Laboratoire 2 - Évaluation d'un document d'exigences

INF8301 : Ingénierie de la qualité en logiciel

Simon Delisle - 1538886
Félix Gingras-Harvey - 1490242
François Pierre Doray - 1529405
Alexandre Vanier - 1525768

École Polytechnique de Montréal
7 février 2014

Facteurs de qualité importants pour le client dans la discipline des requis

Flexibilité

Notre projet nécessite l'utilisation de plusieurs technologies nouvelles dans l'industrie du jeu vidéo et des interfaces naturelles. Étant donné que nous ne pouvons pas aisément juger du potentiel et des limites de chaque technologie en avance, le SRS doit être structuré de manière à permettre des révisions de chaque requis tout au long de la session. Également, le SRS ne doit faire un bon compromis entre clarté et flexibilité: il doit décrire clairement nos objectifs, mais ne pas nous empêcher d'avoir de nouvelles idées tout au long de la session.

La flexibilité est importante pour notre client car il voudra ajuster les objectifs qu'il nous fixe en fonction des nouvelles découvertes technologiques que nous lui présenterons.

Testabilité

Le client voudra suivre l'avancement de notre projet. Il est donc très important de pouvoir lui dire avec assurance quels requis ont été implémentés à la fin de chaque itération.

Pour cela, on doit éviter les requis non testables tel que «le capteur pourra détecter la position exacte des doigts de n'importe quel être humain sur le piano». Ce requis devrait être remplacé par «le capteur indiquera toujours la note attendue après 10 heures d'utilisation par 10 utilisateurs ayant des tailles de main différentes». On doit aussi minimiser l'effort requis pour tester l'accord entre le produit final et le SRS.

Compréhensibilité

Afin que les membres de l'équipe puissent travailler ensemble sur le projet, il est primordial que tous aient une même compréhension des requis. Sinon, il est possible que les idées soient perdues simplement parce certains pensaient que les autres les avaient comprises en lisant les requis. Également, une mauvaise compréhensibilité pourrait mener à du temps perdu à implémenter des fonctionnalités non conformes aux attentes ou incompatibles entre elles.

Pour le client, il est primordial qu'il comprenne bien nos requis s'il souhaite nous conseiller et nous faire profiter de son expérience antérieure de la compétition à laquelle nous participons.

Justesse

Le SRS doit indiquer avec précision le projet que nous souhaitons réaliser. Il faut à tout prix éviter que des fonctionnalités ne soient pas décrites ou que les informations fournies ne soient pas conformes aux objectifs réels (par exemple, «on sait ce qu'on veut faire, mais on ne l'a pas décrit tel quel parce que c'était trop long»).

Le manque de justesse pourrait faire en sorte que notre produit ne soit pas conforme aux attentes du client.

Faisabilité

L'utilisation de technologies nouvelles est un très grand risque pour notre projet. Nous avons tout de même la contrainte de réaliser le produit d'ici le mois d'avril. Le client doit donc savoir que nous avons de grandes chances de réaliser la majorité des requis décrits dans le SRS.

Modèle de qualité basé sur le modèle de référence, et ajusté au pour le client et les requis

Flexibilité

- *Annotated by Relative Stability*
- *Modifiable*
- *Design Independant*

Dans le cas de notre projet, la flexibilité est un élément très important puisque nous prototypons fréquemment avec de nouvelles technologies ou implémentations afin de vérifier si celles-ci sont faisables. Il est donc crucial que l'on puisse dévier du SRS de temps en temps quand un requis est vu comme n'étant pas faisable dans le temps imparti. Il est toutefois important de connaître dès le départ les requis qui sont à risque de changer. Cela est fait grâce à l'annotation des requis selon la stabilité tel que décris par Davis. Il est aussi évident que si l'on veut une flexibilité maximale il faut que le SRS soit fait de façon à ce qu'il soit facilement modifiable. Cela est décrit dans le facteur de qualité "Modifiable" de Davis. Finalement, un des facteurs de qualité de Davis les plus importants dans notre cas est l'indépendance de la conception ("Design Independence"). En effet, il est très important dans notre cas de ne pas forcer une implémentation particulière pour chaque requis étant donné qu'il est très probable que nous changions plusieurs fois d'implémentation pour un requis donné.

Testabilité

- *Traceable*
- *Annotated by Version*
- *Verifiable*

La testabilité est un élément crucial de notre projet étant donné que notre projet sera présenté devant public et qu'il doit performer comme prévu. Pour ce faire, nous avons choisi comme facteur de qualité la traçabilité, l'annotation par version et la vérifiabilité. La traçabilité nous permet de toujours savoir quel requis est testé par nos tests, ce qui permet ensuite de s'assurer plus facilement que tous nos requis sont testés. La vérifiabilité est aussi évidemment très importante puisqu'un requis doit être vérifiable si on veut pouvoir le tester. Finalement, l'annotation par version permet de mieux tenir compte des changements afin, par exemple, d'effectuer des tests de régression.

Compréhensibilité

- *Unambiguous*
- *Understandable*
- *Precise*
- *Not redundant*
- *Concise*

La compréhensibilité est importante pour n'importe quel projet où plusieurs personnes ont à lire le SRS, particulièrement lorsqu'il y a un client. Les facteurs bas niveau de Davis choisis en lien avec la compréhensibilité sont la non-ambiguïté, la compréhensibilité, la précision, la non-redondance et la concision. La compréhensibilité est importante car toutes les classes d'utilisateurs doivent être capables de lire le SRS. La non-ambiguïté l'est aussi car un SRS n'est pas réellement bien compréhensible si on peut l'interpréter de plusieurs façons. La précision permet quant à elle de s'assurer que toutes les informations importantes sont bien lues et comprises par le lecteur. La non-redondance fait que le SRS est moins lourd et plus facile à lire. Finalement, la concision permet d'avoir juste milieu entre une trop grande précision et pas assez d'information.

Justesse

- *Complete*
- *Correct*
- *At Right Level of Detail*

Le SRS se doit d'être juste afin d'éviter des problèmes plus tard dans la conception et l'implémentation (ou même possiblement jusqu'à la fin du projet). En effet, des informations incorrectes peuvent avoir des effets dévastateurs sur un projet en cours de développement. Il est aussi important que le SRS soit complet, c'est-à-dire que chaque information présente dans le document de vision soit traduite en requis. Finalement, il faut avoir le bon niveau de détail afin d'avoir assez de détails pour avoir un requis vraiment juste.

Faisabilité

- *Annotated by Relative Importance*
- *Achievable*

La faisabilité est très importante pour notre projet car tout au long du projet nous devons regarder si telle ou telle option est faisable ou non. Pour ce faire, il est important d'annoter chaque requis par son importance relative afin de s'assurer que ceux ci sont priorisés par rapport aux moins importants. Il faut aussi évidemment que les requis eux-mêmes tel que décrits dans le SRS soient considérés comme ultimement faisable.

Métriques choisies pour évaluer la qualité

Flexibilité

Cette section présente 2 métriques permettant de mesurer la flexibilité du document de spécification des requis.

D'abord, une «proportion de requis ne traitant pas de l'implémentation ou du design» nous permet de répondre à la question «Est-ce que notre SRS nous oblige à implémenter la solution d'une manière plutôt qu'une autre?» Nous visons donc le plus grand ratio (1) possible ici. Cela correspond à une réponse négative à la question, ce que nous souhaitons. Dans le cas contraire, cela veut dire que notre SRS risque de nous limiter à un certain design, qui n'est peut-être pas le meilleur.

Ensuite, le facteur de de facilité de modification nous permet de répondre à la question «Est-ce qu'un changement dans le SRS est facile à effectuer sans y inclure d'incohérences?» Nous cherchons ici le plus grand facteur (1), déterminé par certain éléments du SRS et de l'éditeur de texte utilisé pour le créer. Cela correspond à une réponse positive, ce que nous souhaitons.

Tableau 1: Proportion de requis ne traitant pas de l'implémentation ou du design

a) QME Name	Proportions des requis ne traitant pas de l'implémentation ou du design *(mesure de notre propre cru, adaptée de Davis)*
b) Target entity	SRS
c) Objectives and property to quantity	L'objectif est de cerner à quel point le SRS impose une certaine implémentation lorsqu'il ne devrait pas. Il faut donc calculer la proportion du SRS qui ne spécifie rien sur l'implémentation. Par exemple, si nous spécifions que al kinect doit être positionnée à un tel endroit, cela implique que nous imposons l'utilisation de la kinect et de son SDK au lieu d'un autre capteur comme le Xtion d'ASUS.
d) Relevant Quality measures(s)	Niveau de dépendance à un design.
e) Measurement method	Recensement de chaque requis par la lecture du SRS.
f) List of sub properties related to the property to quantity (optional)	Liste: Requis, Requis imposant un certain design ou implémentation
g) Definition of each sub property (optional)	Requis: Un requis est un élément de niveau 3 du SRS, c'est à dire une section du SRS dont la forme est #.#.#.

	Requis imposant un certain design ou implémentation: Requis qui impose un certain design ou implémentation par sa formulation. Augmente la dépendance du SRS à un certain design.
h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	
j) Numerical rules	La proportion Q de la métrique est obtenue avec Q = 1 - Li / Lt, où • Li = Nombres de requis spécifiant/imposant une certain implémentation ou un certain design. • Lt = Nombres de requis total
k) Scale type	Ratio
l) Context of QME	Cette metrique est choisie pour mesurer la flexibilité su SRS. Si le SRS créer une forte dépendance à un design, certaines solutions qui auraient peut-être été meilleures seront écartées.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Tableau 2: Facteur de facilité de modification

a) QME Name	Facteur de facilité de modification *(Issue de Davis)*
b) Target entity	Spécification des requis logiciels (SRS)
c) Objectives and property to quantity	L'objectif est de cerner à quel point le SRS est facilement modifiable sans répercussion négative. Ici, la condition de base est d'avoir une version électronique du SRS. Comme c'est de nos jours toujours le cas, cet aspect n'entrera pas dans la métrique.
d) Relevant Quality measures(s)	Niveau de facilité à modifier la spécification des requis.

e) Measurement method	Analyse de la présence d'éléments facilitant les modifications
f) List of sub properties related to the property to quantity (optional)	Liste: Requis, Table des matières, Index, Mise à jour automatique.
g) Definition of each sub property (optional)	Requis: Un requis est un élément de niveau 3 du SRS, c'est à dire une section du SRS dont la forme est #.#.#. Table des matières: Table représentant les parties (requis) du document ainsi que leur position dans le document. Index: Liste de mots-clefs trouvés dans le document ainsi que les endroits où ils sont présents. Mise-à-jour automatique: Possibilité de l'éditeur de texte électronique de mettre à jour automatiquement la table des matières et l'index après modifications du document.
h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	Aucune unité de mesure (ratio).
j) Numerical rules	Le facteur Q de la métrique est obtenue avec Q = TM + I + Ma, où • TM = Présence de table des matières (valeur de 0.4) • I = Présence d'un index (valeur de 0.4) • Ma = Présence de la fonctionnalité Mise-à-jour automatique dans l'éditeur de texte (0.2)
k) Scale type	Ratio
l) Context of QME	Cette métrique est choisie pour mesurer la facilité de modification du SRS. Est facilement modifiable, cela permettra les mises-à-jour faciles pour les changements futurs, sans perdre d'information, ou de cohérence. La possibilité de l'éditeur de texte à faire la mise-à-jour automatique de la table des matières et de l'index est pris en compte, car il réduit les risques d'apparition d'incohérences suite aux modifications du SRS.

m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Testabilité

Cette section présente 2 métriques permettant de mesurer la testabilité du document de spécification des requis.

Tout d'abord, nous avons la proportion d'exigences numérotées. Elle nous permet de savoir si toutes nos exigences sont bien identifier afin de mieux les référencer lors des tests. Nous visons ici un ratio de 1. Cela représente un SRS où toutes les exigences sont numérotées.

Ensuite, le nombre de requis annotés par une version nous permet d'augmenter la testabilité, car il est possible de savoir quelles exigences ont été implémenté précédemment. Dans le contexte de notre projet les versions sont peu définies, donc l'annotation sera faible.

Tableau 3: Proportion d'exigences numérotées

a) QME Name	Proportion d'exigences numérotées *(Issue de Davis)*
b) Target entity	SRS
c) Objectives and property to quantity	L'objectif est de faciliter la recherche et les références aux requis. Cela aidera pour le design et les tests.
d) Relevant Quality measures(s)	Niveau de facilité à tracer et tester les requis.
e) Measurement method	Regarder si les paragraphes sont numérotés de façon hiérarchique, s'il y a un seul requis dans chaque paragraphe et si chaque requis sont numérotés.
f) List of sub properties related to the property to quantity (optional)	Liste: Paragraphes numérotés contenant des exigences, paragraphe non numéroté.
g) Definition of each sub property (optional)	Paragraphes numérotés contenant des exigences: C'est un paragraphe bien numéroté hiérarchiquement et qui contient une exigence. Paragraphe non numéroté: Paragraphe qui n'est pas bien numéroté. Diminue la traçabilité des exigences.

h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	Aucune unité de mesure (ratio)
j) Numerical rules	Notre ratio est Q = En / Et, où • En = Exigences numérotées • Et = Exigences totales
k) Scale type	Ratio
l) Context of QME	Cette métrique est choisie pour faciliter le design et les tests. Si le SRS est traçable, il sera plus facile de savoir qu'elle exigence est concernée. S'il n'est pas traçable, le design et les tests ne seront pas de qualité.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Tableau 4: Nombre de requis annotés par une version

a) QME Name	Nombre de requis annotés par une version *(Issue de Davis)*
b) Target entity	SRS
c) Objectives and property to quantity	L'objectif est déterminer quels requis est satisfait dans quelle version du logiciel.
d) Relevant Quality measures(s)	Niveau de facilité à tracer et tester les requis.
e) Measurement method	Vérifier si les exigences sont bien associées à une version du produit.
f) List of sub properties related to the property to quantity (optional)	Liste: Requis sans version, requis associés à une version.

g) Definition of each sub property (optional)	Requis sans version: Requis auquel on ne pas déterminer à quelle version du logiciel il appartient. Requis associés à une version: Requis qu'on peut associer à une version précise du logiciel.
h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	Aucune unité de mesure (ratio)
j) Numerical rules	R_version / R_total où • R_version: nombre de requis associés à une version du logiciel • R_total: nombre total de requis
k) Scale type	Ratio
l) Context of QME	Cette métrique permet de faciliter la traçabilité et les tests. Si tous les exigences sont associés à une version, l'étape des tests sera plus facile, car les exigences déjà implémentés seront connu.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée. On doit minimalement avoir une idée des versions du logiciel.

Compréhensibilité

Cette section présente 2 métriques permettant de juger de la compréhensibilité des requis. D'abord, une grande «proportion de requis ambigus» permet de répondre à la question «Est-ce que les requis sont ambigus?». Si la réponse est «oui», c'est qu'il est difficile de comprendre les requis tel que souhaité par l'auteur. Le «nombre moyen de requis par catégorie» permet de répondre à la question «Est-ce le SRS est organisé de manière à favoriser une bonne compréhension?».

Tableau 5: Proportion de requis ambigus

a) QME Name	Proportion de requis ambigus *(issu de Davis)*

b) Target entity	Spécification des requis logiciels (SRS)
c) Objectives and property to quantity	L'objectif est de trouver les requis pouvant être interprétés de plusieurs manières différentes lors de l'activité de révision du SRS. Ce qui doit être mesuré est le nombre de requis ayant plusieurs interprétations possibles.
d) Relevant Quality measures(s)	Niveau de compréhensibilité du SRS.
e) Measurement method	Demander à 1 représentant de la partie cliente et à 2 développeurs de partager leur interprétation de chaque requis. Compter un requis ambigu chaque fois qu'un désaccord, même minime, survient entre les participants.
f) List of sub properties related to the property to quantity (optional)	Liste: requis, requis ambigu, requis non ambigu.
g) Definition of each sub property (optional)	Requis: Un requis est un élément de niveau 3 du SRS, c'est à dire une section du SRS dont la forme est #.#.#. Requis ambigu: Un requis pour lequel au moins une partie prenante du projet (client ou développeur) est en désaccord avec l'interprétation d'une autre partie prenante. Requis non ambigu: Requis pour lequel toutes les parties prenantes sont en accord avec l'interprétation de toutes les autres parties prenantes consultées.
h) Input for the QME	Spécification des requis logiciels (SRS), liste de personnes aptes à participer à l'évaluation de l'ambiguïté.
i) Unit of measurement for the QME	Aucune unité de mesure (ratio).
j) Numerical rules	La valeur Q de cette métrique est obtenue par la formule: Q = R_ambigu / R_total où • R_ambigu: Nombre de requis respectant la définition de «Requis ambigu».

	• R_total: Nombre total de requis, tel que défini en g). Doit être égal au nombre de requis ambigus additionné au nombre de requis non ambigus.
k) Scale type	Ratio
l) Context of QME	Cette métrique sert à mesurer la compréhensibilité du SRS. Un SRS ayant une forte proportion de requis ambigus indiquent que les parties prenantes ont une compréhension différente de celui-ci, ce qui peut mener à des conflits.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée. L'implication d'au moins 3 personnes du projet est nécessaire.

Tableau 6: Nombre moyen de requis par catégorie

a) QME Name	Nombre moyen de requis par catégorie *(mesure de notre propre cru)*
b) Target entity	Spécification des requis logiciels (SRS)
c) Objectives and property to quantity	L'objectif est de juger de la bonne organisation du SRS. Une organisation avec des titres et sous-titres facilite la compréhension en permettant aux parties prenantes de trouver rapidement l'information recherchée. Ce qui doit être mesuré est le nombre moyen de requis par section.
d) Relevant Quality measures(s)	Niveau de compréhensibilité du SRS.
e) Measurement method	Compter le nombre de requis dans le SRS et diviser le résultat par le nombre total de catégories.
f) List of sub properties related to the property to quantity (optional)	Liste: requis, catégorie.
g) Definition of each sub property (optional)	Requis: Un requis est un élément de niveau 3 du SRS, c'est à dire une section du SRS dont la forme est #.#.#. Catégorie: Une catégorie est un élément de niveau 2 du SRS, c'est à dire une section du SRS dont la forme est #.#.#.

h) Input for the QME	Spécification des requis logiciels (SRS).
i) Unit of measurement for the QME	Requis / section.
j) Numerical rules	La valeur Q de cette métrique est obtenue par la formule: Q = R_total / C_total où • R_total: Nombre total de requis, tel que défini en g). • C_total: Nombre total de catégories, tel que défini en g).
k) Scale type	Ratio
l) Context of QME	Cette métrique sert à mesurer la compréhensibilité du SRS. Un SRS ayant trop peu de requis par catégorie indique qu'il serait bénéfique de faire des catégories moins spécifiques. Les parties prenantes auraient ainsi moins de noms de catégories à parcourir dans la table des matières pour trouver les informations recherchées. À l'inverse, un nombre de requis moyen trop élevé est signe que les catégories sont trop chargées. L'effort cognitif demandé aux parties prenantes est alors sûrement trop élevé.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être divisé en catégories. Il n'est pas nécessaire d'attendre que toutes les catégories du SRS soient rédigées pour prendre cette mesure.

Justesse

Cette section présente 2 métriques permettant de juger de la justesse des requis.

Tout d'abord, le nombre d'exigences contenant des valeurs numériques permet d'ajouter de la justesse aux requis. En effet, elle permet d'avoir un niveau de détail plus élevé donc d'avoir une plus grande justesse dans notre logiciel.

Puis, nous avons le ratio du nombre d'exigences correctes sur le nombre d'exigences non validés. Ceci nous permet d'avoir une idée globale de l'utilité de nos exigences. Si nous avons un score de 1, ça veut dire que tous nos exigences sont correctes et utiles.

Tableau 7: Nombre d'exigences contenant des valeurs numériques

a) QME Name	Nombre d'exigences contenant des valeurs numériques *(issu de Davis)*
b) Target entity	Spécification des requis logiciels (SRS)

c) Objectives and property to quantity	L'objectif est de bien préciser les exigences de façon à augmenter la justesse et la compréhensibilité du SRS.
d) Relevant Quality measures(s)	Niveau de justesse du SRS.
e) Measurement method	Compter le nombre d'exigences qui contiennent une valeur numérique et que celle-ci soit précise.
f) List of sub properties related to the property to quantity (optional)	Liste: Valeur numérique, valeur numérique précise.
g) Definition of each sub property (optional)	Valeur numérique: une valeur numérique, par exemple le nombre de seconde. Valeur numérique précise: valeur numérique qui a un niveau de précision raisonnable.
h) Input for the QME	Spécification des requis logiciels (SRS).
i) Unit of measurement for the QME	Aucune unité de mesure
j) Numerical rules	Le nombre d'exigence
k) Scale type	
l) Context of QME	Cette métrique sert à mesurer la justesse du SRS. Un SRS ayant peu de précision ne sera pas juste, car il ne contiendra pas tous les informations nécessaires.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Tableau 8: Ratio du nombre d'exigences correctes sur le nombre d'exigences

a) QME Name	Ratio du nombre d'exigences correctes sur le nombre d'exigences non validés. *(issu de Davis)*
b) Target entity	Spécification des requis logiciels (SRS)
c) Objectives and property to quantity	L'objectif d'avoir des exigences bien définies et utiles en éliminant les exigences superflues.

d) Relevant Quality measures(s)	Niveau de justesse du SRS.
e) Measurement method	Compter le nombre d'exigences qui sont correct (qui sont utiles au système)
f) List of sub properties related to the property to quantity (optional)	Liste: Requis correctes, requis non validés.
g) Definition of each sub property (optional)	Requis correctes: Un requis qui représente quelque chose d'essentiel dans le produit. Chaque requis satisfait un besoin.. Requis non validés: Requis qui ne sont pas encore validés.
h) Input for the QME	Spécification des requis logiciels (SRS).
i) Unit of measurement for the QME	Aucune unité de mesure (ratio)
j) Numerical rules	Q_3 = n_c / (n_c + n_nv) où • n_c: Nombre d'exigences correctes • n_nv: Nombre d'exigences non validées
k) Scale type	Ratio
l) Context of QME	Cette métrique sert à mesurer la justesse du SRS. Un SRS ayant des requis superflus peut diminuer la justesse.
m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Faisabilité

Les deux métriques que nous avons choisies pour la faisabilité sont le ratio du nombre d'exigences avec lesquelles un membre de l'équipe aurait de l'expérience technique ainsi que le ratio de requis annotés par importance. La première permet d'évaluer le risque de dépassement en termes de temps/coût alors que la deuxième permet de savoir si le SRS est organisé par importance et donc de les classer selon leur faisabilité dans le même temps.

Tableau 9: Ratio d'exigences avec lesquelles un membre de l'équipe possède de l'expérience technique

a) QME Name	Ratio d'exigences avec lesquelles un membre de l'équipe possède de l'expérience technique *(Mesure de notre propre cru)*
b) Target entity	SRS
c) Objectives and property to quantity	L'objectif est de déterminer pour quels requis un ou des membres de l'équipe possède de l'expérience.
d) Relevant Quality measures(s)	Faisabilité des requis du SRS
e) Measurement method	Vérifier si un ou des membres de l'équipe ont déjà travaillé sur un requis similaire pour chaque requis
f) List of sub properties related to the property to quantity (optional)	Liste: Requis avec lesquels un membre de l'équipe a de l'expérience, requis avec lesquels aucun membre de l'équipe n'a d'expérience.
g) Definition of each sub property (optional)	Requis avec lesquels un membre de l'équipe a de l'expérience : Requis pour lequel un ou des membres de l'équipe déjà travaillé avec un requis similaire. Requis avec lesquels un membre de l'équipe a de l'expérience : Requis pour lequel aucun membre de l'équipe déjà travaillé avec un requis similaire.
h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	Aucune unité de mesure (ratio)
j) Numerical rules	R_experience/R_total où • R_experience : Nombre de requis pour lequel un ou des membres de l'équipe déjà travaillé avec un requis similaire. • R_total: nombre total de requis
k) Scale type	Ratio
l) Context of QME	Cette métrique permet d'avoir un aperçu du risque associé à chaque requis. Un requis pour lequel aucun membre de l'équipe n'a d'expérience pratique a un risque plus élevé de dépassement en termes de coût ou de durée.

m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée. Les expériences pratiques de chaque membre de l'équipe doivent être connues.

Tableau 10 : Ratio d'annotation par importance

a) QME Name	Ratio d'annotation par importance *(Issu de Davis)*
b) Target entity	SRS
c) Objectives and property to quantity	L'objectif est de déterminer le pourcentage de requis annotés par importance
d) Relevant Quality measures(s)	Faisabilité des requis du SRS
e) Measurement method	Calculer le nombre de requis annotés par importance
f) List of sub properties related to the property to quantity (optional)	Liste: Requis annotés par importance, requis non-annotés par importance
g) Definition of each sub property (optional)	Requis annotés par importance : Requis possédant une mention déterminant leur degré d'importance. Requis non-annotés par importance : Requis ne possédant pas de mention déterminant leur degré d'importance.
h) Input for the QME	Spécification des requis logiciels (SRS)
i) Unit of measurement for the QME	Aucune (ratio)
j) Numerical rules	R_annotes/R_total où • R_annotes: Nombre de requis annotés par importance. • R_total: nombre total de requis
k) Scale type	Ratio
l) Context of QME	Cette métrique permet d'avoir un aperçu du nombre de requis pour lesquels un degré d'importance a été assigné.

m) Software Life Cycle process(es)	SYSTEM/ SOFTWARE REQUIREMENTS ANALYSIS, révision du SRS.
n) Measurement Constraints (optional)	Le SRS doit être complété avant que cette mesure puisse être évaluée.

Mesures du document de spécification des requis

Flexibilité

Proportions des requis ne traitant pas de l'implémentation

Nous avons calculé la proportion de requis ne traitant aucunement de l'implémentation ou du design. Une première valeur de

$$Q = 33/37 = 0.92$$

a été obtenue. Nous énumérons ici les requis contraignant les choix sur le design ou l'implémentation:

«3.4.1» *Vue à la première personne*
En parlant de la caméra virtuelle du jeu, nous spécifions qu'elle sera le type de caméra fournie dans l'engin de jeu Unity.

«4.5.1» *Langages de programmation*
Nous spécifions que les langages ainsi que l'engin de jeu qui seront utilisés.

«4.5.1» *Librairies utilisées*
Nous spécifions les différentes librairies et kits de développement qui seront utilisés.

«4.5.4» *Achat de composantes*
Nous spécifions que nous utiliserons exactement 2 capteurs Kinect.

Par contre, certains requis sont explicitement ajoutés au SRS pour spécifier les technologies utilisées (les requis de la section 4). Si l'on retire ces requis, un seul requis augmente la dépendance à l'implémentation et nous obtenons donc une valeur améliorée de

$$Q = 36/37 = 0.97$$

Facteur de facilité de modification

Notre SRS contient une table des matières qui peut se mettre à jour automatiquement à l'aide de Microsoft Word. Par contre, notre document ne contient pas d'index. Nous avons donc

$$Q = TM + I + MA = 0.4 + 0 + 0.2 = 0.6$$

Testabilité

Proportion d'exigences numérotées

Ici, nous avons eu un ratio parfait de 1, puisque toutes nos exigences étaient numérotées et donc Q = 37/37 = 1.

Nombre de requis annotés par une version

Dans le contexte de notre projet intégrateur, nous n'avons pas associé nos exigences à des numéros de version de notre logiciel. Ce qui fait que nous avons aucun requis annoté par une version, et donc un ratio de 0.

Compréhensibilité

Proportion de requis ambigus

Nous avons calculé la proportion de requis ambigus tel que décrit dans la description de la mesure. Les développeurs consultés sont François Pierre Doray et Félix Gingras Harvey. Le client consulté est Olivier Gendreau. Notons que la nature de notre projet intégrateur (participation à une compétition) fait en sorte que le «client» est défini comme une «personne extérieure à l'équipe du projet, capable de juger de l'intérêt technologique de la solution développée et présente pour nous guider tout au long de sa réalisation».

Sur les 25 exigences fonctionnelles, 4 ont été jugées ambigües. Sur les 12 exigences non fonctionnelles, 1 a été jugée ambigu. On obtient donc une mesure globale *Q = 5/37 = 0,14*. Cette mesure se rapproche de l'idéal de «0» sans toutefois être parfaite.

Les exigences fonctionnelles pour lesquelles une ambiguïté a été notée sont:

«3.1.1» Choix du mode de jeu
1 évaluateur pensait que le choix du mode de jeu pourrait être modifié à n'importe quel moment, tandis que les autres évaluateurs ont compris que l'utilisateur devait quitter la zone de jeu visible par la Kinect pour que le menu du choix du mode de jeu réapparaisse. Il a été conclu que la première interprétation est la bonne.

«3.1.3» Mode assisté
2 évaluateurs pensaient que l'instrument automatique redevenait actif quelques secondes après que l'utilisateur ait cessé d'utiliser l'instrument manuel. L'autre évaluateur a compris que le joueur devait indiquer explicitement qu'il souhaitait cesser l'utilisation de son instrument manuel, mais n'ont pas su expliquer de quelle manière indiquer ce choix. Il a été conclu que la seconde explication était la bonne et que le choix devait se faire en levant les 2 bras dans les airs face à la Kinect.

«3.3.2» Détection du geste correspondant à chaque instrument
2 évaluateurs pensaient que tous les joueurs devaient choisir leur instrument avant qu'il soit possible de commencer à jouer, tandis que l'autre évaluateur a jugé que chaque joueur pouvait jouer après le choix de son propre instrument. Il a été conclu qu'il était possible de commencer à jouer après avoir choisi son propre instrument. Cependant, la musique automatique du mode assisté ne démarrerait qu'une fois que tous les joueurs aient fait leur choix.

«3.9» Affichage des instruments de chaque joueur sur un même écran
2 évaluateurs pensaient que tous les joueurs voyaient leur instrument dans une scène continu, tandis que l'autre évaluateur pensait à un affichage de type «split screen». Il a été conclu que cela n'était pas clair au moment d'écrire le SRS, mais que pour des raisons techniques, un affichage «split screen» serait préféré.

L'exigence non fonctionnelle pour laquelle une ambigüité a été notée est:

«4.4.1» Calibrage facile
Aucun évaluateur n'a eu la même définition de «calibrage facile». Il a été déterminé qu'un calibrage facile signifie qu'il doit y avoir au plus 10 valeurs numériques à fournir pour calibrer les capteurs. Un affichage en temps réel doit permettre à l'opérateur de valider si les valeurs entrées produisent le bon résultat.

Nombre moyen de requis par catégorie

Le SRS contient 25 requis fonctionnels répartis en 11 catégories et 12 requis non fonctionnels répartis en 6 catégories. Le nombre moyen de requis par catégorie est donc de 2.18 requis par catégorie. Cela est faible comparé à la cible de 4 que nous nous étions fixée avant de prendre la mesure.

Les catégories contenant le plus de requis sont «3.4 Exigences relatives au piano» (5 requis) et «3.5 Exigences relatives à la batterie» (4 requis). Le nombre de requis se devait d'être plus élevé que la moyenne en raison de la complexité de ces catégories.

La catégorie «Instruments souhaitables» contient 1 seul requis, qui indique des instruments qui devraient être ajoutés au jeu seulement si le temps le permet. Il est compréhensible que des requis supplémentaires n'aient pas été ajoutés dans cette section sachant qu'il y a peu de chances qu'ils soient implémentés. Néanmoins, il peut être difficile pour un lecteur de comprendre quels sont nos objectifs en incluant ces requis.

La catégorie «Modes de jeu» comporte 4 requis. Ces requis décrivent des fonctionnalités complexes avec peu de détails. Il aurait sans doute été souhaitable de faire une catégorie pour chaque mode de jeu et d'inclure plus de détails (séparés en requis distincts) pour chaque requis. Avoir des catégories distinctes aurait favorisé la compréhensibilité en indiquant clairement au lecteur quels requis sont reliés à un même mode de jeu.

Justesse

Nombre d'exigences contenant des valeurs numériques

Nous avons compté le nombre d'exigences précises (possèdent des valeurs numérique si applicable). Nous en sommes arrivés à 31 exigences précises. Nous avons au total 37 exigences, donc une proportion de 0,84 exigences sont précises.

Des valeurs numériques auraient dû être présentes pour les exigences suivantes :

« 3.4.4 » Grosseur relative des modèles
Nous ne spécifions pas en quelle proportion les notes de pianos doivent être plus larges que les doigts.

« 3.4.5 » Utilisation des pédales
Nous ne spécifions pas le nombre de pédales à implémenter.

« 3.5.2 »
Nous ne spécifions pas le nombre d'éléments de la batterie à implémenter.

« 3.7 »
Nous ne spécifions pas le nombre d'instruments à percussion à implémenter.

« 3.8 »
Nous ne spécifions pas le nombre d'instruments à vent à implémenter.

« 4.4.1 »
Nous ne spécifions pas combien de personnes ou combien de temps devraient être nécessaires pour calibrer le système sans l'équipe.

Ratio du nombre d'exigences correctes sur le nombre d'exigences non validées

Ici, il est à noter que nous avons explicitement des requis non essentiels dans notre SRS, car nous avons 3 types de requis, soit Essentiel, Souhaitable, ou Optionnel. Si l'on effectue le calcul en comptant uniquement les requis essentiels comme requis corrects, nous obtenons un ratio de 16/25 = 0.64. Si nous considérions uniquement les requis souhaitables comme non valides, nous obtenons donc un ratio de 20/25 = 0.8

Ici, toutes nos exigences non fonctionnelles sont pour nous des contraintes essentielles qui répondent à un besoin. Nos deux ratios deviennent donc 0.76 et 0.86 si nous ajoutons celles-ci au calcul.

Faisabilité

Ratio d'exigences pour lesquelles un membre de l'équipe possède de l'expérience technique

Les membres de l'équipe ne possédaient pas d'expérience pour la réalisation des exigences suivantes :

«3.3.1 Détection du geste correspondant à chaque instrument»
Aucun membre de l'équipe n'a déjà travaillé avec les squelettes de la Kinect. Cependant, cela est assez simple selon les équipes des années passées.

«3.4.3 Notes jouables du piano»
Aucun membre de l'équipe n'a d'expérience avec l'utilisation d'une Kinect pour la détection de la position des doigts. Cependant, nous avons déjà travaillé avec les informations de couleur et de profondeur fournies par une Kinect et 2 des membres de l'équipe ont suivi le cours de traitement de signal.

«3.5.2 Jouabilité de la batterie»
Même commentaire que 3.3.1

«3.6.2 Jouabilité de la guitare»

Même commentaire que 3.3.1

Nous avons possédons de l'expérience pour réaliser 21 exigences fonctionnelles sur 25, soit un ratio de 0,84.

Ratio d'exigences annotées par importance

Toutes les exigences sont annotées par importance à l'exception des exigences non-fonctionnelles. Il aurait sans doute fallu ajouter des annotations d'importances aux exigences non-fonctionnelles même si celles-ci sont plutôt générales.

Conclusions sur l'évaluation

Les résultats de l'évaluation du facteur de qualité «**Flexibilité**» sont satisfaisants. Les requis spécifiant les capteurs et SDK à utiliser sont contraignants (nous aurions pu découvrir de meilleures solutions au cours du projet) mais tout de même souhaitable pour nous permettre de rassembler tôt dans la session le matériel nécessaire. Nous ferions de même dans un futur SRS. Le requis contraignant l'architecture logicielle pour les caméras du jeu n'aurait toutefois pas dû être présent. Également, dans un futur SRS, nous ajouterions un index pour faciliter la recherche des sections à modifier.

Il n'y a pas de problème majeur avec la «**Testabilité**», bien que l'annotation des requis par version aurait facilité l'écriture des plans de tests à chaque itération.

Les résultats de l'évaluation du facteur de qualité «**Compréhensibilité**» ne sont pas satisfaisants. La valeur de la QME «Proportion de requis ambigus» est 0,14 mais aurait aisément pu être 0. L'ambigüité de notre SRS s'explique par l'absence de figures et l'omission de décrire l'effet de chaque action de l'utilisateur. Dans un futur SRS, nous devrions présenter des figures comme des machines à état. Pour ce SRS, une machine à état aurait facilement pu décrire tous les passages possibles entre les différents modes de jeu et les actions associées. En décrivant chaque action possible de l'utilisateur dans chaque mode de jeu, nous n'aurions pas incité les lecteurs à faire des déductions incorrectes et nous aurions éliminé 3 des 5 points de désaccord observés lors de l'évaluation.

En ce qui concerne la «**Justesse**», l'utilisation d'un grand nombre de valeurs numériques a été très bénéfique. Nous devrions faire de même dans un prochain SRS. Le contenu de notre SRS décrit aussi exhaustivement nos objectifs.

Enfin, la «**Faisabilité**» du SRS n'est pas évidente. Beaucoup de requis font place à l'inconnu et nous ne pouvons garantir qu'ils sont faisables. Cela est dû à la nature de notre projet, qui se doit d'explorer des technologies encore inconnues des experts en réalité virtuelle. Cependant, cela implique un très grand risque et a un impact négatif sur d'autres facteurs de qualité comme la justesse (on peut difficilement décrire précisément ce que l'on va faire sans en avoir une idée parfaitement claire). Pour un projet avec un réel client, nous ne ferions pas un tel SRS car le produit final risque d'avoir plusieurs différences avec sa spécification. Dans ce projet, nous avons été encouragés à procéder de la sorte et nous croyons avoir fait un bon travail en précisant ce qui était risqué et en décrivant les solutions de rechange.

Temps passé sur le laboratoire

Nous avons passé 18 heures-personne sur ce laboratoire.