# Approches Opposables au Cycle MVP

Le MVP (Minimum Viable Product) n'est pas qu'un livrable — c'est une philosophie de développement centrée sur l'itération rapide, la valeur utilisateur, et la validation par l'usage réel. Ce document présente une série d'approches qui s'y opposent ou l'alternent sur différents axes du cycle de développement.

---

## Panorama des Paradigmes Opposables

| Paradigme | Logique fondamentale | Différence clé vs MVP | Quand préférer ? |
|---|---|---|---|
| **Big Design Up Front (BDUF)** | Tout concevoir en amont | Préfère **prévoir** plutôt que découvrir | Projets à fort enjeu, réglementaire, spatial |
| **Waterfall / Cycle en V** | Cycle séquentiel rigide | Pas d'itération rapide, **phase unique** | Industrie, critique, certification |
| **Prototypage exploratoire** | Explorer avant tout développement | **Validation sans produit**, parfois sans code | UX précoce, design thinking, R&D |
| **Design to Specs (Contractualisé)** | Répondre à un cahier des charges | Pas de test marché, **exécution d'un contrat** | Projets commandités, marchés publics |
| **Agile (hors MVP)** | Itération continue sans "produit viable" | MVP n'est **pas requis** à chaque sprint | Logiciels internes, équipes expérimentées |
| **Release-Driven Development** | Planification par **version stable** | Préfère les jalons livrables complets | Produits structurants, B2B long terme |
| **Architecture-First / Model-Driven** | Démarre par la modélisation métier/tech | La valeur utilisateur n'est pas prioritaire | Legacy, interopérabilité, domaine complexe |
| **Scientific Approach (Test First)** | Tester des **hypothèses** avant de bâtir | Ne cherche pas de MVP mais des **preuves** | DeepTech, IA, bio, climat |
| **Fake Door / Pretotyping** | Tester sans rien construire | Le MVP est **évité** par simulation | Early market validation, low-cost testing |
| **No Code / Low Code expérimentaux** | Construire sans ingénierie logicielle | **Cycle hyper-rapide**, jetable, non itératif | Prototypes, démonstrateurs business |

## Analyses détaillées

### 1. Big Design Up Front (BDUF) / Cycle en V

- **Cycle** : Analyse → Conception → Dév → Tests → Livraison
- **Opposition au MVP** : Assume que le besoin est **prédictible** ; peu d'espace pour l'itération.
- **Cas d'usage** : Défense, aéronautique, santé, normes ISO, etc.

### 2. Scientific / Hypothesis-Driven

- **Cycle** : Hypothèse → Expérimentation → Analyse → Recalibrage
- **Opposition** : Pas de focus sur le marché, mais sur des **preuves scientifiques**.
- **Usage** : Recherche, IA, biotech, climat, données.

### 3. Design Thinking / Prototypage UX

- **Cycle** : Empathie → Définition → Idéation → Prototype → Test
- **Opposition** : Met l'**expérience utilisateur avant la faisabilité technique**.
- **Utilisé pour** : Innovation, design spéculatif, produits centrés humain.

### 4. Contractualisation / Cahier des charges

- **Cycle** : Spécifications → Contrat → Réalisation
- **Opposition** : Pas d'évolution du besoin ; **obligation de moyens**.
- **Contexte** : Marché public, prestation externalisée.

### 5. Architecture-First / Model Driven

- **Cycle** : Conception → Architecture → Implémentation
- **Opposition** : Vise la **qualité structurelle** avant la mise en marché.
- **Cas** : ERP, systèmes distribués, plateformes interopérables.

### 6. Release Planning / Jalons fonctionnels

- **Cycle** : Roadmap → Version stable → Mise en production
- **Opposition** : Valorise la **cohérence d'ensemble**, pas l'expérimentation rapide.
- **Usage** : Produits B2B, outils internes, logiciels critiques.

### 7. No Code / Prototypes jetables

- **Cycle** : Construction rapide → Test → Abandon ou passage à l'industrialisation
- **Opposition** : **Ignorent volontairement** les contraintes de production logicielle.
- **Idéal pour** : Validation commerciale, démonstrateurs marketing.

---

## 🧠 Résumé par axes de friction

| Axe de friction | MVP | Approche opposable |
|---|---|---|
| **Vitesse de mise en marché** | Privilégiée | Parfois sacrifiée pour rigueur |
| **Incertitude produit** | Acceptée, voire recherchée | Réduite par modélisation ou cadrage |
| **Spécifications** | Evolutives, découvertes | Fixées contractuellement |
| **Validation** | Par usage réel | Par simulation, preuve formelle ou maquette |
| **Priorité** | Valeur utilisateur | Architecture, contrat, robustesse |

## Conclusion

Le MVP est un **outil puissant**, mais pas une réponse universelle. Chaque approche propose une vision du cycle produit qui **hiérarchise différemment** :

- la **vitesse**,
- la **qualité**,
- la **compréhension du besoin**,
- la **structure logicielle**.

Un bon architecte ou responsable produit sait **choisir ou hybrider** ces cycles selon :

- le contexte métier,
- le niveau de risque,
- les parties prenantes,
- la maturité de l'équipe.