

# Formation Domain Driven Design avancé

## Conception de logiciel dirigée par le métier

Référence : DDD-04

Durée : 4 jour(s)

### Présentation

Comprendre le métier et le modéliser est un pré-requis pour concevoir et implémenter avec succès un logiciel. La validation du métier et du logiciel est en outre un facteur clé d'efficacité.

Cette formation vous apprendra les concepts et vous les fera pratiquer afin de réussir la modélisation de votre métier et sa traduction en logiciel à travers l'activité de conception.

### Objectifs

- Comment modéliser le métier ? Intégrer les règles de gestion dans le modèle ?
- Comment bien spécifier le besoin et les fonctions pour faciliter la conception du logiciel ?
- Comment isoler les aspects métiers du logiciel et les aspects techniques ?
- Comment gérer la complexité d'un métier d'envergure et du logiciel associé ?
- Comment intégrer l'agilité dans l'activité de conception ?
- Comment valider la conception ?
- L'activité de conception et de développement : quelles sont leur différences ? comment les intégrer au mieux ?

**Répartition:** 50% Théorie, 50% Pratique

**Public:** Développeur, Architecte, Analyste, Chef de projet

**Pré-requis:** Connaissance en modélisation et conception objet

### Programme

#### Fondamentaux de modélisation

- Aspect statique et dynamique d'un modèle
- Typologie de systèmes
- Les éléments fondamentaux d'un modèle objet
- Les modèles associés aux méthodes, les processus des méthodes et leurs paradigmes
- Le contexte d'un modèle

#### Le langage de modélisation

- Comment modéliser graphiquement ou textuellement ?

#### L'essentiel dans UML

- Exposition des notions essentielles issues du langage de modélisation UML

#### Gestion de la complexité avec des modèles

- Qu'est ce que la complexité ?
- Comment gérer la complexité d'un système par la modélisation ?
- Utilisation de niveaux d'abstraction

#### L'outillage pour modéliser

- Du papier/crayon au modéleur logiciel complet

#### L'ubiquitous language

#### Éléments de construction

- Repositories, Factory, Services, Entity, Value Object, Module, Aggregates, Domain Event
- Exercice de mise en œuvre des Modules, Entity et Value objects et Aggregates, Domain Event

#### La modélisation Agile

- Principes d'organisation favorisant l'efficacité de l'activité de conception avec des modèles.

#### Les règles de gestion

- Comment les décrire ? Comment les tester ?

#### Les patterns d'analyse

- Typologie des patterns d'analyse : Accountability, Observations et Mesures,

#### Valider le modèle et le logiciel

- Des exigences aux user stories
- Des user stories aux scénarios
- Des scénarios aux tests
- Des scénarios pour valider le modèle ET le logiciel
- Notions de Tests-Driven-Design et Test-Driven-Development
- Outillage

#### L'activité de conception

- La conception et le développement : deux activités corrélés, des niveaux d'abstraction différents

#### Concepts permettant la réalisation d'un logiciel testable et maintenable

- Gestion des dépendances
- Gestion des états
- Séparation des responsabilités
- Encapsulation et interface
- Protection des variations (open/closed principle)
- Intention-revealing interface
- Standalone classes
- Side-Effect free function
- Assertions
- Conceptual contours
- Closure on operations
- SOLID
- Loi de Demeter
- Principes de substitution de liskov
- Concepts issu du paradigme fonctionnel : structures de données, combinator functions

#### Modélisation du système logiciel

- Présentation de l'architecture en couches
- Fonctionnement Interne du logiciel : aspect structurel et comportemental
- Fonctionnement Externe du logiciel : comportement, aspect visuel des écrans, interactions

#### Le refactoring

- Techniques et utilisation dans le cadre de l'activité de conception

#### Intégrer les technologies courantes de l'entreprise avec une approche dirigée par le métier

- Isoler les aspects techniques des aspects métiers
- Architecture Orientée Service (SOA) et bus de services d'entreprises avec approche dirigée par le métier
- Mise en œuvre des frameworks courants : Spring, Frameworks de persistance (Hibernate)

#### Intégrer le logiciel dans le système d'information, intégrer les équipes entre-elles

- Notion de contexte du modèle
- Comment intégrer différents modèles ?
- Mapping de contexte
- Typologie des relations entre systèmes et équipes

#### Concepts Avancés

- Aspect stratégique de l'effort de conception
- Le cœur du domaine et les domaines supports
- Les domaines génériques : les progiciels
- Élément de qualité d'une conception

## OFFERT EN INTER-ENTREPRISE

☺ Le petit déjeuner croissants, jus d'orange, café)

☹ Le déjeuner

☹ Une qualification téléphonique si nécessaire avec l'un de nos consultants

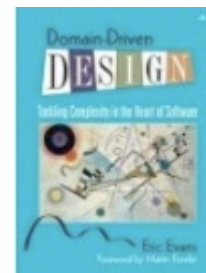
Tel: +33(0)1 45 26 19 15  
Fax : +33(0)1 75 43 49 92  
Email : training@zenika.com

### Auteur du cours



Jérémie Grodziski est un passionné de la conception et du développement de logiciel. Il forme et accompagne les entreprises souhaitant concevoir et réaliser des logiciels de qualité de manière agile. Jérémie a été formé par Eric Evans et a créé le DDD User Group de Paris.

### Livre offert ! (\*)



Domain-Driven Design

(\*) Les livres sont offerts uniquement pour les formations inter-entreprise. Zenika se réserve le droit de changer le livre proposé à tout moment.



zenika  
ARCHITECTURE INFORMATIQUE