TechTrends #10

Studio : De l'idée à l'autonomie en production

Démarrer, Piloter, Pérenniser

BY ENGINEERS *at* PUBLICIS SAPIENT

publicis
sapient

Les TechTrends sont l'expression de notre savoir-faire ; forgé sur le terrain, auprès de nos clients dans le cadre des projets que nous menons avec eux.

Fruit d'un travail collaboratif de nos consultants, vous y trouverez, nous l'espérons, les nouvelles tendances technologiques et méthodologiques ainsi que l'état de l'art de notre profession.

Nous tentons, dans le cadre de ces publications, de vous dispenser des conseils directement opérationnels afin de vous guider dans les décisions stratégiques que vous avez à prendre.

Distribués à plusieurs milliers d'exemplaires tous les ans, la collection des TechTrends s'étoffe régulièrement de nouveaux ouvrages.

Bonne lecture !

Démarrer

Piloter

Pérenniser

Introduction

Historiquement, les services informatiques, qu'ils soient directement hébergés au sein de l'entreprise, ou bien « détachés » via de la prestation de service, ont été considérés comme des « sous-traitants » du métier. Même si, depuis une décennie, les choses évoluent (« Software is eating the world »), cet état de fait a souvent conduit à une gestion de projet protectionniste, en mode client / fournisseur, le plus souvent, en suivant une méthodologie séquentielle dite « Waterfall » et en multipliant les strates de pilotage (AMOA, AMOE). Cette approche a dilué les responsabilités, jusqu'à ce que la politique prime sur la santé du projet et sur la valeur du produit.

Nous croyons qu'il est possible de mener des projets autrement :

- En réalisant des « **produits** » et non des « **projets** » ;
- En plaçant la **qualité au centre des développements** ;
- En accueillant **favorablement le changement**, convaincus et guidés par les **méthodes agiles** ;
- En maîtrisant et en tirant partie des **nouvelles technologies** ;
- En se plaçant dans une relation de **partenaires**, plutôt que d'exécutant et de donneur d'ordres.

Né en 2012, Studio réalise les produits ambitieux de ses clients. Pour y arriver, nous savons qu'il est nécessaire de :

- Mettre en place des équipes expérimentées, pluridisciplinaires, passionnées de technique et de qualité ;
- Proposer un modèle produit et affronter intelligemment les difficultés dès qu'elles surviennent ;
- Anticiper et préparer le passage du mode projet au mode maintenance.

Ce TechTrends est destiné aux équipes et aux décideurs qui souhaitent adopter une démarche similaire ou profiter de ce partage de connaissances pour améliorer leurs pratiques.

Même si notre expérience découle principalement de produits réalisés pour le compte de nos clients, nous sommes convaincus que les recettes exposées ici sont transposables dans un environnement où la direction métier (donneur d'ordres) et la direction informatique (réalisation) appartiennent à la même entreprise.

Quelques chiffres à retenir :

16
projets actifs en 2019

60
Sapients travaillent sur **des projets Studio** en 2019

100%
des projets utilisent le **contrat agile** ou son principe

9M€
de chiffre d'affaires en 2019

Démarrer son projet

Au sein du Studio, le démarrage d'un projet suit toujours le même rituel de la mise en place de l'atelier de Story Mapping, à l'exécution du Sprint 0, en passant par le kick-off et l'atelier des risques.

L'ensemble de ces étapes garantit :

- D'avoir une vision d'ensemble claire du produit à réaliser ;
- De démarrer par les fonctionnalités qui apportent le plus de valeur au produit ;
- D'identifier individuellement les équipiers et de favoriser les interactions directes ;
- De s'assurer de la bonne compréhension des rôles et responsabilités de chacun ;
- D'identifier au plus tôt les risques majeurs du projet, pour pouvoir les traiter ;
- De mettre en place les outils et pratiques d'intégration et de déploiement continu ;

En somme, de démarrer au plus tôt et d'impulser un rythme optimal.

Définir la vision produit

Avant même le démarrage du projet, vous aurez sûrement cherché à clarifier le positionnement exact de votre produit par rapport au marché, vos utilisateurs cibles, son financement, etc. Pour ce faire, l'utilisation d'un **Lean Canvas** peut être appropriée et nous vous invitons à en découvrir ses grands principes dans notre ouvrage : *Product Academy - Le guide des Product Managers et des Product Owners d'élite*[1].

Connaître la vision produit et la partager régulièrement avec l'équipe de réalisation est un facteur clé de réussite d'un projet. Pensez à lui présenter ce Lean Canvas de temps à autre, ce qui l'aidera à mieux saisir les tenants et aboutissants de votre produit, pour une meilleure implication sur le long terme.

Travailler son Story Mapping

L'atelier de Story Mapping consiste à dégager **les étapes clés des parcours utilisateurs**, en se focalisant sur les personas identifiées (i.e : des archétypes d'utilisateurs cibles) et à les optimiser pour influencer positivement l'usage du produit.

Durant cet atelier, qui aboutit à une **User Story Map**, il est important que toutes les parties prenantes soient présentes : marketing, métier, design, technique, etc. Cela permet d'une part de bien prendre en compte les attentes et les contraintes de chacun, et d'autre part d'obtenir un consensus et un engagement sur la Roadmap. Nous vous recommandons de ne pas faire durer cet atelier plus d'une demie-journée au démarrage du projet. Le Story Mapping sera régulièrement mis à jour avec le Product Owner durant la réalisation du projet.

Pour plus d'informations sur cet atelier de Story Mapping, nous renouvelons notre invitation à consulter notre ouvrage : *Product Academy - Le guide des Product Managers et des Product Owners d'élite.*

D'un point de vue opérationnel, et avec le recul, nous n'envisageons plus de commencer un projet Studio sans cet atelier tant il est **structurant pour le produit et le dimensionnement de l'équipe**. En effet, celui-ci permet :

- De faire émerger au plus tôt un premier backlog priorisé pour l'ensemble du produit, donnant une vision d'ensemble de la Roadmap et des releases majeures ;
- De dimensionner l'équipe à mettre en place au démarrage du projet après un atelier de Magic Estimation* ;
- D'identifier le « Minimum Viable Product » (MVP) ;
- De donner une première estimation budgétaire en déterminant le nombre minimal d'itérations nécessaires (Sprints).

L'ensemble de ces éléments est ensuite mis à jour Sprint par Sprint, afin d'aider à arbitrer si nécessaire sur une réduction du périmètre pour tenir les délais, une augmentation du budget ou un décalage de release pour couvrir plus de fonctionnalités produit.

1. Disponible en téléchargement sur publicissapient.fr/services/engineering

* Focus sur l'atelier de Magic Estimation

Au début d'un projet ou avant même de le démarrer, il est naturel d'avoir besoin d'une idée, même imprécise, de l'effort qu'il représente.

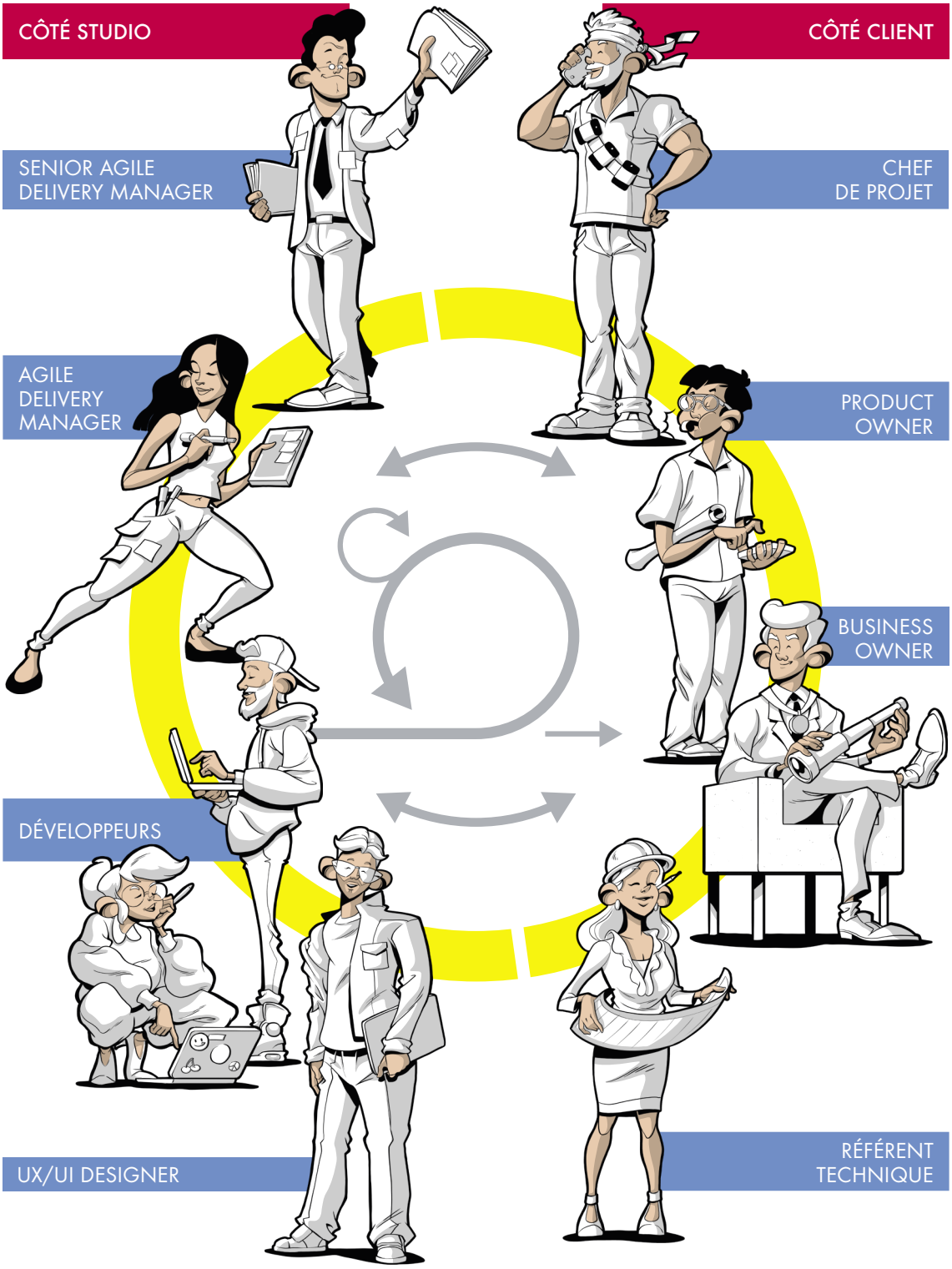
Ce type d'estimation est un exercice très difficile : le besoin n'est pas encore précisément défini, il va évoluer dans le temps et nous ne pouvons pas présumer des difficultés que l'équipe va rencontrer. Il convient d'y consacrer le temps suffisant pour obtenir un livrable qui sera perfectible.

Magic Estimation (ou Extreme Quotation) est un atelier qui permet de répondre à ce besoin : estimer l'intégralité d'un backlog dans une durée courte, environ 2 heures. Il permet de conserver une estimation comparative et collaborative sans pour autant se lancer dans un planning poker intégral qui serait long, fastidieux et pas forcément plus précis à cette étape du projet.

Constituer et organiser son équipe

À ce stade, vous avez donc une première idée de l'effort à fournir pour mener à bien votre projet et il est temps de constituer votre équipe et proposer une organisation. Ci-dessous, quelques éléments issus de notre expérience au Studio.

" L'équipe de développement doit être pluridisciplinaire et dispose de l'ensemble des compétences techniques nécessaires. "



Rôles et organisation d'une équipe sur un projet Studio

Côté équipe de réalisation (Studio)

L'équipe de développement

L'équipe de développement doit être **pluridisciplinaire** et dispose de l'ensemble des **compétences techniques** nécessaires à la bonne réalisation du produit dans son ensemble. Vous vous attacherez également à construire une équipe **mixte en terme de séniorité** :

- La pluridisciplinarité permettra de faciliter l'entraide entre coéquipiers et également le remplacement ou le renfort en cas de « rush » ou de congés ;
- Mixer la séniorité facilitera généralement le partage d'expérience, et instaurera une dynamique vertueuse sur le moyen / long terme.

Plus classiquement, si vous êtes familier avec Scrum, l'équipe sera autonome dans ses choix techniques tout en assumant la responsabilité de la qualité des développements qu'elle produit.

L'Agile Delivery Manager (ADM)

L'ADM est un rôle hybride - pierre angulaire du dispositif - que nous avons fait émerger au fil des projets et des situations que nous avons rencontrées. Il est le **Scrum Master de l'équipe.** Il est également le représentant du Product Owner du client / donneur d'ordres au sein de l'équipe (**Proxy PO)** et assure le **pilotage du projet** éventuellement accompagné du Senior Agile Delivery Manager (SADM). Il est notamment responsable :

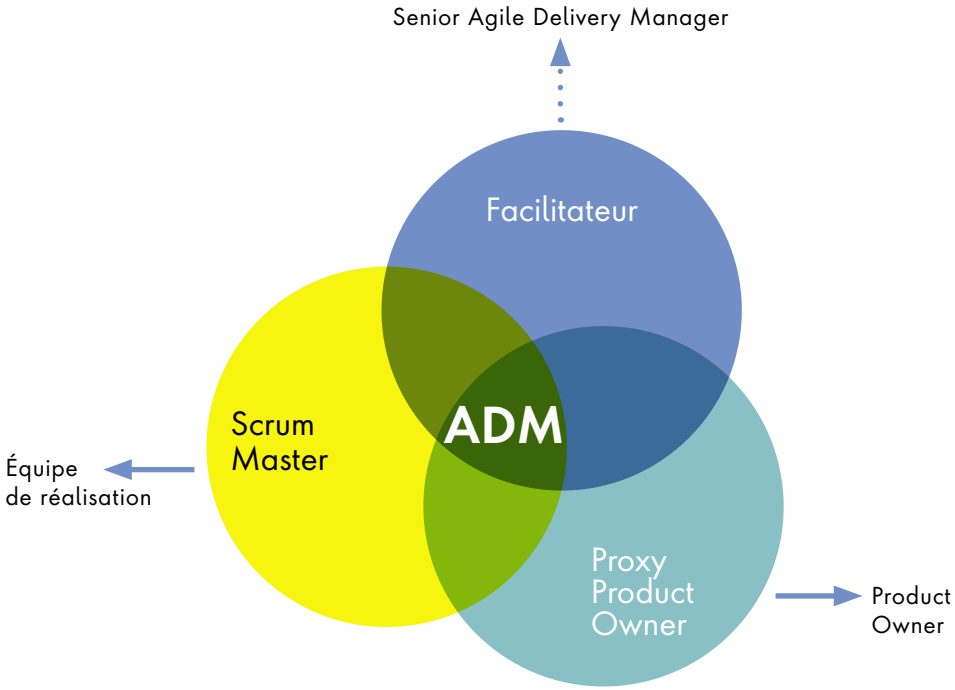
- De garantir le cadre agile du projet et la fluidité des différentes cérémonies, de faciliter les interactions entre l'ensemble des parties prenantes ;
- D'aider voire même suppléer le Product Owner dans la gestion de son Back-log (rédaction des User Stories, priorisation, etc.) ;
- De fournir, en se basant sur les indicateurs de prédictibilité et d'avancement, une date actualisée, Sprint par Sprint, des prochains jalons ;
- De suivre les indicateurs de qualité fonctionnelle et technique ;
- De rédiger les éléments des comités de pilotage / kick-off (avec le SADM si nécessaire) ;
- D'être un catalyseur de l'amélioration continue au sein de l'équipe.

Les plus agilistes d'entre vous trouveront sans doute surprenant ce rôle multifacettes attribué à l'ADM, à la fois **Scrum Master et Proxy Product Owner**. Ce dernier œuvre pour « protéger » l'équipe, tout en défendant les « intérêts » de son client / donneur d'ordres.

En France, il arrive fréquemment de rencontrer des Product Owners qui débutent dans l'Agilité, n'ont pas encore acquis l'autorité requise pour arbitrer sur leur produit, n'accordent pas suffisamment de temps à la bonne définition ou au bon suivi de ce dernier. L'ADM est là pour palier ce manque. Ce rôle est clé pour :

- **Faciliter et optimiser l'implication** de toutes les parties prenantes ;
- **Provoquer les prises de décision au plus tôt**, dans la synthèse des attentes de chacun ;
- **Expliciter le formalisme attendu** pour les livrables, de part et d'autre ;
- **Fournir de manière itérative la date de livraison des prochains jalons,** tout en conservant l'exigence de qualité fixée.

Bien sûr, cela nécessite le recul nécessaire et une expérience significative dans la réalisation de projets agiles. En général, il travaille sur 1 à 2 projets.



Le Senior Agile Delivery Manager (SADM)

Selon le produit à développer, la taille de l'équipe (ou des équipes) qui le développe et le contexte de notre client, nous veillons à proposer le dispositif de pilotage nécessaire et suffisant. Concrètement, pour une équipe réduite qui travaille dans un contexte favorable, nous préférons confier l'intégralité du cadre que nous décrivons ici à une seule personne, l'Agile Delivery Manager. Dans ce cas, l'ADM couvre également le rôle et les responsabilités du SADM.

Pour des projets où l'équipe est conséquente (plus de 5 développeurs sans dépasser 8), où le produit met en oeuvre plusieurs équipes ou dans un contexte qui nécessite d'avoir plus de hauteur qu'une personne impliquée à plein temps sur le sujet, nous pilotons nos projets en binôme. Au-delà d'une situation qui pourrait vous sembler classiques (Chef de Projet / Directeur de Projet), cela permet :

- D'appeler à l'aide un collègue senior pour régler au plus vite des situations délicates ;
- De solliciter un pair pour faciliter les cérémonies et ainsi proposer à nos collègues des approches différentes tout en étant alignées ;
- De prendre des décisions fortes avec l'ADM, l'équipe et le client pour gérer les aléas rencontrés durant la vie du projet ;
- Parfois, de faire porter un message difficile par une personne qui n'est pas impliquée au quotidien dans le projet.

Également, il assure le bon déroulement des projets Studio sur le plan opérationnel, contractuel et financier :

- Il facilite et garantit la mise en place d'un cadre agile satisfaisant, et l'adoption de bonnes pratiques de développement, avec l'aide de l'ADM ;
- Il est responsable de la mise en place et de la tenue du kick-off (réunion de lancement du projet), puis, périodiquement, du COPIL (Comité de PILotage) ;
- Plus classiquement, il gère les aspects administratifs et financiers (contrat, facturation, etc.).

Dans votre organisation, il pourrait s'agir d'un Directeur de Projet qui dispose des connaissances théoriques et pratiques du Craftsmanship et des méthodes agiles. Au studio, il ne gère pas plus de 3 projets.

Au-delà d'un pur dispositif de pilotage transparent et d'un cadre de projet assurément agile, nous aimons accompagner nos client vers l'agilité, en les incubant dans nos équipes. En montrant par l'exemple, nous formons des Product Owners et nous accompagnons les autres parties-prenantes. Il arrive que nos projets / produits deviennent source d'inspiration sur les plateaux de développement client que nous côtoyons.

Côté client / donneur d'ordres

Côté client / donneur d'ordres, les rôles sont plus classiques. Nous en rappelons néanmoins les grands principes.

Product Owner

Le Product Owner formalise et **priorise les fonctionnalités** à développer. Pour l'équipe, il incarne la vision produit et est responsable de sa bonne transmission. En cas de problèmes (technique, timing, etc.), il a le pouvoir et le devoir de choisir la meilleure orientation pour son produit parmi les options proposées par l'ensemble de l'équipe. Avec l'Agile Delivery Manager, il est responsable de la bonne transposition des fonctionnalités sous forme de User Stories et de tests d'acceptation.

Le Chef de Projet

Le Chef de Projet gère les aspects contractuels, facturation, planning et l'ensemble des aspects opérationnels côté donneur d'ordres. Il a une bonne connaissance de la société et met en relation facilement les bons interlocuteurs avec l'équipe pour la résolution des problèmes.

Référents techniques

Idéalement, les référents techniques sont en **relation directe avec l'équipe de développement** et aident à valider les choix techniques en conformité avec les règles et contraintes du système d'informations cible. Ils sont disponibles de manière régulière en début de projet et à la demande par la suite.

Organiser un atelier des risques

Démarrer un nouveau projet, c'est se préparer au meilleur tout en prévoyant le pire. L'atelier des risques sert à identifier au plus tôt les problèmes potentiels que pourrait rencontrer l'équipe, en pondérant ces derniers d'une sévérité, et d'une probabilité d'occurrence, pour établir un **plan d'actions ciblé, réaliste et suivi**.

Idéalement organisé en début de projet, cet atelier cible un large panel de participants (marketing, métier, design, technique, etc.), réunit une dizaine de personnes maximum, suit un format *timeboxé* d'une heure, et ne nécessite que quelques post-it et stylos. L'atelier se déroule en cinq étapes, d'environ 10 minutes chacune.

Durant la première étape, chaque participant **note en silence les risques qu'il identifie**, un par post-it. Tout est éligible : l'organisation projet, la technique, le produit, les intervenants, etc. À l'issue de cette étape, chacun présente ses idées au reste du groupe.

La deuxième étape est l'occasion pour l'animateur de **regrouper les idées similaires**, en intitulés synthétiques, que le groupe valide. Ces intitulés forment la première colonne du tableau d'analyse des risques.

Durant la troisième étape, l'animateur demande au groupe de **déterminer les impacts projet** pour chacun des risques identifiés (par exemple dépassement budgétaire, date de livraison retardée, réduction du périmètre fonctionnel, etc.).
Ces éléments viennent alimenter la deuxième colonne du tableau.

Quatrième étape, pour chacun des risques / impacts identifiés, les participants :

- Évaluent de 1 à 3 la probabilité que le risque se réalise (1=faible, 2=moyenne, 3=forte) ;
- Notent de 1 à 3 la gravité des impacts (échelle identique).

L'animateur renseigne dans la troisième colonne **la dangerosité de chaque risque** (dangerosité = probabilité x gravité).

Finalement, la cinquième étape permet de se concentrer sur le plan d'actions à tenir pour limiter les risques considérés comme les plus « dangereux ». L'animateur note les stratégies imaginées dans la quatrième colonne, et obtient un **engagement des personnes désignées** comme responsable sur le suivi de chacune des actions et le respect des deadlines associées.

Quelques clés pour réussir vos ateliers

Personnes présentes : ces ateliers sont menés collectivement par une équipe représentant l'ensemble des spécialités (produit, création, développement, recette, production, etc.). Chaque profil pose des questions, qui permettront de faire avancer le projet dans sa globalité. Bénéfice supplémentaire, les membres de l'équipe apprennent à se connaître et à interagir ensemble.

La méthode : il est indispensable d'énoncer en début d'atelier son objectif (incluant les livrables éventuels), le déroulement de ce dernier, les grandes étapes, le timing associé. Ensuite, canaliser les discussions avec discernement, afin de bien timeboxer l'ensemble de l'atelier.

Démarrer le développement par le Sprint 0

La phase de développement démarre par une étape un peu particulière : le Sprint 0.

Ce dernier a pour objectif de **préparer le backlog, les outils et le cadre agile,** afin de permettre à l'équipe d'enchaîner efficacement les Sprints de développement à venir.

Sur le plan technique, l'équipe **définit la base de l'architecture logicielle**, s'accorde sur les outils de test et les met en place. Elle construit également son usine logicielle dans l'optique d'un déploiement en continu (i.e : pousser un « Hello World » du développement à la production).

Sur le plan produit, l'Agile Delivery Manager et le Product Owner rédigent les premières User Stories priorisées dans la User Story Map. La description de ces US pourra suivre le formalisme **BDD (Behaviour Driven Development)**, sans que cela soit une obligation.

Sur le plan méthodologique, le Sprint 0 permet la mise en place des premières briques du cadre agile : planification des cérémonies, travail sur la DoD (Definition of Done) et la DoR (Definition of Ready), mise en place du management visuel, etc.

Bien contractualiser

Fixer un cadre vertueux, se donner des objectifs ambitieux aussi bien sur le plan technologique que sur celui du Time-To-Market, embrasser sans complexe les méthodes agiles ne doit pas nous faire oublier l'importance d'une contractualisation **adaptée** et **équilibrée**. Avec le temps, nous nous sommes construits quelques certitudes sur ce point.

Pas de « Forfait » ...

Contractualiser « au forfait » ne renseigne ni sur le processus de développement, ni sur les pratiques d'ingénierie utilisées dans la réalisation du projet. Dans l'esprit des contractants, cela permet essentiellement de fixer les contours exacts de leur relation commerciale en termes de coûts, délais et périmètre.

Il est rare que le périmètre soit clairement défini et maîtrisé au début du projet. Il est aussi rare - et c'est souhaitable - que ce dernier ne doive pas évoluer au cours de la réalisation du produit pour s'adapter à un marché qui, lui, évolue.

La contractualisation au forfait empêche en fait de **se concentrer sur ce qui est vraiment important au final : le produit**, sa pertinence et son timing. Elle n'accepte pas favorablement le changement, n'encourage pas à produire de la qualité, ne priorise pas le travail à effectuer et distrait l'équipe réalisatrice qui se concentre davantage sur les clauses contractuelles, que sur l'excellence de la réalisation même.

Une contractualisation au forfait n'incite pas non plus à donner une visibilité régulière sur l'avancement des développements, masquant d'éventuels risques importants tout au long du projet (périmètre, qualité des développements, budget, etc.).

Nous pensons qu'un mode de contractualisation plus équilibré, tout en restant engageant, adapté aux méthodes agiles et à un développement de qualité, c'est-à-dire durable, est possible.

... Mais le Contrat Agile

La contractualisation agile est un sujet brûlant. L'adoption massive des méthodes agiles a en effet forcé les fournisseurs de service d'un côté et les Directions Achats de l'autre à se poser la question d'un cadre contractuel ad-hoc. Ce cadre peut aussi bien sûr se décliner en dehors de toute relation commerciale, à l'intérieur d'une même entreprise. Il est la preuve de l'engagement de l'équipe envers son donneur d'ordres, et réciproquement.

Pourquoi un contrat Agile ?

Porteuses de grandes promesses, les méthodes agiles impliquent des droits et des devoirs pour les deux parties, qu'il est nécessaire de consigner dans un cadre légal adéquat.

En 2011, Publicis Sapient Engineering (anciennement Xebia) a offert à la communauté agile un document de référence, fruit de notre expérience et d'un travail collaboratif auquel a été associé un cabinet d'avocats spécialisés. **Ce contrat est disponible en Creative commons (CC), à l'adresse suivante : http://www.contrat-agile.org**

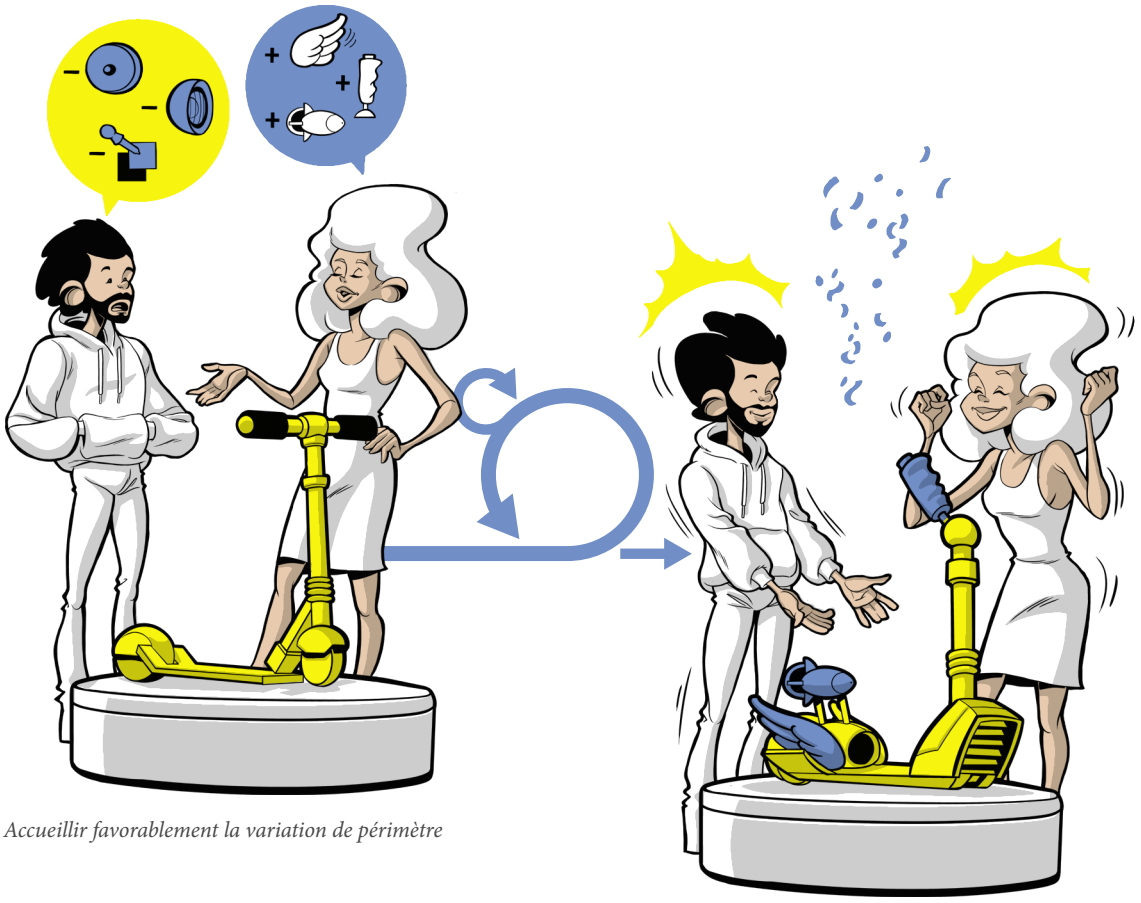
Aujourd'hui, il est utilisé dans l'ensemble des secteurs, que ce soit dans l'e-commerce, le luxe, la banque / finance, l'industrie, les médias, etc. 100% des projets Studio sont régis par ce contrat ou par les grands principes de ce dernier.

Quelques principes du contrat Agile

Sans être exhaustif, rappelons quelques grands principes contenus dans ce contrat Agile.

L'esprit du contrat tout d'abord : il se base sur l'acceptation des deux parties d'utiliser les méthodes agiles pour la réalisation du produit concerné, dans un rapport équilibré (une relation de partenaires, plutôt que de client / fournisseur).

Accueillir favorablement le changement en donnant un cadre contractuel compatible avec la flexibilité du périmètre. La variation du périmètre suit un mécanisme de « trade-in / trade-out » : il est possible de rajouter une fonctionnalité dans le périmètre, dans la mesure où l'on retire des fonctionnalités dont la somme est équivalente en nombre de points de complexité.



Accueillir favorablement la variation de périmètre

Une souplesse contractuelle permettant la rupture du contrat sous un mois, sans motif. Des points sont effectués régulièrement afin d'arbitrer si l'on souhaite continuer des incréments de Sprint ou si la valeur du produit atteinte est suffisante.

Pas d'engagement au forfait (budget, délai, à périmètre fixe) mais un engagement sur un certain nombre d'indicateurs, fixés après une période d'étalonnage (au moins trois Sprints en général) :

- **Qualité technique :** a minima, couverture de code (non généré) par les tests unitaires et complexité cyclomatique.
- **Qualité fonctionnelle :** nombre d'anomalies (y compris régressions) découvertes après la livraison, rapporté au nombre de fonctionnalités développées durant l'incrément produit.
- **Prédictibilité :** nombre de fonctionnalités démontrées en fin d'itération par rapport au nombre de fonctionnalités prévues initialement.
- **Capacité de l'équipe :** nombre de « Story Points » qui ont été implémentés rapporté à la charge de l'équipe du Sprint.

Et le suivi d'indicateurs plus subjectifs, mais tout aussi essentiels :

- **Satisfaction du client :** une note de 0 à 5 indiquant la satisfaction du client pour le travail réalisé par l'équipe, à chaque Sprint.
- **Satisfaction de l'équipe :** une note de 0 à 5 indiquant la satisfaction de l'équipe pour la collaboration menée avec son client, à chaque Sprint.

La transparence incite chacune des parties à partager ses réussites, comme ses difficultés, au plus tôt.

*" **L'esprit du contrat :** il se base sur l'acceptation des deux parties d'utiliser les méthodes agiles pour la réalisation du produit concerné, dans un rapport équilibré (une relation de partenaires, plutôt que de client / fournisseur). "*

Take away

Démarrer son projet

Préparer

Préparer les développements en exposant clairement la vision produit et les différents jalons du projet à l'aide d'un atelier de Story Mapping.

Constituer

Constituer une équipe efficace côté développement comme côté produit, et la guider au plus tôt, en identifiant les personnes clés comme l'Agile Delivery Manager et, si nécessaire, le Senior Agile Delivery Manager.

Contractualiser

Contractualiser dans l'intérêt du produit, en trouvant le juste équilibre entre vos engagements et ceux de l'équipe de réalisation.

Piloter son projet

Vous avez clarifié votre vision produit et l'avez partagée avec l'équipe, vous avez mis en place les premières briques techniques et l'usine logicielle, tout en posant des bases contractuelles équilibrées, il est temps de passer à la réalisation.

Notre méthodologie Projet

Avec le temps, nous avons organisé l'ensemble de nos projets Studio en **Scrumban**[2] : l'équipe de développement travaille en Scrum, tandis que nous suivons les tâches amonts et avals sous forme de flux. Pour le développement, en **Scrum**, nous vous invitons à le découvrir en détails sur le guide développé et maintenu par Ken Schwaber et Jeff Sutherland[3].

Concernant le travail amont et aval du développement (conception de maquettes, rédaction des premières User Stories, préparation des environnements de production, etc.), il est parfois difficile d'utiliser **Scrum** « à tous les étages ». En effet, en réalisant une dizaine de projets par an, pour une dizaine de clients, nous devons nous adapter à l'hétérogénéité de leurs connaissances agiles, s'inscrire dans leur modèle organisationnel et acter qu'un travail préparatoire et / ou une organisation adaptée seront nécessaires. Ce travail va être organisé en flux et suivi par l'Agile Delivery Manager.

2. Scrumban : Initialement, cette méthode a été pensée pour faciliter la transition entre Scrum et Kanban. Dans les faits, il est courant d'arriver à une hybridation de ces 2 méthodes.

3. http://www.scrumguides.org

Quelques exemples de mise en oeuvre d'un flux de travail, en parallèle de l'organisation des développements en Scrum.

Il nous arrive fréquemment de rencontrer des Product Owners débutants ou peu à l'aise avec l'agilité. Nous allons les accompagner dans la rédaction de leurs User Stories (le format attendu, la granularité, l'anticipation, etc.) et les aider à organiser leur travail de co-conception avec l'UX et l'UI Designer. Typiquement, ce travail est réalisé sous forme de flux, en amont du développement, avec des points d'étapes s'assurant de la qualité des stories produites et de la cadence de leur rédaction.

Un autre exemple concerne les projets Big Data, pour lesquels il faut préparer l'ingestion et le stockage des données en amont de la réalisation des Use Cases métiers. La structure des données, leur accès limité / sécurisé, les contraintes d'infrastructure sont des tâches candidates à ce type de modélisation sous forme de flux.

En aval des développements, il arrive que le rythme de validation, par le Product Owner, des User Stories développées dans un Sprint soit insuffisant, engendrant parfois des retards de « conformité » sur deux ou trois sprint. Dans ce cas, nous mettons en place un flux dédié, en parallèle de la réalisation, qui permet de rappeler l'importance de cette validation en continu, et de mieux la suivre.

Organiser son Comité de Pilotage

Nous cultivons une grande transparence et les cérémonies agiles comme la revue d'itération et la rétrospective sont des moments privilégiés dans la relation client. Cela permet notamment de faire un point d'ensemble sur notre mode de fonctionnement, notre organisation et nos difficultés comme sur nos axes concrets d'amélioration.

Néanmoins, l'expérience montre qu'il est nécessaire de sacraliser et de formaliser un temps dédié au suivi global de l'exécution du projet : le Comité de Pilotage. Ce dernier, plutôt classique, est l'occasion de revenir sur quelques points clés du projet, tout en assurant une bonne communication du travail effectué aux parties prenantes, moins présentes au quotidien.

Mensuel, il aborde classiquement les points suivants :

- **Statut du projet :** son avancement, les dernières livraisons, les prochaines échéances à court et moyen terme ;
- **Gestion des risques :** suivi du plan d'actions établi durant l'atelier des risques ;
- **Projections :** présentation du contenu et de la date des prochaines releases en prenant en compte l'avancement et la vélocité de l'équipe ;
- **Indicateurs qualité :** présentation des indicateurs de qualité technique et fonctionnelle ;
- **Facturation :** suivi de la facturation et des règlements.

" Nous cultivons une grande transparence et les cérémonies agiles comme la démonstration et la rétrospective sont des moments privilégiés dans la relation client. "

Piloter son projet par le budget / planning

" Nous fournissons en continu des projections qui prennent en compte la capacité à avancer de l'équipe, de manière réaliste. "

Nous ne pratiquons pas les méthodes agiles pour « satisfaire » les développeurs ou nous soustraire aux contraintes budgétaires et de planning. Au contraire, nous prenons très au sérieux ces aspects, en nous attachant à délivrer de manière optimale la valeur produite.

Comme nous l'avons vu précédemment, l'atelier de Story Mapping permet d'avoir au plus tôt une première estimation du budget et du planning pour le produit. Sprint par Sprint, nous continuons à nous approprier le contexte organisationnel de notre client, son domaine fonctionnel, les contraintes liées à son système d'informations, ce qui nous permet d'affiner ces estimations. Nous fournissons ainsi en continu des projections qui prennent en compte la capacité à avancer de l'équipe, de manière réaliste.

En cas de difficultés, par exemple parce que les projections mènent à une date de livraison trop tardive, ou encore à un coût trop important, nous re-priorisons au plus tôt le travail restant. Plusieurs méthodes pour ce faire :

- **Réduction du périmètre :** le Product Owner accepte de réduire le périmètre fonctionnel attendu pour la prochaine release.
- **Simplification :** l'équipe de développement et le Product Owner cherchent à simplifier un certain nombre de User Stories : ne pas développer certaines animations, réduire le nombre de filtres de recherche, etc.

Maîtriser la qualité de son produit, de son pilotage, et le bien-être de ses équipes

Une des valeurs de Publicis Sapient Engineering (anciennement Xebia), et de tout Software Craftsman, est la qualité sans compromis. La qualité des produits que nous réalisons passe par la mise en œuvre de l'ensemble des pratiques Craft. Si vous souhaitez en savoir plus, nous vous recommandons notre Techtrends Craftsmanship[4].

Concernant les pratiques, nous introduisons régulièrement dans le Sprint des stories dédiées à la réduction de la **dette technique**[5]. À ce titre, les projets tiennent à jour un backlog technique. Ce dernier permet de suivre au quotidien le niveau de la dette. Notre parti pris est **d'injecter au fil de l'eau ces stories techniques**, plutôt que de leur dédier des Sprints complets. L'idée est de conserver une production en continu de nouvelles fonctionnalités tout en **gardant un niveau de dette technique acceptable**. Selon les priorités fonctionnelles, nous estimons qu'introduire entre 10 et 20% de ces stories techniques dans un Sprint est raisonnable.

La **documentation** est un point qui revient également souvent. Nous croyons en une documentation légère, la plupart du temps portée par le code et les commentaires. Celle-ci est complétée par la mise en place de tests exhaustifs, bien nommés, et mis à jour régulièrement au gré de l'évolution du produit. Enfin, elle est aussi modélisée par un formalisme rigoureux des User Stories.

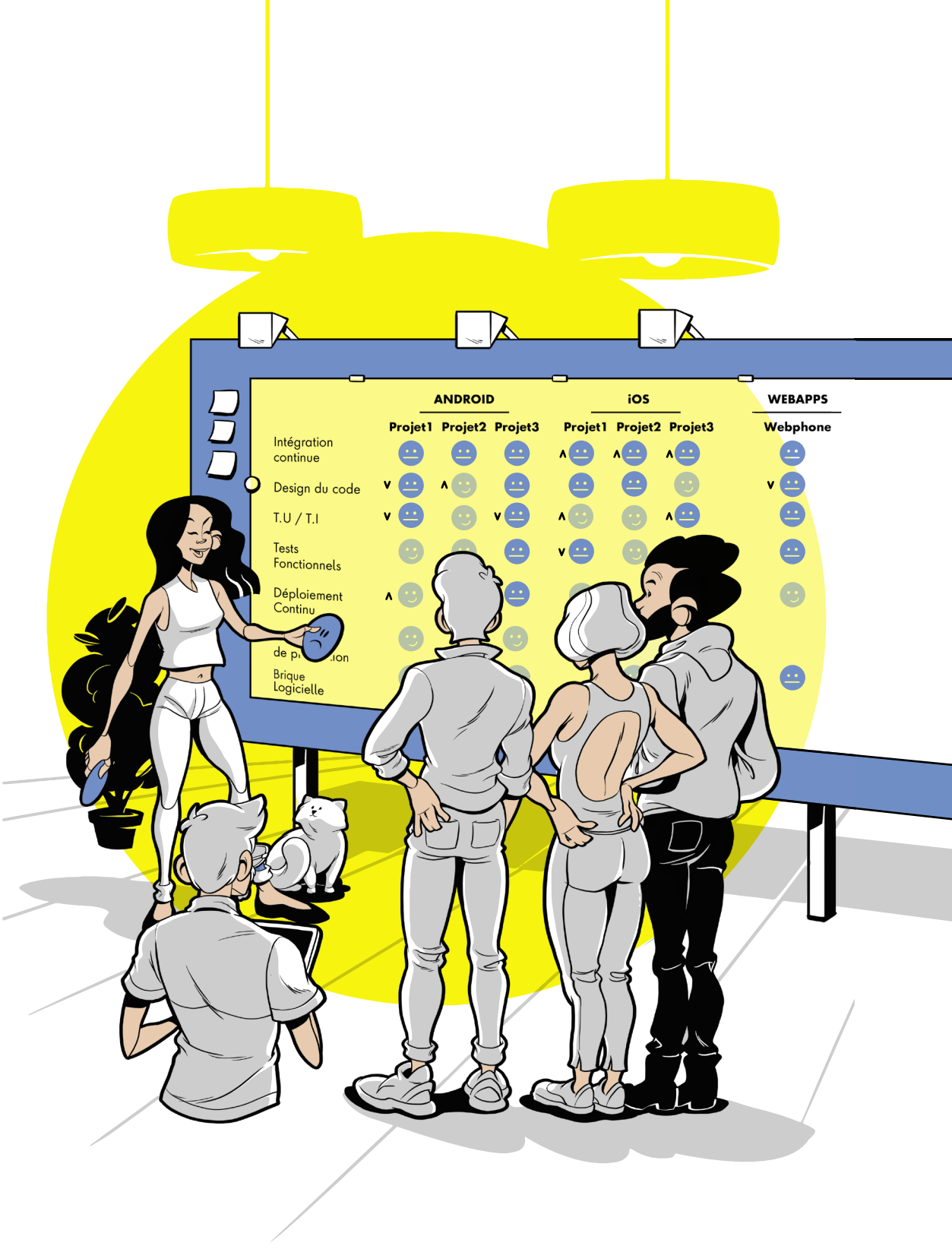
Au delà de la rétrospective de Sprint, nous mettons en oeuvre une rétrospective dédiée à la qualité : **Le Quality Health Check (QHC)** fortement inspiré du « Squad Health Check »[6] d'Henrik Kniberg chez Spotify. Ce dernier, effectué tous les 2 à 3 mois, permet de prendre du recul sur nos pratiques de développement et aboutit à un plan d'action complémentaire aux points d'améliorations soulevés Sprint par Sprint.

Cette liste de critères est mise à jour régulièrement pour s'adapter aux situations rencontrées sur le terrain.

4. publicissapient.fr/services/engineering

5. http://blog.engineering.publicissapient.fr/2011/09/30/livre-blanc-maitrisez-votre-dette-technique/ (https://bit.ly/2lxRTzb)

6. https://labs.spotify.com/2014/09/16/squad-health-check-model/ (https://bit.ly/1qbKnPt)



Le Quality Health Check

Évaluer régulièrement la satisfaction des équipiers : un vecteur d'efficacité

Nous ne nous contentons pas de suivre l'évolution d'indicateurs rationnels (tels la vélocité) pour nous assurer du bon fonctionnement de l'équipe. À l'occasion de la rétrospective, nous avons l'habitude d'évaluer subjectivement le bien-être de toutes les parties prenantes.

Le facilitateur (en général l'ADM) demande à chacun de noter de 1 à 5 sa satisfaction personnelle sur le Sprint qui vient de se terminer. En tour de table, chacun illustre cette note de son ressenti conjoncturel sur le projet. Idéalement, cela se complète par une question du type « Qu'aurait-il fallu faire pour donner 1 point de plus ? » (une source d'inspiration provenant du Perfection Game).

Creuser ces questions, et ensuite y apporter des réponses concrètes, permet d'identifier des axes de progression « de l'intérieur ». Cela apporte un double bénéfice :

- Ce sont les équipiers qui trouvent collectivement des solutions à leurs problèmes, et les exposent aux décideurs qui participent à la rétrospective ;
- La motivation individuelle et globale de l'équipe s'améliore, car les problèmes sont pris en compte et traités au plus tôt. Cela peut avoir un impact très concret sur l'efficacité de l'équipe.

Un suivi homogène des projets avec le Studio Health Check

D'année en année, nous avons souhaité nous améliorer pour permettre plus de cohérence dans les pratiques et outils mis en oeuvre sur les projets Studio. Nous cherchons également à suivre ces derniers facilement, selon quelques critères essentiels. Enfin, nous voulons aider les nouvelles équipes projet à s'organiser simplement et rapidement autour de bonnes pratiques issues de notre expérience, tout en laissant la part de souplesse nécessaire à la bonne mise en oeuvre de l'agilité.

Dans ce but, nous avons créé le Studio Health Check, également inspiré du Squad Health Check. À l'occasion d'un point projet spécifique, tous les 2 mois, les projets évaluent leur maturité selon les critères suivants :

Critères	Bon	Quelques problèmes	Mauvais
Bien-être équipe	Le rythme de l'équipe est soutenu et soutenable. L'équipe s'enrichit. intellectuellement (techniquement, fonctionnellement, humainement etc.).	Rythme supportable mais sera un problème sur la durée. L'équipe apprend mais ne se sent pas challengée.	Le rythme est problématique. L'équipe n'apprend plus.
Environnement de travail	L'équipe aime le produit sur lequel elle travaille. L'équipe est pleinement satisfaite des conditions de travail (mobiliers, locaux, fournitures).	Pas de réel attrait pour le produit. Travail gêné par les conditions de travail.	L'équipe n'aime pas le produit. Travail régulièrement pénalisé par les conditions.
Cadre agile	Les artefacts sont suivis. Les cérémonies ont tout le temps lieu. Les rôles sont identifiés et sans équivoque.	Artefacts suivis sporadiquement, quand on y pense. Les cérémonies ont lieu à quelques exceptions près. Les rôles pas clairs.	Artefacts non suivis. Les cérémonies ont lieu de temps en temps, quand on a le temps. Les rôles posent problème : il y a de l'overlap et/ou des trous dans la raquette.
Qualité du backlog	Le Backlog est peuplé et détaillé pour occuper 2 sprints minimum.	Le Backlog se constitue au dernier moment, il arrive de commencer le sprint avec des stories non prêtes. Nous avons une vague idée du contenu de la release, mais ne sommes pas capable de dire exactement ce qu'on y veut.	Il nous arrive de commencer le développement de stories qui ne sont pas prêtes.
L'avancement du sprint /release est suivi	Burndown chart et/ou daily meeting avec identification claire des ralentisseurs/ accélérateurs. La date de déploiement et l'avancement sur le périmètre souhaité sont connus et partagés.	Il nous arrive de découvrir des blocages après quelques jours. La date d'atterrissage est ambiguë, et/ou non communiquée au client.	Nous avons clairement identifié des effets tunnels. Nous ne savons pas quand nous serons capable de déployer.
Critères de qualité	Les critères de qualité mesurables du projet sont identifiés avec le client et l'équipe et partagés à tous. Ils sont suivis régulièrement et automatisés.	Critères ambiguës et/ ou non communiqués au client.	Critères inexistants et/ou non suivis.
Suivi de l'usine logicielle	L'état de l'usine logicielle est affiché en permanence et remonte une alerte en cas d'anomalie.	Etat régulièrement regardé par les membres de l'équipe, mais pas d'alerte.	Etat suivi en asynchrone.

Qualité des US	Les Definition Of Ready / Definition Of Done sont affichées et respectées Pas de remontée d'US non ready lors des 2 derniers rétrospectives. Une formalisation des stories a été définie entre l'équipe et le PO et est respectée.	DOD / DOR existantes, mais pas toujours respectées. Cas exceptionnel d'US non ready. Formalisation non existante/respectée, mais contenu présent.	DOD &/ou DOR inexistantes. Nous avons régulièrement des US non ready. Contenu des stories aléatoire.
Les rétrospectives sont utiles	Les actions et décisions sont suivies et accomplies d'un sprint à l'autre.	Il nous arrive de ne pas suivre une partie des actions.	Actions régulièrement non suivies.
Déploiement	Le déploiement d'une version ne nécessite presque pas d'effort (<2h) voire le déploiement continu est en place jusqu'à l'environnement de recette.	Déploiement compliqué/ complexe. Il nous arrive d'avoir des problèmes.	Déploiement fastidieux et/ ou régulièrement sujet à erreurs.
Collaboration PO/Equipe	Le PO écoute les propositions fonctionnelles de l'équipe. Le PO suit les réalisations au long du sprint et donne du feedback quasi-quotidiennement.	Le PO prend peu en compte les propositions de l'équipe. Présence et feedback sporadique au cours du sprint.	PO directif et/ou absent au cours du sprint.
Maîtrise / Gestion de la dette technique	Gestion dans le backlog, un backlog dédié. Fréquence de gestion : revue à chaque sprint planning. Priorisation relative ou allocation de buffer.	Pas de priorisation des éléments de dette. Eléments dépilés à un rythme variable.	Pas de gestion. Nous ne connaissons pas notre dette Nous la réglons quand nous pouvons.
Partage des pratiques	Application des pratiques suivantes : Pair programming, revue de code, kata, dojo Application des principes de design du code (Domain Driven Design, SOLID).	Application occasionnelles, quand le besoin se présente.	Application rare.
Testing	Niveau des pratiques mises en places (Test Driven Development, Behaviour Driven Development, stratégies de mocks). Maîtrise des outils.	Pratique régulière mais pas systématique. Outils utilisés non maîtrisés par une partie de l'équipe.	Pratiques et/ou outils non maitrisés/appliqués.

Cette liste de critères est mise à jour régulièrement pour s'adapter aux situations rencontrées sur le terrain.

Take away

Piloter son projet

Choisir

Choisir une méthodologie adaptée à chacun des aspects du projet : Scrum pour les développements au quotidien, en flux pour les tâches amont et aval.

Piloter

Piloter le projet en s'aidant continuellement des projections fournies par l'équipe.
Organiser avec méthode les arbitrages entre budget, planning et périmètre fonctionnel.

Maintenir

Maintenir la qualité technique et du pilotage sur le projet est garant de son succès sur le long terme.

Pérenniser son produit

S'améliorer au quotidien est une chose.

Tenir dans la durée oblige à prendre toujours plus de recul sur le travail effectué par l'équipe.

Récolter les feedbacks plus globaux des donneurs d'ordres, s'enrichir et partager auprès d'autres équipes participent à cet objectif.

S'améliorer en continu

Recueillir les Customer Feedbacks

En plus de l'amélioration continue favorisée au sein des projets, une à deux fois par an, nous prenons un peu de recul sur la relation d'ensemble que nous avons avec nos clients.

Nous les interrogeons ainsi sur quelques questions ciblées :

- Comment qualifiez-vous la qualité de nos livrables ?
- Comment qualifiez-vous la coopération entre les équipes ?
- Comment qualifiez-vous le respect de nos engagements ?
- Comment qualifiez-vous notre devoir de conseil ?
- Comment qualifiez-vous notre réactivité ?
- Comment qualifiez-vous notre capacité d'anticipation ?
- Comment qualifiez-vous notre pilotage de la prestation (budget / risque) ?
- Comment qualifiez-vous la stabilité de notre équipe ?

Les réponses à ces questions sont autant de vecteurs de feedbacks qui nous permettent d'améliorer notre gestion et notre réalisation des projets. Le changement d'interlocuteurs et le fait de rendre formelle cette initiative nous permettent d'obtenir des retours parfois différents de ceux captés régulièrement sur le terrain. Ceci empêche qu'une éventuelle frustration s'installe durablement dans la relation avec nos donneurs d'ordres.

Encourager les Communautés de pratique

Voici comment Étienne Wenger, le père fondateur du concept, définit la notion de communauté de pratique :
« Une communauté de pratique est un groupe dont les membres s'engagent régulièrement dans des activités de partage de connaissance et d'apprentissage à partir d'intérêts communs. ».

Au sein du Studio, des **communautés de pratique** ciblées existent (Android, iOS,ADM, Architectures réactives, etc.). Ces dernières sont autonomes et chacune a son mode de rassemblement : de manière régulière ou à la demande. Le principe est de proposer un ordre du jour et de se réunir autour de formats variés (présentation formelle, revue de code, débat, etc.).

Elles permettent aux équipes **d'échanger sur des problématiques communes** entre projets, sur des bonnes pratiques de développement et d'architecture, de déploiement continu, etc. C'est également un moyen de faire monter en compétence ses pairs, de partager des astuces ou encore d'échanger sur les difficultés spécifiques rencontrées sur les projets.

À noter que selon les sujets abordés sur les projets, il n'est pas rare de voir des communautés naître et d'autres se dissoudre.

Elles font partie intégrante de notre amélioration continue, en permettant une plus grande mixité entre les équipiers, au delà des frontières du projet.

" Une communauté de pratique est un groupe dont les membres s'engagent régulièrement dans des activités de partage de connaissance et d'apprentissage à partir d'intérêts communs. "

Il est temps de se poser la question du devenir de votre produit, tandis que ce dernier rentre dans une phase de maintenance évolutive et corrective. Il est alors nécessaire de mettre en place une organisation spécifique pour assurer ce nouveau mode de développement ou bien organiser le transfert du travail réalisé vers un tiers.

La maintenance évolutive et corrective

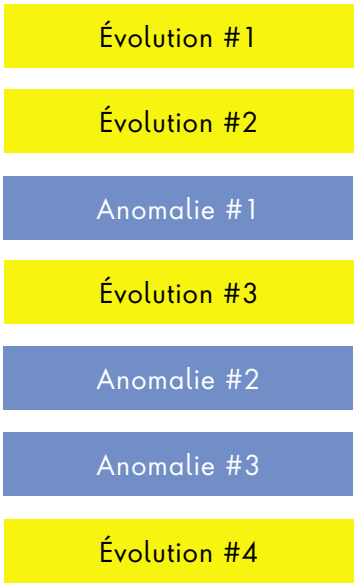
Après les premières livraisons majeures en production, le développement du produit entre en maintenance évolutive et corrective.

Au sein du Studio, nous gérons cette dernière de la manière suivante :

- Pour gérer les corrections sur le produit et les incidents en production, **nous fixons par Sprint une contingence de points dédiés** (par exemple, 20% de la capacité de production moyenne de l'équipe) ;
- Cette contingence est réévaluée Sprint par Sprint afin de s'ajuster à la réalité des corrections et anomalies rencontrées en production ;
- À chaque Sprint, le Product Owner priorise les corrections par rapport aux évolutions, en fonction de ses attentes.

Ce mode de fonctionnement permet :

- De **mobiliser l'ensemble de l'équipe sur les corrections** (nous souhaitons éviter que certains équipiers ne traitent que des évolutions, et d'autres que des corrections avec le risque de créer un contexte démotivant pour ces derniers) ;
- De **continuer à livrer régulièrement** en production de nouvelles fonctionnalités, tout en apportant des correctifs sur le produit ;
- D'assurer un **bon maintien en condition opérationnelle** grâce à une taille d'équipe suffisante.



Maintenance évolutive et corrective

Transfert vers une équipe tierce : la réversibilité

Nous menons cette réversibilité de deux manières différentes. L'une classique, l'autre plus originale.

Réversibilité classique

Une réversibilité classique se déroule généralement en trois phases et dure environ un mois (bien sûr, cela dépend de la taille du projet et de la quantité de travail effectuée avant d'envisager le transfert).

Phase 1 - Préparation de la réversibilité et formation

Cette phase dure environ une semaine et est sous la responsabilité du Studio. Elle permet la préparation des supports nécessaires à la **formation théorique** de la future équipe de développement sur l'application (client ou autre partenaire).
Ces supports présentent notamment :

- L'architecture du produit développé et son interaction avec des composants externes ;
- Le fonctionnel de la solution ;
- Les dernières évolutions réalisées ;
- L'environnement de développement et l'ensemble des accès aux outils ;

- La chaîne de build et de déploiement continu (de la modification d'une ligne de code à la mise en production).

Phase 2 - Maintenance en double

Cette phase dure environ deux semaines et est toujours sous la responsabilité du Studio. Elle permet une prise en main concrète du produit, des environnements de développement et du cadre méthodologique mis en place par le tiers. La réversibilité s'attache en effet trop souvent à ne transférer que le savoir-faire fonctionnel et technique.

Ici, **l'équipe en place et la future équipe travaillent sur le développement de fonctionnalités** représentatives des prochaines évolutions du produit en pair-programming (un développeur de l'équipe actuelle avec un développeur de la future équipe). Ceci permet de s'approprier plus facilement les pratiques de développement et normes de codage en vigueur sur le projet, tandis que le déroulé du Sprint suit son cours habituel afin que le tiers s'imprègne du cadre agile mis en place.

Phase 3 - Maintenance sous contrôle

Cette phase dure environ deux semaines et est sous la responsabilité du tiers. Ce dernier gère désormais les développements et l'ensemble des aspects opérationnels. Néanmoins, l'équipe du Studio reste disponible sur demande, en cas de problème.

À la fin de cette phase, la validation de la réversibilité est prononcée et le tiers est désormais autonome sur le projet.

Réversibilité en immersion

Quand cela est possible, nous privilégions une autre forme de réversibilité que nous mettons en œuvre depuis quelques années au Studio. Il s'agit d'embarquer au sein de l'équipe - au plus tôt idéalement - un ou plusieurs développeurs du tiers, afin qu'ils montent en compétence *in situ*.

Les **équipiers du client / repreneur sont intégrés dans l'équipe**, de manière transparente et en font pleinement partie. Ils participent bien sûr à l'ensemble des cérémonies. Nous attendons de ces équipiers, comme de tous les autres, qu'ils s'inscrivent dans une logique de développement de qualité et d'amélioration continue.

Cette approche apporte plusieurs bénéfices :

- Une **prise en main optimale** du code et des environnements, au plus tôt ;
- La **participation aux choix techniques** de la solution et donc une meilleure maîtrise ;
- La pratique *in situ* du **cadre méthodologique** mis en oeuvre sur le projet, pour une appropriation facilitée.

Le remplacement progressif des équipiers Sapient, par des équipiers du tiers, aboutit finalement à une équipe autonome sur le produit.

" Quand cela est possible, nous privilégions une autre forme de réversibilité que nous mettons en oeuvre depuis quelques années au sein du Studio. Il s'agit d'embarquer au sein de l'équipe - au plus tôt idéalement - un ou plusieurs développeurs du tiers, afin qu'ils montent en compétence in situ. "

Take away

Pérenniser son produit

S'améliorer

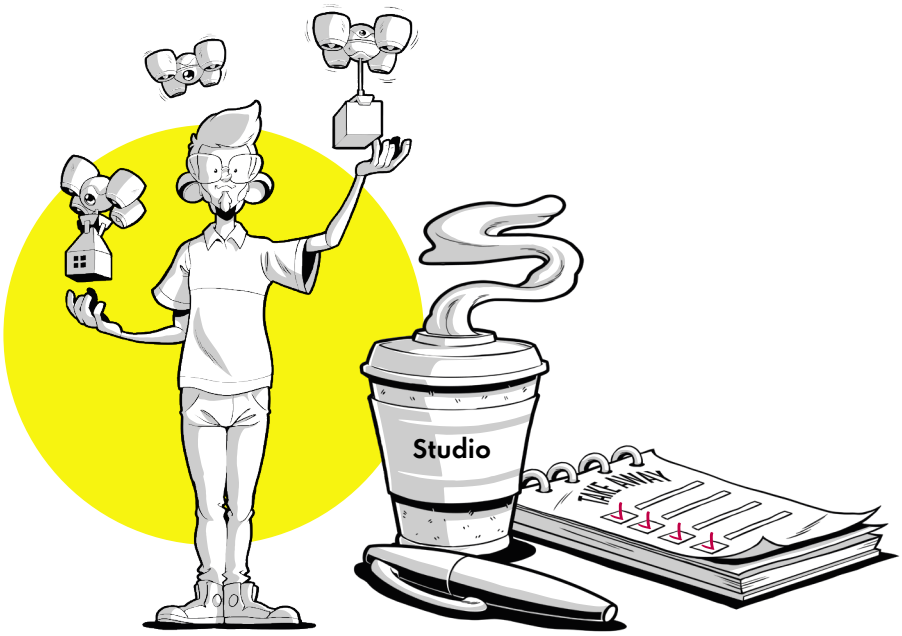
S'améliorer implique aussi une confrontation organisée avec ses donneurs d'ordres et d'autres équipes projets.

Savoir

Savoir modifier son organisation est nécessaire pour aborder efficacement une phase de maintenance évolutive et corrective.

Intégrer

Si les développements sont transférés vers un tiers, intégrer au plus tôt les futurs équipiers est un facteur de succès.



Les facteurs de succès du Studio

Conclusion

Studio existe depuis maintenant quatre ans et a réalisé une cinquantaine de projets. Nous avons beaucoup appris en marchant. À travers ce TechTrends, nous avons voulu partager avec vous les quelques recettes qui font le succès de Studio. **Nous croyons qu'il est possible de réaliser des produit autrement, et nous œuvrons pour que l'agilité, l'excellence technique et la qualité trouvent leur place dans une relation équilibrée entre donneur d'ordre et exécutant.**

Cette recherche débute dès le démarrage du projet. À ce stade, il est essentiel de mener une préparation méthodique et pragmatique de la vision produit et de la partager régulièrement avec l'équipe.

Il faut également structurer son projet, et identifier les profils qui se chargeront de **faciliter le travail des développeurs**, tout en s'assurant de la qualité de la formalisation de la vision produit. L'Agile Delivery Manager (ADM) et le Senior Agile Delivery Manager (SADM) répondent à ce besoin.

Enfin, il est indispensable de mettre en place un cadre contractuel équitable. Le contrat Agile est issu de notre expérience, et combine les engagements que doivent prendre l'équipe et son donneur d'ordres pour une relation saine, durable et centrée sur l'intérêt du produit.

La phase de réalisation doit aussi être aménagée. Scrum n'est pas la solution à toutes les problématiques et vous préférerez, dans certains cas, gérer les tâches amont et aval sous forme de flux.

Dans tous les cas, il est normal que vous obteniez de la visibilité sur l'avancement de votre produit ainsi que **le planning prévisionnel et le budget associés aux prochaines releases**. Appuyez-vous sur des **indicateurs projets** adaptés à vos attentes et construits avec l'équipe.

La qualité des développements et de vos pratiques doit aussi être un de vos objectifs, car elle garantit la solidité et la pérennité de votre produit.

Aménagez-vous le temps et le recul nécessaire pour mettre en place une stratégie de test, d'amélioration continue et de partage de connaissance.

Arrive maintenant la fin de votre projet, ou du moins, son entrée en phase de maintenance évolutive et corrective. Si vous êtes habitué à faire appel à un tiers pour gérer cette partie, nous ne pouvons que vous conseiller d'anticiper ce jalon au plus tôt et d'accueillir dès le démarrage de la réalisation, un ou plusieurs développeurs du repreneur. Ceci leur permettra de s'imprégner correctement des pratiques méthodologiques et de développement en vigueur sur le projet.

Si vous décidez de « garder la main », l'organisation du travail quotidien de votre équipe devra sûrement être revue. Vous devrez **prioriser les prochaines évolutions** tout en conservant la place nécessaire pour traiter efficacement les anomalies rencontrées en production.

Nous tirons notre expérience des produits que nous développons au service de nos clients.

Notre ambition quotidienne, notre **recherche de la qualité et de l'excellence**, nous inscrivent plutôt dans une relation de partenaires / d'alliés. Ainsi, nous croyons que les recettes de ce Techtrends s'appliquent très bien à vos équipes internes, et les projets que vous menez ou mènerez par exemple entre une DSI et les entités métiers qui la sollicitent.

Parce que nous restons strictement dans une logique d'amélioration continue, nous imaginons que certains éléments de cet ouvrage seront amenés à évoluer au gré de nos futures expériences, de nos différents tâtonnements et essais. Nous sommes convaincus que les aménagements que nous apporterons tireront nos projets - et les produits qui les sous-tendent - vers le haut. D'ailleurs, nous sommes preneurs de vos retours, remarques, trucs & astuces, à l'adresse engineering@publicissapient.com.

Take away

Studio

Démarrer son projet

- Préparer les développements en exposant clairement la vision produit et les différents jalons du projet à l'aide d'un atelier de Story Mapping.
- Constituer une équipe efficace côté développement comme côté produit, et la guider au plus tôt, en identifiant les personnes clés comme le Senior Agile Delivery Manager (SADM) et, si nécessaire, le Senior Agile Delivery Manager.
- Contractualiser dans l'intérêt du produit, en trouvant le juste équilibre entre vos engagements et ceux de l'équipe de réalisation.

Piloter son projet

- Choisir une méthodologie adaptée à chacun des aspects du projet : Scrum pour les développements au quotidien, en flux pour les tâches amont et aval.
- Piloter le projet en s'aidant continuellement des projections fournies par l'équipe. Organiser avec méthode les arbitrages entre budget, planning et périmètre fonctionnel.
- Maintenir la qualité technique et du pilotage sur le projet est garant de son succès sur le long terme.

Pérenniser son produit

- S'améliorer implique aussi une confrontation organisée avec ses donneurs d'ordres et d'autres équipes projets.
- Savoir modifier son organisation est nécessaire pour aborder efficacement une phase de maintenance évolutive et corrective.
- Si les développements sont transférés vers un tiers, intégrer au plus tôt les futurs équipiers est un facteur de succès.

À lire et à relire

Pour télécharger l'un des TechTrends en version éléctronique (pdf), rendez-vous sur le site publicissapient.fr/services/engineering
Pour demander une version papier, envoyez un mail à techtrends@publicissapient.com

Pour en savoir plus sur le Data, le Cloud, le Web, les architectures Java, la mobilité et l'agilité, rendez-vous sur blog.engineering.publicissapient.fr

Techtrends #8
Mobile
Comprendre, Façonner, Innover

Techtrends #9
Cloud
Préparer sa migration, Séléctionner son offre, Choisir ses technologies

Techtrends #12
Conteneurs
Projeter, Implémenter, Exploiter

Techtrends #13
Produits Data Science
Explorer, S'organiser, Fluidifier

Techtrends

#11 Internet of Things
#7 Back
#6 DataLab
#5 Front
#4 Craftsmanship
#3 Agilité
#2 DevOps
#1 Data

Autres parutions

Scrum Master Academy
Le Guide du Scrum Master d'élite

Les Communautés de Pratique en Pratique
Le Mini Guide

Product Academy
Le guide des Product Managers et des Product Owners d'élite

Les auteurs

François LAURAIN

Thibaud CAVIN

Ludovic PEROT

Nicolas JOZWIAK

Meriem EL AABOUDI

Christophe HEUBÈS

Pablo LOPEZ

Un remerciement tout spécial à Hugo GEISSMANN (ancien responsable de Studio) et Jean-Laurent DE MORLHON (ancien CTO de Xebia devenue Publicis Sapient Engineering) pour avoir fait naître Studio dans le souci de la qualité, de l'excellence technique et de la transparence.

Direction artistique : **Publicis Sapient Engineering**
Design graphique : **micheletmichel.com** / Illustrations: **lasaintepaire.com**

Publicis Sapient
94 Avenue Gambetta, 75020 Paris
+33 (0)1 53 89 99 99
engineering@publicissapient.fr

Toutes les informations sur :
publicissapient.fr/services/engineering