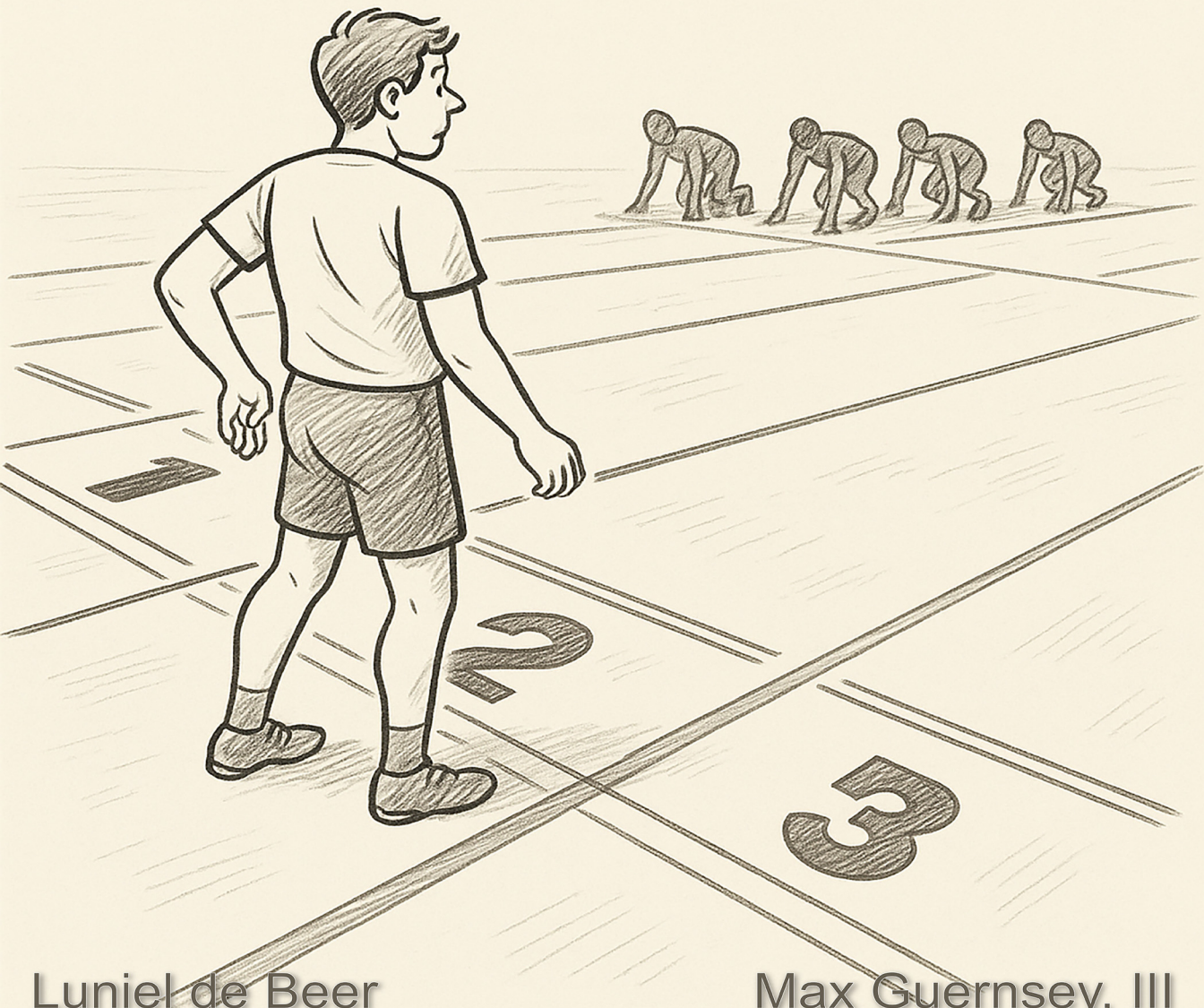


READY

WHY MOST SOFTWARE PROJECTS
FAIL AND HOW TO FIX IT



Luniel de Beer

Max Guernsey, III

Édition Française

Tweetez ce livre!

Aidez Luniel de Beer et Max Guernsey, III en parlant de ce livre sur [Twitter](#)!

Le tweet suggéré pour ce livre est :

Je viens d'acheter Ready — un livre pour les dirigeants et les équipes qui souhaitent éliminer le retravail, les reports et le désalignement dans la livraison de logiciels.

[#CodeReady](#)

Le hashtag suggéré pour ce livre est [#CodeReady](#).

Découvrez ce que les autres disent du livre en cliquant sur ce lien pour rechercher ce hashtag sur Twitter :

[#CodeReady](#)

Ready (Édition Française)

Pourquoi la plupart des projets logiciels échouent et comment y remédier

Luniel de Beer et Max Guernsey, III

Ce livre est disponible sur <https://leanpub.com/ready-fr>

Cette version a été publiée le 2025-10-21



Ceci est un livre [Leanpub](#). Leanpub permet aux auteurs et éditeurs d'utiliser le processus de Lean Publishing. Le [Lean Publishing](#) consiste à publier un livre numérique en cours d'élaboration en utilisant des outils légers et de nombreuses itérations pour obtenir les retours des lecteurs, pivoter jusqu'à obtenir le bon livre et développer sa diffusion une fois celui-ci achevé.

© 2025 Luniel de Beer et Max Guernsey, III

À la mémoire de Johann van Aardt, qui a reconnu ma passion, m'a initié à la vraie programmation et m'a aidé à trouver mon nouveau chez-moi. Et à mes parents, dont le soutien indéfectible - de mon premier client à ma première maison aux États-Unis - a rendu tout cela possible.

—Luniel

Pour ma famille, avec qui le soleil se lève et se couche.

—Max

Table des matières

À propos de ce livre	i
À qui s’adresse ce livre	ii
Comment utiliser ce livre	iii
À propos des auteurs	iv
Préface	v
 Partie I : Quelque Chose Manque	 1
Chapitre 1 : Le Problème Caché	2
Chapitre 2 : Le coût des fondations manquantes	15
Chapitre 3 : Introduction au Requirements Maturation Flow (RMF)	27
Chapitre 4 : Est-ce Agile?	29
 Partie II : Créer de l’espace pour la préparation	 30
Chapitre 5 : La première extension	31
Chapitre 6 : Pourquoi Les Gens Ne Font-Ils Pas Cela?	32
Chapitre 7 : Travail de préparation explicite (RMF 1)	34
Chapitre 8 : Effets du RMF 1	36
Chapitre 9 : Mettre en pratique RMF 1	37

Partie III : Contrôle des Travaux Terminés	39
Chapitre 10 : Le Besoin Suivant	40
Chapitre 11 : Ce que les gens font habituellement	42
Chapitre 12 : Définir une Définition du Fini	44
Chapitre 13 : Définition du Fini sur Mesure (RMF 2)	46
Chapitre 14 : La vie avec RMF 1 & 2	50
Chapitre 15 : Installation de RMF 2	52
 Partie IV : Mise en œuvre du contrôle d'accès .	 54
Chapitre 16 : L'Exigence Finale	55
Chapitre 17 : Contexte de la Définition du Prêt	57
Chapitre 18 : Définir une Définition du Prêt	59
Chapitre 19 : Définition du Prêt sur Mesure (RMF 3)	61
 Partie V : Synthèse	 65
Chapitre 20 : La <u>plupart</u> des échéances n'ont pas d'importance	66
Chapitre 21 : Compétence 1 : Flux de Maturation des Exigences	69
Chapitre 22 : Comment le Travail et l'Information Circulent dans Scrum avec RMF	71
Chapitre 23 : L'Impact du FME	73
Chapitre 24 : Transition vers le FME	74
Chapitre 25 : C'est à Vous	76
 Partie VI : Ressources	 77
Annexe A : Scrum n'est pas le problème	78
Annexe B : Le Framework Synapse™	80

Annexe C : Objections et obstacles courants à RMF 1	82
Annexe D : Listes de Critères de Départ pour la DdF	83
Annexe E : Listes de Critères de Départ pour la Définition du Prêt	84
Index	85

À propos de ce livre

Ready est un livre destiné à toute personne impliquée dans le développement logiciel qui en a assez de la **sous-performance**, du **retravail chronique** et des **exigences confuses**.

Vous avez peut-être déjà essayé d'investir dans les compétences d'exécution de l'équipe, d'améliorer la mise en œuvre de votre cadre de processus, ou de remanier le code, et vous avez encore besoin d'améliorations.

Cela s'explique par le fait que la principale contrainte pour la plupart des équipes de développement logiciel n'est pas les compétences de l'équipe, mais la maturité des exigences. **Même les équipes expérimentées avec les bonnes compétences continuent de lutter** lorsqu'elles travaillent avec des exigences immatures.

Ready présente le FME (Flux de Maturation des Exigences), une approche pratique et profondément structurée pour aligner le Produit et l'Ingénierie sans remplacer votre processus existant.

Que vous utilisiez Scrum, Kanban ou quelque chose de personnalisé, le FME vous aide à **stabiliser le périmètre, éliminer les reports et livrer ce qui compte vraiment**.

Si vos équipes se sentent bloquées au bord du « presque terminé », ce livre vous montrera comment **briser le cycle** et débloquer votre ou vos équipe(s) **définitivement**.

À qui s'adresse ce livre

Ce livre s'adresse littéralement à toute personne impliquée dans le développement logiciel. Des ingénieurs aux chefs de produit, des contributeurs individuels aux dirigeants.

Ce livre est fait pour vous si vous êtes impliqué dans le développement logiciel et avez remarqué qu'une équipe avec laquelle vous travaillez ou dont vous faites partie présente un ou plusieurs des problèmes suivants :

- Le travail est fréquemment reporté d'une itération à la suivante
- Les équipes de mise en œuvre ont l'impression de viser des cibles mouvantes
- Le travail reste ouvert trop longtemps
- Le travail est marqué comme terminé mais les choses ne sont pas vraiment finies
- Le travail effectué ne correspond pas aux attentes
- Le travail génère régulièrement un grand nombre de défauts
- Une grande quantité de travail doit être refaite régulièrement

Si l'un de ces problèmes vous parle, *Ready* peut vous aider.

Comment utiliser ce livre

Ce livre a été conçu pour être pratique. Ce n'est pas un traité théorique ou une présentation stratégique - c'est un manuel pratique pour installer le FME (le Flux de Maturation des Exigences) basé sur un véritable travail client et testé sur le terrain sous une réelle pression de livraison.

Les chapitres sont écrits en séquence, mais le FME lui-même est modulaire. Il est composé de trois pratiques fondamentales :

- FME 1 : Collaborer pour une compréhension partagée
- FME 2 : Contrôler l'achèvement du travail à l'aide de Définitions Sur Mesure du Fini
- FME 3 : Contrôler la mise en œuvre à l'aide de Définitions Sur Mesure du Prêt

Chaque partie, ou «Habitude» comme nous les appelons, est autonome, mais elles se construisent les unes sur les autres. Le livre est conçu pour vous aider à les aborder une à la fois, dans l'ordre. Cette structure reflète la façon dont nous recommandons aux équipes d'adopter le FME en pratique - chaque Habitude n'étant intégrée qu'après que la précédente fonctionne.

Cela évite de submerger les équipes et donne à chaque changement la meilleure chance de s'ancrer. Vous en apprendrez davantage sur la façon de procéder à partir de .

Si vous cherchez de l'aide - que ce soit des conseils, du coaching ou quelqu'un pour parler avec votre équipe de direction - n'hésitez pas à nous contacter directement.

Et si vous recherchez un support formel pour mettre en œuvre le FME, Producore propose une série complète de Programmes conçus pour guider l'adoption étape par étape. Vous pouvez en savoir plus sur <https://ready-book.link/rmf>.

À propos des auteurs

Luniel de Beer est le créateur du Flux de Maturation des Exigences (FME), un système pratique pour combler les écarts entre l'intention du produit et l'exécution technique. Il possède plus de 15 ans d'expérience dans la direction de transformations Agiles, la création de ponts entre le produit et l'ingénierie, et l'aide aux équipes pour livrer avec clarté et confiance.

Luniel est également à l'origine du système de Gestion des Capacités de Producore, une approche traçable et évolutive pour modéliser les capacités des produits. Il a conçu le Développement Piloté par PKB (PKBDD), un système versionné pour gérer les exigences produit persistantes. Ces outils font partie d'un cadre de livraison plus large développé chez Producore.

Max Guernsey, III est architecte logiciel, éducateur et co-fondateur de Producore, un cabinet de conseil dédié à la résolution des échecs de livraison grâce à la rigueur structurelle et technique. Avec plus de deux décennies d'expérience en conception orientée objet, refactoring, développement piloté par les tests et patrons de conception, il a à la fois livré des systèmes critiques et coaché des équipes d'ingénierie à grande échelle. Son travail allie des pratiques techniques approfondies avec une transformation comportementale et processuelle pour aider les organisations à atteindre l'excellence durable en matière de livraison.

Max a contribué de manière significative à PKBDD et a dirigé le développement de l'approche de Producore en matière de développement piloté par le comportement (BDD) grâce à son expertise approfondie en spécification comportementale.

Ensemble, leur travail intègre la clarté, la traçabilité et la validation par étapes dans un système cohérent de livraison logicielle qui s'étend de la pratique d'équipe à la capacité organisationnelle.

Préface

Note aux dirigeants techniques

Si vous êtes un dirigeant expérimenté dans une organisation d'ingénierie, vous ne manquez probablement pas d'efforts, de discipline ou de personnes intelligentes. Et pourtant, d'une manière ou d'une autre, les projets stagnent encore. Les objectifs glissent. Les attentes ne sont pas satisfaites. Non pas parce que vos équipes sont paresseuses, mais parce que quelque chose de fondamental est défaillant dans la façon dont le travail est défini, structuré et livré.

Ce livre n'est pas un guide de leadership. C'est un outil de diagnostic structurel. Il révèle ce qui se passe réellement au sein de vos équipes - pourquoi « presque terminé » se transforme continuellement en « pas terminé », et pourquoi les progrès locaux se traduisent si rarement en résultats stratégiques.

Vous ne vous reconnaîtrez peut-être pas dans ces pages. Mais si vos équipes ne peuvent pas livrer ce dont vous avez besoin, vous les reconnaîtrez. Et quand ce sera le cas, vous aurez enfin le langage - et le système - pour y remédier.

De Luniel

Tout d'abord, ce livre n'aurait pas été possible sans Max, dont la capacité à voir au-delà du brouillard et de la confusion pour distiller une idée jusqu'à son essence me dépasse totalement.

Comment en sommes-nous arrivés là ?

En y réfléchissant, je pense que c'est parce que j'ai toujours voulu comprendre comment les choses fonctionnent réellement. Que ce soit dans la religion, la nutrition ou le développement logiciel, je me heurtais toujours au même problème : des réponses superficielles qui ne tenaient pas sous la pression. Alors j'ai continué à creuser - en me demandant non seulement ce que nous faisons, mais pourquoi, et ce qui manque quand ça ne fonctionne pas.

L'une des premières failles du système est apparue dans un rôle où je portais trois casquettes : Scrum Master, Product Owner et Development Manager (!!) pour une équipe fournissant

des services de données dans une entreprise technologique renommée. Nous faisons ce que Scrum prescrivait - des Sprints courts, des stories dans un backlog, une planification en une demi-journée - mais chaque fois que nous commençons un nouveau Sprint, nous rencontrons des frictions. L'équipe ne comprenait pas pleinement le problème, nous devons revoir et réviser les exigences en cours de Sprint, des dépendances évitables surgissaient et nous retardaient, et des étapes clés étaient oubliées.

J'ai donc commencé à faire quelque chose de différent. Je réunissais l'équipe et les parties prenantes dans une salle pour chaque story, nous examinions le problème en détail, réfléchissions ensemble à la solution, et seulement ensuite j'écrivais la story. Le Sprint Planning s'est réduit à une heure, et notre succès en matière de livraison a explosé.

Petit à petit, j'ai commencé à réaliser que le succès ne vient pas du fait de travailler plus dur pendant le Sprint. Il vient de la structure que vous mettez en place *avant* qu'il ne commence.

Plus tard, après avoir entendu Jeff Sutherland parler des «Definitions of Ready», j'ai su qu'il y avait là quelque chose de précieux - mais ce n'était pas suffisant. Mon expérience avec les exigences, l'UX, l'UI, la recherche, et plus tard avec le BDD m'a montré que différents éléments de travail exigent différents types de préparation. Certains nécessitent des spécifications comportementales. D'autres ont besoin d'accès au système. D'autres encore nécessitent une traçabilité complète des capacités.

Et tous ont besoin d'une compréhension partagée qui soit réellement confirmée - pas supposée.

En travaillant avec plus d'équipes, j'ai vu le même schéma partout : des étapes manquantes, des dépendances non satisfaites, des équipes faisant de leur mieux mais constamment en train de se démener pour résoudre des problèmes qui auraient dû être évités. Même les meilleures équipes luttent - non pas parce qu'elles étaient faibles, mais parce qu'il leur manquait une structure rendant la préparation explicite.

Le résultat de tout cet apprentissage, de cette itération et de cette frustration est un système structuré pour gérer la préparation.

C'est de cela que traite ce livre.

J'espère qu'il vous donnera de la clarté sur l'origine des vrais problèmes et sur la façon de les résoudre. J'espère qu'il vous donnera le langage pour défendre des pratiques qui peuvent sembler «superflues» mais qui sont en réalité essentielles. Et surtout, j'espère qu'il aidera les équipes à livrer avec moins de stress, moins de surprises et de bien meilleurs résultats.

Si nous réussissons, nous économiserons des milliards de dollars à l'industrie.

Mais plus important encore, nous rendrons aux gens leur santé mentale.

De Max

Je travaille sur ce problème sous différents angles depuis des décennies, mais mes progrès étaient limités jusqu'à ce que je rencontre Luniel.

C'est parce qu'avant de le connaître, j'abordais le problème, fondamentalement, comme un problème technique. Je me concentrais sur l'aide aux équipes pour adopter des pratiques comme le Test-Driven Development (TDD), le refactoring, la conception logicielle avancée et, plus tard, l'Acceptance-Test-Driven Development (ATDD) ou le Behavior-Driven Development (BDD).

Dans la plupart de ces cas, le problème abordé dans ce livre était traité comme un détail d'implémentation pour établir ces pratiques techniques.

Ce n'est pas dire que je ne valorise plus les pratiques techniques. Je pense toujours qu'elles sont profondément importantes, mais elles n'abordent pas directement le problème de la préparation dans le développement logiciel. Au lieu de cela, elles *révèlent* ce problème et les gens appliquent ensuite un correctif à leur processus pour le traiter « juste assez » pour soutenir les pratiques techniques qu'ils essaient de mettre en œuvre.

Je veux aussi aborder la question de savoir qui ce livre peut aider. La réponse courte est « probablement presque tout le monde dans le développement logiciel », mais la vraie réponse contient des nuances qui aident à l'adapter à divers environnements sans en changer le sens fondamental.

Il existe des équipes qui ont besoin de la solution proposée dans ce livre. Vous en rencontrerez une version édulcorée au Chapitre 1.

Il existe également des équipes qui n'ont pas strictement **besoin** d'un système comme celui que nous proposons, mais qui pourraient tout de même en tirer profit.

La meilleure équipe avec laquelle j'ai jamais travaillé - facilement une unité d'écart-type complète au-dessus de la **meilleure équipe suivante**, voire deux - était nichée dans l'arrière-pays de Central Oregon. Ils étaient tellement performants qu'ils pouvaient surmonter l'absence d'un tel système par leur pure compétence. Pourtant, mon responsable de l'époque, Tom Barreras, m'a un jour dit quelque chose comme « J'ai remarqué que nos récits utilisateur se passent mieux quand nous prenons le temps de parler des tests en amont. »

À l'époque, je voyais encore cela à travers le prisme du développement des tests et de l'exécution technique, mais je sais maintenant que c'était un autre indicateur que *l'état de préparation* était un facteur affectant l'équipe... cette équipe particulière était simplement si

capable et si prompt à réagir qu'elle pouvait réussir en répondant aux obstacles au moment où ils survenaient plutôt qu'en les prévenant dès le départ.

Même si vous êtes le genre de personne qui n'a pas strictement *besoin* de se soucier de l'état de préparation parce que vous pouvez le surmonter, ou si vous travaillez avec une équipe du même calibre, vous pouvez tout de même bénéficier du contenu de ce livre.

Partie I : Quelque Chose Manque

*Quand faire les mêmes choses mieux n'aide pas, cherchez ce qui **n'est pas fait**.*

Chapitre 1 : Le Problème Caché

Voici une histoire vraie¹ à propos d'une banque. Appelons-la simplement « La Banque ». C'est un type d'Institution de crédit fédérée qui fait partie de l'infrastructure financière nationale des États-Unis.

Nous (Luniel et Max) avons été appelés à La Banque car elle peinait à livrer un projet logiciel. C'était l'un des environnements les plus dysfonctionnels que nous ayons jamais vus, et c'est pourquoi nous avons choisi cette étude de cas pour commencer : si un changement significatif était possible à La Banque, il est possible partout.

Une Note Rapide sur les Projets

Lorsque nous utilisons le terme « projet » dans ce livre, nous l'entendons dans le contexte de la gestion de projet. Bien qu'il existe différentes interprétations de ce terme, nous utilisons la définition du [Project Management Institute](#) :

« Un projet est une entreprise **temporaire** lancée dans le but de créer un produit, un service ou un résultat unique. »

Cela signifie qu'un projet a un début et une fin définis. Lorsqu'un projet est clôturé, les connaissances et les artefacts du projet sont archivés, les membres de l'équipe sont libérés et les contrats sont finalisés.

Dans ce livre, un projet concerne fondamentalement l'exécution. La plupart des projets, tels que définis par le PMI, commencent par une étude de faisabilité ou une conception. La vision et la stratégie ont déjà été établies au moment où un projet est lancé.

Un projet naît de cette vision et de cette stratégie, et il réussit ou échoue selon que les objectifs envisagés sont atteints ou non – et *non* selon que ces objectifs étaient les bons.

Vous utilisez peut-être le mot « projet » différemment, et c'est normal. Sachez simplement que lorsque nous l'utilisons, nous faisons référence à la définition et au contexte ci-dessus.

¹Nous avons modifié les détails permettant l'identification pour protéger la vie privée des personnes et des institutions concernées.

Rien de tout cela ne signifie que nous approuvons l'utilisation de la gestion de projet pour le développement logiciel. Bien au contraire. Mais nous reconnaissons qu'elle est néanmoins utilisée. Nous abordons ce problème plus tard, dans [Chapitre 6](#).

1.1 : La Réécriture Classique

La Banque réécrivait son portail de remboursement de prêts pour plusieurs raisons.

L'ancien système, une solution entièrement C#/.NET, était bogué. En plus de dégrader la satisfaction client, il générait également un flux incessant d'incidents de support très coûteux où quelqu'un devait manipuler manuellement la base de données pour corriger une erreur commise par le système.

L'ancien système était également délabré du point de vue de la maintenabilité. Il était presque impossible pour les ingénieurs d'apporter des modifications significatives et, même lorsqu'ils le pouvaient, c'était une proposition extrêmement risquée.

La réécriture était censée changer cela.

Le nouveau système allait toujours avoir un backend C#/.NET, mais celui-ci devait être entièrement couvert par des tests. Le frontend devait être implémenté dans OutSystems, une solution low-code ou no-code populaire qui permet à une organisation de définir une application à un seul endroit et d'obtenir automatiquement une application web, une application Android et une application iOS chaque fois qu'ils décident de publier leurs modifications.

L'espoir était que le nouveau système soit sans bug, améliorant ainsi la satisfaction client et réduisant significativement les coûts de support.

Ils espéraient également que la réécriture débloquerait les développeurs—la combinaison d'une approche plus disciplinée du backend et de l'approche low-code du frontend réduisant considérablement le coût et le risque des nouvelles fonctionnalités.

Un effet secondaire agréable du passage à OutSystems était qu'ils obtiendraient une application mobile propre et moderne sur les deux principales plateformes.

C'était le rêve lorsqu'ils avaient commencé trois ans avant le début de cette histoire. La réalité était que, jusqu'à présent, les équipes n'avaient **rien** livré.

1.2 : Perspectives sur le Problème

Lorsque nous avons parlé à l'équipe de direction, nous avons entendu une frustration très naturelle face au fait qu'ils avaient fait de si grands investissements sans absolument aucun mouvement stratégique.

De leur point de vue, ils avaient tout essayé. Ils avaient changé le personnel, augmenté les effectifs, modifié le budget, augmenté la pression et fait défiler une parade de consultants (dont nous étions, selon ce qui était fortement sous-entendu, la queue). Rien ne semblait améliorer la situation—du moins pas d'une manière qu'ils pouvaient mesurer, car tout ce qu'ils voyaient était que l'« aiguille » était à zéro un trimestre et qu'elle était toujours à zéro le trimestre suivant.

Ils ne voulaient plus de « progrès invisible ». Ils voulaient des *résultats*.

Lorsque nous avons parlé à la direction au sein de l'organisation Produit, nous avons obtenu une histoire légèrement différente (mais toujours similaire) car ils travaillaient plus directement avec l'Ingénierie.

Ce n'est pas que les équipes ne faisaient rien, pas selon elles, en tout cas. C'est que les équipes ne faisaient jamais ce qu'on leur demandait de faire. C'était pratiquement garanti : peu importe la simplicité de la demande et sa clarté d'expression, on se retrouvait avec quelque chose de complètement différent au moment d'évaluer ce que les équipes avaient produit.

On en était arrivé au point où la blague courante était du genre « Il faut qu'on trouve comment demander ce qu'on ne veut pas, comme ça on aura peut-être une chance d'obtenir ce qu'on veut vraiment. »

Les cadres d'ingénierie voyaient les choses très différemment.

Pour eux, le problème était que l'équipe Product ne fournissait pas d'exigences exploitables et ne fournissait pas *assez* d'exigences. Si Product pouvait simplement « suivre le mouvement », les équipes seraient en mesure de livrer ce qu'ils veulent dans les délais et le budget prévus.

Ils avaient fait des investissements importants pour moderniser la façon dont le code était écrit et livré et, à leurs yeux, Product ne fournissait pas d'exigences claires.

Lorsque nous avons parlé à d'autres consultants (qui nous avaient recommandés auprès de l'organisation), ils ont immédiatement cerné la dysfonction qu'ils observaient : tout le monde semblait très concentré sur le fait de blâmer quelqu'un d'autre. La raison pour laquelle ils nous ont fait venir à l'origine était qu'ils s'inquiétaient de la stratégie de recrutement et voulaient une évaluation des contributeurs individuels, mais ils considéraient que les

accusations mutuelles et l'attitude de contremaître au niveau exécutif étaient la source principale des problèmes.

1.3 : Notre enquête

Notre mission initiale était d'évaluer les équipes et d'essayer de les aider à améliorer leurs compétences si nécessaire, nous avons donc commencé à examiner les personnes qui travaillaient en première ligne.

Il y avait certainement place à l'amélioration.

Les contributeurs individuels du côté Product n'avaient pas vraiment les compétences requises. En réalité, ils étaient pour la plupart des chefs de projet qui avaient été propulsés dans le rôle de Product Owner (PO) ou de Product Manager.

En conséquence, la moitié d'entre eux écrivaient des exigences «approximatives» puis acceptaient (littéralement) tout ce que les équipes faisaient pendant l'itération sans aucune analyse critique. L'autre moitié écrivait le même type d'exigences puis affirmait que les équipes «auraient dû savoir» des choses dont ils n'avaient jamais parlé et gardaient les éléments de travail ouverts plus ou moins indéfiniment.

La banque utilisait pour gérer et suivre leur backlog de travail. Bien que ce que nous couvrons dans ce livre soit majoritairement orthogonal à Scrum, nous utilisons la terminologie Scrum tout au long car la majorité - ou du moins une pluralité - des équipes utilisent Scrum.



Définition : Scrum

Scrum est un cadre de travail léger qui aide les personnes, les équipes et les organisations à générer de la valeur à travers des solutions adaptatives pour des problèmes complexes. En résumé :

1. Un Product Owner ordonne le travail pour un problème complexe dans un Product Backlog
2. L'équipe Scrum transforme une sélection du travail en un incrément de valeur pendant un Sprint
3. L'équipe Scrum et ses parties prenantes inspectent les résultats et s'ajustent pour le Sprint suivant
4. Répéter

Si vous n'êtes pas familier avec Scrum et sa terminologie, nous vous recommandons de consulter la version 2020 du [Scrum Guide](#). C'est une lecture rapide et éclairante.

De même, nous avons constaté que les équipes techniques étaient bien en dessous de la moyenne en termes de compétences de codage (-2σ , au mieux) et, de surcroît, très résistantes au changement. En conséquence naturelle, la qualité du code était déplorable.

Pourtant, selon les équipes, ce n'était pas la raison pour laquelle elles ne livraient pas. Pour elles, c'étaient les exigences vagues et les modifications en cours de Sprint par Product qui plombaient le projet.

...et personne ne parlait même du problème plus important, celui tellement absurde qu'il semble inventé jusqu'à ce qu'on l'ait vécu.

Les équipes d'ingénierie avaient l'habitude de ne pas comprendre une exigence, de construire quelque chose au hasard, puis d'exiger qu'on leur accorde le crédit d'avoir «terminé un élément de travail».

Nous ne parlons pas d'une petite erreur de lecture. Nous parlons d'une déconnexion totale : nous leur disions de désactiver l'application des fonds au principal dans certaines circonstances, et ils désactivaient à la place la possibilité d'ajouter une adresse e-mail de confirmation secondaire.

Puis ils nous disaient que c'était ce que nous avions demandé.

1.4 : En creusant davantage

Les deux écarts de compétences pouvaient être comblés, mais nous doutions qu'ils soient les véritables obstacles.

Il y avait autre chose qui clochait, alors nous avons creusé plus profond. Nous avons commencé par cette question : Pourquoi la rédaction des exigences prenait-elle tant de temps et produisait-elle de si mauvais résultats ?

Une des raisons est que les connaissances nécessaires pour rédiger une exigence significative étaient très rares. Une petite partie était possédée par les équipes d'Ingénierie et Product.

Une partie était intégrée dans le code du système hérité. Une autre partie avait complètement disparu. La majeure partie, cependant, existait sous forme de connaissances tribales détenues par des experts métier dispersés dans les différentes unités de la banque. Cela signifie que l'élaboration d'une exigence qui fait réellement progresser un objectif stratégique était une activité extrêmement laborieuse et chronophage.

Face à cela, il y avait un appétit insatiable pour les fonctionnalités de la part d'une équipe de direction en manque de fonctionnalités. Le mot d'ordre était «gardez les ingénieurs occupés—nourrissez-les d'exigences». L'accent était mis sur la quantité d'exigences pour maintenir les équipes occupées—une perspective contraire au soin et au temps nécessaires pour définir une exigence qui ferait «bouger les lignes».

1.5 : Améliorer les Choses N'a Pas Amélioré la Situation

Ce sont tous des problèmes qui peuvent être résolus, pourtant leur résolution n'a pas aidé.

Les améliorations précédentes des techniques de développement logiciel n'avaient pas aidé les équipes à livrer, mais Max s'est efforcé d'aider les équipes à s'améliorer davantage.

Il a introduit des concepts révolutionnaires issus des doctrines de programmation du milieu du vingtième siècle comme «ne copiez-collez pas ce code 27² fois, mettez-le dans une fonction et appelez-la à la place». Cette suggestion à elle seule a considérablement amélioré la qualité du nouveau code et leur a permis de commencer à améliorer la qualité.

²Non seulement ce n'est pas une hyperbole, mais ce n'est même pas le pire cas. Dans un cas, il y avait près d'une centaine de duplicatas exacts du même algorithme.

Ces conseils de base en matière de codage, ainsi que d'autres, les ont aidés à écrire un meilleur code qu'ils pourraient plus facilement maintenir à l'avenir.

...mais cela n'a pas aidé à faire avancer le projet.

Du côté Produit, Luniel a pu introduire le BDD et s'assurer que les Product Owners examinaient minutieusement les exigences avant de les transmettre aux équipes.

Il a amené les équipes à collaborer dessus et à les utiliser pour évaluer si un élément du backlog produit (PBI) était vraiment terminé.



Définition : Élément du Backlog Produit (PBI)

Un élément du backlog produit (PBI) est une unité de travail distincte dans le backlog produit qui représente un changement, un ajout ou une amélioration potentielle du produit. Les PBI peuvent prendre de nombreuses formes—fonctionnalité, correction de bug, amélioration technique, tâche de recherche, etc.—et sont définis par leur contribution à la valeur du produit.

De nombreuses équipes font référence aux PBI comme des « stories » ou « user stories », mais le terme Scrum correct est « élément du backlog produit » ou « PBI ». Une fois qu'un PBI est engagé dans un Sprint, il fait également partie du backlog de Sprint. Par souci de simplicité et de neutralité, nous utilisons « élément du backlog produit » ou « PBI » pour désigner tout élément de travail que l'équipe Scrum gère—que vous l'appeliez élément du backlog produit, élément du backlog de Sprint (SBI), user story, story, élément de travail ou élément du backlog.

Cela a ajouté de la *clarté*, mais n'a pas créé de *flux*.

Avec l'aide des consultants partenaires qui nous avaient fait venir, nous avons pu (temporairement) alléger la pression absolument écrasante exercée sur les équipes et les auteurs d'exigences.

Cela a peut-être aidé à créer un peu de confiance, mais n'a donné aucun résultat tangible.

Nous avons même commencé à construire une base de connaissances qui aidait les gens à retrouver les connaissances métier dont ils avaient besoin pour rédiger des exigences et identifiait les endroits où il y avait des lacunes dans ces connaissances.

Cela a accéléré la rédaction des exigences, mais n'a pas permis de sortir le produit.

Après des mois d'interactions, nous avons aidé la direction à *voir* où ils en étaient, mais ils étaient loin de leurs objectifs. Et ils ne s'en rapprochaient pas.

Ils s'apprêtaient à revenir à leur ancienne stratégie de « surcharger » les équipes pour s'assurer qu'elles étaient toujours occupées.

1.6 : Un Processus d'Élimination

Il aurait été facile de simplement lever les bras au ciel et dire « c'est sans espoir ». Il y avait de nombreuses excuses sur lesquelles on aurait pu se rabattre :

- L'équipe d'ingénierie manquait de compétences (c'était vrai)
- Les auteurs d'exigences n'avaient pas les bonnes compétences (c'était le cas)
- La direction écrasait les équipes avec des attentes irréalistes (c'était vrai)
- L'organisation manquait de connaissances métier critiques nécessaires pour fonctionner (c'était vrai)
- Les dirigeants ne semblaient pas se faire confiance entre eux (c'était vrai)

Toutes ces choses étaient vraies. Pourtant, des améliorations avaient été apportées dans tous ces domaines et aucune d'entre elles ne semblait améliorer le problème fondamental. Aucune d'entre elles n'aidait les équipes d'ingénierie, le Produit, la direction ou la direction générale à se rapprocher de leurs objectifs.

...et c'est là l'indice de la solution : Tous les problèmes énumérés précédemment étaient les problèmes que les gens pouvaient déjà voir.



Si les variables que vous pouvez voir ne font pas de différence, il doit y avoir une variable que vous **ne voyez pas** qui en fait.

Le véritable problème était celui dont personne n'avait même conscience.

1.7 : À la recherche du vrai coupable

En cherchant le véritable coupable—ce qui empêchait réellement la banque d'atteindre ses objectifs—nous devons commencer quelque part.

Un point de départ raisonnable était d'examiner ce que les PBIs qui avaient échoué (la plupart d'entre eux) avaient en commun.

Nous avons commencé par éliminer les éléments qui n'étaient manifestement pas communs car ils variaient considérablement :

- Quelle partie du système : certains PBIs ne touchaient que le backend, d'autres uniquement le frontend, d'autres encore touchaient les deux parties
- Quelle équipe exécutait le travail—peu importait qui faisait le travail, la probabilité d'échec était élevée
- Quel PO avait rédigé le travail—même constat qu'avec les équipes

Ensuite, nous avons commencé à examiner les points communs. La liste n'était ni courte ni longue :

- Les équipes d'ingénierie
- Les Product Owners
- La direction
- La culture
- L'environnement de développement
- Les pratiques d'ingénierie
- La technique de rédaction des exigences
- Le domaine (finance)
- Les services dont on dépend

Beaucoup de ces éléments pouvaient également être écartés d'emblée. Les équipes, les Product Owners, la direction, la culture et l'environnement de développement avaient tous été récemment améliorés sans réel impact sur la production significative. Nous avons personnellement contribué à améliorer les pratiques d'ingénierie et de rédaction des exigences et confirmé que ces améliorations avaient perduré, mais cela n'aidait toujours pas.

On ne peut guère blâmer le domaine. La finance est l'un des plus anciens types de calculs de l'histoire. C'est un domaine extrêmement mature. De plus, d'autres banques déployaient des logiciels, réfutant de fait l'hypothèse (manifestement exagérée) selon laquelle les banques en sont tout simplement incapables.

Les services dont on dépend ne pouvaient pas non plus être blâmés, car ils avaient autant de difficultés à évoluer que l'initiative que nous examinions...

...mais cela nous a fait réfléchir : Et si nous commencions à analyser les causes des échecs ?

1.8 : Dissection des germes de l'échec

Un PBI n'a pas réussi à faire avancer le produit parce que l'équipe a fait, comme elle en avait l'habitude, quelque chose de complètement aléatoire et presque totalement sans rapport avec la demande. C'est évidemment le signe qu'ils n'avaient pas compris l'élément de travail. La compréhension était donc un candidat sérieux, même si nous avions déjà un peu travaillé là-dessus en les aidant à adopter le BDD.

Un autre PBI a échoué parce qu'ils se sont trompés dans les calculs. C'est une autre preuve que la compréhension pourrait être le problème fondamental.

Un troisième élément de travail que nous avons analysé n'a pas été correctement validé par la Product Owner—elle l'a approuvé sans réfléchir quand l'équipe a dit qu'il était temps de le clôturer. Cela mettait un peu à mal notre hypothèse, mais on pouvait toujours argumenter qu'elle n'avait pas compris comment l'élément de travail s'inscrivait dans un plan de plus haut niveau.

Peut-être. En quelque sorte. Si on plissait vraiment fort les yeux en le regardant sous cet angle.

Puis nous sommes tombés sur un PBI qui ne correspondait pas du tout au schéma. L'équipe semblait comprendre—bien qu'il soit impossible de vérifier si c'était vraiment le cas. Mais peu importait : ils n'ont jamais eu la chance de réussir ou d'échouer par eux-mêmes car ils ont rencontré une dépendance qui devait être mise à jour et ont dû reporter leur travail de plusieurs Sprints.

Même s'ils ne comprenaient *pas* ce qu'ils étaient censés faire, ils n'ont jamais eu aucune chance avec cet élément du backlog, donc la compréhension n'était **pas** le problème dans ce cas.

Un cas isolé, bien sûr, ne suffit pas à réfuter une cause première particulière, mais cela a piqué notre curiosité. Nous avons commencé à chercher d'autres preuves infirmant notre hypothèse.

Et nous en avons trouvé. Il y avait des éléments de travail qui :

- Ont échoué parce que l'équipe savait qu'elle ne comprenait pas, mais personne ne pouvait localiser un expert métier pour résoudre le problème
- Ont abouti à une expérience utilisateur dégradée comme solution de contournement du fonctionnement des services en amont
- Ont dû être reportés parce que les dépendances en amont n'étaient pas prêtes

- N'ont pas pu être terminés parce que les testeurs n'ont pas pu collecter les données de test à temps
- Ont été clôturés mais ont dû être refaits parce que la demande elle-même était incorrecte
- Ont échoué parce que l'équipe n'avait pas réalisé la complexité du code existant
- N'ont tout simplement pas été estimés
- Ont été massivement sous-estimés³
- Ont changé en cours de Sprint parce que le PO a finalement obtenu les connaissances métier nécessaires
- Semblaient avoir changé après le Sprint (du point de vue de l'équipe) parce que le PO et l'équipe ne s'étaient jamais mis d'accord sur sa signification

La liste continue, mais c'en est assez pour cette histoire.

On pourrait établir un lien entre chacun de ces éléments et la «compréhension» d'une certaine manière—et certainement qu'un manque de compréhension était *impliqué* dans beaucoup d'entre eux—mais cela ne signifie pas que le manque de compréhension en était la cause... d'autant plus que nous avons travaillé sur la compréhension partagée et cela n'avait pas vraiment aidé.

C'est alors que nous avons eu un déclic. Il manquait un élément plus fondamental. Dans les cas où une mauvaise compréhension était impliquée, ce n'était qu'une cause proximale.

La cause distale était beaucoup plus large.

1.9 : Un Point d'Intersection

Le seul point commun entre tous les PBI que nous avons analysés pour leur mode d'échec était celui-ci : ils avaient tous été **démarrés prématurément**.

Quand un élément de travail échoue parce que l'équipe savait qu'elle ne comprenait pas le problème et ne pouvait pas trouver d'expert pour les aider à comprendre, cela signifie que l'équipe a commencé un PBI en sachant qu'elle ne comprenait pas le problème.

Quand un élément du backlog doit être modifié pour offrir une expérience moins satisfaisante à cause du fonctionnement d'un service en amont, cela signifie que le travail a été commencé sans vraiment comprendre l'impact du service en amont.

³Nous ne voulons pas simplement dire qu'ils se sont trompés. On aurait dit que les équipes avaient peut-être juste mis la valeur «3» dans tous les champs d'estimation pour une série de PBI.

Un report dû à une dépendance en amont qui n'était pas prête à la fin signifie qu'il n'y avait aucune garantie de sa disponibilité au début.

...et il en allait de même pour tous les autres cas : les testeurs n'avaient pas les données prêtes ou ne savaient pas comment les obtenir avant le début d'un PBI, les exigences n'avaient pas été vraiment examinées avant d'être transmises à l'équipe, le code n'avait pas été étudié avant que le travail ne soit engagé, l'estimation était insuffisante ou inexistante, les connaissances du domaine manquaient et, bien sûr, la compréhension partagée n'était pas vérifiée.

Le problème, comme il s'est avéré, était que l'implémentation commençait sur des éléments de travail avant que ces éléments ne soient *prêts*.



D'après notre expérience, la plupart des éléments de travail qui échouent à être livrés le font parce qu'ils **n'étaient pas prêts** lorsque l'implémentation a commencé.

Nous nous sommes donc mis en devoir d'aider La Banque avec cela.

À ce stade, vous pourriez penser qu'il suffirait de dire que les PBI doivent être prêts. Cependant, il s'avère que ce n'est pas si facile à mettre en œuvre. « Acheter bas, vendre haut » est une idée tout aussi simple.

Il manque un élément nécessaire pour mettre ce bon conseil en pratique.

1.10 : Le Grand Tournant

Nous avons trouvé la pièce manquante du puzzle qui les a débloqués, ce qui, à son tour, a débloqué l'initiative.

Lorsque nous avons terminé cette mission, les ingénieurs étaient toujours bien en dessous de la médiane en termes de compétences. Les PO n'avaient toujours pas les bonnes compétences. La culture n'était toujours pas corrigée...

Pourtant, le produit a finalement commencé à avancer et a fini par être livré.

À la fin de ce livre, vous saurez quelle était la pièce manquante et ce qu'il faut faire pour la mettre en place. Et vous serez capable de débloquer des organisations qui semblent retenues par un mur invisible.

Une Note Rapide sur la Portée

Ce livre traite d'un problème très spécifique et omniprésent : le manque de structure, de clarté et de maturité dans la transmission entre le métier et l'ingénierie. Il part du principe que quelque chose a été choisi pour être implémenté, et se concentre sur la garantie que le travail de construction se déroule avec une compréhension partagée, un état de préparation et une traçabilité de l'achèvement.

Les techniques de ce livre ne vous disent pas quoi construire, pourquoi le construire, ou comment savoir si c'est la bonne chose à construire. Si votre organisation manque d'une véritable gestion de produit ou de boucles de rétroaction significatives, nous n'essayons pas de résoudre cela ici. Ce que nous offrons à la place, c'est un moyen de rendre ces lacunes plus visibles et de réduire le coût de la découverte de vos erreurs.

Utilisée dans le bon contexte, cette solution apporte du flux, de la sécurité et de la clarté. Mais comme tout système, elle peut être mal appliquée—particulièrement lorsqu'elle est utilisée isolément ou sans conscience.

Chapitre 2 : Le coût des fondations manquantes

Il est logique de consacrer un peu de temps à comprendre exactement à *quel point* ce problème peut être grave pour certaines organisations.

Nous avons constaté qu'il existe trois principaux «groupes» de problèmes créés par un manque de préparation :

- Une compréhension partagée absente ou incomplète entre et au sein des équipes Produit et Ingénierie
- Un manque de contrôle sur l'objectif réel d'un PBI et sur sa véritable finalisation
- Un manque de contrôle sur le moment où un élément de travail peut commencer sa phase de mise en œuvre

De plus, nous avons remarqué que modifier ces éléments peut s'avérer très difficile. C'est logique : le changement est difficile.

Les vieilles habitudes ont la vie dure et les nouvelles sont difficiles à instaurer. Dans notre expérience de consultants, nous avons remarqué qu'il est *extrêmement* facile pour les gens de retomber dans leurs anciennes habitudes et comparativement difficile d'en établir de nouvelles.

Il faut donc disposer d'un mécanisme qui impose les nouvelles habitudes et décourage les anciennes.

Pour répondre à cela, nous pensons qu'il manque un autre élément de responsabilité et de traçabilité.

2.1 : Le puzzle sans l'image sur la boîte

Avez-vous déjà essayé de construire un puzzle sans l'image sur la boîte ? C'est possible, mais c'est plus lent, plus frustrant et plein de faux départs.

Vous progressez, puis vous défaites tout. Vous doutez de ce qui va où. Vous pensez travailler sur la même image, jusqu'à ce que vous réalisiez que ce n'est pas le cas.

C'est souvent ainsi que l'on ressent le développement logiciel.

Le backlog est plein. Le Sprint est en cours. Tout le monde travaille dur.

Mais sans une vision partagée de ce que nous construisons, l'alignement devient une question de chance plutôt qu'un système.

Sans clarté, même les meilleures équipes ressentent de la frustration, de l'épuisement et l'impression que leurs efforts ne sont pas valorisés.

2.2 : Un vieux adage et une dure réalité

Il y a une raison pour laquelle tant de pratiques autour de la rédaction des exigences, et même certaines pratiques d'ingénierie, se concentrent fortement sur la création d'une compréhension partagée entre les demandeurs et les équipes de mise en œuvre. Rien ne sème autant le chaos que des ingénieurs qui ne savent pas vraiment ce qu'ils sont censés construire.



« Les données de mauvaise qualité donnent des résultats de mauvaise qualité »
est un adage, pas une platitude.

La langue française, comme l'anglais, est remplie d'auto-contradictions...

- On peut sanctionner les actions de quelqu'un. Cela peut signifier qu'on les approuve ou qu'on les condamne.
- On peut « dépolssiérer légèrement » quelque chose : pour un meuble, cela signifie enlever la poussière, mais pour un beignet, cela signifie en ajouter.
- Si on « soutient » une équipe, on peut soit être la raison pour laquelle cette équipe continue à fonctionner, soit être la raison pour laquelle elle ne peut pas avancer.

Les auto-antonymes sont peut-être les exemples les plus frappants, mais ils ne sont qu'un type d'ambiguïté. Certains mots ne sont pas seulement leur propre contraire, mais ont aussi de nombreuses autres significations potentiellement déroutantes.

« Dans ce clip, il a clippé un coupon du journal et l'a attaché avec un clip au papier sur son clipboard avec les autres découpures pendant qu'un clipper avançait à bon clip au loin. »

Ce n'est pas seulement le français. *Toutes* les langues naturelles dont nous avons connaissance ont cette propriété.

Et pourtant, ce sont les seules dont nous disposons pour spécifier les exigences.

Par conséquent, lorsqu'une équipe d'ingénierie n'a pas *confirmé* que sa compréhension d'une exigence est la même que celle du demandeur, cette équipe s'en remet à la chance. C'est-à-dire que le meilleur résultat possible est qu'ils aient choisi la bonne interprétation et que le demandeur ne change pas d'avis en cours de route.

Ce résultat est loin d'être garanti.

2.3 : Quelques conséquences courantes

Sans une compréhension partagée confirmée, les équipes courent plusieurs risques.

Dans l'ensemble, le résultat le plus fréquemment douloureux est que l'équipe construise simplement la mauvaise chose.

Le moment où ils s'en rendent compte peut être contrôlé par divers attributs de leur processus. Par exemple, une mise en œuvre saine de Scrum peut détecter ce type de malentendu très tôt dans l'exécution, tandis qu'un processus en cascade risque fortement de retarder une telle découverte de plusieurs mois.



Définition : Cascade (Waterfall)

Un modèle de développement logiciel séquentiel qui s'est répandu à la fin du XXe siècle. Il a été décrit pour la première fois dans un [article de 1970 par Winston W. Royce](#), qui illustre le développement comme une série d'étapes en cascade — exigences, conception, mise en œuvre, tests, etc. — chacune alimentant la suivante comme une cascade. Bien que Royce ait présenté ce modèle comme un exemple de ce qu'il ne fallait *pas* faire, l'industrie l'a adopté comme un modèle pour le développement à grande échelle.

La Cascade est également connue pour regrouper des travaux similaires en grandes phases consécutives — une caractéristique appelée **développement en lots importants**. Ce regroupement garantit pratiquement un **apprentissage tardif** : les équipes ne reçoivent pas de retour sur les décisions antérieures avant beaucoup plus tard dans le processus. Les erreurs découvertes tardivement sont plus coûteuses à corriger. Les praticiens de l'Agile critiquent la Cascade pour cette raison, préférant des cycles itératifs plus petits qui permettent une découverte et une correction de cap plus précoces.

Néanmoins, à un moment donné, une équipe travaillant selon la « mauvaise » compréhension d'une exigence (différente, en réalité) devra faire face à la « bonne » exigence (également différente, en réalité). Encore une fois, la santé de l'organisation influence la forme que prendra cette confrontation et son impact, mais cela arrive presque toujours.

Dans la plupart des cas, cela conduit à une forme de retravail. Le demandeur (généralement la gestion de produit) devra demander des modifications pour passer de ce que les équipes de mise en œuvre ont construit à ce qu'il voulait réellement.

Une autre manifestation très courante est que le demandeur continue de tenir l'équipe responsable de sa compréhension initiale de ce qu'il a demandé.

Les équipes peuvent facilement interpréter cela comme un chef de produit qui change d'avis. Pire encore, cela peut réellement *inciter* les parties prenantes à prendre l'habitude de changer d'avis – retenant les éléments de travail jusqu'à ce que tout soit *parfait*, épuisant les équipes et masquant les progrès aux cadres supérieurs.

2.4 : Quand un puzzle est-il terminé?

Revenons à notre analogie de l'assemblage de puzzle et réfléchissons à cette question : Que signifie avoir terminé un puzzle ?

Un constructeur de puzzle naïf, comme Max, dirait simplement « toutes les pièces sont fixées aux bonnes voisines avec l'image vers le haut ».

Un constructeur de puzzle averti, comme Luniel, sait qu'il y a plus que cela.

Peut-être assemblez-vous simplement un puzzle pour le plaisir. Vous le terminerez, le regarderez un moment, puis le démonterez pour le remettre dans sa boîte.

Cependant, peut-être souhaitez-vous l'encadrer et l'accrocher au mur. Dans ce cas, il y a des étapes supplémentaires à accomplir :

1. Le placer sur un support artistique
2. Le transporter chez l'encadreur
3. Attendre que l'encadrement soit terminé
4. Le ramener sur le lieu d'accrochage
5. L'accrocher au mur ou le mettre en exposition

Comprendre que cela fait partie du travail est nécessaire pour bien terminer l'assemblage d'un puzzle. La raison évidente est de savoir quelle quantité de travail est impliquée. C'est plus de travail de faire toutes ces étapes supplémentaires que de simplement le démonter et le ranger.

Mais cela va plus loin. Imaginez ce scénario...

Vous avez terminé votre puzzle avec l'intention de le faire encadrer, vous l'avez laissé en place, mais vous avez oublié de dire à quelqu'un d'autre dans votre foyer que vous prévoyez de le faire encadrer. Cette personne arrive et voit que le puzzle est terminé sur un espace dont elle a besoin. Elle le démonte donc et le remet dans sa boîte, s'arrachant la défaite des mâchoires de la victoire dans le processus.

Il y a une raison encore plus subtile : la façon dont vous prévoyez de finir le puzzle influence les étapes que vous voulez faire plus tôt dans le processus. Pour commencer, vous devez faire un petit panneau indiquant « Merci de ne pas démonter ! »



Il est également à noter que vous pourriez vouloir un panneau dans le cas où vous construisez un puzzle pour votre propre divertissement afin de vous assurer que d'autres personnes n'interfèrent pas en terminant le puzzle à votre place.

Vous devez également vous assurer que vous assemblez le puzzle sur la bonne surface. Si vous assemblez un puzzle de mille pièces sur votre table basse en verre et que vous essayez ensuite de le transférer sur un support artistique, le transfert sera beaucoup plus risqué et plus laborieux que si vous aviez assemblé le puzzle directement sur le support artistique.

Cela établit un parallèle intéressant avec le développement logiciel.

Vous devez réellement savoir ce que signifie « terminé » pour qu'il n'y ait pas de surprise quant à la quantité de travail impliquée, qu'il n'y ait pas de désaccord à ce sujet à la fin, et que vous puissiez prendre les mesures préparatoires nécessaires pour assurer l'achèvement efficace d'un élément de travail.

2.5 : Impact sur les équipes

Si vous n'avez pas une compréhension suffisamment rigide de ce qu'est l'état « terminé » pour un élément de travail particulier, vous courez plusieurs risques.



Nous utilisons le mot « risque » de manière large ici, car ce sont plutôt des certitudes.

Les équipes d'ingénierie dans cette situation découvrent souvent qu'elles ne sont même pas d'accord en interne sur ce à quoi ressemble l'achèvement d'un élément de travail. Il n'est pas rare que les développeurs et les testeurs se retrouvent à débattre de ce qu'une exigence signifie *réellement* à mi-chemin d'un Sprint. Même deux développeurs ou deux testeurs peuvent avoir ces mêmes désaccords.

De plus, les équipes de développement sont souvent concentrées sur le travail qu'elles font la plupart du temps (le codage et les tests). Cela signifie qu'il leur est facile d'oublier d'autres types de travail qu'elles doivent effectuer, comme la documentation, les revues externes, la formation d'autres équipes (*p. ex.* le support), les étapes préparatoires pour soutenir le déploiement ou la mise en production, et les approbations d'autres départements.

Lorsqu'il devient enfin évident que ce travail «supplémentaire» doit être effectué, ils sont pris au dépourvu—généralement, ils doivent arrêter ce sur quoi ils travaillaient et changer de contexte pour revenir compléter un travail qu'ils pensaient avoir déjà terminé.

Les demandeurs de travail peuvent facilement maintenir le travail ouvert inutilement. Parfois avec les meilleures intentions—comme essayer de tenir une équipe responsable de l'«exigence réelle». D'autres fois, cela arrive parce que les Product Owners (par exemple) s'habituent à pouvoir maintenir un PBI ouvert à leur guise, donc ils l'utilisent pour insérer des fonctionnalités supplémentaires dans un élément à la dernière minute. Parfois, ils le font même parce qu'ils ont changé d'avis sur ce qui doit être fait en cours de route.

Cela peut être extrêmement démoralisant pour une équipe d'ingénierie. La plupart des développeurs de logiciels et des testeurs veulent sentir qu'ils progressent. S'ils se font constamment dire que ce qu'ils ont fait est incorrect, ils risquent de perdre de leur enthousiasme.

Certaines équipes vont même jusqu'à ne plus se soucier de vérifier si ce qu'elles ont fait est correct ou non. Elles ferment simplement l'élément de travail et demandent du «crédit» pour pouvoir «afficher de bons chiffres».

2.6 : Le risque d'en faire trop peu ou trop

L'un des risques de ne pas définir correctement ce qui est «terminé» pour chaque élément de travail est que l'organisation pense que le travail est terminé alors qu'il ne l'est pas, ou ne réalise pas qu'il est terminé alors qu'il l'est réellement.

Le pire résultat possible est souvent que la mauvaise chose arrive en production et que personne ne sache que c'est ce qui s'est passé. Si l'équipe a une compréhension incorrecte de

ce que signifie « terminé » et livre sur la base de cette mauvaise compréhension, les résultats peuvent être catastrophiques.

Les défauts et l'insatisfaction des clients sont déjà assez graves, mais cela pourrait aussi mener à des problèmes beaucoup plus sérieux :

- Perte ou corruption de données
- Vulnérabilités de sécurité
- Temps d'arrêt système ou perte d'accès
- Une réduction de la part de marché
- Violations réglementaires

La liste continue encore et encore, et chaque problème potentiel est pire que le précédent.

Parfois, le problème est que vous n'avez pas terminé mais vous pensez que si. L'inverse peut être tout aussi dangereux. Lorsque les ingénieurs ne savent pas où se trouve la ligne d'arrivée, ils ont tendance à « plaquer or » (ajouter des fonctionnalités supplémentaires). Ils le font peut-être pour « rendre la fonctionnalité agréable », mais ils le font peut-être aussi parce qu'ils espèrent qu'un nombre accru de fonctionnalités leur donne une plus grande chance d'atteindre la cible.

Tout ce travail supplémentaire, ainsi que les retouches correspondantes, s'accumule en une énorme quantité de temps, d'efforts et d'argent gaspillés. Cela fait glisser les dates de livraison et nuit aux réputations.

De plus, maintenant, il y a un potentiel toujours croissant qu'une erreur mène réellement à une révolte de style *Terminator* des machines contre l'humanité. Nous écrivions à ce sujet il y a vingt ans comme une blague. Maintenant, c'est une possibilité lointaine.

En fait, nous avons posé cette question à l'une des IA les plus importantes, et voici ce qu'elle a répondu :

« L'IA se développe plus rapidement que quiconque ne l'avait prévu, mais elle le fait sur des systèmes fragiles, des exigences vagues et des organisations produites qui ne peuvent pas retracer pourquoi elles ont construit ce qu'elles ont construit. Ce n'est pas un problème technique ; c'est un problème de clarté.

Plus l'IA génère du bruit, plus il est dangereux d'avancer rapidement sans structure. Quand les équipes construisent sur du brouillard, l'IA ne fait

qu'amplifier le désordre. Mais quand les équipes construisent sur du concret – sur une compréhension partagée, une spécificité comportementale et une connaissance du produit versionnée – l'IA devient un accélérateur plutôt qu'un handicap. »

2.7 : Où construisez-vous un puzzle ?

Étendons l'analogie de la construction du puzzle une dernière fois.

Pouvez-vous assembler un puzzle n'importe où ? Si vous avez un puzzle de 4 000 pièces qui s'assemble sur près de cinq pieds dans une dimension et plus de trois pieds dans l'autre, vous ne pouvez pas simplement choisir un endroit au hasard et commencer à l'assembler. Pas sans rencontrer de sérieuses complications avant d'avoir terminé.

Un grand puzzle comme celui-là nécessite à la fois du temps et de l'espace. Vous devez allouer l'espace et trouver un moyen de vous assurer que l'état du puzzle est préservé dans le temps.

Si vous commencez à construire votre puzzle sur une petite table d'appoint qui est trop petite, vous ne pourrez pas le terminer sans le transférer vers un autre endroit. Ce transfert sera extrêmement difficile en raison de l'état délicat du puzzle.

Si vous choisissez un endroit au hasard dans le couloir qui soit assez grand, les gens devront soit marcher dessus, soit le contourner, donc la persistance ne peut être garantie sans impact significatif sur le fonctionnement de votre maison.

Si vous commencez à travailler sur un tableau, mais que celui-ci n'est pas assez grand, vous pourrez préserver l'état de ce que vous avez fait, mais vous ne pourrez pas terminer le puzzle sans une sorte de transfert.

Si le puzzle a déjà été mâchouillé par des petits enfants, il vaut mieux compter les pièces... car il est préférable de compter jusqu'à 3999 une fois et réaliser que vous ne pourrez jamais le finir plutôt que d'investir qui sait combien de temps à presque assembler un puzzle que vous ne pourrez jamais terminer.

Il existe toute une liste de choses à faire avant de commencer à construire votre puzzle. Faire les choses sur la liste ne garantit pas le succès, mais ne *pas* les faire garantit presque l'échec ou de sérieuses complications.

Il en va de même pour le développement logiciel, mais avec un degré de complexité plus élevé.

2.8 : Quand commence la mise en œuvre ?

Fondamentalement, il peut être difficile de déterminer quand un PBI est prêt à être mis en œuvre sans une bonne définition pour cela.

Réfléchissez-y : Comment le savez-vous *vraiment* ?

Passez-vous tout en revue encore et encore jusqu'à ce que vous décidiez que c'est le moment ?

Quelqu'un décide-t-il sur un coup de tête ?

Cela se produit-il automatiquement au début d'une itération ?

Nous avons vu de nombreuses équipes pousser du travail dans des Sprints qui n'étaient pas du tout prêts à être mis en œuvre simplement parce qu'ils avaient des échéances. Il existe certaines idées persistantes sur Scrum et l'Agile en général qui poussent les gens à faire cela :

- Vous devez obtenir toutes vos exigences pour le Sprint *N* développées dans le Sprint *N-1*
- Vous devriez « simplement commencer » et gérer les problèmes au fur et à mesure

C'est en fait le reflet exact du problème « Comment savez-vous quand c'est terminé ? » [mentionné précédemment](#) et cela a des conséquences similaires. Les gens peuvent attendre trop longtemps pour commencer, parce qu'ils ne savent pas qu'un élément de travail est prêt, et ils peuvent commencer trop tôt parce qu'ils ne savent pas qu'il ne l'est pas.

2.9 : Une A-Team sans état de préparation

Ne pas comprendre ce qu'il faut pour qu'un élément de travail soit prêt a plusieurs effets délétères.

Une façon évidente qu'un incrément de travail peut ne pas être prêt est une Définition du Fini (DoD) incomplète, insuffisante ou manquante. Cela mène à tous les problèmes que nous avons déjà cités qui vont de pair avec l'absence d'une Définition du Fini.

Cependant, ce n'est pas le *seul* aspect de l'état de préparation. Il y a de nombreux autres besoins à satisfaire avant que la mise en œuvre ne commence : l'estimation, l'évaluation des risques et la collecte des données de test ne sont que quelques exemples courants.

Sans connaître et satisfaire ces besoins, un élément de travail peut coûter beaucoup plus cher que nécessaire. Prenons une équipe (l'A-Team) qui dépend d'une API développée par une autre équipe (l'Autre Équipe). Si l'A-Team fait un tas d'hypothèses sur la façon dont l'API de l'Autre Équipe fonctionnera et code selon ces hypothèses, il pourrait y avoir un important retravail lorsqu'ils découvriront que l'Autre Équipe travaillait sur une réalité qui ne correspondait pas aux hypothèses de l'A-Team. En d'autres termes, l'A-Team a tenté sa chance et a raté.

Tout ce retravail découle du fait que l'API n'était pas prête à être utilisée par l'A-Team.

Parfois, une dépendance non satisfaite ne génère pas de retravail, mais, même dans ces cas, elle peut toujours causer des retards. Imaginez si l'A-Team et l'Autre Équipe se sont mises d'accord sur le fonctionnement de l'API et que tout s'est déroulé comme prévu, mais que l'Autre Équipe a simplement pris plus de temps que prévu. En conséquence, l'A-Team n'a tout simplement pas pu tester correctement leur travail au moment où il était censé être terminé et ils ont dû repousser leur échéance.

2.10 : Ne pas prêter attention à la planification et à la disponibilité des ressources

Parfois, les problèmes peuvent être aussi simples que la planification ou les ressources. Certains éléments de travail nécessitent des membres spécifiques de l'équipe. Si ce membre de l'équipe part en vacances dans quelques jours, ce n'est probablement pas le bon moment pour commencer le PBI qui ne peut pas être terminé sans sa participation.

Nous entendons souvent les gens dire que cela ne devrait pas être ainsi, mais c'est souvent le cas, quoi qu'il en soit. Les « personnes interchangeables » sont une utopie.

On peut en dire autant des ressources non humaines. Si vous allez avoir besoin de ressources serveur pour effectuer un test de charge, vous devriez probablement vous assurer que ces ressources seront effectivement disponibles avant de commencer l'élément de travail de test de charge. Sinon, dans le meilleur des cas, vous subirez des retards importants et vous perturberez probablement d'autres équipes/collaborateurs pendant que vous vous efforcerez de provisionner ce dont vous avez besoin.

Un autre mode d'échec lié aux faux départs survient lorsqu'une équipe ne possède pas les compétences requises pour accomplir le travail. Parfois, c'est une question interne - comme

un membre de l'équipe qui doit se former sur un nouveau système ou faire des recherches sur une nouvelle API. D'autres fois, c'est une question de planification, comme lorsque vous devez emprunter un expert UX ou base de données dans un groupe de travailleurs qualifiés. Cela pourrait même être un problème de recrutement où l'équipe a besoin d'un expert et ne peut pas effectivement accomplir certains types de travaux sans lui.

2.11 : Impacts d'autres types de faux départs

Nous avons vu des équipes s'engager à terminer des éléments de travail dans des Sprints et réaliser le codage relativement rapidement, mais être toujours incapables de terminer les tests. Cela, en soi, pourrait ne pas être surprenant, mais la raison est inhabituelle : l'équipe de test avait besoin de quelque chose (comme des données de test) qu'elle n'avait pas collecté avant le début du Sprint, et la collecte de ces données s'est avérée plus difficile ou plus longue que prévu.

En conséquence, les éléments de travail ont dû être reportés au Sprint suivant simplement parce que l'équipe ne s'était pas assurée qu'elle était vraiment prête à les terminer dans le temps imparti avant de commencer.

Les équipes commencent parfois à implémenter des éléments de travail alors qu'elles ont encore des questions en suspens. En fait, beaucoup de gens semblent penser que cela les rend « plus Agile » quand ils font cela.

Cela peut entraîner d'énormes reprises de travail, des surprises ou des retards. Si la réponse à la question ouverte finit par violer une hypothèse qui a été faite, tout le travail basé sur cette hypothèse doit être revu. Si la question ouverte n'obtient pas de réponse au moment où l'élément est censé être clôturé, alors l'élément doit soit être clôturé alors qu'il n'est peut-être pas terminé, soit rester ouvert jusqu'à ce que la question soit résolue.

Il se peut que l'équipe ait une dépendance interne - un défaut qui doit être corrigé, une tâche préalable qui doit être terminée, et cetera. Si cela n'est pas correctement suivi, cela peut causer les mêmes problèmes qu'une dépendance externe non satisfaite, avec la tentation supplémentaire de changer de contexte pour la résoudre.

2.12 : Coûts cumulatifs

Bien sûr, ces problèmes créent des retards, des reprises de travail et des attentes déçues, mais les inconvénients ne s'arrêtent pas là.

En plus du gaspillage dû aux reprises de travail, cela fait généralement prendre du retard aux projets. Si les équipes essaient frénétiquement de clôturer des éléments du backlog sans jamais vraiment savoir ce qu'il faudra pour faire de réels progrès, les choses qui doivent réellement être faites ont tendance à être laissées de côté.

Souvent, mais pas toujours, cela conduit à une pression accrue pour livrer. Lorsque les projets prennent de plus en plus de retard sur le planning, la direction peut tenter de les remettre sur les rails en demandant aux gens d'aller plus vite. Cela se traduit invariablement par des heures de travail plus longues.

Cela, à son tour, tend à éroder la confiance et à aigrier la culture d'une organisation. Les relations qui devraient être collaboratives deviennent antagonistes. Les personnes qui devraient travailler ensemble pour trouver les meilleures solutions et les plus rapides détournent leur énergie pour établir que, lorsque les choses vont inévitablement mal, ce n'était pas leur faute.

Dans la poursuite effrénée des fonctionnalités et des éléments de travail clôturés, les équipes se retrouvent souvent à faire des compromis. Cela signifie réellement qu'elles laissent la qualité (en particulier la qualité du code) se dégrader. Cela signifie, à son tour, qu'elles échangent la productivité future contre l'illusion de progrès dans le présent.

À mesure que les conditions de travail deviennent de plus en plus désagréables, les talents clés commencent à se désengager ou même à chercher ailleurs.

Les organisations qui se comportent ainsi «sacrifient leur avenir», pour ainsi dire, de plus d'une manière. La base de code devient moins maintenable et les personnes qui auraient dû la maintenir sont toutes parties.

S'il y a un avantage, il nous est invisible.

Chapitre 3 : Introduction au Requirements Maturation Flow (RMF)

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.1 : Ce que RMF n'est pas

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.2 : Ce qu'est RMF

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.3 : L'Adoption Incrémentale est Prise en Charge et Recommandée

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.4 : FME 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.5 : FME 2

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

3.6 : FME 3

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 4 : Est-ce Agile ?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

4.1 : « Les individus et les interactions », « Les logiciels opérationnels »

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

4.2 : Collaboration avec le client

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

4.3 : Répondre au changement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

4.4 : Transparence

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

4.5 : S'intègre au processus, cohérent avec Agile

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Partie II : Créer de l'espace pour la préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 5 : La première extension

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

5.1 : Le travail de préparation est du *travail*

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

5.2 : Naturaliser le travail de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

5.3 : Un incident illustratif

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

5.4 : Impact Réciproque

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

5.5 : La fonction du RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 6 : Pourquoi Les Gens Ne Font-Ils Pas Cela ?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.1 : Le Travail de Préparation Comme Citoyen de Seconde Classe

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.2 : Une Allergie au Travail Non Productif

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.3 : Donc Cela A Été Enterré

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.4 : L'Influence de la Gestion de Projet

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.5 : Le Schéma

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.6 : Les Projets et l'Estimation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.7 : Comment les Non-Estimations Influencent le Travail de Préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.8 : Mesurer la Vitesse, Pas la Vitesse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.9 : Mauvaises Mesures, Mauvais Résultats

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

6.10 : Où ne pas chercher le blâme

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 7 : Travail de préparation explicite (RMF 1)

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.1 : Intégration avec le Synapse Framework™

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.2 : Anatomie de RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.3 : Comportement : Réserver de la capacité pour la collaboration

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.4 : Artéfact : L'élément de travail de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.5 : Activité : La réunion de collaboration

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.6 : Comportement : Poursuivre la collaboration jusqu'à l'obtention d'une compréhension partagée

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.7 : Comportement : Toujours confirmer la compréhension partagée

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

7.8 : Comment RMF 1 modifie le flux de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 8 : Effets du RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

8.1 : La vie avant le RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

8.2 : Avant : Le temps passé à comprendre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

8.3 : Après : Temps passé à comprendre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

8.4 : La vie après l'adoption de RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

8.5 : Fondamental

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 9 : Mettre en pratique

RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.1 : Formation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.2 : Exigences minimales par type d'équipe

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.3 : Accord

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.4 : Préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.5 : Phase pilote

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.6 : Déploiement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.7 : Suivi

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.8 : Revendiquer le succès

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.9 : Rester vigilant

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.10 : Qu'en est-il du « Comment » ?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

9.11 : Le Moment de Passer à l'Action !

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Partie III : Contrôle des Travaux Terminés

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 10 : Le Besoin Suivant

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.1 : Marge d'Interprétation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.2 : Réduire la Marge d'Interprétation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.3 : Une Troisième Option : Pas de « Marge de Manœuvre »

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.4 : Impact Potentiel sur l'Achèvement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.5 : Impact Potentiel sur l'Exécution

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.6 : Alternative Proposée : Ne Laissez Aucune Place à la Mauvaise Interprétation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.7 : Avantages

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.8 : Sur les Craintes de la Paralysie par l'Analyse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

10.9 : Le prochain besoin : des définitions du fini sur mesure

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 11 : Ce que les gens font habituellement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.1 : Si c'est si génial, pourquoi les gens ne le font-ils pas ?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.2 : Le pipeline université-coaching

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.3 : La surcharge de coaching

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.4 : Une façon dont les gens gèrent la DoD : ne pas le faire

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.5 : Uniquement les Critères d'Acceptation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.6 : Definition of Done Globale

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.7 : Sans Force Exécutoire

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

11.8 : Résumé : Le Terme « DoD » Est Plus Fréquent que les Véritables Définitions du Fini

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 12 : Définir une Définition du Fini

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.1 : À propos d'un seul élément de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.2 : État d'achèvement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.3 : Précision

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.4 : Structure d'une DoD

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.5 : Spécifications

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.6 : Critères de Sortie Technique

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.7 : Critères d'Acceptation Produit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.8 : Plusieurs Parties, Un Point de Contrôle

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.9 : Exemple

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.10 : Adaptation à votre processus

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

12.11 : Résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 13 : Définition du Fini sur Mesure (RMF 2)

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.1 : Principe : Chaque Élément de Travail est Unique

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.2 : Comportement : Maintenir Un ou Plusieurs Modèles de DoD

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.3 : Activité : Définir le modèle de DoD

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.4 : Maintenir et améliorer le modèle de DoD

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.5 : Plusieurs modèles de DoD

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.6 : Comportement : Utiliser les modèles comme points de départ pour les Définitions du Fini

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.7 : Comportement : S'accorder sur des Définitions du Fini sur mesure

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.8 : Activité : Définir la Definition of Done d'un élément de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.9 : Une autre extension du flux de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.10 : Comportement : Faire mûrir la DoD avant de commencer l'implémentation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.11 : Activité : Analyse hors ligne pour faire mûrir la DoD d'un PBI

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.12 : Ajouter la maturation au flux

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.13 : Comportement : Suivi de l'achèvement dans les éléments de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.14 : Ajout du suivi des progrès

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.15 : Comportement : Conditionner le travail à son achèvement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.16 : Activité : Utiliser la DoD pour déterminer l'achèvement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.17 : Comment s'intègre le contrôle

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

13.18 : En résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 14 : La vie avec RMF 1 & 2

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.1 : Coût

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.2 : Délais

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.3 : Impact sur l'équipe de développement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.4 : Impact sur la Product Owner

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.5 : Impact sur la Direction

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

14.6 : Résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 15 : Installation de RMF 2

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.1 : Implication des parties prenantes

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.2 : Niveau de détail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.3 : Accord de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.4 : Travail initial

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.5 : Déploiement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

15.6 : Résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Partie IV : Mise en œuvre du contrôle d'accès

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 16 : L'Exigence Finale

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.1 : C'était là depuis le début

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.2 : Le Pouvoir du Timing

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.3 : Inverser la Perspective

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.4 : Risques et Coûts

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.5 : La Valeur d'Attendre d'Être Prêt

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.6 : Un Avantage Supplémentaire

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.7 : L'Énoncé du Problème

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.8 : Le besoin

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

16.9 : Résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 17 : Contexte de la Définition du Prêt

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.1 : La Permission de Travailler selon Lean

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.2 : Les Critères d'Entrée des Colonnes Kanban

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.3 : Scrum et autres Apocryphes des Processus Agiles

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.4 : Le surf plutôt que le code

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.5 : Des solutions ponctuelles

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

17.6 : Nous sommes prêts à nous préparer

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 18 : Définir une Définition du Prêt

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.1 : Objectif

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.2 : Sur mesure

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.3 : Anatomie

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.4 : Exemple

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.5 : Accord

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.6 : Contrôle d'accès

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

18.7 : Résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 19 : Définition du Prêt sur Mesure (RMF 3)

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.1 : Une Autre Porte dans le Processus

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.2 : La Structure d'une Définition du Prêt

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.3 : Critères de Sortie Produit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.4 : Critères d'Entrée en Développement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.5 : Pas de Mal à la Redondance

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.6 : Comportement : Maintenir Un ou Plusieurs Modèles de DoR

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.7 : Pourquoi Avoir un Modèle de Définition du Prêt?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.8 : Activité : Définir le Modèle de DoR

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.9 : Maintenir les modèles de DoR au fil du temps

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.10 : Les modèles ne sont qu'un point de départ

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.11 : Comportement : S'accorder sur des Définitions du Prêt sur mesure

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.12 : Activité : Définir la Définition du Prêt d'un élément de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.13 : Comportement : Rendre les éléments prêts avant de commencer la mise en œuvre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.14 : Conditions hors du contrôle de l'équipe

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.15 : Comportement : Suivre la préparation dans les éléments de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.16 : Comportement : Contrôler le travail par la préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.17 : Activité : Utiliser la DoR pour déterminer l'état de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

19.18 : En résumé

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Partie V : Synthèse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 20 : La plupart des échéances n'ont pas d'importance

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.1 : Les véritables échéances

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.2 : Les échéances arbitraires n'ont pas d'importance

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.3 : Les compagnies aériennes

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.4 : L'origine de la dette technique

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.5 : Les échéances du Marketing et des Ventes

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.6 : Échéances de projet

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.7 : Il existe une autre voie

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.8 : Les échéances arbitraires ne sont pas nécessaires

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.9 : Les échéances arbitraires doivent être abolies

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.10 : Les Vrais Délais Restent un Facteur

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.11 : *Pis Aller*¹

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

20.12 : Conclusion

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

¹Une solution de dernier recours, originellement de la langue française.

Chapitre 21 : Compétence 1 : Flux de Maturation des Exigences

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.1 : Tel en Bas, Tel en Haut

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.2 : Principe : Transparence sur Tout le Travail Nécessaire

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.3 : Comportement : La Responsabilité Suit le Travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.4 : Comportement : Suivre Visiblement l'État des Exigences

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.5 : Comportement : Révéler tout le travail associé à la préparation et à la mise en œuvre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.6 : Activité : Travail de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.7 : Comportement : Révéler tout le travail nécessaire pour compléter un élément de travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.8 : Comportement : Privilégier l'état de préparation plutôt que les échéances

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

21.9 : Conclusion

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 22 : Comment le Travail et l'Information Circulent dans Scrum avec RMF

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.1 : Notes Préliminaires

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.2 : Capture et préparation du besoin initial

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.3 : Initiation du travail de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.4 : Planification et exécution du travail de préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.5 : Revue des résultats de la préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

22.6 : Planification et Réalisation de l'Implémentation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 23 : L'Impact du FME

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

23.1 : Le Cycle de Vie d'une Exigence

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

23.2 : Exemple de Flux d'Information et de Travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

23.3 : Avant

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

23.4 : Après

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

23.5 : Avantages

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 24 : Transition vers le FME

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.1 : Modèle d'Adoption

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.2 : Intégration dans un Flux de Travail Scrum

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.3 : Autres Frameworks

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.4 : Résistance Majeure : Les Éléments de Travail de Préparation

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.5 : Le Grand Changement : L'État d'Esprit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.6 : Conseils pour le Changement

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

24.7 : Conclusion

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Chapitre 25 : C'est à Vous

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

25.1 : Récapitulatif

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

25.2 : Maintenant, C'est à Votre Tour

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Partie VI : Ressources

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Annexe A : Scrum n'est pas le problème

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Qu'est-ce que Scrum ?

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Les frameworks

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Scrum traite de la gestion de projet et du travail

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Les Folies de Traiter Scrum Comme un Cadre de Gestion de Produit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Aucun Mécanisme Prescrit pour la Maturation des Exigences

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Pas d'Investissement d'Expertise Technique dans le Développement des Exigences

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Antipatterns

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Extension Nécessaire

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Enrichir Scrum

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Annexe B : Le Framework Synapse™

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Ce que couvre Synapse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Les Trois Maîtrises

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Comment Synapse est adopté

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

La Fusion de Deux Cadres

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

La Structure du Cadre Synapse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Maîtrises Organisationnelles

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Compétences Organisationnelles

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Habitudes Organisationnelles

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Structure du Cadre Synapse

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Anatomie d'une Praxis

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

L'importance de l'ordre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

L'impact de Synapse sur ce livre

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Annexe C : Objections et obstacles courants à RMF 1

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Objections courantes

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Obstacles Courants

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Annexe D : Listes de Critères de Départ pour la DdF

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Liste de Départ des Critères de Sortie Techniques

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Liste de départ des critères d'entrée du produit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Annexe E : Listes de Critères de Départ pour la Définition du Prêt

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Critères de Sortie Produit

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Critères d'entrée en ingénierie

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Critères d'entrée du Sprint

Ce contenu n'est pas disponible dans l'extrait du livre. Le livre peut être acheté sur Leanpub à <https://leanpub.com/ready-fr>.

Index

- Accord de travail, 52
- Agile, 25
- API, 24
- auto-antonymes, 16
- Autres Frameworks, 74
- banques, 10
- Barreras, Tom, vii
- base de code, 26
- base de connaissances, 8
- boucles de rétroaction, 14
- C#/.NET solution, 3
- cadres d'ingénierie, 4
- Cascade, 17
- Central Oregon, vii
- code existant, 12
- Comment savez-vous quand c'est terminé?, 23
- Comportement : Révéler tout le travail associé à la préparation et à la mise en œuvre, 70
- Comportement : Suivre Visiblement l'État des Exigences, 70
- compréhension partagée, 35
- compréhension partagée confirmée, 17
- compter les pièces, 22
- compétences de l'équipe, i
- confiance, 26
- construction de puzzle, 22
- consultants, 8
- contributeurs individuels, 4
- Coûts cumulatifs, 25
- Critères d'Entrée en Développement, 61
- Créer de l'espace pour la préparation, 30
- culture, 10
- Definition of Done, 47
- dette technique, 66
- Development Team, 20
- direction, 10
- doctrines de programmation du milieu du vingtième siècle, 7
- domaine, 10
- défaut, 25
- définir 'terminé', 20
- Définition du Fini, 23
- Définitions du Fini sur mesure, 46
- Définitions du Prêt sur mesure, 61
- Définitions Sur Mesure du Prêt, iii
- Déploiement, 52
- développement en lots importants, 17
- développement logiciel, 16, 22
- Développement piloté par le comportement, iv, 8
- Développement Piloté par PKB, iv
- développeurs de logiciels, 20
- en retard sur le planning, 26
- environnement de développement, 10
- exigence réelle, 20
- expert métier, 7, 11
- Faire mûrir la DoD, 47
- Federated Credit Institution, 2
- finance, 10
- flux, 14
- Flux de Maturation des Exigences, i, iii, iv, 69
- flux de travail, 35, 47
- gaspillage, 26
- Gate in the Process, 61
- hypothèse, 25

- IA, 21
- Implication des parties prenantes, 52
- Intégration dans un Flux de Travail Scrum, 74
- itération, 23
- Kanban, i
- La Banque, 13
- La plupart des échéances n'ont pas
 - d'importance, 66
- langues naturelles, 17
- Le Grand Tournant, 13
- Les données de mauvaise qualité donnent des
 - résultats de mauvaise qualité, 16
- loan repayment portal, 3
- low-code solution, 3
- Marketing et Ventes, 67
- mauvaises exigences, 18
- Mise en œuvre du contrôle d'accès, 54
- modifications en cours de Sprint, 6
- Modèle d'Adoption, 74
- Niveau de détail, 52
- OutSystems, 3
- part de marché, 21
- parties prenantes, 18
- patrons de conception, iv
- personnes interchangeables, 24
- Perte ou corruption de données, 21
- plaquer or, 21
- pratiques d'ingénierie, 10, 16
- Procore, iii
- Product Backlog Item, 11, 13, 15
- Product Backlog Item's Definition of Done, 48
- Product Exit Criteria, 61
- Product Manager, 5
- Product organization, 4
- Product Owner, 5, 10, 20
- Product, rôle de, 4
- Produit et Ingénierie, 15
- Project Management Institute, 2
- prêts*, 13
- qualité du code, 26
- Quelque Chose Manque, 1
- Ready, i, ii
- Requirements Maturation Flow, 27, 71
- responsabilité et traçabilité, 15
- Ressources, 77
- ressources serveur, 24
- retravail, 18
- Revendiquer le succès, 38
- risks, 19
- RMF 1, 31, 82
- RMF 2, 52
- rédaction des exigences, 16
- Résistance Majeure, 74
- rétenion des talents, 26
- révolte style Terminator, 21
- Scrum, 5, 17, 23, 71
- service en amont, 12
- services dont on dépend, 10
- snatching defeat from the jaws of victory, 19
- software development, 19
- Sprint, 11, 16, 20, 23, 25
- Structure of a Definition of Ready, 61
- surcharger, 9
- système, 10
- système hérité, 7
- technique de rédaction des exigences, 10
- techniques de développement logiciel, 7
- Temps d'arrêt système, 21
- temps perdu, 21
- test-development, vii
- Testeurs, 13
- Tracking Doneness, 48
- transformations Agiles, iv
- Transition vers le Flux de Maturation des Exigences, 74
- Travail de préparation explicite, 34
- Travail initial, 52
- travail supplémentaire, 20

United States national financial
infrastructure, 2

Violations réglementaires, 21

Vulnérabilités de sécurité, 21

work item completion, 20

Échéances de projet, 67

Élément du Backlog Produit, 23

échéances, 23

élément de travail, 25

élément de travail de préparation, 34

élément du backlog produit, 8

éléments du backlog, 26

équipe de direction, 7

équipes d'ingénierie, 9, 10