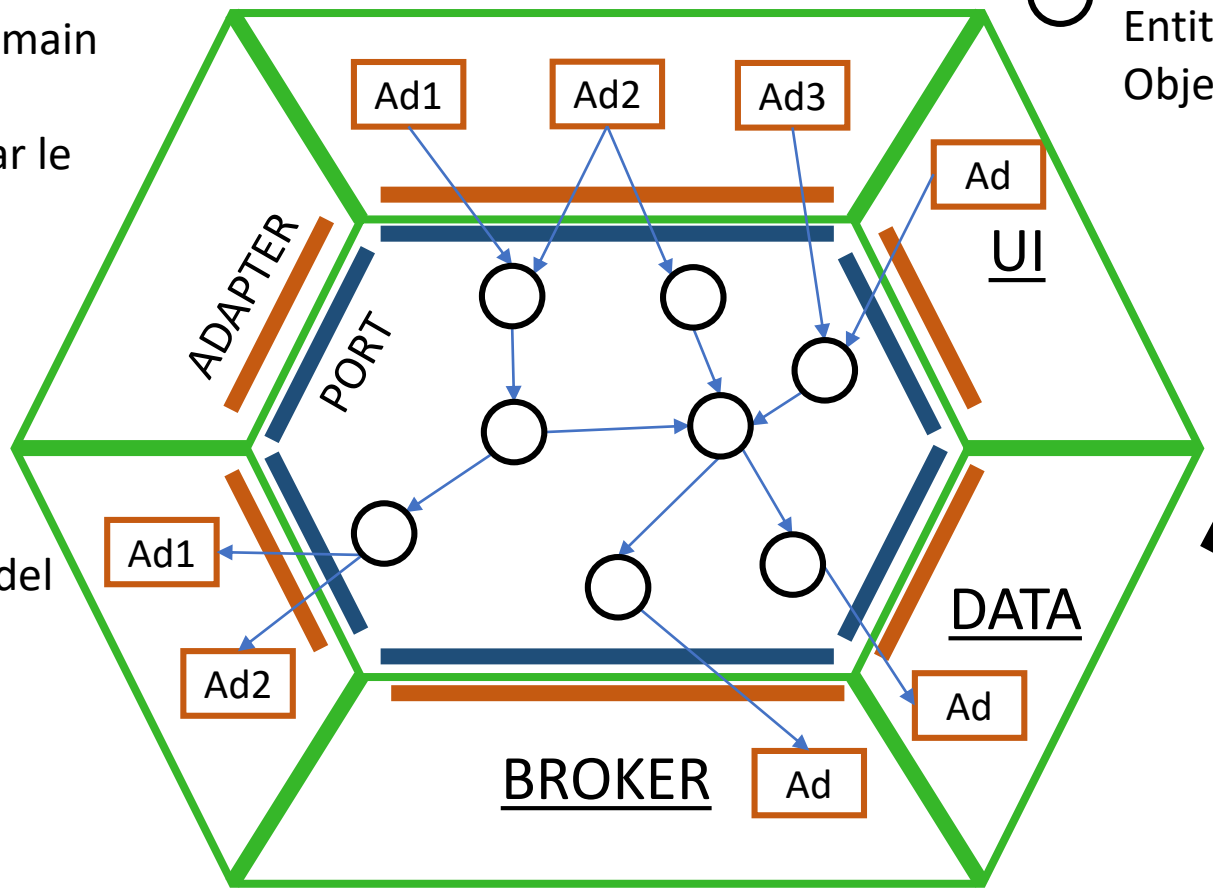
Domain Driven Design

TECH WEEK KAIZEN

Architecture Hexagonale (Ports & Adapters)

Port (Interface):
API => expose le domain model (use case) ;
SPI => Invoquées par le model

Adapter :
obtient des informations du model (appelle l'API) ;
implémente une SPI

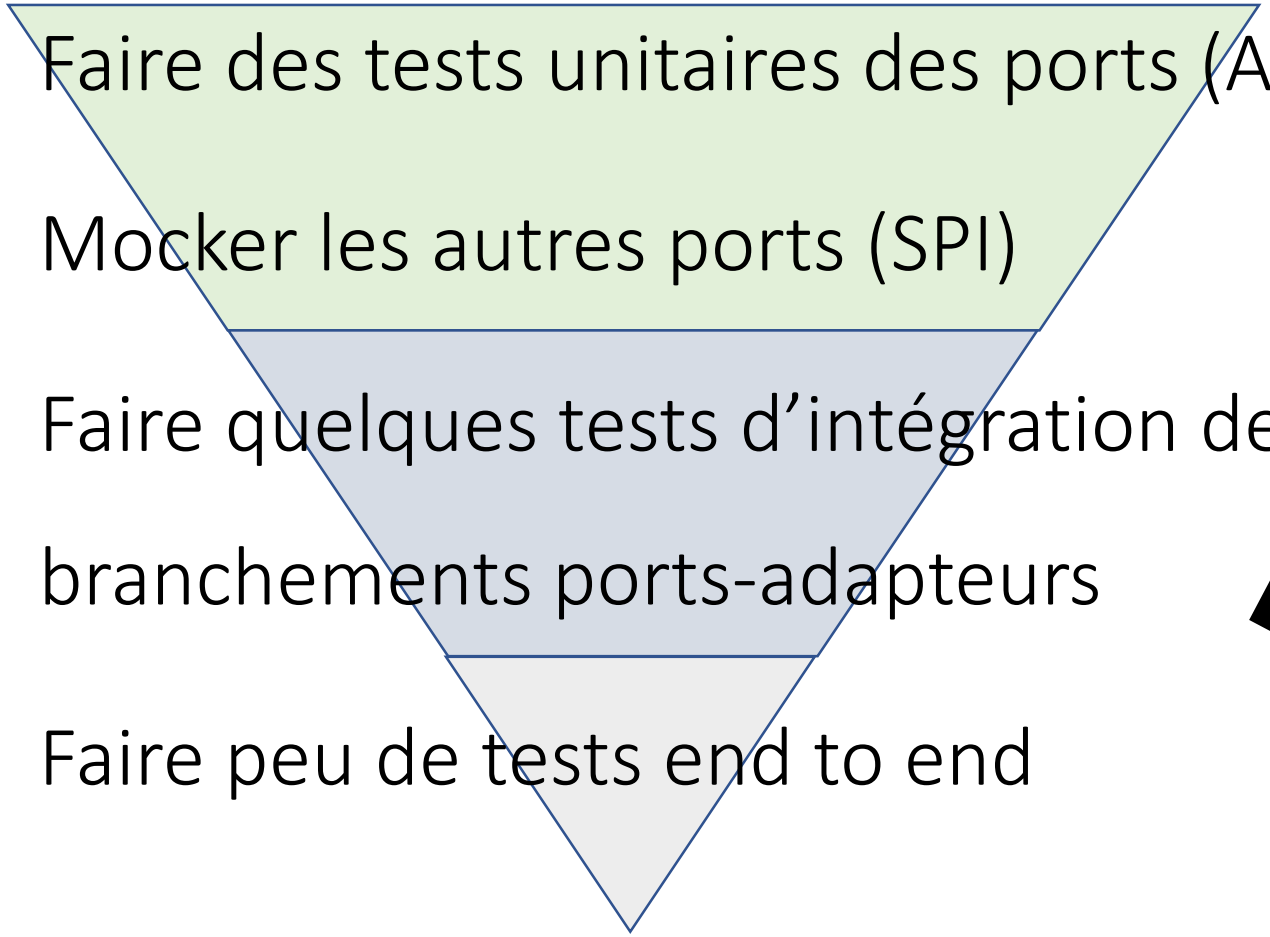


Aggregates, Services, Entities, Value Objects, Etc.

Domain Driven Design

TECH WEEK
KAIZEN

Ben, ce qu'on TEST



- Faire des tests unitaires des ports (API)
- Mocker les autres ports (SPI)
- Faire quelques tests d'intégration des branchements ports-adapteurs
- Faire peu de tests end to end

Domain Driven Design

TECH WEEK KAIZEN

2. Kaizen Tech Beer

Kaizen Tech Beer

TECH WEEK

KAIZEN

Philippe T.

23 Avril

24 Avril

25 Avril

« Raisonnons » sans DDD

Kaizen Tech Beer

Nous avons « mal » modelé de manière intentionnelle 

Interface utilisateur

- Calcul les bières restantes à boire
- Affiche le résultat

Logique métier

- Service qui récupère les données

Données

- Bières attribuées / jours / personnes (3)
- Bières bues / jours / personnes

TECH WEEK
KAIZEN

Raisonnons avec DDD

Kaizen Tech Beer

Identifions les entités (en l'occurrence l'aggregate root)

- **bière** : elle peut changer d'état, de « disponible » à « réclamée »

Identifions les repositories

- **bièreRepository** : récupère les bières associées à un jour

Identifions les services métier

- **bièreService :** afin de consommer la première bière disponible le jour J

Domain events ?

- Déclenchés par le changement d'état des bières

Value Objects ?

- La composition de la bière

TECH WEEK KAIZEN

Conclusions

TECH WEEK

KAIZEN

Avantages du DDD

- Le code métier est centralisé (isolé) : le métier est pérennisé
- On sait ce qu'il faut tester
- La maintenance et l'évolution du code
- Les nouveaux développeurs comprennent le métier avec le code
- Ecrire du code en français n'est pas mal vu
- Mène à des échanges et des choix intellectuels entre membres de l'équipe afin de délimiter le contexte
- Le tout, même si le résultat est invisible aux yeux des utilisateurs finaux

Conclusions

TECH WEEK
KAIZEN

Transition vers le DDD

- A partir de la théorie, il n'est pas évident d'identifier les aggregates, les entités, les value objets, et les autres objets du domaine
- Le modèle ne cesse pas d'évoluer
- Il faut ***éviter les choix précipités*** : discuter pour mieux identifier et modéliser
- Développer : des petites fonctions ; des interfaces qui exposent leurs intentions
- Faire du DDD mène à créer plus de classes
- Attendez-vous à entendre : « on a toujours fait comme ça » ; « cela marchait mieux avant »

Conclusions

TECH WEEK KAIZEN


Le DDD Dchire !
TECH
WEEK
KAIZEN

Merci !

En espérant avoir levé votre curiosité ;)

Juan-Pablo SUAREZ-COLOMA

Jean-Baptiste DELPECH

TECH WEEK

KAIZEN

Web/Biblio - graphie

- Eric Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software. 2003
- Eric Evans. Domain-Driven Design Reference, Definitions and Pattern Summaries. 2015
- http://alistair.cockburn.us/Hexagonal+architecture

TECH WEEK

KAIZEN