Ingénierie et Conduite de projets numériques

2. Pilotage opérationnel

Planifier, découper, affecter les ressources, et structurer des dispositifs numériques interactifs

2. Pilotage opérationnel Structuration

1. Structurer un projet dans le temps

- Savoir découper un projet numérique en étapes, phases et jalons structurés pour mieux piloter l'exécution.

- Un projet numérique, ce n'est pas un gros bloc qu'on lance puis qu'on livre. C'est une suite **d'étapes interdépendantes** à organiser finement.
  Mal structurer, c'est le meilleur moyen de dériver.

- **Une bonne structuration permet de :**
  - Visualiser l'enchaînement logique des tâches
  - Définir des priorités et des dépendances
  - Identifier les **jalons critiques**
  - Gérer le **temps, les personnes, le budget**
  - Rassurer les parties prenantes (commanditaire, équipe…)

2. Pilotage opérationnel Structuration temporelle

1. Structurer un projet dans le temps

Un projet, c'est aussi un rythme, un calendrier et des livrables :

- Certaines phases sont **linéaires** (ex : définition, mise en ligne)
- D'autres sont **itératives** (ex : prototypage, test, amélioration)
- Des **jalons-clés** permettent de valider les étapes avant de continuer

Semaine	Phase principale	Livrables associés
1–2	Cadrage	Note d'intention, fiche projet
3	Empathie	Personae, empathie map, benchmark, carte de positionnement
3-4	Définition	Problématique, objectifs UX/stratégiques, critères
4	Idéation	Pistes, concepts, wireframes, mind map, moodboard
5–6	Prototypage & tests	Maquettes UI. Prototype interactif, feedback utilisateur, version testée
7–10	Développement	Contenus validés, intégration / développement
11	Mise en ligne	Site/app fonctionnel, livraison, publication, plan de communication
12	Suivi initial	Recueil de feedbacks, rapport de bugs, statistiques de consultation, plan d'amélioration

Exemple de répartition des phases (projet sur 12 semaines)

2. Les grandes phases d'un projet numérique

- Les projets numériques varient, mais la plupart suivent un cycle assez universel.

Phase	Objectifs principaux	Exemples de livrables
1. Cadrage	Définir le besoin, les cibles, les contraintes	- Note d'intention, - Fiche projet
2. Recherche / Conception	Comprendre les usages, les attentes, les contextes. Imaginer des solutions, prototyper, tester	- Cahier des charges fonctionnel - Dossier de conception (interviews, benchmark, personae, empathy maps, scénarios, user journey map, design system, arborescence, wireframes) - Prototype interactif
3. Développement	Construire techniquement le produit	- Code, documentation, intégration
4. Tests / Recette	Vérifier, corriger, valider	- Rapport de tests, checklists
5. Déploiement / Livraison	Mise en ligne, communication, adoption	- Lancement, formation, suivi
6. Maintenance	Corriger, améliorer, mettre à jour	- Correctifs, évolutions, support

3. Définir les priorités et les dépendances

- Structurer un projet, ce n'est pas juste empiler des tâches dans l'ordre logique. Il faut aussi **savoir ce qui est prioritaire**, et **ce qui dépend d'autre chose** pour pouvoir démarrer.

- **Prioriser les tâches**
  Tout ne peut pas être fait en même temps.
  Il faut identifier ce qui est :
  - **Essentiel** (must-have)
  - **Important** (should-have)
  - **Optionnel** (could-have)
  - **Repoussable** (won't have)

- Cette logique est souvent utilisée dans les **méthodes agiles**, via la matrice **MoSCoW** :



MoSCoW	Signification	Exemple (appli mobile)
Must have	Obligatoire pour livrer le produit	Carte interactive, horaires en temps réel
Should have	Important, mais pas bloquant	Profil utilisateur
Could have	Bonus si temps/dispo	Playlist personnalisée
Won't have	Hors périmètre pour cette version	Partage social intégré

3. Définir les priorités et les dépendances

- **Gérer les dépendances**
  Certaines tâches **ne peuvent pas commencer** tant qu'une autre n'est pas terminée.

- **Types de dépendances :**
  - **Tâche bloquante** : le dev attend que l'UX soit validée
  - **Livrable dépendant** : les tests utilisateurs ne peuvent avoir lieu qu'après le prototype
  - **Décision dépendante** : la com ne peut planifier le lancement sans date de livraison fixée

- **Représentation visuelle :**
  - **Diagramme de Gantt**
  - Outils : GanttProject, Notion, MS Project permettent de **visualiser les dépendances critiques**

- Savoir ce qui est prioritaire et ce qui est dépendant, c'est ce qui permet :
  - de **ne pas tout démarrer en même temps**,
  - de **minimiser les risques de blocage**,
  - et de **rester réaliste** dans son pilotage.

3. Notion de jalons projet

- Un jalon est un **point clé dans le déroulement du projet**, souvent associé à un livrable ou à une validation intermédiaire.

- **Exemples typiques :**
  - Validation du prototype
  - Livraison d'une version beta
  - Mise en production
  - Livraison finale / clôture

- Un bon jalon = une **date + un livrable clair + une validation**

- **À éviter :**
  - Trop de micro-jalons (illisible)
  - Jalons sans contenu associé
  - Dates symboliques sans impact réel

4. Planning du projet

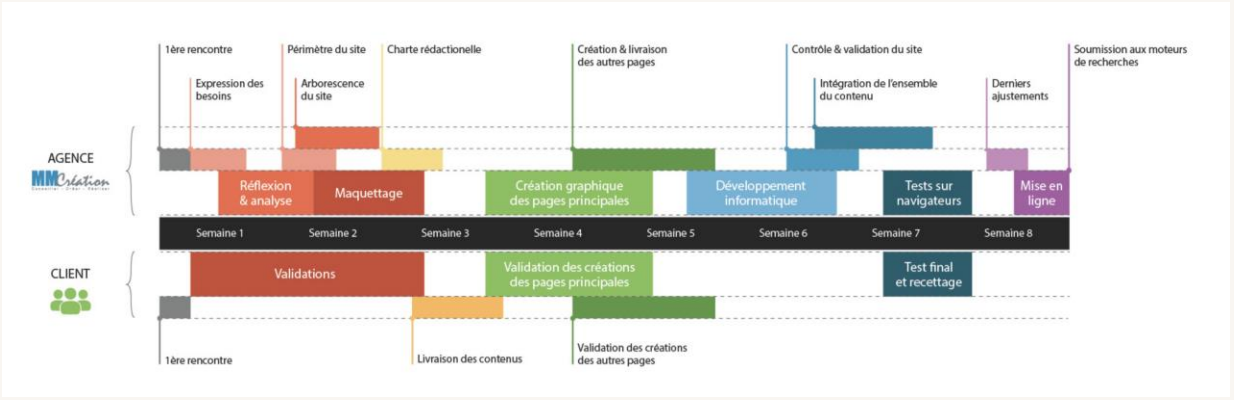
Cette section présente la planification complète du projet, décrivant les étapes majeures, les délais associés, et les livrables pour chaque phase. Elle aide à structurer le déroulement des activités et à anticiper les étapes suivantes.

Phases, Délais et Livrables Clés

. Description : Un tableau récapitulatif des phases principales du projet, avec les dates de début et de fin, et les livrables attendus.

Ce planning est conçu pour donner une vue d'ensemble claire des grandes étapes, de manière à ce que chaque phase puisse être validée avant le passage à la suivante.

. Exemple : **Timeline horizontale** pour montrer les phases en cours et leurs échéances.



timeline des phases et échéances du projet

4. Planning du projet

Prévisions des étapes futures

. Description : Une projection des étapes futures, telles que le prototypage, le webdesign, et le développement. Chacune de ces étapes est estimée en termes de durée, de manière à anticiper les ressources et les approbations nécessaires pour la suite du projet.

. Exemple :
Un diagramme de **Gantt simplifié et développé** (selon la phase du projet) pour illustrer toutes les étapes de l'implémentation future.

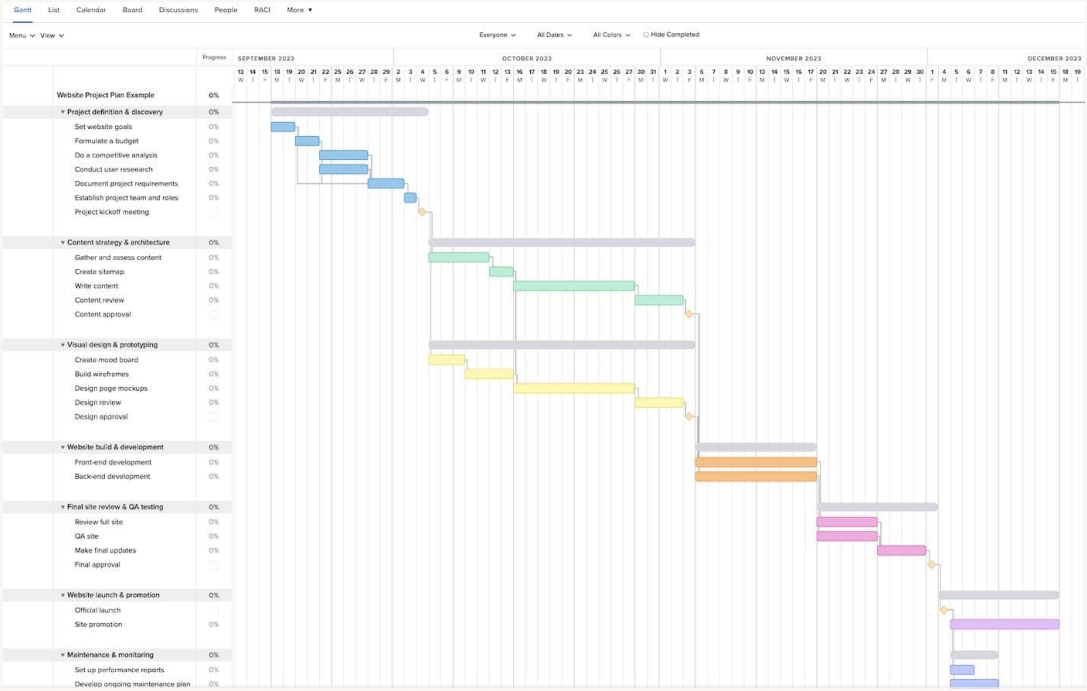


diagramme de Gantt

5. Les outils de pilotage projet (classiques & numériques)

- Un projet numérique génère beaucoup de **tâches, dépendances et validations**.
- Les outils de pilotage permettent de **visualiser, prioriser et suivre** l'avancement.

Rappel clé : l'outil n'est qu'un support → c'est la méthode qui guide son usage.

1. Tableau Kanban (suivi des tâches en flux)

> Idéal pour la gestion agile, maintenance et évolutions.

Principe : tâches représentées sous forme de cartes, qui passent de colonnes "À faire → En cours → Terminé.

- **Trello** (simple, visuel, collaboratif).
- **Jira** (plus complexe, adapté aux grosses équipes tech).

Utile pour : visualiser la charge de travail et éviter les blocages.



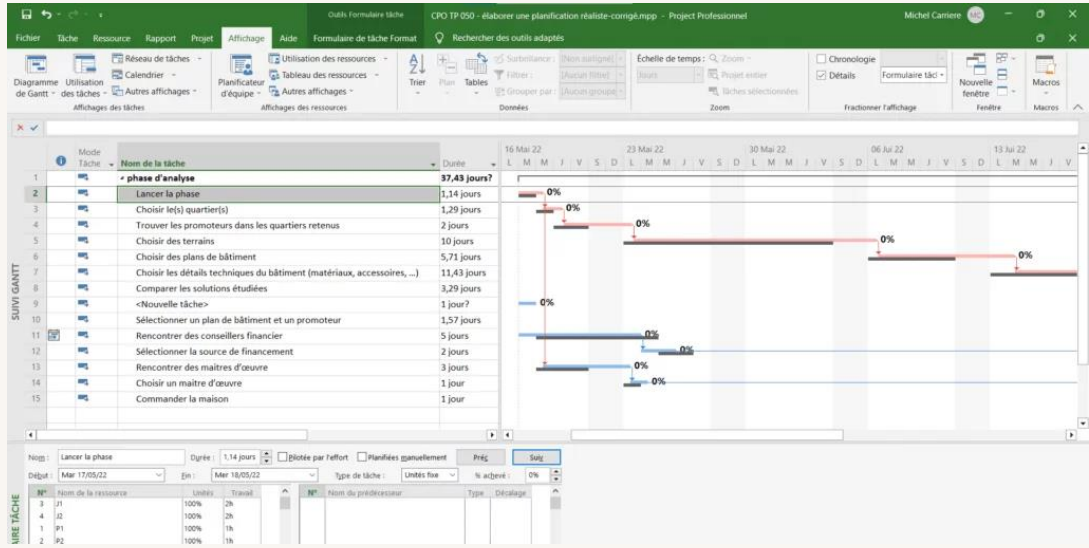
5. Les outils de pilotage projet (classiques & numériques)

2. Diagrammes de Gantt (planification dans le temps)

> Outil classique de gestion de projet prédictive (cycle en V, hybrides).

Principe : chaque tâche est représentée dans une chronologie, avec dépendances.

- **MS Project** (standard historique en gestion projet, très complet, mais lourd et coûteux. Standard en entreprise).
- **Notion** ou **ClickUp** (outils modernes avec timeline intégrée, gestion des tâches et collaboration en temps réel)
- **GanttProject** → simple, gratuit, idéal pour s'initier ou pour des projets académiques/petits projets.



Utile pour : planifier jalons, anticiper retards et dépendances.

4. Les outils de pilotage projet (classiques & numériques)

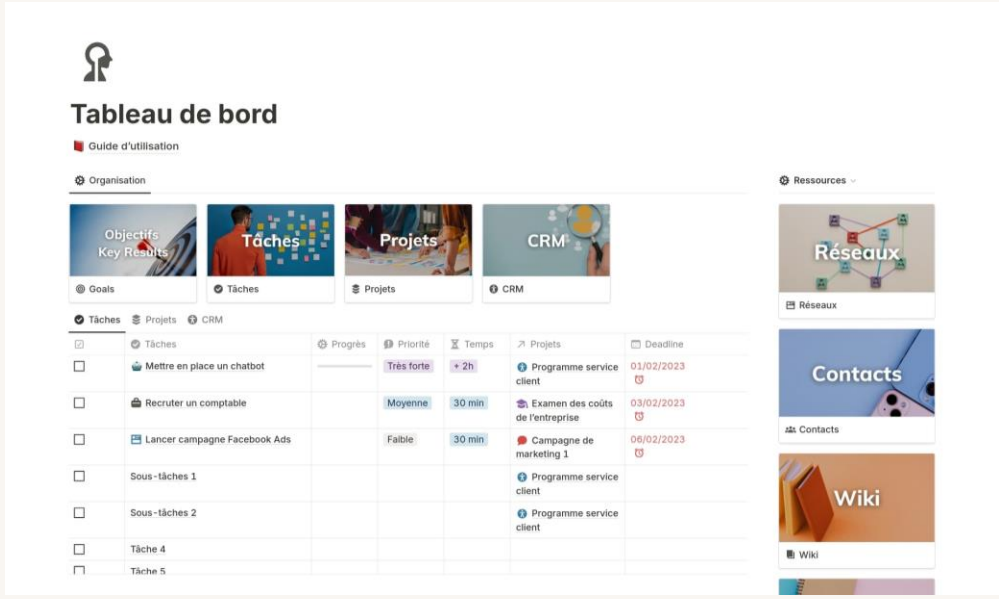
3. Tableaux de bord (suivi global & KPIs)

> Donnent une vision synthétique du projet, utile pour le chef de projet et les parties prenantes.

Principe : regrouper infos clés (avancement, budget, KPIs, risques).

- **Airtable** (base de données visuelle, suivi flexible).
- **Notion** (centralisation tâches, documents, reporting).

Utile pour : piloter la valeur du projet, pas seulement les tâches.



5. Indicateurs de réussite d'un projet numérique (KPIs)

- **Pourquoi mesurer ?**
  - Un projet **réussi** ≠ un projet simplement **livré**.
  - Les indicateurs permettent de savoir si le produit a réellement **atteint ses objectifs** : adoption, usage, valeur, qualité.
  - Ils servent aussi à **ajuster le pilotage** (agile ou non) au fil du temps.

KPIs principaux (Key Performance Indicator)

1. **Taux d'adoption** (utilisateurs actifs / téléchargements).
   - Mesure la mise en usage réelle du produit.
   - % d'utilisateurs actifs vs. téléchargements (apps).
   - % d'inscriptions vs. visiteurs (plateformes).
   - Nombre de comptes créés, onboarding complété.

Exemple : une appli téléchargée 10 000 fois mais avec 500 utilisateurs actifs → adoption faible.

5. Indicateurs de réussite d'un projet numérique (KPIs)

KPIs principaux (Key Performance Indicator)

2. Rétention

- Mesure la fidélité et la réutilisation.
- % d'utilisateurs qui reviennent après 1 jour (D1), 7 jours (D7), 30 jours (D30).
- Courbe de churn (taux d'abandon).

Exemple : un service de streaming vise une rétention mensuelle > 60 %.

3. Satisfaction utilisateur

Évalue la qualité perçue.

- **SUS** (System Usability Scale) → score de 0 à 100 sur la facilité d'usage.
- **NPS** (Net Promoter Score) : % promoteurs – % détracteurs.
- Feedback qualitatif (entretiens, verbatims).

Exemple : NPS = +40 est considéré comme excellent.

5. Indicateurs de réussite d'un projet numérique (KPIs)

KPIs principaux (Key Performance Indicator)

4. Vélocité des sprints (Agile)

- Mesure la capacité de l'équipe à livrer régulièrement.
- Nombre de "story points" ou tâches complétées par sprint.
- Suivi de la tendance dans le temps (stabilité = maturité).

Exemple : vélocité stable = équipe performante ; vélocité instable = problèmes de priorisation.

5. Respect budget & délai

- Dimension classique mais toujours essentielle.
- % d'écart entre budget prévisionnel et réel.
- % d'écart entre roadmap prévue et livraisons effectives.

Exemple : +10 % de dépassement toléré, mais au-delà le projet perd sa viabilité.

> Un projet réussi n'est pas seulement en ligne. Il est adopté, utilisé, apprécié et durable.

6. Gestion des risques d'un projet numérique

- **Tout projet numérique comporte des incertitudes.**
  Les risques doivent être **anticipés et suivis** → sinon ils deviennent des problèmes coûteux.

- La gestion des risques est **complémentaire** au choix de méthode (Cycle en V, Scrum, hybride…).

- **Principaux risques :**

  1. Retards (planning & dépendances)
  - Dépendances mal anticipées (ex. dev bloqué en attente du design).
  - Décisions client tardives.

  > Conséquence : dérive du calendrier, perte de crédibilité.

  2. Dérapage budgétaire
  - Ajout de fonctionnalités non cadrées ("effet tunnel" ou "scope creep").
  - Sous-estimation initiale des ressources.

  > Conséquence : dépassement financier, tensions avec le commanditaire.

6. Gestion des risques d'un projet numérique

- **Principaux risques :**

3. Risques techniques

- Dette technique accumulée (code non maintenable).
- Incompatibilités (navigateurs, appareils, intégration API).

> Conséquence : livrable inutilisable, surcoûts de correction.

4. Adoption faible

- Besoin utilisateur mal compris → produit livré mais pas utilisé.
- Absence de tests ou de phase pilote.

> Conséquence : échec stratégique malgré la livraison.

5. Risques humains

- Rotation d'équipe (turnover).
- Manque de compétences sur une techno clé.
- Mauvaise communication entre métiers (UX / Dev / Marketing).

> Conséquence : ralentissement, perte de cohésion.