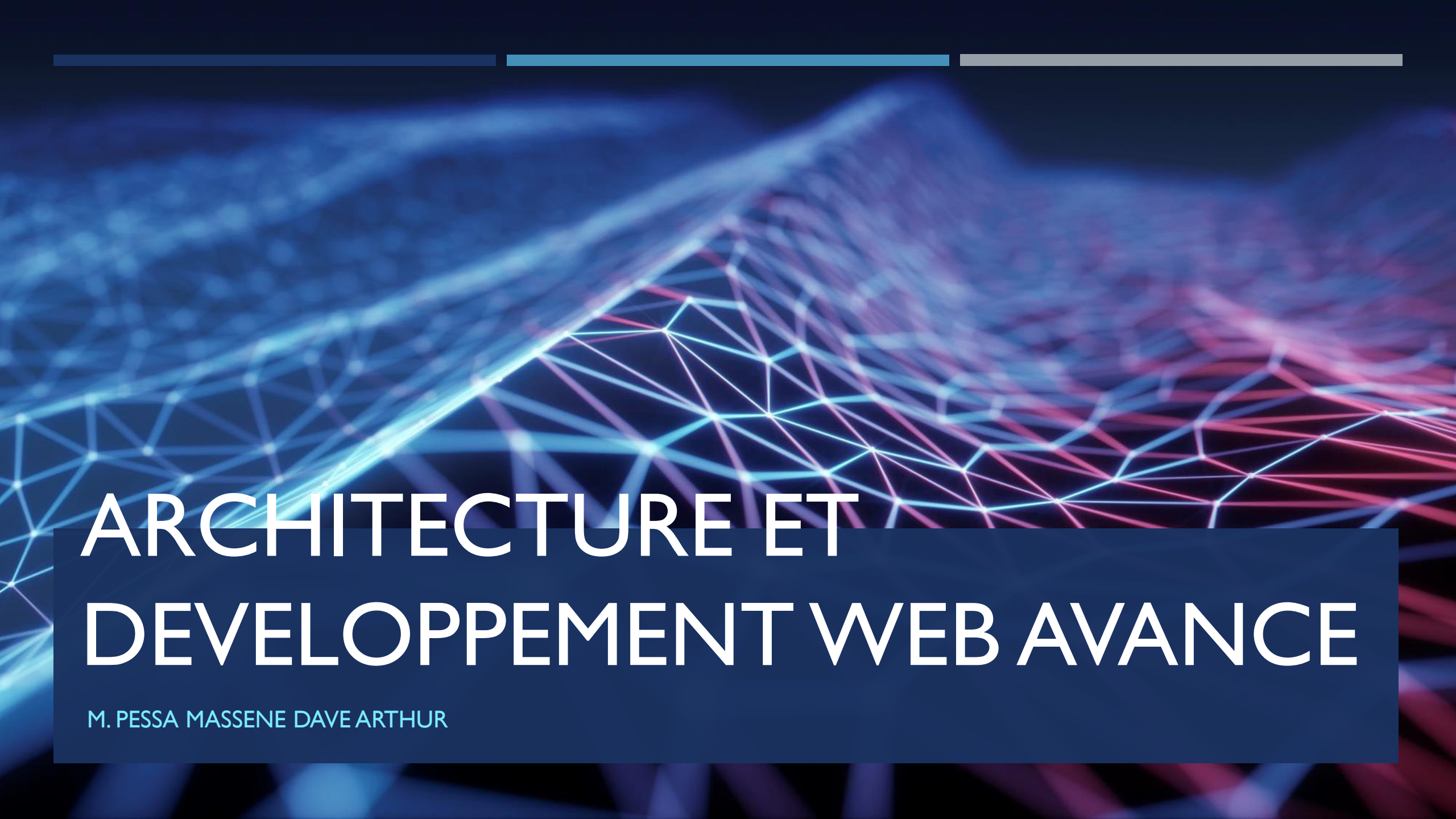


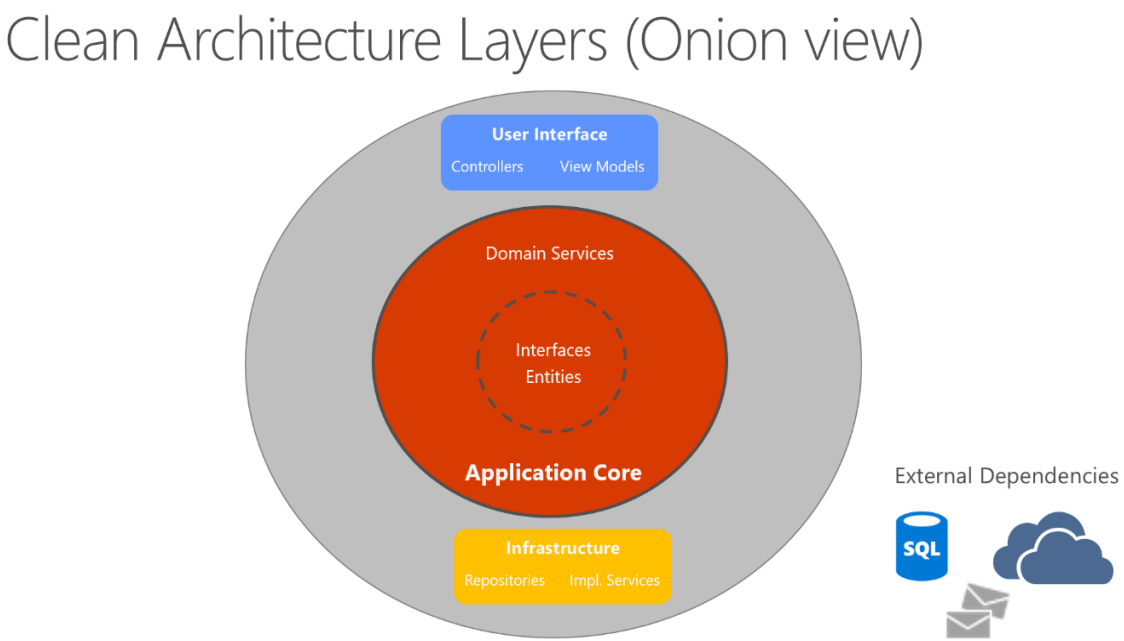
ARCHITECTURE ET DEVELOPPEMENT WEB AVANCE

M. PESSA MASSENE DAVE ARTHUR

PRESENTATION DU CHAPITRE I

Architectures des applications complexes

CHAP 1. ARCHITECTURE DES APPS COMPLEXES



- **Applications tout en tout-en-un**
- **Application monolithique**
- **Structurer une application en couches**
- **Applications avec une architecture en N couches conventionnelle**
- **Architecture propre**

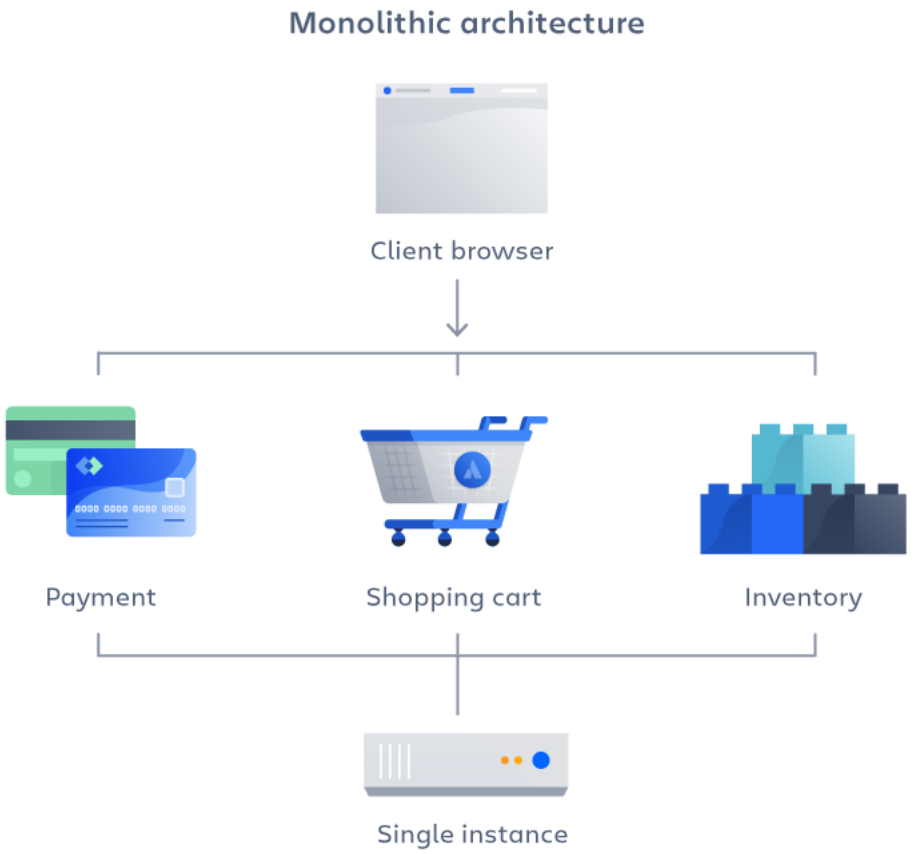
INTRODUCTION (RAPPELS)

- Les concepts sur une architecture générale (client –serveur)
- Les concepts sur les architectures n-tiers
- Les services web
- Logiciel, Les composants et les modules

APPLICATION TOUT EN UN

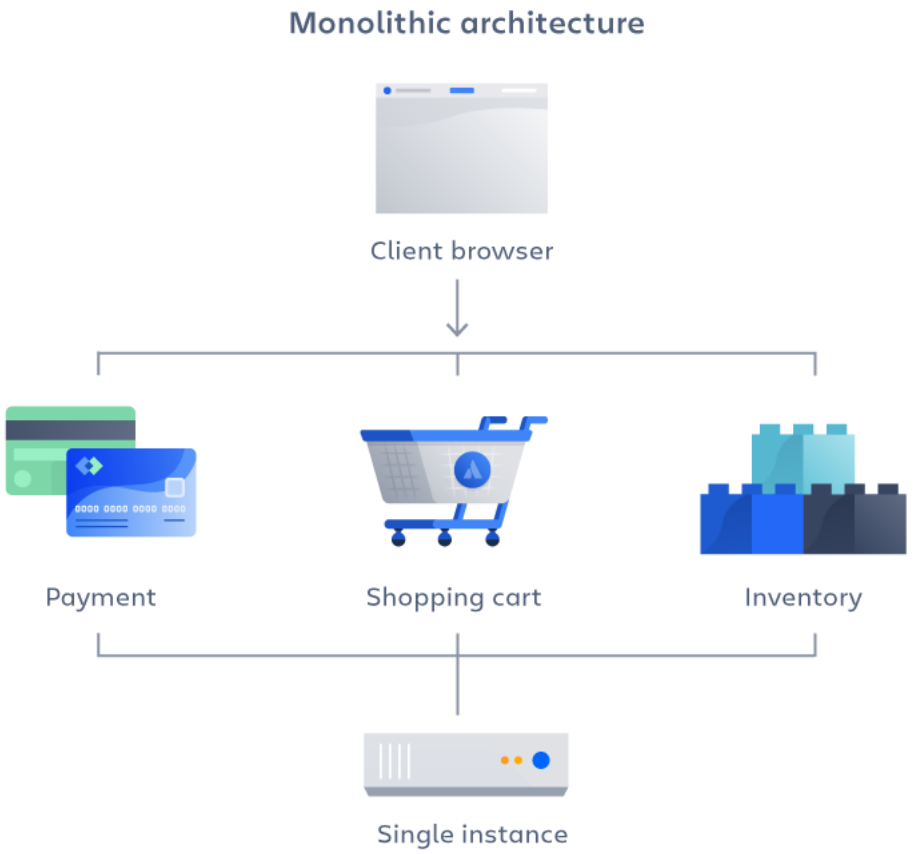
- L'ensemble du projet se réduit en une unité;
- Les logiques (les métiers) sont implémentés au sein d'un même projet;
- L'ensemble des ressources (modèles, services, vues) sont implémentés au sein d'une même unité;
- La solution d'un projet unique est simple mais elle peut devenir contraignante pour un projet plus complexes

APPLICATION MONOLITHIQUE



- Est une application qui est comporte de façon autonome (indépendante) bien qu'elle peut avoir des interactions avec des services ou des magasins de données
- L'ensemble de la logique (fonctionnalités) est implémenté dans un seul programme

APPLICATION MONOLITHIQUE



- Facile à déployer, à tester et à débugger car il s'agit d'une seule unité basée sur une seule base de code;
- Par contre la vitesse de développement est plus lente, la difficulté à faire évoluer et le manque de fiabilité pour des projets volumineux

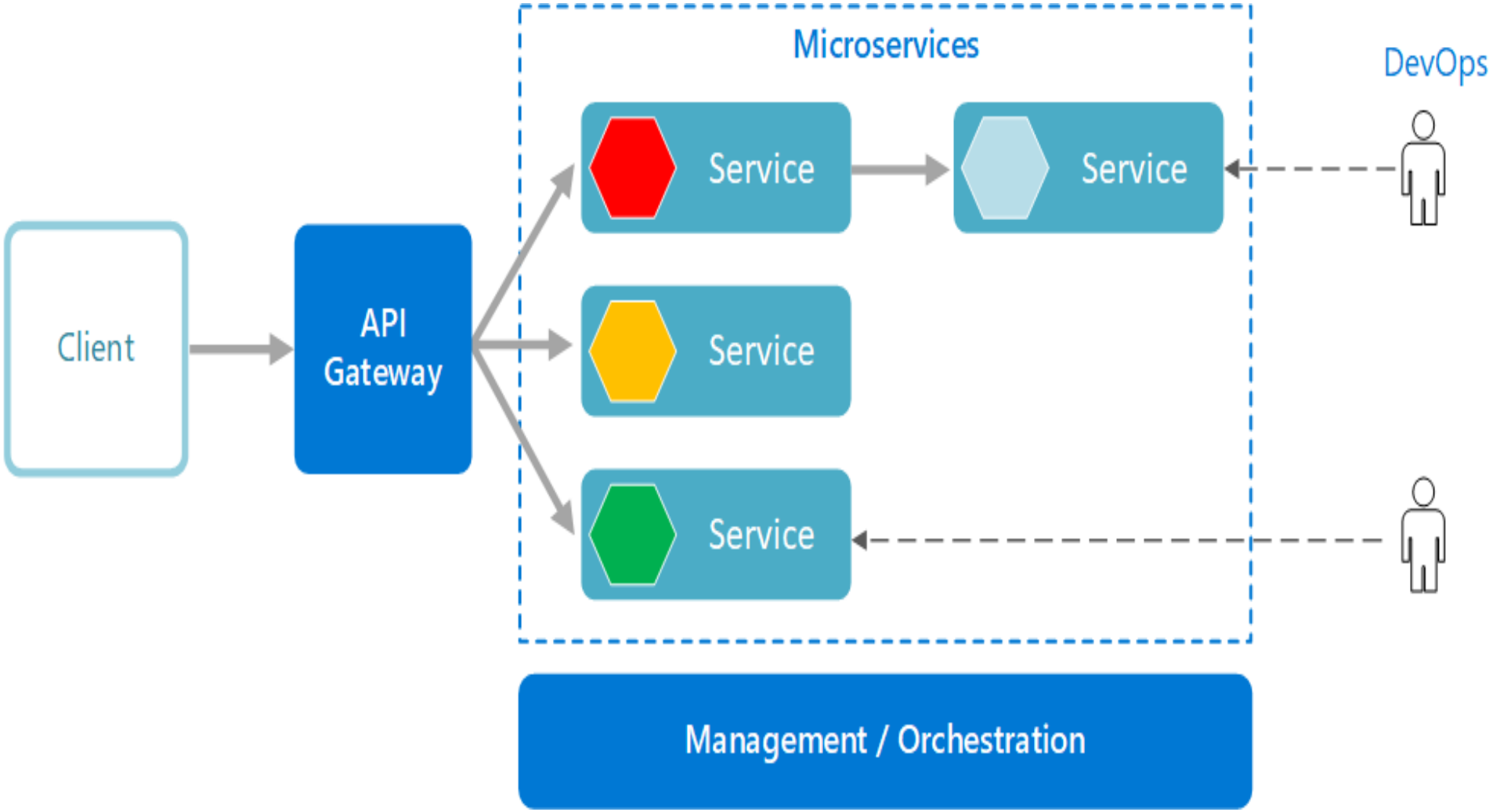
STRUCTURER UNE APPLICATION EN COUCHES

- Contrairement aux architectures précédentes, la complexité est gérer en scindant l'application en des parts responsables et indépendantes;
- Le but est de séparer les aspects (métiers) indépendants les uns des autres mais pouvant être réutilisable et facilement évolutive;
- Ceci permet de réduire la quantité du code à écrire et appliquer le principe « DRY »;

STRUCTURER UNE APPLICATION EN COUCHES

- Cette approche est adéquate pour l'implémentation des micro services : **architecture orienté micro services**
- Ici les services sont autonomes et doivent implémenter une fonctionnalité précise dans un contexte encadré
- Un service peut être implémenté suivant une code base distinct. *Ex. sercice 1 => Java, service 2 => Python, etc*: **programmation polyglotte**
- Chaque service est responsable de la persistance de leur propre données

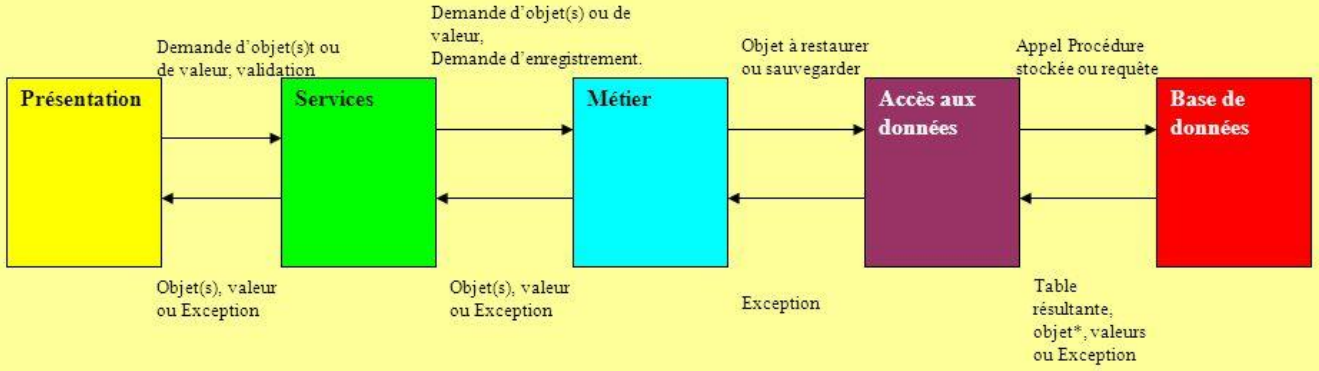
STRUCTURER UNE APPLICATION EN COUCHES

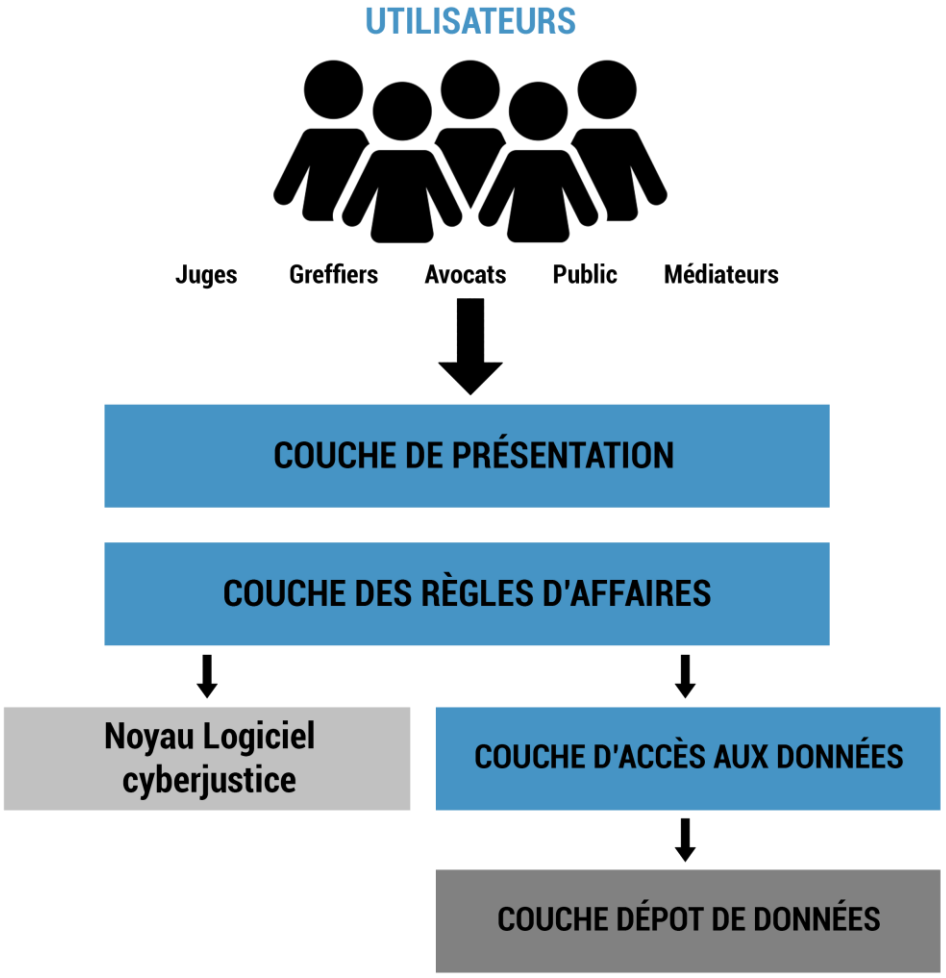


STRUCTURER UNE APPLICATION EN COUCHES

Architecture

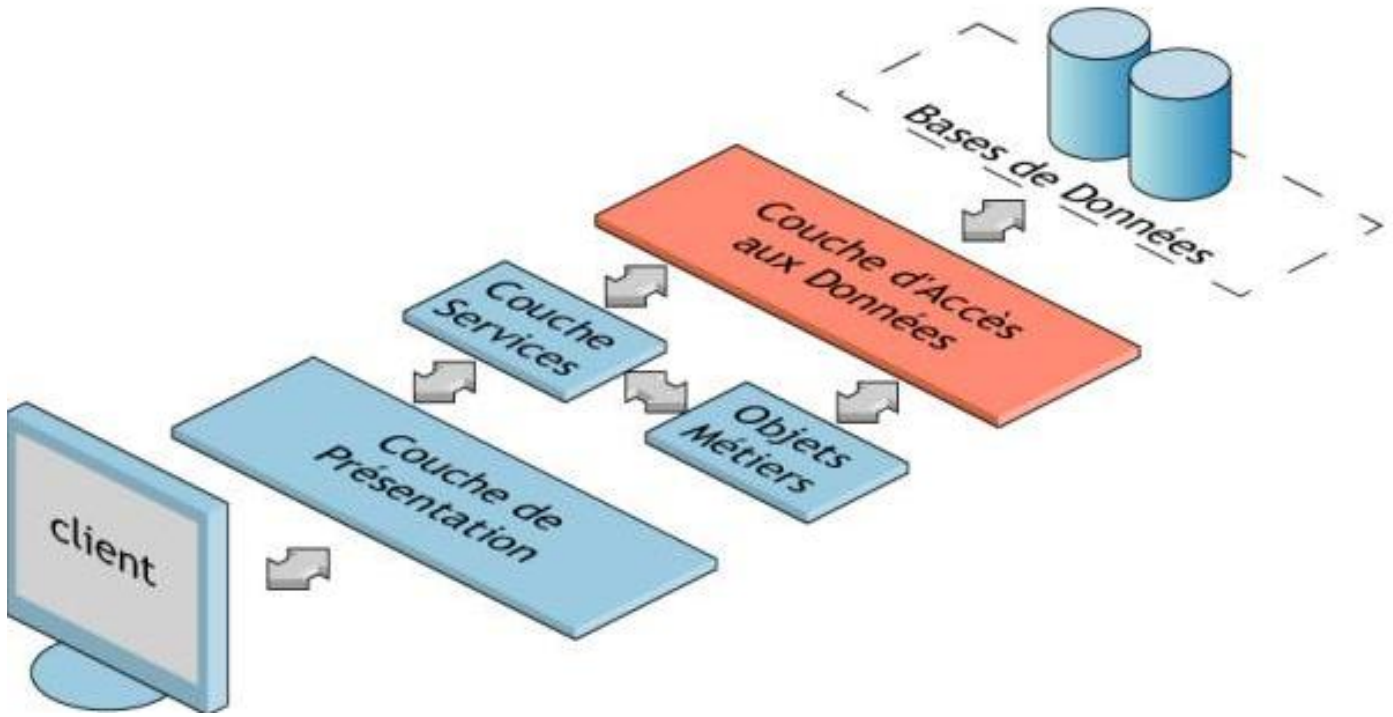
Les Couches





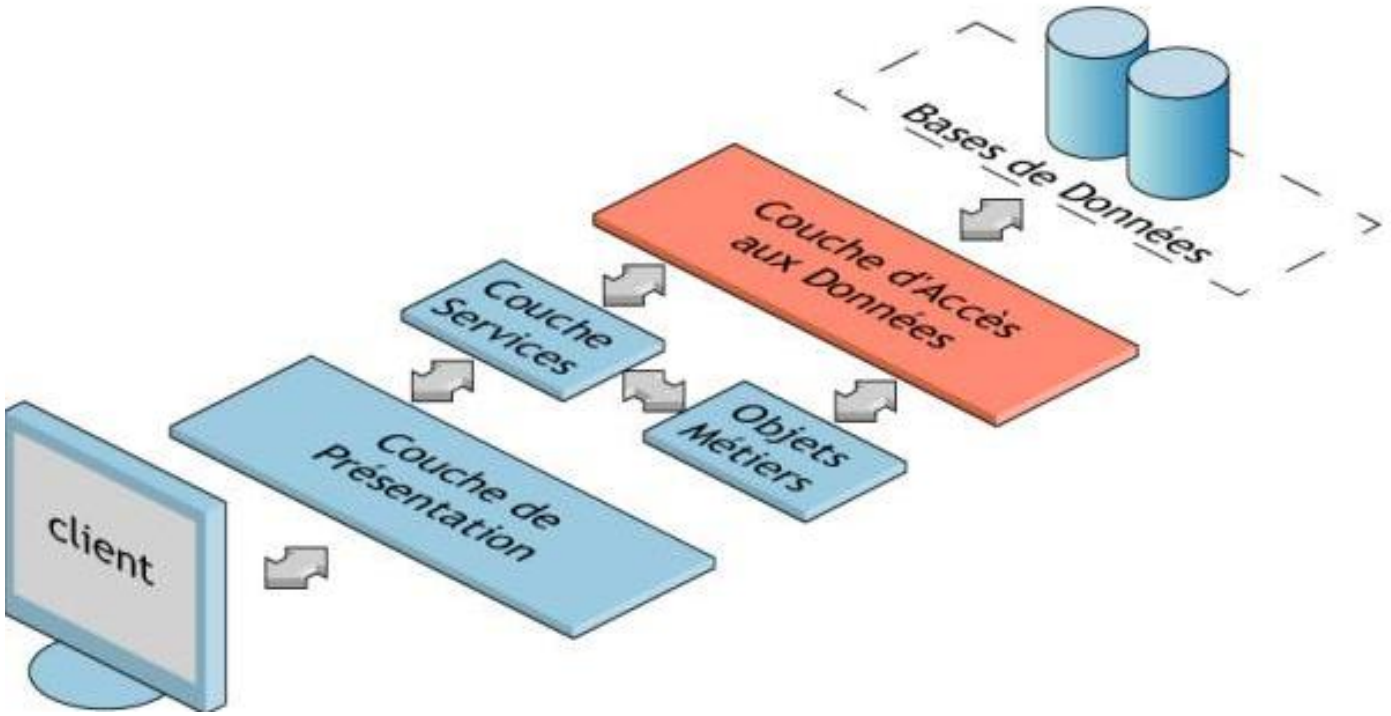
APPLICATIONS AVEC UNE ARCHITECTURE EN N COUCHES

- Les couches interagissent les unes des autres. Via le UI, un utilisateur peut émettre une requête depuis son client (*Ordinateur, Smartphone, etc.*) la réponse retournée est relative à la ressource demandée.



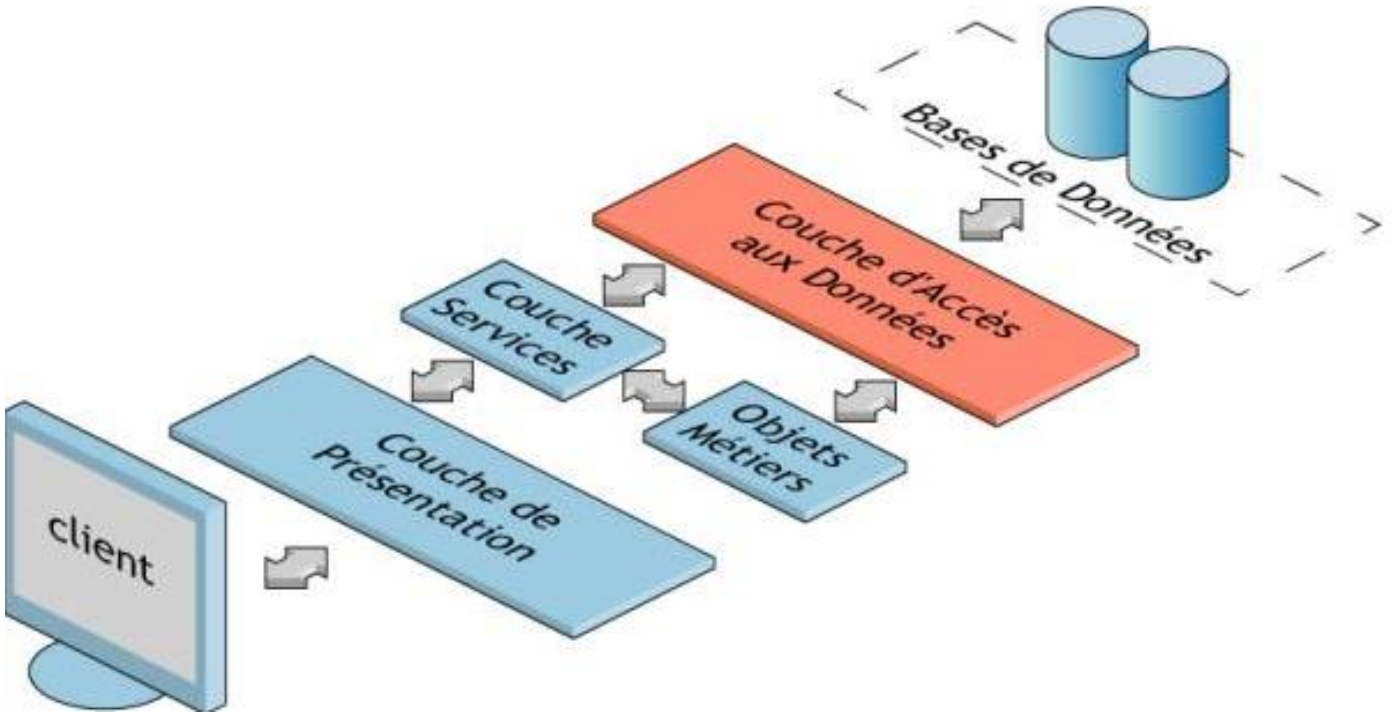
APPLICATIONS AVEC UNE ARCHITECTURE EN N COUCHES

- Les couches peuvent être tester séparément **sans altérer l'opérabilité** des autres couches.
- La couche d'accès aux données peut être changer (l'usage d'autres SGBD) sans affecter le projet de manière générale



APPLICATIONS AVEC UNE ARCHITECTURE EN N COUCHES

- Les couches peuvent être tester séparément **sans altérer l'opérabilité** des autres couches.
- La couche d'accès aux données peut être changer (l'usage d'autres SGBD) sans affecter le projet de manière générale



APPLICATIONS AVEC UNE ARCHITECTURE EN N COUCHES

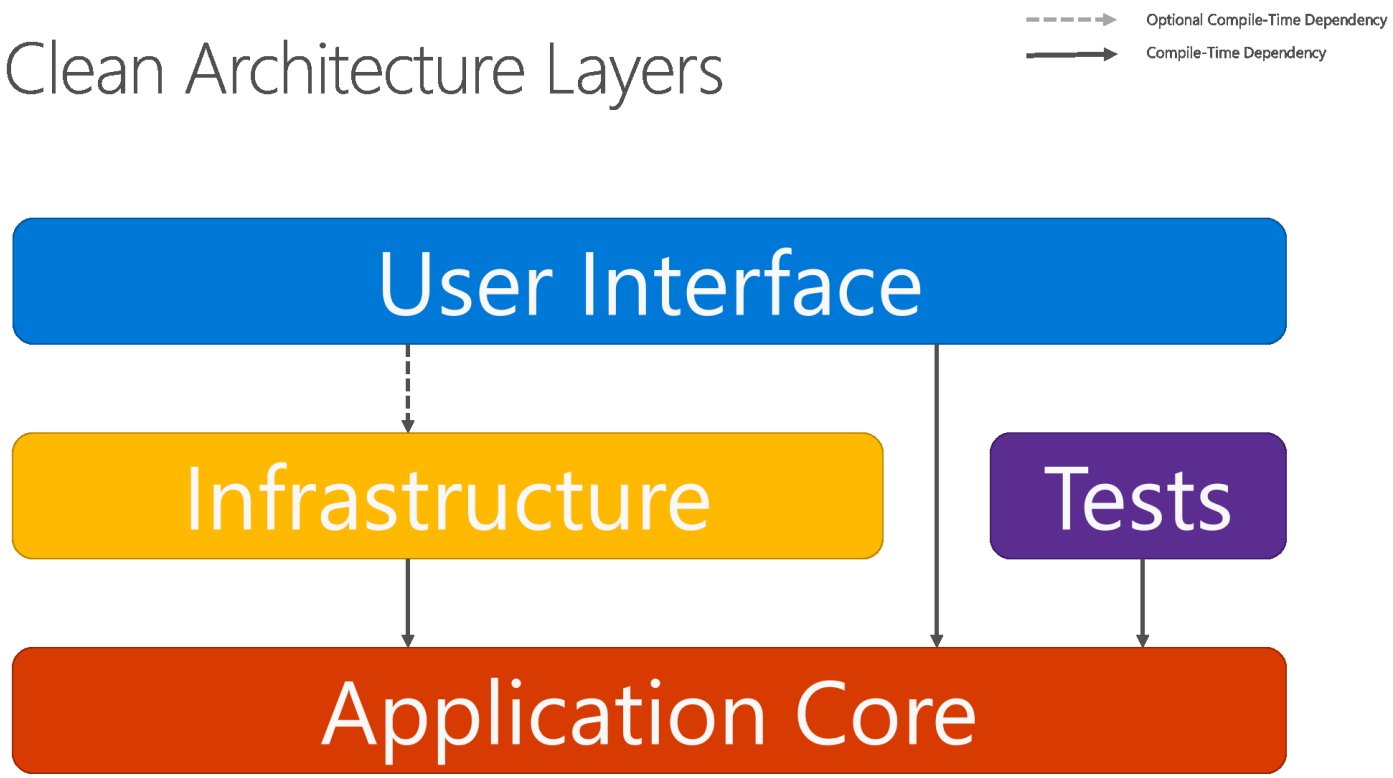
- Cependant l'inconvénient de cette approche « architecture en couche », réside sur leurs dépendances entre elles;
- La couche UI a besoin des couches inférieures pour obtenir un résultat;
- Les tests sont de plus en plus coûteux vu la multiplicité des micro services

ARCHITECTURE PROPRE

- Est une architecture en couches
- Inventé par Robert C. Martin, via son livre « *Architecture logicielle propre* », 2020
- Le terme propre renvoie à « coder proprement ».

ARCHITECTURE PROPRE

En d'autres termes implémenter à la lettre l'architecture logicielle qui a été défini de plus réduire au maximum les dépendances pour faciliter l'évolution du logiciel.



CONCEPTION ORIENTEE COMPOSANT OU MODULE

- La couche « **application core** » : Les entités et les interfaces de l'application (implémentées par les services domaine) se trouvent au centre même du diagramme.
- Tandis que les couches « **infrastructures** » et « **interface utilisateur** » dépendent de la couche « application core ».
- Dans la couche infrastructures on peut retrouver des « repositories » et « d'autres services ou métiers implémentés »

THANK YOU

ARTHURPESSA@INSTITUTSAINTJEAN.ORG