Clean architecture dans le monde .NET

-

Introduction

Clean architecture & .NET 9

Pour moi cette formation sera un succès si ... ?

Organisation de la formation

- **6 modules de 3h30 divisés entre:**
  - **Présentation théorique des concepts**
  - **Application pratique sous forme de cas pratiques**
  - **Mise en commun des travaux effectués & commentaire**

Principaux concepts abordés au cours de la formation

Introduction aux principes d’architecture

Bonnes pratiques et stratégies de tests automatisés

Mise en place dans un projet d’exemple

Pourquoi l’architecture est importante pour votre business ?

Quels sont les axes de la qualité d'un produit?

1. L'adéquation entre le besoin et la façon dont le logiciel répond à celui-ci
2. L'absence de dysfonctionnement
3. La performance
4. L'usage naturel
5. La transparence de fonctionnement

Pourquoi mesurer la qualité?

"You can't control what you can't measure."

(Tom DeMarco)

Comment mesure-t-on la qualité?

Pour aborder la notion de qualité, comme beaucoup de notions abstraites positives, on peut la définir par l'absence du concept opposé.

De même qu'on définit la paix comme l'absence de guerre, on peut définir la qualité par l'absence de défauts.

Qu’est-ce qu’un défaut?

Un défaut est toute caractéristique qui n’est pas présente de façon souhaitée ou optimale.

Il peut se trouver dans tout aspect d’un logiciel. Y compris les aspects esthétiques.

Pourquoi mesurer la qualité?

- Un code mal maîtrisé entraîne mécaniquement une augmentation régulière et constante de la **dette technique** et de l'**entropie logicielle.**
- Surveillance des métriques permet une diminution du **coût de maintenance** à moyen et long terme

Pourquoi mesurer la qualité?

- Accélération de la **prise en main du code** : un code homogène sera mieux pris en charge par l'équipe, et l'intégration de nouveaux arrivants s'en retrouve également facilitée, on diminue l'**effet « boîte noire »**.
- Qui dit approche quantitative dit **processus automatisable** et donc moins coûteux !

Quels sont les principaux axes de qualité d'un logiciel?

Le Consortium for IT Software Quality (CISQ) organise l'analyse de la qualité logicielle autour de **5 axes**:

1. Fiabilité
2. Efficacité
3. Maintenabilité
4. Sécurité
5. Taille

Fiabilité

- Un attribut de résilience et de solidité structurelle.
- Elle mesure le niveau de risque d’échec de l’application.
- Le but de l’augmenter est de réduire les temps où elle n’est pas accessible ainsi que les défauts qui affectent les utilisateurs.
- Elle améliore aussi l’image du département IT au sein de l’entreprise.

Efficacité

- Le code source et l'architecture sont ce qui garantit l'efficacité du logiciel en phase d'exploitation.
- Spécialement important dans les contextes où la rapidité est importante.
- Une analyse de l'efficacité et de la scalabilité du code fournit une image claire des risques business liés aux dégradations des temps de réponse.

Sécurité

- Une mesure de la probabilité de faille due à des pratiques de code ou d’architecture
- Quantifie les vulnérabilités critiques qui sont dommageables au métier. (Perte d’informations stratégiques, fuite de données à caractère personnel, etc.)

Maintenabilité

- Inclut la notion d’adaptabilité, de portabilité et de transférabilité (d’une équipe de dev à une autre par exemple)
- C’est un aspect essentiel pour les applications critiques pour le métier. En particulier quand le « time to market » est un aspect important de la compétitivité de l’organisation.
- Il est également très important de garder sous contrôle les coûts de maintenance.

Taille

- Bien que ce ne soit pas un attribut à part entière, cette caractéristique impacte tous les autres aspects.
- Combinée avec les autres caractéristiques de qualité, la taille peut être utilisée pour estimer le volume de travail effectué ou à effectuer par les équipes.
- Elle permet également de mesurer ou d’estimer la productivité.

Métriques logicielles

Quelques caractéristiques à mesurer pour estimer la qualité d'un logiciel

Quelles caractéristiques peut-on mesurer pour estimer la qualité d'un logiciel?

Il y a 3 principales catégories de métriques dans la qualité logicielle:

- Qualité applicative
- Maintenance applicative
- Respect des processus de développement

Quelles caractéristiques peut-on mesurer pour estimer la qualité d'un logiciel?

D'un point de vue technique, toutes les métriques de qualité logicielle sont liées à au code et à son adéquation avec le besoin métier qu'il est censé remplir.

Par souci de simplification nous allons donc distinguer 2 catégories de caractéristiques mesurables dans un logiciel:

- Les bugs, qui représentent un comportement qui n'est pas celui qui est attendu.
- Les défauts, qui sont des caractéristiques qui rendent plus difficile l'évolution du logiciel.

Maintenabilité du logiciel

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.

(Martin Fowler)

IZQuotes

Quelques métriques liées à la maintenabilité

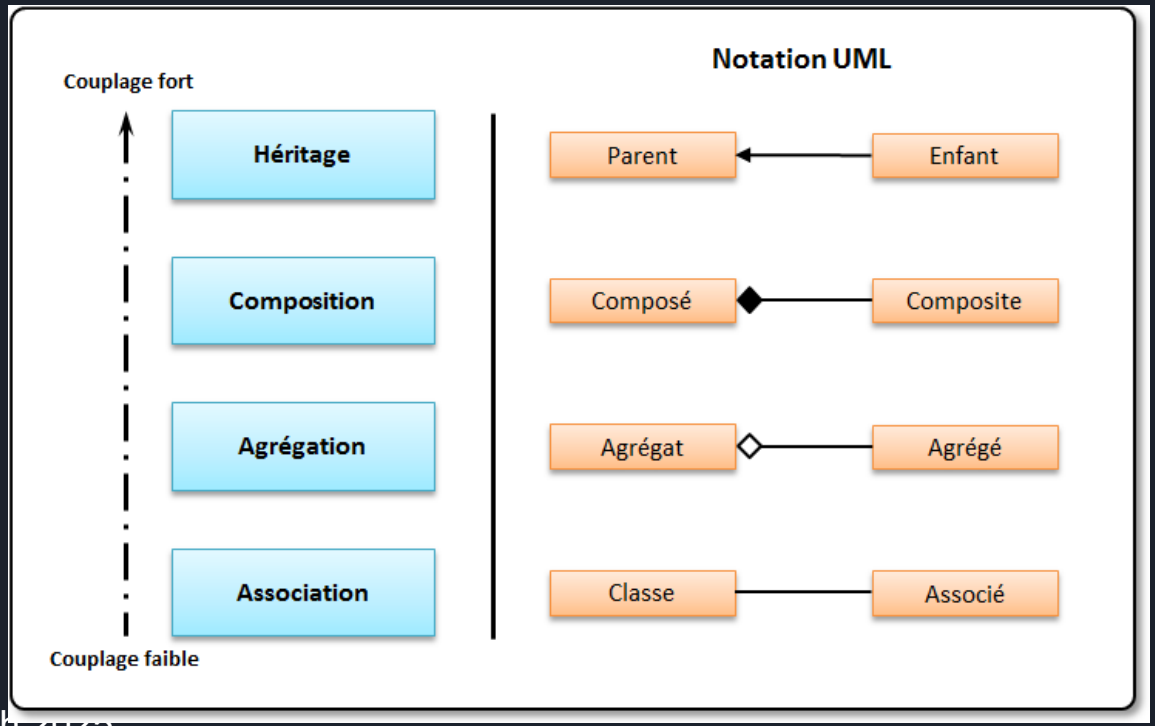
- Nombre de lignes de code
- Couplage
- Complexité cyclomatique
- Cohésion
- Densité de défauts (code smells)

Métriques liées à la maintenabilité:
Couplage

Définition :

Le couplage représente une relation entre deux éléments, classes ou modules. Il peut être « fort » ou « faible ». Son degré influe sur les répercussions des modifications d'un élément sur l'autre.

Métriques liées à la maintenabilité:
Couplage

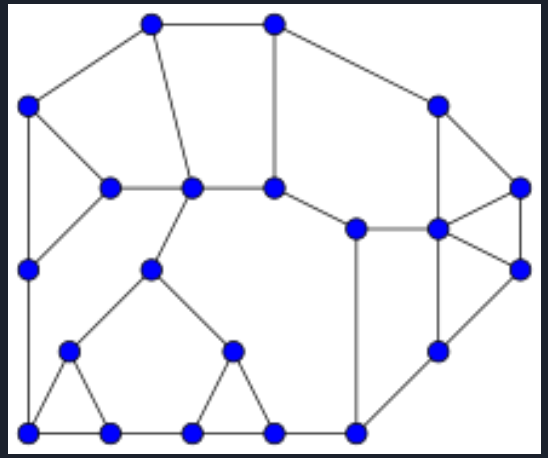


Métriques liées à la maintenabilité:
Couplage

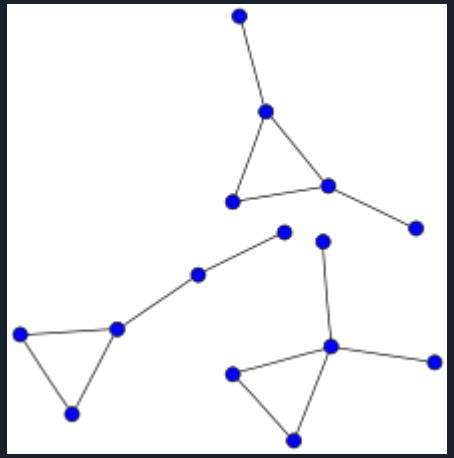
Ce qu'il faut en retenir :

Plus on a un couplage fort entre les composants, plus le logiciel est difficile à maintenir. On favorise **toujours** un couplage **faible**.

Métriques liées à la maintenabilité:
Complexité cyclomatique



Graphe connexe



Graphe non connexe avec trois composantes connexes

Métriques liées à la maintenabilité:
Complexité cyclomatique

Ce qu'il faut en retenir :

Plus on a de chemins algorithmiques, plus la complexité augmente.

Pour garantir un code de **qualité**, il faut **diminuer les tests conditionnels**.

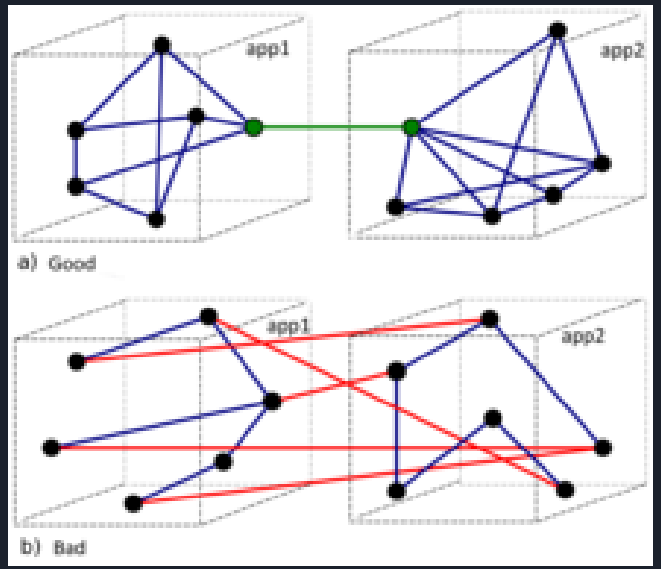
Métriques liées à la maintenabilité:
Cohésion

Définition :

Elle représente le degré d'accord entre les différents éléments d'un module. Elle peut servir à mesurer le degré d'encapsulation d'un module et le masquage de l'information.

Métriques liées à la maintenabilité:
Cohésion

Le niveau le plus élevé est atteint lorsque la classe ou le module est dédié à une seule et unique tâche bien spécifique.



Métriques liées à la maintenabilité:
Cohésion

Ce qu'il faut en retenir :

Le niveau accidentel est celui de plus **faible cohésion**, le niveau fonctionnel celui de plus **forte cohésion.**

Une bonne architecture logicielle nécessite **la plus forte cohésion possible**.

Métriques liées à la maintenabilité:
Densité de défauts: les « **code smells** »

Définition :

Ils représentent une caractéristique qui indique la probabilité d'un problème plus important.

Défini par Kent Beck dans les années 90, le terme a gagné en popularité après l'utilistion du concept par Martin Fowler dans son célèbre ouvrage « Refactoring ».

Métriques liées à la maintenabilité:
Densité de défauts: les « **code smells** »

Pour aller plus loin :

Le célèbre ouvrage de Martin Fowler : « Refactoring ».

Le site https://refactoring.guru/

Métriques liées à la maintenabilité:
Densité de défauts: les « **code smells** »

Pour aller plus loin :

Le célèbre ouvrage de Martin Fowler : « Refactoring ».

Le site https://refactoring.guru/

Merci pour votre attention