Chapitre 5
Tests non fonctionnels

1. Introduction

Sur une grosse majorité des projets rencontrés, il s'avère que les tests non fonctionnels sont éludés. À l'évocation de cette catégorie de test, un soupir sonore « *Aaaaah* » un peu gêné est en général l'unique Exigence Non Fonctionnelle (ENF ou NFR en anglais - *Non Functional Requirements*) connue de tous. Dans certains cas, on aura des tentatives telles qu'un essai de l'application sur X navigateurs différents ou une équipe spécialisée qui fait des tests de charge ou de sécurité.

Les anomalies ne proviennent des fonctionnalités qu'à hauteur de 16 % [Beizer 1994], mais **si certaines contraintes rencontrées en exploitation ne sont pas respectées, le produit ne sera qu'un prototype ou une expérience de laboratoire**. C'est pourquoi une stratégie de test ne doit pas faire l'économie de tests non fonctionnels avec une **prise en compte dès la conception**.

Exigences non fonctionnelles dans la mobilité

Dans le domaine de la mobilité, il existe une ENF connue de tous : l'énorme choix de combinaisons OS/Version/Matériel et si l'utilisation du produit sur différentes plateformes mobiles n'est pas prise en compte dès le début, on risque de développer pour iOS et de devoir reprendre tous les développements pour Android ou WinPhone !

Remarque

On pourrait multiplier dans ce livre les informations sur l'importance des ENF, mais ce ne sera que par le vécu et un partage au sein de l'équipe que la maturité émergera pour impliquer réellement ces contraintes. La ***culture est un facteur majeur*** *dans la prise en compte des ENF. Lorsqu'une ENF est bien ancrée dans celle-ci, les solutions techniques et les moyens de tests sont impliqués et budgétés sans avoir ni à réfléchir, ni à convaincre :*

- une technologie multi-plateforme est impliquée : Ionic (https://fr.wikipedia.org/wiki/Ionic_(framework)), Cordova (https://fr.wikipedia.org/wiki/Apache_Cordova), Xamarin (https://fr.wikipedia.org/wiki/Xamarin), NativeScript (https://en.wikipedia.org/wiki/NativeScript), ReactNative (http://www.reactnative.com/), Titanium (https://fr.wikipedia.org/wiki/Appcelerator_Titanium) ou Flutter (https://flutter.dev/) ;

- une ferme de serveur est montée avec OpenStf (https://openstf.io/) ou louée à un prestataire dans le Cloud comme BrowserStack (https://www.browserstack.com/) ou SauceLab (https://saucelabs.com/) pour avoir autant de combinaisons que son marché le nécessite.

On l'aura compris, **cette culture provient de contraintes opérationnelles** (voir chapitre Versant technique du test) qui imposent une **vision proche du terrain**.

2. Tests de sécurité

Depuis le 26 mai 2018, l'Europe a promulgué une loi (RGPD - voir plus bas dans ce chapitre) pour lutter contre le faible niveau de sécurité lié au traitement et à la protection des données personnelles. Cela a fait beaucoup de bruit au vu des sanctions annoncées (https://fr.wikipedia.org/wiki/R%C3%A8glement_g%C3%A9n%C3%A9ral_sur_la_protection_des_donn%C3%A9es) et n'est qu'un prétexte de plus pour mettre au centre des préoccupations les tests de sécurité déjà abordés par l'industrie des cartes bancaires [PCI-DSS], l'Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI) qui a produit le « Référentiel général de sécurité » [RGS] des standards internationaux de sécurité [ISO27002] ou encore ceux liés aux sociétés cotées en bourse [SAS70].

Cette section se penche avec ambition sur un domaine souvent obscur.

2.1 Généralités sur les tests de sécurité

La sécurité est un domaine des plus vastes, car il **touche toutes les parties du produit et les personnes qui interagissent avec**. Cette configuration fait du test de sécurité :

– une activité **pluridisciplinaire** avec des Experts qui comprennent les règles, les lois, l'organisation sous-jacente, les opérations, les processus et les technologies utilisées ;

– une exploration des dangers **dans l'obscurité** avec pour seul éclairage l'expertise afin de trouver les failles avant qu'elles ne soient exploitées ;

– une activité de contrôle **perpétuel** des failles connues pour éviter qu'elles n'impactent (à nouveau) le système ;

– une tâche qui ne saurait être simple…

Astuce

Le MVP pour les tests de sécurité

*Ce sujet est très difficile à aborder, car **techniquement complexe, culturellement ambitieux et politiquement délicat**. Aussi, comme à chaque fois avec l'agilité, on tentera d'identifier le MVP de la sécurité ; ce MVP peut être complètement empirique (on fait avec les moyens et la connaissance du bord), sous-traité à un tiers compétent dans le domaine, ou pris en main dès qu'on a un peu de temps devant soi, par exemple en lisant ce chapitre pour, au moins, aborder le « SHU » de la sécurité.*

Les problèmes de sécurité ne sont pas seulement réservés au domaine de l'infrastructure. La communauté « *Common Weakness Enumeration* » (énumération des failles récurrentes) recense une liste de failles directement liées au développement avec près de 700 vulnérabilités identifiées (voir https://cwe.mitre.org/). Selon Jeffery Payne, **les failles liées au code représentent 50 % des menaces** [Payne 2016].

Astuce

Challenge « 30 jours sur les tests de sécurité »

Pour commencer l'acculturation d'une équipe aux tests de sécurité, Melissa Eaden propose un challenge de 30 jours sur ce thème avec un défi par jour pour avancer sur différents axes [Eaden 2016] :

1. Lire un blog sur la sécurité.

2. Sélectionner et lire un livre sur le test de sécurité.

3. Utiliser un outil de test de sécurité tel que ZAP (https://www.zaptest.com/) ou BurpSuite (https://fr.wikipedia.org/wiki/Burp_suite).

4. Apprendre quoique ce soit sur le scan de vulnérabilité.

5. Connaître le modèle de menace (voir par exemple le modèle « STRIDE »).

6. Explorer ces sites : Google gruyere; HackYourself First; Ticket Magpie; The Bodgelt store.

7. Apprendre une chose ou plus sur les tests de pénétration.

8. Utiliser un outil de type proxy pour observer le trafic d'une application web ou mobile.

9. Découvrir le processus et les procédures liées à l'audit de sécurité.

10. Apprendre ce qu'est le Hacking Ethique (White Hat).

11. Tenter de se représenter une posture de cybersécurité pour une application.

12. Se documenter sur le test de sécurité et établir une discussion sur le meilleur endroit du cycle de développement logiciel où on pourrait en introduire.

13. Réaliser une analyse de sécurité dans une US.

14. Faire l'ingénierie d'un plan de test incluant des tests de sécurité.

15. Écrire et partager des idées sur les tests de sécurité sur Twitter ou un blog.

16. Faire des recherches sur comment construire une « Tiger Box ».

17. Faire des recherches sur une faille de sécurité ou un hack récent.

18. Se documenter sur la sécurisation des Headers.

19. Se documenter sur les Script Kiddies et/ou les Packet monkeys.

20. Se documenter sur les attaques DoS/DDoS. Partager quelques exemples sur un réseau social.

21. Se documenter sur les vulnérabilités d'un réseau et les rechercher dans son organisation.

22. Se documenter sur la sécurité des systèmes et l'appliquer sur son organisation.

23. Répondre à la question : quel est le top 10 des menaces de sécurité cette année ?

24. Utiliser une suggestion de la check-list OWASP pour les applications web.

25. Identifier et utiliser un outil de test de sécurité pour mobile.

26. Faire un comparatif sur les réseaux sociaux des tests web et mobile.

27. Comment le BYOA (Bring Your Own Application - Amenez votre propre application) peut jouer un rôle dans la sécurité ?

28. Partager des idées de tests de sécurité pour un domaine particulier.

29. Faire des recherches sur les contraintes réglementaires de sécurité pour un domaine particulier.

30. Décrouvrir la différence entre le Hacking White Hat, Grey Hat et Black Hat.

31. BONUS : participer à un Bug Bounty.

2.2 Orientations en termes de sécurité

Pour tenter de lutter contre la menace liée aux problèmes de sécurité, plusieurs standards existent et Aleatha Shanley propose une comparaison de différents modèles [Shanley 2015] :

- BSIMM - *Building Security in Maturity Model* : propose différentes pratiques liées à la sécurité (dont les tests de pénétration), mais aucun outil ;
- ISSAF - *Information Systems Security Assessment Framework* : un framework de test de pénétration avec des processus et des outils ;
- MSF - *Metasploit Framework* : palette d'outils de tests de pénétration et de détection d'intrusion ;
- OSSTMM - *Open Source Security Testing Methodology Manual* : méthode d'audit de sécurité ;
- OWASP - *Open Web Application Security Project* : outils, guides et méthodes de travail pour les applications web qui couvrent tout le cycle de développement ;
- PTES - *Penetration Testing Execution Standard* : standard de tests de sécurité qui regroupe différentes sources dont le OWASP [Hayes 2012].

Aleatha Shanley utilise d'autres critères de comparaison :

- l'ISO25010 comme base de son *benchmark* ;
- la longévité, le nombre de révisions et la date de dernière mise à jour ;
- le facteur de lisibilité tel que le *Gunning Fog Index* - GFI (voir chapitre Facteurs de succès) pour guider le choix d'orientation pour une organisation.

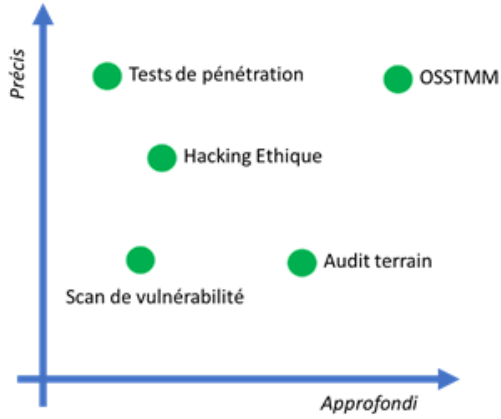

Figure 1 : Positionnement de l'OSSTMM par rapport aux pratiques usuelles de tests de sécurité [Leita 2011]

Le positionnement de l'OSSTMM semble être le meilleur, étant donné la représentation choisie ; en revanche, sur la perspective de la mise en pratique, ce standard se situe loin derrière des approches basées sur les scans de vulnérabilité comme l'OWASP ou MSF qui fournissent des outils.

Définir sa stratégie de test de sécurité

Même si Aleatha Shanley ne va pas jusqu'au bout de la démarche en montrant l'étude comparative des six standards, cela peut donner des pistes pour vous aider à choisir le vôtre ; c'est ce que fait Corrado Leita en comparant à sa façon les frameworks OSSTMM, ISSAF et NIST.

La norme ISO 27001 (voir https://fr.wikipedia.org/wiki/ISO/CEI_27001) aborde la complexité de la sécurité suivant l'approche du progrès continu, le PDCA - Prévoir/Donner corps/Contrôler/Améliorer (en anglais : « Plan/Do/Check/Act » - voir https://fr.wikipedia.org/wiki/Roue_de_Deming) avec les activités suivantes :

- *Prévoir :*
  - *on délimite le périmètre du Système de Management de la Sécurité de l'Information (SMSI) ;*
  - *on s'engage sur une politique de SMSI en fonction des enjeux métier ;*
  - *on s'engage sur une approche du risque ;*
  - *on réalise une analyse de risques ;*
- *Donner corps :*
  - *sensibilisation et communication ;*
  - *gestion de l'exploitation du SMSI ;*
  - *gestion des ressources du SMSI ;*
  - *définition des procédures de détection des incidents de sécurité ;*
  - *définition de procédures de gestion de la documentation ;*
- *Contrôler :*
  - *mesure de l'efficacité du SMSI ;*
  - *révision de l'analyse des risques ;*
  - *conduite d'audits internes ;*
  - *archivage des incidents ;*
  - *collecte et analyse des enregistrements ;*
- *Améliorer :*
  - *mise en place des améliorations identifiées ;*
  - *communication des améliorations ;*
  - *vérification de l'efficacité des améliorations.*

Chapitre 4
Pantesting appliqué au cycle de développement

1. Introduction

La façon dont les activités de développement sont conduites est décisive dans la testabilité d'une solution. Chaque activité est une opportunité d'introduire des pratiques de test et en améliore l'efficacité.

La difficulté à laquelle le test agile à l'échelle est confronté est due :

– aux cadences de mise en œuvre des besoins des clients

– à la quantité d'éléments produits tels que les besoins, le code ou les documents

– aux activités et compétences décloisonnées des organisations pleinement agiles de type C [Nonaka 1986], car tout est effectué « *en même temps* », ce qui pose la question du moment où se fait le test. Dans [Moustier 2019a], la question trouve des réponses, mais comment limiter (voire éviter) de générer un phénomène d'étagement des actions entre chaque équipe et le client ? En effet, il est facile de concevoir des équipes agiles qui livrent leur produit à une équipe d'intégration qui assemble les différentes parties pour livrer la solution complète pour des tests de bout en bout. Hélas, cet exemple décrit clairement une organisation de type B, voire A.

Loin d'apporter une baguette magique, cette section donne des éléments pour s'approcher de cet idéal.

2. Rôle de la vision

Partir dans la bonne direction est le premier point à observer pour arriver à bon port. La vision, qu'elle soit pour une entreprise, une solution, un besoin ou une idée, vient résumer en quelques mots une direction dans laquelle on souhaite engager le plus de personnes possible, ou simplement une équipe. Pour faciliter cette transformation, l'adoption par les membres d'une organisation d'un objectif primordial est d'une grande aide : celui-ci donne une direction, une source de motivation et de cohérence, ainsi que de cohésion.

Cet objectif aide à la prise de décision sur des choix que chacun est amené à faire et donne une raison d'être, plutôt que de procéder à des actions mineures du niveau de la maintenance.

Cet objectif doit [Bradford 1984] :

– refléter l'objectif central pour un groupe de personnes,
– être faisable,
– être difficile, mais stimulant,
– être porteur d'un sens supérieur.

D'autres auteurs, comme [Kotter 1996], aiment aussi ajouter des critères tels que :

– imaginable,
– désirable,
– concentré,
– flexible,
– communicable.

Selon Simon Sinek, commencer par le "*Pourquoi*" de l'objectif [Sinek 2011] est aussi un facteur qui permet à chacun de s'identifier dans une vision. Peu importe les critères, qui peuvent être multipliés ([Appelo 2010] en propose 15 autres), ce qui importe, c'est que ces critères arrivent à faire naître de l'inspiration, créer du sens, ne pas intimider ni démoraliser ou proposer trop de directions [Heathfield 2019], et surtout que tous s'en rappellent !

Pour construire la vision, on se base sur ce que l'organisation fait pour ses clients, sur ses compétences et ce sur quoi elle s'engage. Il ne faut pas hésiter à faire plusieurs propositions pour vote et affinage en atelier, les mélanger, laisser reposer quelques jours. Les quelques propositions restantes peuvent alors être exposées à d'autres équipes sans hésiter à modifier, si nécessaire, la vision résultante. Son partage en sera d'autant plus pertinent pour toutes les personnes qui côtoient cette organisation. On peut aussi se reporter à la section sur la "Planification à l'échelle" où un atelier sur la vision est proposé.

Exemple de vision d'une équipe de SAV

Voici une vision possible pour une équipe de SAV :

« Nous sommes ceux qui maintiennent la réputation de MyCorp en matière de qualité et fiabilité en testant de façon proactive tous nos produits ».

Pour aider à structurer sa vision, [Tichy 2004] propose une structure narrative percutante en quatre parties.

- ***Idée** : on expose l'idée*
- ***Valeurs** : cette idée conduit à des valeurs*
- ***Energie** : l'ensemble des valeurs sollicitées doivent faire vibrer le lecteur par sa cohérence et sa pertinence*
- ***Précipice** : la fin du texte doit laisser le lecteur comme au bord d'un précipice dans lequel il est forcément engagé*

Si on reprend l'exemple de l'équipe de SAV :

- *Idée : « En plus de toute action corrective, nous voulons maintenir au plus haut la réputation de nos produit…/… »*
- *Valeurs : « …/… par une approche proactive centrée sur la qualité intégrée au plus tôt afin de prévenir les défaillances et démontrer la fiabilité de nos produits. … /… »*
- *Energie : « …/… Nous allons sans relâche partager, traquer dans les moindres recoins du produit et de l'organisation toute opportunité d'amélioration. … /… »*
- *Précipice : « …/… Tel est l'engagement de l'équipe de SAV et celui de MyCorp. ».*

Beaucoup de confusions peuvent planer sur les notions de vision, mission, objectif, but ; ce qui nous intéresse ici, c'est l'intention, et surtout le système qui conduit une organisation dans une direction.

Astuce

Les buts sont pour les perdants

Scott Adams, le célèbre dessinateur humoriste de Dilbert, a un regard pertinent sur le concept de vision [Adams 2013] : "Les buts sont pour les perdants", car si on étudie les personnes qui réussissent, elles suivent plutôt des systèmes, c'est-à-dire des attentes générales ; par exemple, perdre 10 kg est un but tandis que mieux manger est un système.

Tandis que les buts sont rigides, les systèmes sont flexibles et mènent à la curiosité, à de meilleures façons de réaliser les choses et développent les compétences. Toutefois, les objectifs restent pertinents, pourvu qu'ils soient restreints, simples et prédictibles.

Par ailleurs, en matière de buts, ne pas atteindre un objectif est un échec tandis qu'en matière de systèmes, ce n'est qu'une tentative pour converger vers un résultat fiable. Le succès n'est qu'une question de chances combinées aux compétences.

3. Cycles de développements de solutions Lean

Dans les approches liées à l'agilité, au-delà des frameworks liés à la gestion des équipes, on trouve aussi des cycles de développements de solutions. Cette vision à haut niveau fournit une stratégie de développement de la solution adaptée au contexte du business ; en effet, une start-up ne connaît ni sa solution ni son marché, tandis qu'une solution mature développée dans une entreprise de grande taille est déjà assise sur un certain nombre de certitudes qui font d'elle un colosse aux pieds d'argiles (voir l'exemple de l'entreprise SpaceX, qui est venue bouleverser le marché de la mise en orbite de satellites qui était pourtant fortement établi) [Blank 2016].

Pour tenter de répondre à un contexte business changeant et Lean-agile, Geert Claes propose un tour d'horizon des cycles de solutions Lean en fonction de la maturité du modèle économique [Claes 2017]. Le schéma ci-dessous propose une déclinaison de sa vision enrichie du Lean UX.

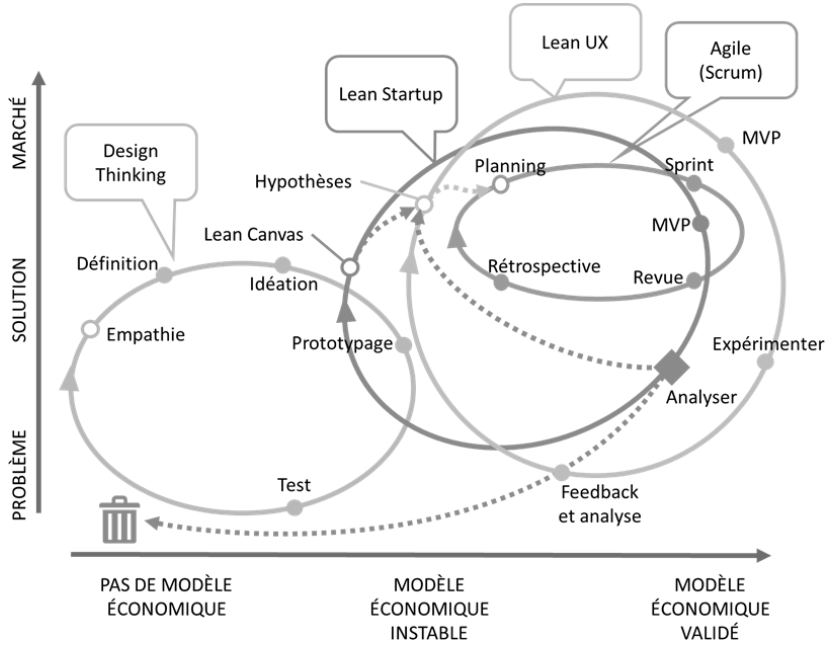

Figure IV-1 : Cycles de développements de solutions Lean en fonction du modèle économique

Sur ce schéma, on peut noter en pointillés les transitions possibles entre les types de cycles de développement. Ces transitions sont comparables aux effets de mémoire qui se produisent en phase Ω.

Vous pouvez aussi vous représenter d'autres transitions en sens inverse et à tout moment, lorsque les retours d'expériences sont tellement significatifs qu'ils imposent parfois une remise en question (phase α) du modèle économique, qui subit alors une révolution.

Malgré leurs formes, ces différents cycles sont en fait des écocycles. Si on reprend la forme en selle de cheval d'un écocycle en 3D vue au chapitre Pantesting, il suffit de voir cette forme depuis le dessous (afin de conserver le sens d'évolution des cycles ci-dessus) pour voir une forme arrondie :

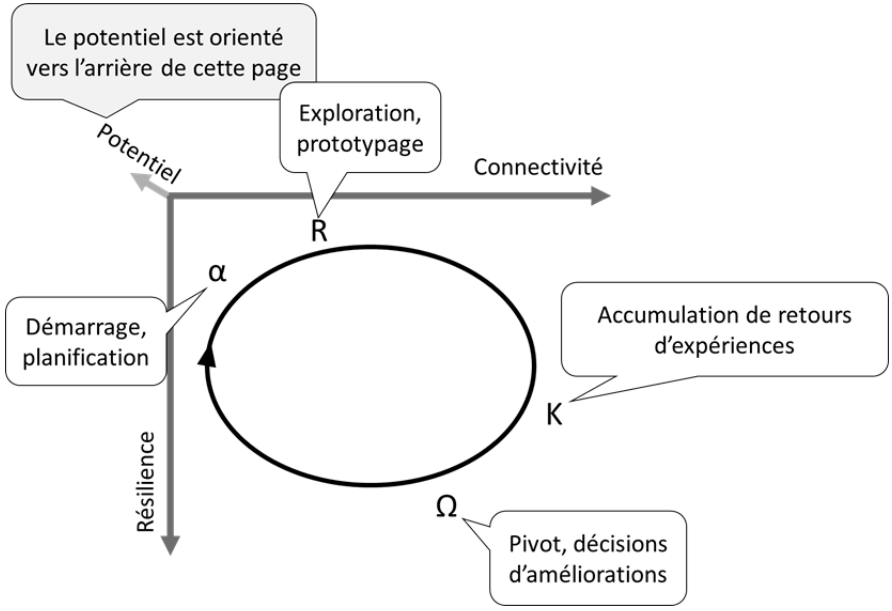

Figure IV-2 : Cycles de développements de solutions Lean du point de vue panarchique

Ainsi, le modèle d'écocycle se calque, comme par magie, sur un cycle de développement. Si on en croit le modèle, la connectivité avec d'autres écocycles doit croître avec la maturité du modèle économique, et plus on est dans le problème, plus il faut de la résilience, qui s'accroît notamment par des tests afin d'en apprendre davantage sur la pertinence de la solution.

Ce que nous enseigne le modèle panarchique est cette double nécessité de constitution de connexions avec d'autres écocycles et d'impératif d'expérimentation par le test.

3.1 Design Thinking

Le Design Thinking (démarche inspirée par la conception) est une méthode de gestion de l'innovation qui trouve ses racines dans les années 1970 à 90 avec les travaux de [McKim 1980] et [Faste 1993]. C'est une approche de l'innovation par observation directe de ce que veulent les utilisateurs et le marché. Elle est constituée de différentes étapes, depuis la compréhension jusqu'au prototype et au test, avec des allers-retours entre les activités.

Voici la liste des activités identifiées du Design Thinking, avec quelques pratiques associées.

- **L'empathie**
  - Utiliser des personae [Moustier 2019a]
  - Générer une carte d'empathie [Gray 2009]

Carte d'empathie

Dave Gray [Gray 2009] propose un outil popularisé par Alex Osterwalder [Osterwalder 2010] et décliné notamment par [Boukobza 2017]. Cet outil est utile pour tenter de découvrir un maximum de choses sur les besoins de « l'autre » et se glisser dans la peau des clients.

Pour cela, il faut organiser une session de brainstorming avec des clients de différents segments de marché que l'on souhaite adresser avec la solution. On peut aussi utiliser ce type d'atelier pour caractériser un type de client, pour mieux se représenter une persona, avec des informations complémentaires telles que sa situation maritale, sa classe de revenus, etc. [Moustier 2019a].

Le schéma ci-dessous est dessiné sur un tableau blanc et les personnes présentes remplissent chaque cadran à l'aide de post-it dans l'ordre des questions.

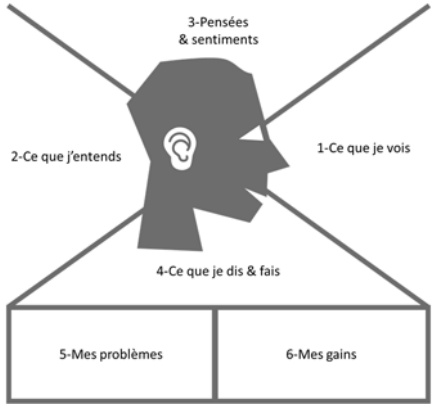

Figure IV-3 : Carte d'empathie

  - ***1-Ce que je vois dans mon environnement*** *- À quoi cela ressemble-t-il ? Qu'est-ce qui est autour de moi ? Qui sont mes alliés et amis, les détracteurs et ennemis ? Quelles sont les offres auxquelles je suis exposé ? Quels problèmes est-ce que je rencontre ?*
  - ***2-Ce que j'entends*** *(ce qui peut m'influencer) - Que disent mes amis ? Qui m'influence réellement et comment ? Quels sont les supports et canaux qui m'influencent ?*
  - ***3-Ce que je pense et ressens réellement*** *- Qu'est-ce qui est vraiment important pour moi ? Qu'est-ce qui m'émeut ? Qu'est-ce qui me tient éveillé la nuit ? Quels sont mes rêves et aspirations ?*
  - ***4-Ce que je dis et fais*** *(ce que je peux dire ou comment je peux me comporter en public - Attentions aux dissonances cognitives) - Quelle est mon attitude ? Qu'est-ce que je pourrais dire aux autres ?*
  - ***5-Mes points de douleur*** *- Quelles sont mes plus grandes frustrations ? Quels sont les obstacles que je rencontre entre ce que je recherche et ma situation actuelle ? Quels sont les risques que j'encours ?*
  - ***6-Mes bénéfices*** *- Qu'est-ce que je souhaite réellement ? Comment je mesure le succès ? Quelles stratégies vais-je employer pour atteindre mes objectifs ?*

- **La définition** (du problème et de ses conditions d'acceptation)
  - Utiliser le framework « *Hooked* » inspiré de [Eyal 2014] pour maximiser les chances de viralité de la solution [Lewrick 2018] (voir l'avertissement "BurnOps", plus bas dans ce chapitre pour en comprendre les mécanismes et les dangers).
  - Identifier les fonctions les plus critiques (ex. à partir de benchmarks, de son expérience, voire des idées les plus saugrenues pour se démarquer de la concurrence) [Lewrick 2018].
  - Utiliser un format tel que QQCOQP*, un « *Elevator pitch* » (courte description partageable le temps d'un voyage dans un ascenseur), le format des Epics de SAFe ou le Lean Canvas (voir plus bas).

(* au lieu de QQQOCP, je préfère dire "QQCOQP", prononcez cela en imitant Johnny Hallyday dans ses toilettes… l'effet mnémotechnique est bien plus fort ! Mais si vous préférez la version érudite, cela s'appelle "*l'hexamètre mnémotechnique de Quintilien*")