Stratégie d'assurance qualité

1.1 LE BUT:

Développer et tester chacun des requis, établir des bonnes stratégies de travail pour sauver des ressources.

1.2 LES ACTIONS CHOISIES:

- 1- À chaque fois qu'on développe une nouvelle fonction, quelqu'un fait un test case en parallèle pour cette fonction et, à la fin, tester ladite fonction.
- 2- Établir une bonne traçabilité, prioriser le «*Bug tracking*» et bien commenter le code.
- 3- Tester chacun des requis du cahier des charges et prévoir des tests cases/intégration.
- 4- Une personne différente du développeur fera les tests.
- 5- Établir des schémas bloc détaillés et s'y référencer tout au long du développement

1.3 LA STRATÉGIE:

Faire de l'assurance qualité l'élément qui englobe toutes les parties du projet. On divisera alors l'AQ en trois parties principales : Le Hardware, le Software et la gestion. Pour s'assurer de la qualité globale du produit, un plan de test sera développé avec un test unitaire par élément dans le cahier des charges.

Pour appliquer l'assurance qualité en gestion, il se doit de mettre à jour fréquemment les courbes en S et de regarder le diagramme de Gantt pour s'assurer de ne pas dépasser le budget. Une personne sera responsable de cette tâche. Une autre personne sera aussi responsable de préparer les réunions, pour s'assurer que celles-ci soient efficaces pour améliorer le procédé du projet. La planification sera aussi révisée par le responsable en assurance qualité, celui-ci verra avec le gestionnaire du projet ce qu'il pourrait être amélioré.

Pour la partie d'assurance qualité en Hardware, il faudra que le responsable de l'assurance qualité surveille les manipulations et les planifications pour s'assurer de la robustesse du produit. Une revue sera prévue à ce sujet pour évaluer les meilleures méthodes de conception pour éviter d'obtenir un produit final qui se brise ou se débranche facilement.

Assurance Qualité

Hardware

Robustesse des capteurs

Les capteurs devront détecter clairement les différents points appuyés

Facilitié de branchement

Le client devrait être en mesure de brancher facilement le système

Robustesse physique

Le déplacement du système ne devrait pas être risquée

Relativement portatif

Être portatif dans la mesure du possible, mais demande tout de même plusieurs branchements

Aspect visuel

Cacher les piezos

Gestion

Requis/Spécifications Cahier de charges

Plan de test qui couvre tous les requis

Budget en heures

Assurer une gestion qui ne dépasse pas le budget

Mise à jour fréquente des outils de gestion

Utiliser et mettre à jour les outils tout au long du projet

Réunions efficaces

Maximiser l'efficacité des réunions et offrir la possibilité à chaque membre d'exprimer son opinion

Planification

Se fier au Gantt, prévoir un calendrier, bien suivre tous les outils de gestion

Quantification des requis

Bien analyser tous les requis pour pouvoir avoir des objectifs de qualité à atteindre

Software

Robustesse stabilité

Le programme devrait pouvoir rouler sans crash. Minimiser les bugs avec le «Bug tracking»

Robustesse en traitement de signal

Le système ne devrait être capable de détecter deux touchés distincts sans les mélanger

Corrélation

Avoir un algorithme consistant pour comparer les signaux

Commentaires dans le code

Bien commenter les lignes de code pour que tout le monde comprenne le programme

Stratégie des documents de code

Bien répartir les fichiers de code dans plusieurs fichiers distincts

Rapidité de l'exécution

Obtenir une exécution la plus rapide possible (~>50ms)

Assurer le maximum de nombre d'actions

Pouvoir avoir au moins 5 à 6 points d'appuie possibles