GUIDE PRATIQUE

Cadrer un projet logiciel sur mesure en 5 étapes

Comment passer d'une idée floue à une feuille de route claire, sans devis prématurés ni mauvaises surprises.

Pour qui ?	Dirigeants, responsables IT, chefs de produit et fondateurs qui envisagent un développement logiciel sur mesure (application métier, API, automatisation, intégration).
Pour quoi ?	Éviter les pièges classiques du cadrage projet : sous-estimation du périmètre, dette technique précoce, dépendance excessive à un prestataire, livrables qui ne correspondent pas au besoin réel.
Temps de lecture	Environ 12 minutes.

Pourquoi le cadrage est l'étape la plus rentable d'un projet

La majorité des projets logiciels qui dérapent en coût, en délai ou en qualité ne souffrent pas d'un problème technique. Ils souffrent d'un problème de cadrage. Une portée mal définie, des priorités implicites, des hypothèses jamais explicitées, et le projet glisse, d'abord doucement, puis brutalement.

La bonne nouvelle : le cadrage est aussi l'étape la moins chère à bien faire. Quelques heures de clarification structurée évitent souvent des semaines de retravail. Ce guide vous donne une méthode en cinq étapes pour passer d'une intention à une feuille de route exploitable, que vous travailliez avec un prestataire externe ou en interne.

Vous pouvez l'utiliser tel quel comme grille de cadrage, ou vous en servir comme base de discussion lors d'un premier échange avec une équipe de développement.

ÉTAPE 01

Clarifier le problème, pas la solution

Le premier réflexe en cadrage est presque toujours mauvais : on décrit la solution imaginée (« il nous faut un outil qui fait X, Y, Z »). Or à ce stade, la solution est encore une hypothèse. Le vrai cadrage commence par expliciter **le problème métier** que cette solution doit résoudre.

Posez-vous (et faites poser autour de vous) trois questions simples : qui souffre aujourd'hui d'une situation, en quoi cette souffrance se traduit concrètement, et qu'est-ce qui empêche aujourd'hui d'y répondre avec les outils existants. Une fois le problème formulé en une à deux phrases sans jargon technique, vous pouvez à nouveau parler de solutions.

Ce travail évite l'écueil le plus coûteux : développer un logiciel qui résout le mauvais problème.

À vérifier avant de passer à la suite :

- Le problème peut s'énoncer en deux phrases compréhensibles par un non-technicien.
- Les personnes directement concernées (utilisateurs, opérationnels) ont été consultées.
- Les conséquences chiffrées du problème actuel sont estimées (temps perdu, erreurs, coûts).
- Au moins une alternative au développement sur mesure a été envisagée et écartée pour de bonnes raisons.

ÉTAPE 02

Distinguer l'essentiel de l'accessoire

Tout projet contient trois catégories de fonctionnalités : celles qui résolvent vraiment le problème, celles qui apportent du confort, et celles qu'on ajoute « parce que tant qu'on y est ». La troisième catégorie est responsable d'environ la moitié des dépassements budgétaires.

La méthode **MoSCoW** est un classique éprouvé : classez chaque fonctionnalité en *Must have* (le projet n'a pas de sens sans), *Should have* (forte valeur mais contournable), *Could have* (sympa mais reportable) et *Won't have* (explicitement hors périmètre). L'exercice paraît trivial, il ne l'est pas. Forcer un choix entre

Must et *Should* fait remonter les vrais désaccords entre parties prenantes, exactement au moment où ils coûtent le moins cher.

Une règle de prudence : si plus de 30 % des fonctionnalités sont classées *Must have*, c'est que le tri n'a pas été fait honnêtement. Recommencez.

À vérifier avant de passer à la suite :

- Chaque fonctionnalité est classée Must / Should / Could / Won't.
- Le périmètre « Must have » seul forme un produit utilisable (pas juste une démo).
- Les Won't have sont explicitement listés (pour éviter les attentes implicites).
- Les arbitrages ont été validés par les décideurs, pas seulement par l'équipe technique.

ÉTAPE 03

Cartographier les contraintes et les dépendances

Un logiciel sur mesure ne vit jamais isolé. Il doit s'intégrer dans un écosystème : outils existants, bases de données, contraintes légales, processus métier, équipes en place. Les **dépendances externes** sont la principale source d'imprévus dans les projets, loin devant la difficulté technique pure.

Listez systématiquement : les systèmes avec lesquels le logiciel devra dialoguer (ERP, CRM, outils comptables, APIs tierces), les contraintes réglementaires applicables (RGPD, secteur régulé, archivage), les personnes dont l'accord ou la collaboration sera nécessaire en cours de projet, et les délais imposés de l'extérieur (clôture comptable, salon professionnel, fin de licence).

Pour chaque dépendance identifiée, posez une question simple : que se passe-t-il si elle n'est pas prête à temps ? Si la réponse est « le projet s'arrête », c'est un risque majeur à traiter dès le cadrage, pas pendant la réalisation.

À vérifier avant de passer à la suite :

- Tous les systèmes à intégrer sont listés avec leurs propriétaires côté client.
- Les contraintes légales et réglementaires applicables sont identifiées.
- Les disponibilités des personnes-clés sont confirmées sur la durée du projet.
- Les jalons externes imposés (dates butoir non négociables) sont connus.

ÉTAPE 04

Définir ce qu'est « réussir »

Un projet sans critère de réussite explicite ne peut littéralement pas réussir : personne ne saura quand le projet est abouti, quand célébrer, quand corriger. C'est pourtant l'étape la plus souvent sautée du cadrage, parce qu'elle force à se mouiller.

Les bons critères de succès sont **mesurables** et **datés**. Pas « améliorer la productivité » mais « réduire de 50 % le temps de traitement d'une commande, mesuré sur les commandes de mars ». Pas « rendre les utilisateurs satisfaits » mais « atteindre un taux d'adoption de 80 % des utilisateurs cibles dans les trois mois suivant le déploiement ».

Distinguez trois horizons : les critères de livraison (le logiciel fait ce qu'il doit faire), les critères d'adoption (les utilisateurs s'en servent réellement) et les critères d'impact (le problème métier initial est résolu). Les trois doivent être suivis, pas seulement le premier.

À vérifier avant de passer à la suite :

- Au moins un critère chiffré par horizon (livraison / adoption / impact) est défini.
- Les modalités de mesure sont précisées (qui mesure, quand, sur quel périmètre).
- Les seuils ont été validés par les décideurs avant le démarrage.
- Un mécanisme de revue est prévu (par exemple à 1, 3 et 6 mois après mise en production).

ÉTAPE 05

Construire un plan par paliers, pas un grand bond

Le mythe du projet « en une livraison » est responsable d'à peu près toutes les catastrophes logicielles connues. La règle d'or : préférer une succession de paliers livrables et utilisables à un grand chantier monolithique de plusieurs mois.

Pour chaque palier, deux questions : qu'est-ce que les utilisateurs peuvent faire à la fin qu'ils ne pouvaient pas faire au début, et quelle décision pourrons-nous prendre une fois ce palier livré ? Si la réponse à la première est « rien d'utile en pratique », c'est un sous-palier, pas un palier. Si la réponse à la seconde est « aucune nouvelle décision », le découpage est trop fin.

Cette approche par paliers a un effet secondaire précieux : elle permet de **réorienter ou d'ajuster** le projet à chaque étape, plutôt qu'au bout de six mois quand il est trop tard. C'est probablement la meilleure protection budgétaire qui existe.

À vérifier avant de passer à la suite :

- Le projet est découpé en paliers de 2 à 6 semaines maximum.
- Chaque palier livre une valeur utilisable, pas juste un avancement technique.
- Un point de décision « continuer / réorienter / ajuster » est prévu entre chaque palier.
- Le premier palier livre le « Must have » minimum, pas une infrastructure invisible.

Et après ?

Une fois ces cinq étapes franchies, vous disposez d'un cadrage qui tient debout : un problème clair, un périmètre arbitré, des dépendances identifiées, des critères de réussite mesurables et un plan par paliers. C'est généralement assez pour engager une discussion concrète avec une équipe de développement, et pour obtenir des estimations qui ne soient pas des chiffres en l'air.

Si à un moment de votre cadrage vous bloquez, particulièrement sur les étapes 3 (dépendances) ou 5 (découpage), c'est exactement le type de discussion qu'un premier échange avec Synappli est conçu pour débloquer. Ce pré-cadrage est gratuit : pas de devis prématuré, pas d'engagement, juste un regard extérieur sur votre cadrage avant que les décisions coûteuses ne soient prises.

Discuter de votre cadrage avec Synappli

E-mail : aurelien.lacaud@synappli.com

Web : synappli.com/contact

LinkedIn : linkedin.com/company/synappli

> Le pré-cadrage est gratuit (60 à 90 minutes cumulées). Si votre sujet est transverse, sensible ou encore flou, un cadrage structuré va plus loin (à partir de 900 € HT). Les niveaux et leurs prix : synappli.com/offres#cadrage