

Chapitre 3 – SUIVI, COMMUNICATION, RISQUES ET DETTE TECHNIQUE

- Objectifs

- Mettre en place un suivi simple et utile du projet.
- Adapter la communication aux différents interlocuteurs.
- Identifier, évaluer et traiter les risques et imprévus.
- Comprendre la dette technique et son impact sur le projet.



- En résumé

Piloter un projet informatique consiste à suivre régulièrement son avancement et à comparer la situation réelle avec les objectifs définis au départ. Ce suivi permet de détecter rapidement les écarts de délai, de charge, de budget ou de qualité, de rendre les difficultés visibles et de prendre des décisions adaptées avant qu'elles ne deviennent critiques.



Le pilotage repose donc autant sur la communication, le suivi des indicateurs et la collaboration entre les acteurs que sur la maîtrise des aspects techniques. L'objectif est de maintenir le projet sous contrôle tout en conservant suffisamment de souplesse pour réagir aux imprévus et aux changements.

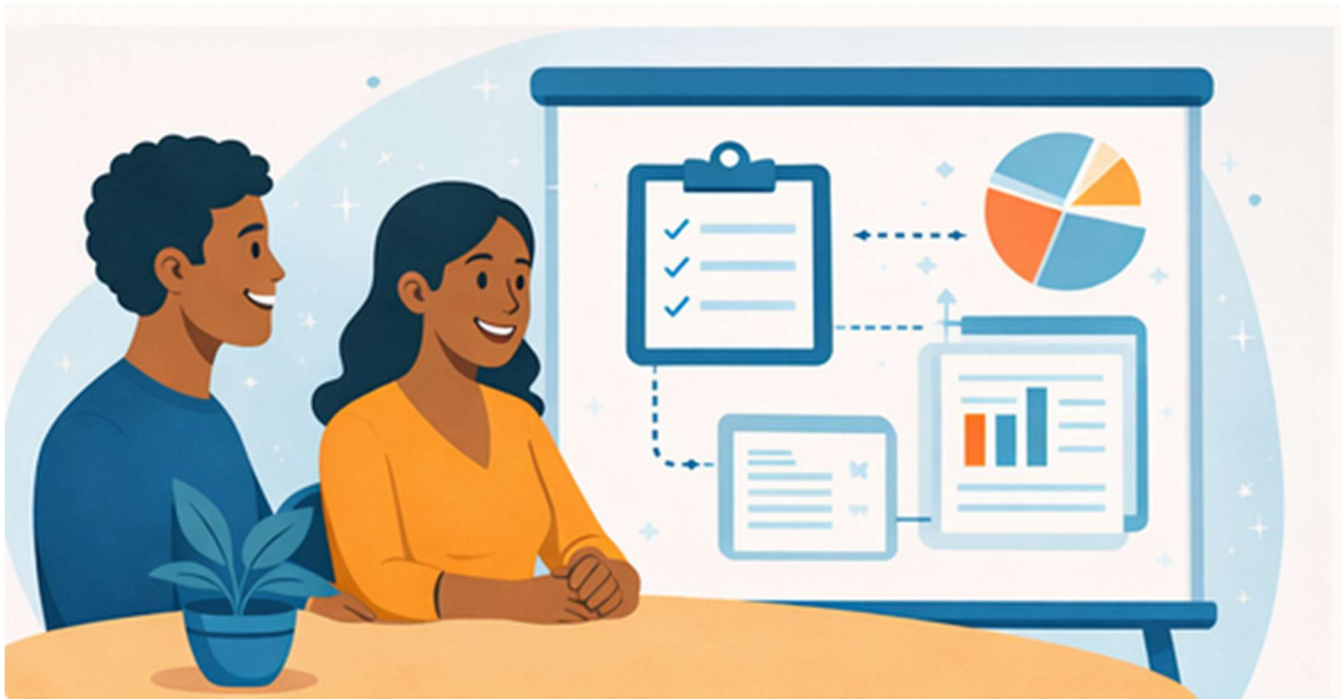
A- Suivi du projet et indicateurs



Le suivi doit permettre de répondre à trois questions : où en sommes-nous, quels obstacles rencontrons-nous et quelles décisions devons-nous prendre ? Les indicateurs doivent donc être peu nombreux, compréhensibles et réellement utilisés.

- Avancement des tâches ou fonctionnalités.
- Respect des jalons et dates clés.
- Charge consommée par rapport à la charge prévue.
- Nombre et criticité des anomalies.
- Vitesse d'équipe dans un contexte Scrum.
- Burndown chart pour visualiser le travail restant dans un sprint.

Précision : Un indicateur n'est utile que s'il conduit à une action. Multiplier les tableaux de bord sans décision associée ajoute du travail sans améliorer le pilotage.



Exemple : dans un projet de développement d'une application mobile, un tableau de bord hebdomadaire peut afficher : nombre de fonctionnalités développées, pourcentage de bugs corrigés, budget consommé, et satisfaction des utilisateurs testeurs.

B- Communication et collaboration

Un projet informatique rassemble des personnes ayant des langages et priorités différents. Le client parle de besoin métier, le développeur de faisabilité technique, la direction de délai et de budget. Le rôle du pilotage est de faire circuler une information adaptée à chacun.

- Réunions courtes et préparées.
- Compte rendu des décisions importantes.
- Canal unique pour les informations de référence.
- Daily ou point d'équipe pour synchroniser le travail.
- Reporting synthétique à la direction ou au sponsor.
- Démonstrations régulières au client ou aux utilisateurs.



C- Risques et imprévus



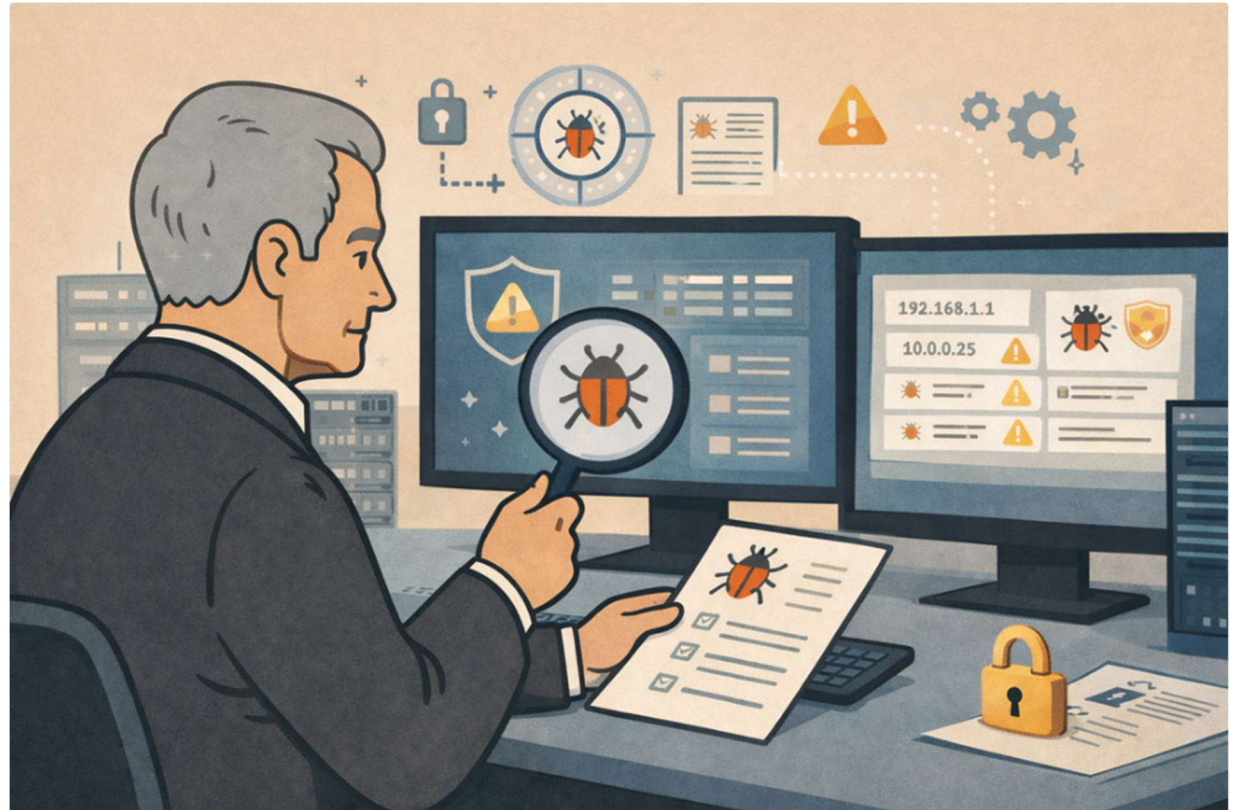
Un risque est un événement incertain qui pourrait affecter le projet. Un problème ou incident est un événement qui s'est déjà produit. L'objectif de la gestion des risques est de réduire les surprises et de préparer les réactions possibles.

- Risques techniques : technologie immature, performance, incompatibilité, sécurité.
- Risques humains : absence, turnover, manque de compétences.
- Risques organisationnels : décisions tardives, dépendance à un autre service.
- Risques externes : fournisseur, réglementation, interruption de service tiers.
- Évaluation : probabilité, impact et criticité.
- Réponses : éviter, réduire, transférer ou accepter.

D- Dette technique

La dette technique correspond au coût futur créé par des choix techniques rapides, incomplets ou difficiles à maintenir. Elle peut être volontaire, par exemple pour tenir une échéance, ou involontaire en raison d'un manque de connaissance, de tests ou de documentation.

- Code dupliqué ou difficile à comprendre.
- Architecture provisoire devenue permanente.
- Absence ou insuffisance de tests automatisés.
- Dépendances obsolètes.
- Documentation absente.
- Report permanent des travaux de refactoring.





Précision : La dette technique n'est pas nécessairement mauvaise si elle est connue, mesurée et remboursée.

Elle devient dangereuse lorsqu'elle est invisible ou systématiquement ignorée.

Exemple : Pour livrer une démonstration urgente, l'équipe crée une intégration rapide avec un service externe sans gérer correctement les erreurs.

Après la démonstration, une tâche de refactoring est ajoutée au backlog afin de fiabiliser cette intégration.