Ingénierie et Conduite de projets numériques

1. Fondamentaux

Concevoir, organiser, produire et piloter des dispositifs numériques interactifs

1. Qu'est ce qu'un projet numérique ?

- Un projet numérique, c'est une **organisation temporaire** qui mobilise des compétences variées pour livrer une **expérience interactive utilisable** dans un contexte réel.

- Trois critères permettent de distinguer un vrai projet numérique d'une simple expérimentation ou d'un objet créatif :
  - **objectif clair :** résoudre un problème, atteindre une cible précise
  - **public identifié :** pas seulement des spectateurs, mais des utilisateurs actifs
  - **cadre institutionnel ou professionnel :** il y a un commanditaire, un financeur, parfois un cahier des charges, et surtout des comptes à rendre.

- Il est destiné à être **mis en usage réel** : le public **n'est pas spectateur mais acteur**. Le projet doit donc être pensé **pour l'usage** et pas seulement pour être vu.

- Un projet numérique est donc différent d'une œuvre artistique libre : on ne cherche pas seulement l'expression, mais l'usage.
  Il est différent d'un prototype technique : le prototype prouve une faisabilité, le projet doit être **mis en usage réel**.

1. Qu'est ce qu'un projet numérique ?

Exemples de projet :

- **Webdoc** → projet numérique si diffusé avec un objectif de médiation et une cible définie
- **Installation immersive** → projet numérique si pensée pour un musée ou un festival, avec un parcours utilisateur.
- **Application mobile** de médiation culturelle → projet numérique car elle a un objectif, un public, un cadre institutionnel, un déploiement.

Exemple : Développement d'une application mobile de covoiturage étudiant

- **Objectif** : mettre en relation les étudiants d'une université pour organiser des trajets partagés.
- **Fonctionnalités clés** : création de compte, recherche de trajets, système de réservation, messagerie interne, notifications.
- **Livrables** : prototype UX, maquettes UI, application mobile (iOS/Android), documentation technique.
- **Équipe type** : chef de projet, product owner, UX/UI designers, développeurs front/back, DevOps, équipe QA/test, marketing digital.
- **Processus** : cadrage → étude des besoins → prototypage → développement → tests → déploiement → communication/lancement.
- **Spécificité** : importance de la gestion des contraintes techniques, de la sécurité des données et de la priorisation des fonctionnalités.

1. Qu'est ce qu'un projet numérique ?

- Quand on parle de **projet numérique**, on ne parle pas juste de produire quelque chose en ligne.

- On parle d'un **processus structuré** qui vise à **construire une expérience interactive**, avec :
  - des **étapes à respecter**,
  - des **ressources à gérer**,
  - des **contraintes à intégrer**,
  - et surtout : un **public final à considérer**.

- Ce qui le distingue, c'est la **nécessité :**
  - **de coordination** entre plusieurs expertises,
  - de **rendre des comptes** (à un commanditaire, à une institution, à un public),
  - et de **livrer un résultat fonctionnel**, qui marche, et qui touche ses utilisateurs.

2. Rôles et équipes projet

- Un projet numérique est toujours dans une **dynamique collaborative**, où chaque rôle a sa part de responsabilité.

- La difficulté est de **comprendre comment son travail impacte celui des autres** :
  - Un choix éditorial peut bouleverser l'UX.
  - Un persona mal défini peut rendre des semaines de développement inutiles.
  - Une contrainte technique peut amener à revoir toute la direction artistique.

- C'est pour ça qu'il est essentiel de comprendre les **interdépendances** dans une équipe projet.

- **Une équipe projet, ce n'est pas une pyramide, c'est un écosystème :**
  - Composée de **profils complémentaires**
  - Organisée selon le **cadre méthodologique choisi**
  - Fonctionne souvent de manière **itérative** et **adaptative**

- Le rôle du chef de projet, c'est **de fluidifier tout ça** : écouter, anticiper, arbitrer, relancer, faire circuler l'info, sans prendre toute la charge sur soi.

2. Rôles et équipes projet

Un projet numérique réunit une équipe pluridisciplinaire, c'est un **écosystème collaboratif** où chaque rôle dépend des autres.

On y trouve des rôles « classiques » et d'autres, plus spécifiques, qui s'organisent dans une logique agile :

- **Chef de projet / Scrum Master** : coordonne, gère budget et planning, assure le lien entre tous les acteurs, facilite l'organisation.
- **Product Owner (PO)** : porte la vision produit, priorise les fonctionnalités, valide les livrables fonctionnels (souvent côté client).
- **Client / Commanditaire** : exprime les besoins, valide les grandes orientations, finance le projet.
- **Directeur Artistique (DA)** : définit l'identité visuelle et créative globale, garantit la cohérence graphique et le respect de la charte.
- **Auteur / Concepteur-rédacteur** : produit les textes, scénarise les contenus, adapte le ton et le discours.
- **UX Designer** : mène les recherches utilisateurs, définit les parcours, conçoit des prototypes.
- **UI Designer** : transforme les parcours en maquettes visuelles cohérentes avec l'identité.
- **Développeurs (Front-end / Back-end / Mobile)** : construisent l'application et intègrent les fonctionnalités prioritaires.
- **DevOps** : assure la mise en production, l'intégration continue et la stabilité technique.
- **Marketing / Communication** : prépare la promotion, rédige les contenus marketing, assure la visibilité du projet.
- **Sécurité / RGPD** : valide la conformité (gestion des données personnelles, sécurité des paiements, etc.).
- **Utilisateurs** : testent les prototypes, donnent du feedback, participent à l'amélioration continue.

2.1 Rôle : Chef de projet

Rôle
- Garantir la **bonne exécution du projet** en coût, qualité, délai.
- Coordonner les équipes et gérer les parties prenantes.

Responsabilités
- Planification (jalons, roadmap).
- Suivi (avancement, risques, budget).
- Animation des réunions et reporting.
- Gestion des contrats et prestataires.

Livrables
- Plan de projet / Gantt.
- Tableaux de bord d'avancement.
- Rapports de suivi.

Points de vigilance
- Risque de doublon avec PO (produit vs projet).
- Trop de reporting = perte de temps.



2.2 Rôle : Product Owner – PO -

Rôle

- Porte la **vision du produit**.
- Garantit la **valeur délivrée** à l'utilisateur et au métier.
- Définit et **priorise le backlog**.

Responsabilités

- Porte la **vision du produit**.
- Garantit la **valeur délivrée** à l'utilisateur et au métier.
- Définit et **priorise le backlog**.

Livrables

- **Backlog produit** (priorisé, mis à jour).
- **User Stories** avec critères d'acceptation.
- **Vision produit** / roadmap fonctionnelle.

Points de vigilance

- Risque de se transformer en "gestionnaire de tickets".
- Nécessite **autorité réelle** pour arbitrer → sinon blocages.



2.3 Rôle : UX Designer

Rôle
- Comprendre les **utilisateurs**, leurs contextes, leurs besoins.
- Concevoir une **expérience fluide** et adaptée.

Responsabilités
- Recherche utilisateur (interviews, observations, personae).
- Création de parcours utilisateurs et scénarios.
- Prototypage (wireframes, maquettes).
- Tests utilisateurs et itérations.

Livrables
- **Personae, User Journey Maps**.
- **Wireframes, prototypes interactifs**.
- **Rapports de tests utilisateurs**.

Points de vigilance
- Risque d'être réduit à "faire des écrans" sans recherche réelle.
- Nécessite un **temps dédié** pour tester et itérer.

2.4 Rôle : Développeur (Frontend, Backend, Fullstack, Mobile, etc.)

Rôle
- Transformer des spécifications en **code robuste, performant et maintenable**.

Responsabilités
- Écrire du code conforme aux standards (revues, tests, clean code).
- Intégrer l'UI/UX fournie par le design.
- Gérer la performance, la sécurité et la scalabilité.
- Documenter et transmettre (README, API docs).

Livrables
- **Code source** versionné (Git).
- **Tests automatisés**.
- **Documentation technique** (APIs, schémas).

Points de vigilance
- Tension possible avec PO/design sur la faisabilité ou les délais.
- Nécessité de **communiquer clairement** avec non-tech.

2.5 Rôle : UI Designer

Rôle
- Traduire l'expérience pensée en UX en **interface visuelle claire, esthétique et cohérente**.

Responsabilités
- Créer des maquettes haute fidélité.
- Définir un **design system** (couleurs, typographies, composants).
- Garantir l'accessibilité visuelle (contrastes, tailles, responsive).

Livrables
- **Maquettes visuelles** (Figma, Sketch…).
- **Design system** documenté.
- **Guidelines UI** pour développeurs.

Points de vigilance
- Dépendance forte avec les devs (intégration fidèle).
- Risque de conflit esthétique / contrainte technique.

2.6 Rôle : Dev Ops

Rôle
- Assurer la **mise en production rapide et sécurisée**.
- Gérer les environnements (dev, staging, prod).

Responsabilités
- Automatiser CI/CD.
- Déployer en toute sécurité (rollback, monitoring).
- Gérer la scalabilité et la résilience (cloud, microservices).

Livrables
- **Pipelines CI/CD**.
- **Scripts d'automatisation** (Terraform, Ansible).
- **Dashboards de monitoring**.

Points de vigilance
- Souvent sous-estimé en début de projet → dette d'infra.
- Risque de silos entre devs et ops si pas d'outils partagés.

2.7 Rôle : Expert Sécurité / RGPD

Rôle

- Assurer que le produit respecte les normes de **sécurité et de protection des données**.

Responsabilités

- Audits de sécurité.
- Mise en conformité RGPD (consentement, droit à l'oubli).
- Gestion des incidents (plan de réponse, logs).

Livrables

- **Registre des traitements**.
- **Rapports d'audit**.
- **Politique de sécurité**.

Points de vigilance

- Tension entre sécurité et rapidité de développement.
- Risque de "cocher des cases" sans réelle intégration des enjeux.

2.8 Rôle : Responsable Communication / Marketing Digital

Rôle
- Préparer et accompagner la **mise sur le marché**.
- Gérer l'**adoption** et la communication autour du projet.

Responsabilités
- Définir le positionnement produit.
- Organiser la stratégie de lancement.
- Mesurer l'adoption (KPIs marketing).

Livrables
- **Plan de communication**.
- **Campagnes digitales** (social media, newsletters).
- **Dashboards marketing** (acquisition, conversion).

Points de vigilance
- Risque de décalage entre discours marketing et réalité produit.
- Collaboration tardive = adoption faible.

2.9 Rôles et équipe de projet

- Le rôle du chef de projet n'est pas de "tout faire" ou de "tout contrôler", mais de **fluidifier les échanges**, éviter les silos, anticiper les blocages, et maintenir la cohérence.

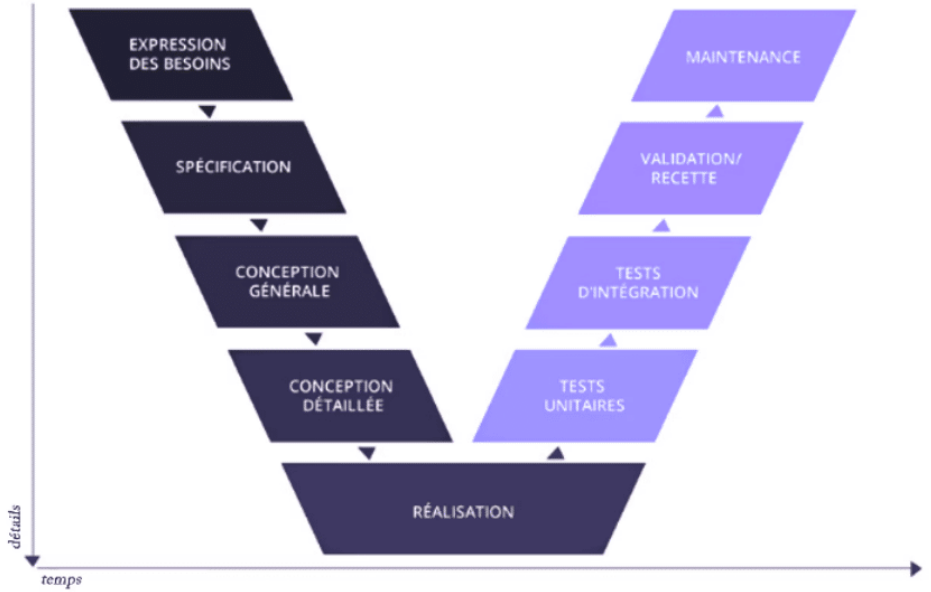
- **Soft Skills du chef de projet numérique :**
  - Communication claire et adaptée (équipes techniques / non-tech).
  - Capacité d'arbitrage et de négociation.
  - Leadership collaboratif (écouter, motiver, relancer sans imposer).
  - Gestion du temps et du stress.
  - Culture numérique transversale (UX, dev, marketing).

> Un chef de projet n'est pas qu'un planificateur, c'est un **facilitateur**.

3. Différents modèles

A. Cycle en V (modèle traditionnel) :

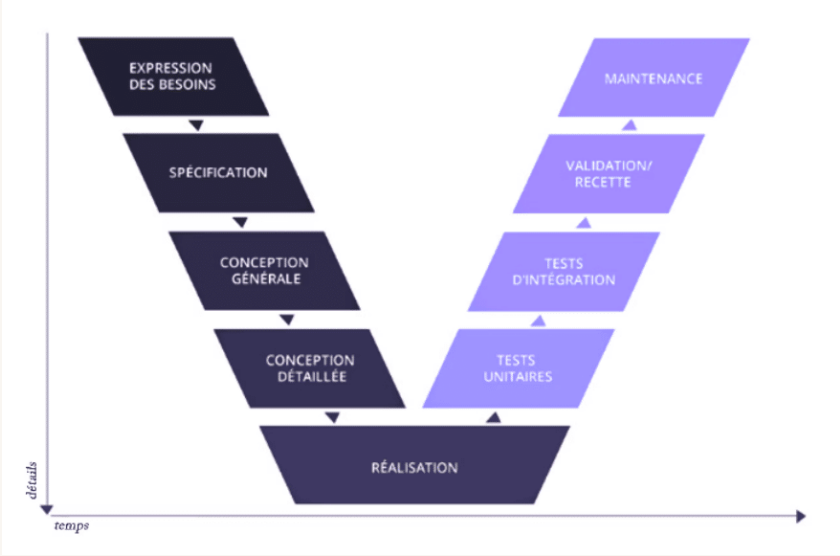
- Tout est planifié dès le départ
- Peu d'interaction avec les utilisateurs pendant le développement
- **Le côté gauche** correspond aux phases de conception, **la pointe** représente la réalisation,
  et **le côté droit** englobe les étapes de validation.
- Approche **linéaire et progressive**.
  Les différentes phases du projet s'enchaînent selon une séquence prédéfinie.
- Cette linéarité implique que chaque étape doit être finalisée avant de passer à la suivante, car les décisions prises conditionnent la suite du projet.



3. Différents modèles

A. Cycle en V

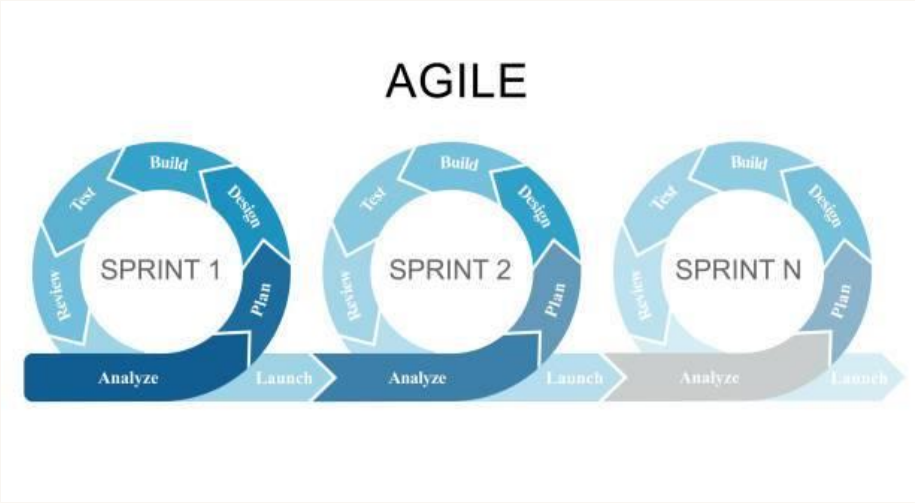
- Modèle prédictif, linéaire
- On avance étape par étape : spécifications, conception, développement, tests, validation.
- Très utilisé dans les **marchés publics** ou les systèmes critiques (santé, transport).
- L'avantage : grande traçabilité, tout est documenté, chaque étape validée.
- L'inconvénient : peu de flexibilité, on ne revoit pas les choix facilement en cours de route.
- Si un problème utilisateur émerge tard, corriger coûte extrêmement cher.



3. Différents modèles

B. Agile : Scrum

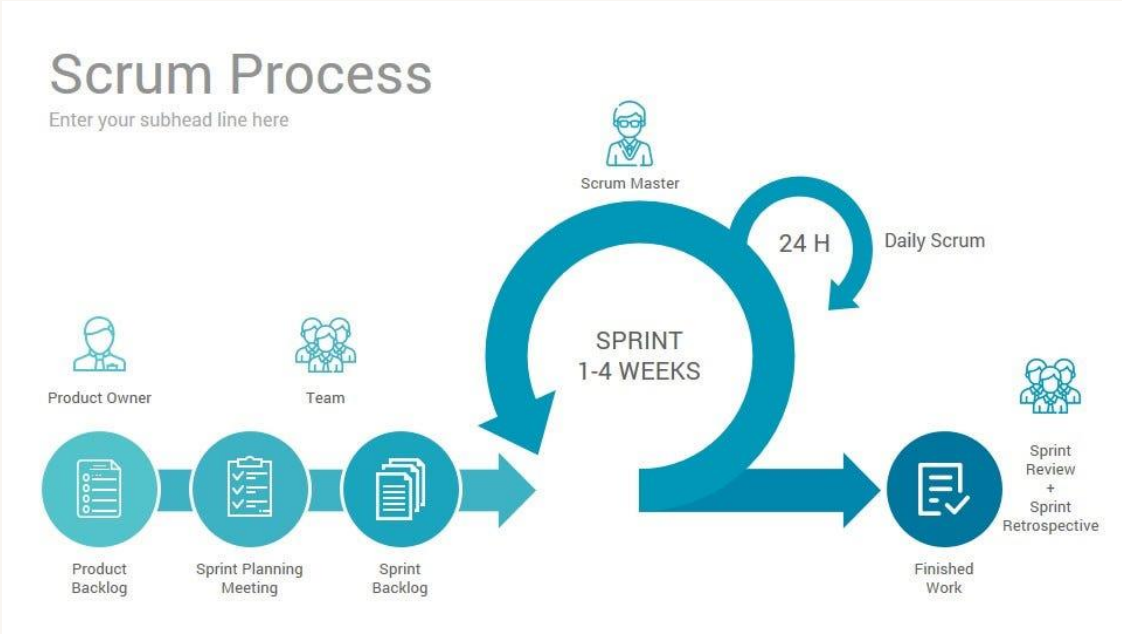
- Les approches **Agiles** comme Scrum reposent sur des cycles **incrémentaux et itératifs**.
- On avance par petits pas : des **sprints de 2 à 3 semaines**.
- À chaque sprint, on produit un incrément utilisable, qu'on **teste avec les utilisateurs**.
- On **ajuste en continu** grâce aux retours utilisateurs
- On **priorise ce qui a le plus de valeur** dès le début
- **Scrum** repose sur des sprints (itérations courtes) et des réunions quotidiennes pour suivre l'avancement du projet.



3. Différents modèles

B. Agile : Scrum

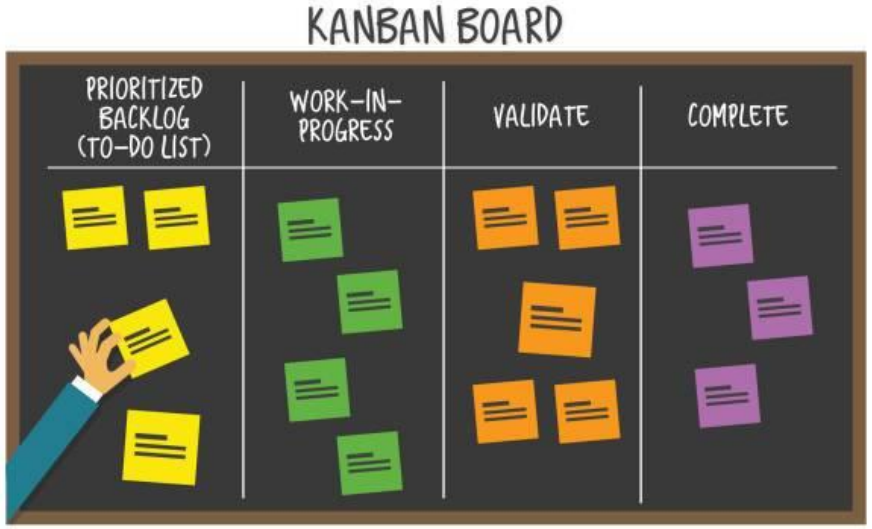
- Avantages : flexibilité, capacité à s'adapter, feedback utilisateur rapide.
- Inconvénients : il faut une équipe disponible, un PO fort, et une organisation réactive.
- **Exemple :**
  Développement d'une nouvelle application culturelle dont on ne connaît pas encore l'usage
  → Scrum permet de tester rapidement des hypothèses et d'ajuster.



3. Différents modèles

C. Agile : Kanban

- Kanban est une **approche visuelle** qui utilise un **tableau pour suivre l'état des tâches** et faciliter la communication entre les membres de l'équipe.
- **Les tâches sont représentées par des cartes qui se déplacent entre différentes colonnes (à faire, en cours, terminé**
- Avantage : visualisation claire, détection des goulots d'étranglement.
- Utilisé pour la **maintenance**, les évolutions continues, ou des projets où le flux est permanent.
  Par exemple une équipe support qui gère des bugs et demandes utilisateurs
  → Kanban pour suivre le flux.



3. Différents modèles

D. Méthodes Hybrides

- Dans la pratique, peu de projets numériques suivent un modèle pur :
  - Le cycle en V est trop rigide pour l'innovation.
  - Scrum est trop léger pour des contextes réglementés.
  - Kanban ne couvre pas les phases de conception.

- Résultat : la plupart des équipes mettent en place des **méthodes hybrides**, qui combinent plusieurs approches selon les besoins.

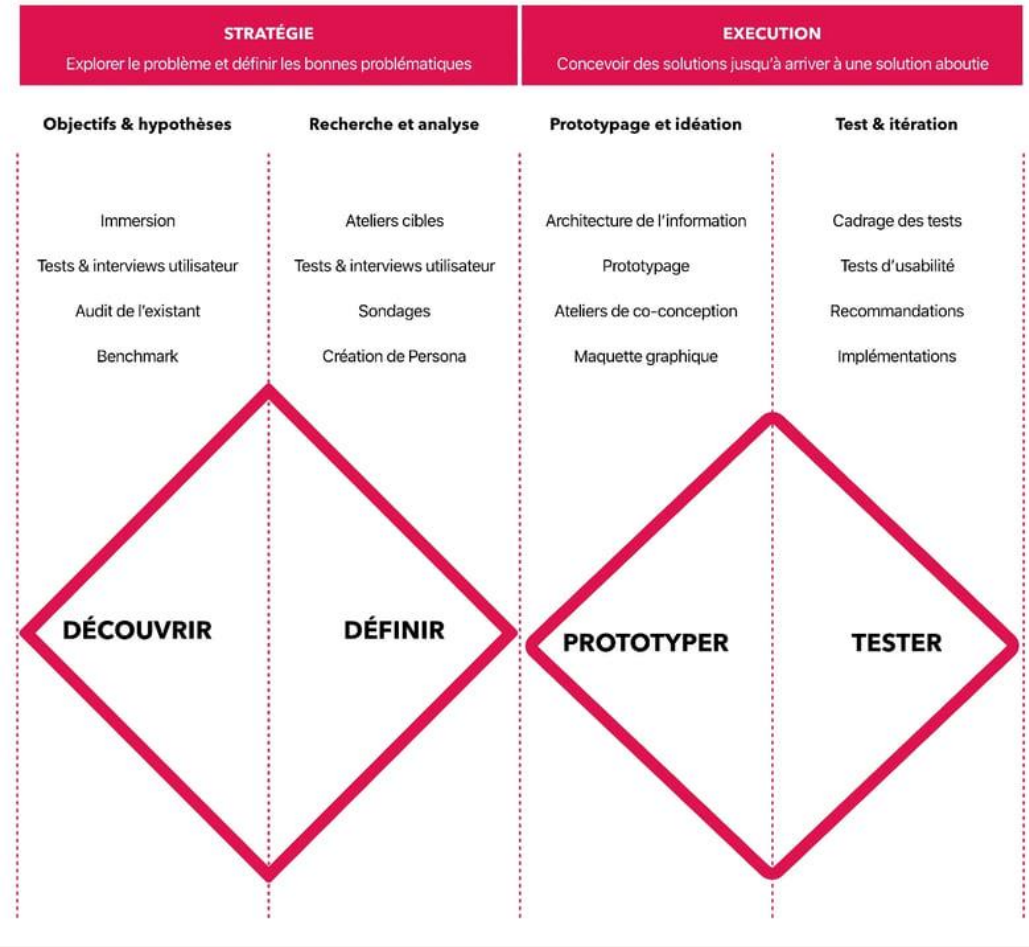
- Principe : choisir le modèle **non pas par préférence**, mais en fonction :
  - du **degré d'incertitude**,
  - du **coût du changement**,
  - du **cadre contractuel** et des **parties prenantes**.

3. Différents modèles

D. Méthodes Hybrides

1. Design Thinking & Double diamant

- Le double diamant n'est pas une méthode de gestion de projet, mais une **démarche de conception**.
  On peut le placer **avant** ou **à l'intérieur** d'un cycle projet, pour s'assurer que la solution répond à un besoin réel.
- Démarche centrée utilisateur.
- Deux grands cycles divergence/convergence :
  1. **Comprendre / Définir** le problème réel.
  2. **Idéation / Prototype & test** de solutions.
- Livrables : personae, journey maps, prototypes testés.

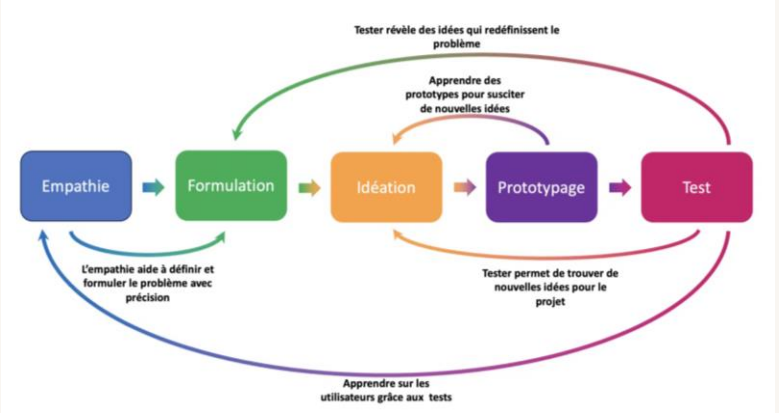


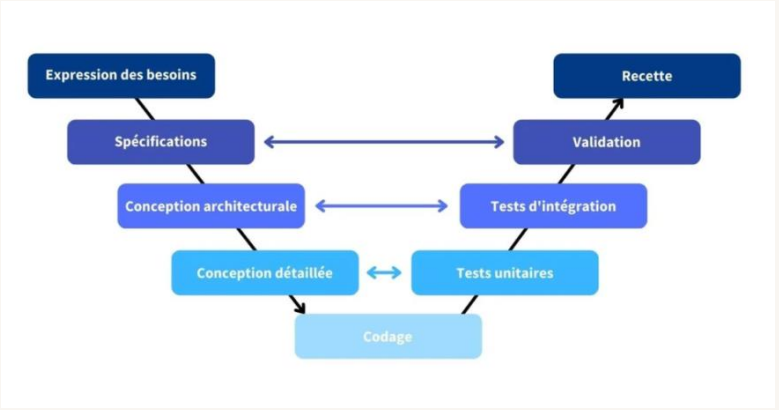
3. Différents modèles

D. Méthodes Hybrides

2. Cycle en V + Design Thinking

- On garde le **cycle en V**, avec son côté prédictif et très structuré, avec livrables validés, on l'enrichit d'une **phase amont de Design Thinking** (double diamant)
- Démarre par la **rédaction d'un cahier des charges** détaillé.
  Cahier des charges mal cadré = projet compromis.
  Ajouter une **phase de Design Thinking**, pour :
  - Observer et interviewer les utilisateurs,
  - Définir clairement la problématique,
  - Imaginer plusieurs pistes,
  - Prototyper rapidement,
  - Tester ces prototypes auprès des utilisateurs.
- On ne lance le cycle en V qu'après cette exploration.
  > cahier des charges solide, s'appuie sur des **besoins réels**.



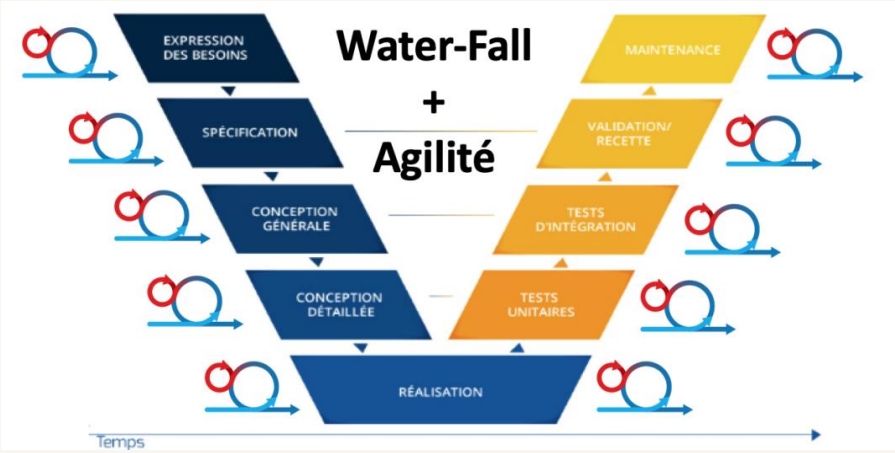


3. Différents modèles

D. Méthodes Hybrides

3. Cycle en V + Scrum > Cycle en V modulaire

- On ne change pas la logique du V, mais on évite l'effet tunnel en le **découpant en plusieurs mini-V**.
- Chaque module suit le même cycle : spécifications → conception → développement → tests → validation. Au lieu d'attendre 18 mois pour livrer tout le projet, on livre des morceaux successifs.
- Avantage :
  - on réduit le risque de blocage,
  - on livre de la valeur plus tôt,
  - on peut apprendre des premiers modules pour améliorer les suivants.



3. Différents modèles

D. Méthodes Hybrides

4. Scrum + Kanban = Scrumban

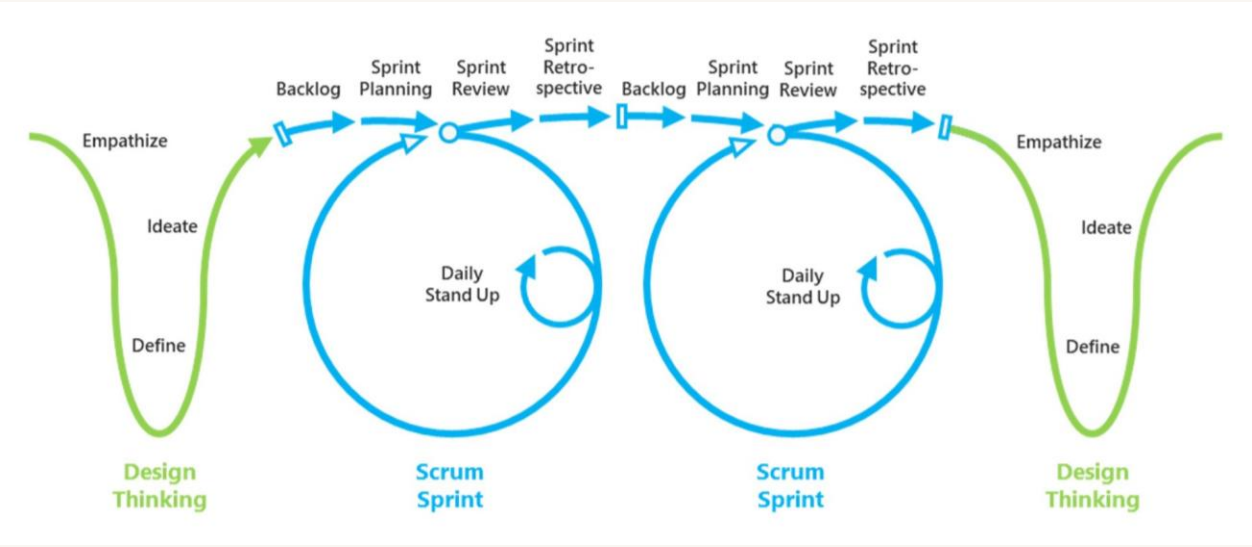
- Les fonctionnalités majeures suivent Scrum (sprints).
- Les petites demandes, bugs, incidents passent en Kanban (flux continu).
- **Exemple :** une app mobile en exploitation : nouvelles fonctionnalités en sprint, maintenance quotidienne gérée en Kanban.

3. Différents modèles

D. Méthodes Hybrides

5. Design thinking + Scrum

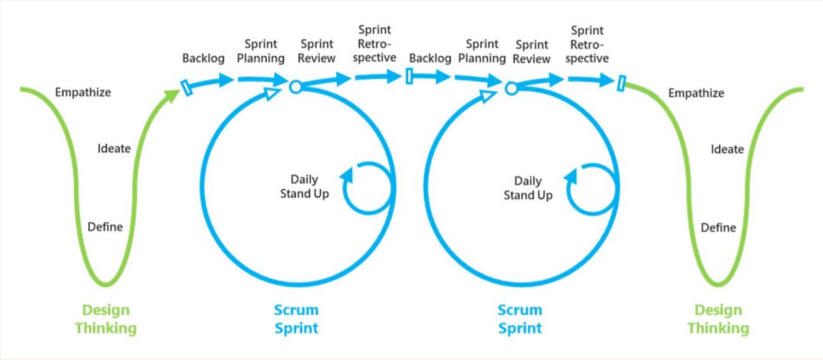
- Ateliers d'idéation et de prototypage
  → Design thinking.
- Implémentation technique
  → Agile (Scrum).
- **Exemple :** création d'un service innovant → exploration créative avant exécution agile.



3. Différents modèles

D. Méthodes Hybrides

5. Design thinking + Scrum



- **Modèle hybride le plus utilisé** combine :
  - Une **phase amont en Design Thinking** (recherche, idéation, prototypage, test)
  - Suivie d'un **développement en Scrum** (sprints, backlog, livraisons itératives)
- Le Design Thinking permet de **partir du besoin utilisateur réel**, sans solution prédéfinie
  > entretiens utilisateurs, cartographie des parcours, prototypage rapide.
- Scrum permet de **livrer rapidement**, en s'adaptant en cours de route
  > développement par sprints avec tests réguliers.

> Centré utilisateur + Flexible + Livrable rapidement
Modèle **adapté à l'innovation encadrée**, avec des enjeux d'usage, de valeur, et de qualité.

3. Différents modèles - Comparatif

Modèle	Avantages	Inconvénients	Contexte adapté
Cycle en V	- Prévisible - Documenté - Séquentiel	- Peu flexible - Peu d'interactions avec les utilisateurs	Projets réglementés, marchés publics
Scrum (Agile)	- Itératif - Feedback rapide - Centré valeur	- Exige autonomie - Forte discipline agile	Nouveaux produits, innovation, MVP
Kanban	- Fluide - Visuel - Gestion des flux	- Pas de cadrage initial - Manque de rythme	Maintenance, support, flux continu
Design Thinking	- Centré utilisateur - Créatif - Exploration ouverte	- Non structurant à long terme	Phase d'exploration, innovation
Scrumban	- Souple - Adapté aux cycles mixtes	- Moins clair qu'un Scrum pur	Produit en production + évolutions
Double diamant + Scrum	- Cadrage solide - Dév. Agile - Centré utilisateur	- Nécessite de bien séparer les temps	Projets numériques avec usage à valider
Cycle en V + Design Thinking	- Précis + basé sur besoin réel	- Lourd à piloter - Long à initier	Projets institutionnels avec innovation

4. Livrables clés d'un projet numérique

- Un livrable, c'est ce qui **prouve qu'on avance**.
  Ce n'est pas juste un fichier, c'est un **élément que l'on partage, valide ou livre**.

- Ce sont eux qui permettent :
  - à l'équipe de **partager une vision commune**,
  - au client **de valider ou de réorienter**,
  - à l'utilisateur **d'expérimenter et de donner du feedback**.

- Un bon livrable est :
  - **lisible**,
  - **partagé** au bon moment,
  - **adapté** à son destinataire (client, dev, DA, utilisateur, etc.)

- Ces livrables sont des **jalons,** ils marquent la fin d'une étape, et permettent de décider si on avance ou si on ajuste.