**1. Division de travail**

On souhaite que le produit final de ce travail soit un jeu de Morpion déployé dans le robot KUKA youBot et commande par une interface graphique, comme on a déjà décrit ci-dessus.

Sur base de ces objectifs, on peut séparer le travail en deux phases différentes que seront développés de manière séquentielle:

* Phase 1: à propos du système robotique dans un environnement de simulation, lequel on doit traiter et solutionner les problèmes de commande du système;
* Phase 2: à propos de la mise en oeuvre du jeu en environnement physique, lequel on doit construire les différentes interfaces du système: robot ⇔ PC et PC ⇔ utilisateur.

Les actions et les sous-produits de la première phase du projet seront séparés en deux structures et développés en parallèle: une concernant l’étude et génération de la trajectoire exécuté par l’effecteur du robot (structure A) et l’autre concernant l’étude et mise en oeuvre de l’asservissement de l’effecteur (structure B).

Après la conclusion de chaque structure, on doit faire l’intégration des parties qui ont déjà été développés. Ainsi, on prévoit observer dans l’environnement de simulation Matlab l’effectuer réaliser une trajectoire stipulée.

De la même manière, la deuxième phase sera séparé en deux structures: une concernant la communication du robot avec l’ordinateur (structure A) et l’autre concernant la création de l’interface graphique de communication de l’ordinateur avec l’utilisateur (structure B).

De la répartition du projet en deux phases en deux phases bien définies, on a pu séparer le projet en deux différentes domaines: d’automatique et d’informatique. Ainsi, lorsqu'on complète la première phase à propos de la problématique de contrôle, on aura déjà obtenu résultats bien satisfaisanr référents les objectifs d'apprentissage de ce projet.

Le diagramme en arbre suivant décrit les différentes phases et structures qu’on a réparti le projet:

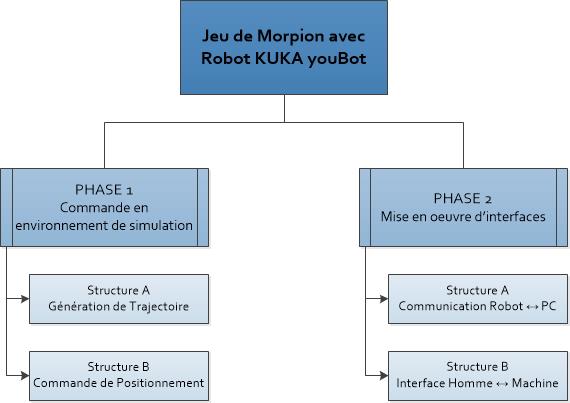


Figure X: Diagramme en arbre de la structure de découpage du projet

**1.1 Définition des tâches à entreprendre**

Ci-dessous, on a défini les tâches à entreprendre en chaque structure de découpage du projet. Dans la section BLA METODOLOGIA on a décrit les tâches de façon plus détaillée.

**1.1.2.B Interface Homme ⇔ Machine**

Comme point de départ du développement de l’IHM, on aura l’obtention et l’utilisation de la fonction concise obtenue en tant que produit final de l’intégration effectuée à la fin de la phase 1 (tâches 1.B.8 et 1.B.9).

Dans un premier temps, on doit définir les caractéristiques, les contraintes et  les dimensions de l’environnement de travail de l’effectuer (tableau du jeu). Alors, on doit créer les fonctions qui exécuteront les deux mouvements que le robot réalisera: “X” et “O”. Après, on doit définir les paramètres d’entrée et les données nécessaires pour le développement de l’interface et aussi la création et l’administration des objets graphiques de l’interface. Enfin, on va faire des tests, corrections nécessaires, améliorations dans l’interface et l’intégration de ses fonctionnalités avec d'autres produits finaux du projet.

Ensuite, on a ordonné ces tâches et leurs temps estimé pour l'exécution de chaque activité:

* 2.B.0: Fonction concise qui détermine le paramètre d’entrée du robot (couple) à partir de la position finale et le type de trajectoire désirée (obtenue dans les tâches 1.B18 et 1.B.9): créneaux;
* 2.B.1: Définition de les caractéristiques et dimensions d’environnement de travail (tableau du jeu): 2 créneaux;
* 2.B.2: Définition et création de la structure de données et paramètres nécessaires pour l’IHM: 2 créneaux;
* 2.B.3: Création des fonctions de dessin “X” et “O”: 2 créneaux;
* 2.B.4: Définition de la disposition graphique et des objets de l’interface: 2 créneaux;
* 2.B.5: L’insertion des éléments dans l’interface: 2 créneaux;
* 2.B.6: Création et gestion des événements associés à les objets: 2 créneaux;
* 2.B.7: Tests, dépannage et intégration: 3 créneaux

Ainsi, on prévoit 6 créneaux pour la réalisation de la tâche 2.B.0 qui on doit obtenir comme résultat de l’intégration des activités dans la première phase du projet et plus 15 créneaux pour l’exécution des autres tâches de cette structure.

**1.2 Désignation des tâches**

À partir du découpage et définition de las différentes structures du projet, on a défini que les membres Rafael Accacio et Rafael Eller seront responsables de l'exécution de la structure A du projet dans les deux phases, tandis que les membres Karoline et Tiago seront responsables de la structure B. Pour faire cette désignation des tâches, on a considéré les jours disponibles de projet (série) de chaque membre, afin que dans les jours lesquels seulement deux membres seront au laboratoire, chacun travaille en structures différentes. Ainsi, on prévoit obtenir une progression plus rapide et égal pour chaque structure.

Cette organisation n’empêche pas les membres d’aider, discuter et exécuter différentes activités prévues, en prenant en compte la conclusion des tâches respectives. De plus, au fin de chaque phase on prévoit l’intégration des structures, ainsi la relation et le partage d’informations entre l’équipe sera considérée pour l’obtention de bons résultats.

**1.3 Calendrier du projet**

Afin d’analyser et gérer le progrès de l’équipe et l'exécution des différentes tâches, on a fait l’estimation de la durée d’exécution des activités nécessaires pour la conclusion du projet. La table suivante décrit le calendrier de réalisation du projet:

CRONOGRAMA AQUI

Les unités du temps sont considérés en créneaux, dans lequel chaque créneau a 1h30 de durée. Les jours dans lesquels seulement moitié de l’équipe sera présente dans le laboratoire, on a considéré la moitié du temps de chaque créneaux.

On aura 40 créneaux pour la séquence 8 et 54 créneaux au total pour la conclusion du projet.

**2. Méthodologie**

**2.2.B Interface Homme ⇔ Machine**

L’objectif de cette structure du travail est obtenir une interface graphique entre l’utilisateur et l’ordinateur qui permet deux joueurs jeter séquentiellement  leurs mouvements qui seront réalisés par le robot sur le tableau réel. Ainsi, l’interface doit recevoir comme paramètre d’entrée le mouvement souhaité par l’utilisateur et remettre comme sortie la variable qui va agir sur le robot.

En prenant en compte cet objectif et les limitations de temps prévus pour l’exécution de ces tâches, on souhaite construire une interface fonctionnelle, intuitif et déployé de façon simple. Alors, on a choisi développer l’interface par programmation en langage Matlab et de manière modularisée, à savoir, la séparation des différentes tâches en différentes fonctions et avec haut niveau d'indépendance. La figure ci-dessous illustre l’idée sur la structure de programmation de l’interface qui sera développée:

Legenda da figura: Estrutura de programação e níveis das funções desenvolvidas na construção da IHM

**2.2.B.0 Fonction concise des trajectoires et commande**

On prévoit obtenir cette fonction comme produit d'intégration des résultats obtenus des structures A et B dans la première phase du projet (tâches 1.B.8 et 1.B.9).

Comme produit d’étape de génération de trajectoire on attend obtenir un algorithme qui calcule le signal de consigne à partir de la position initiale de l'effecteur du robot,  de la position finale et du type de trajectoire souhaitée. Tandis que dans l’étape d’asservissement, on prévoit obtenir le calcul du couple qui va agir au fil du temps dans le robot à partir de la consigne.

Ainsi, en faisant l’intégration de ces deux algorithmes, on pourra obtenir une fonction qui calcule le couple à partir de la position finale souhaité et du type de trajectoire qui le robot va exécuter. Cette fonction sera appelée chaque fois que l’utilisateur, à partir des événements généré dans l’interface, demander un mouvement du robot.

**2.2.B.1 Définition d'environnement de travail (tableau)**

Au début, on doit déterminer l’origine du repère outil, à savoir le repère de l’environnement de travail du robot.

À partir de cette spécification, on doit définir les dimensions et divisions du tableau du jeu du Morpion. Ainsi, on pourra définir les distances parcourues par l'effecteur du robot quand il se déplace sur les carrés du tableau.

Comme alternatif de cette détermination fixe de la taille du tableau, on peut aussi paramétrer la dimension du tableau. Ainsi, on laisse l’option pour l’utilisateur choisir la dimension souhaitée à partir de l’entrée de données de l’interface. Alors, on pourra aussi créer une fonction dans laquel le robot va faire le dessin du tableau.

**2.2.B.2 Création des fonctions "X" et "O"**

À partir de la fonction concise obtenue dans la tâche 2.B.0, on peut créer deux fonctions dans lequel le robot va faire le dessin du mouvement de “X” et “O”. La dimension des dessins devra être d’accord avec la dimension du tableau, ainsi cette fonction devra recevoir cette dimension comme un paramètre.

**2.2.B.3 Définition de la disposition graphique d’IHM**

Dans cette tâche, on va faire l’ébauche de la séquence des actions qu’on attend que l’interface reçoive et quels objets et dispositions graphiques l’interface doit contenir.

**2.2.B.4 Définition de données d'entrée nécessaires IHM**

À partir des définitions de quels objectifs devront contenir dans l’interface, on va savoir quels les paramètres d’entrée que l’interface recevra. Ainsi, on pourra définir quels sont les autres paramètres et informations nécessaires pour l'exécution totale du jeu.

Alors, on pourra démarrer et nommer les variables qui iront recevoir les paramètres d’entrée du jeu. Afin de faciliter le stockage et gestion de ces paramètres, on doit utiliser variables du type structure.

**2.2.B.5 L'insertion des objects graphiques**

À partir de la définition précédent, on doit créer et insérer les éléments dans l’interface et déterminer aussi les propriétés respectives des objets.

**2.2.B.6 Création et gestion d'événements**

À partir de la création des objets, on va faire l’édition de chaque fonction *callback* associée aux événements générés dans l’interface.

**2.2.B.7 Tests, dépannage et intégration**

Au début, on va tester les fonctionnalités de l’interface créée et, si besoin, on va faire le dépannage et amélioration dans l’algorithme de création.

Alors, on prévoit d'intégrer la communication obtenue entre le PC et le robot (résultat de travail de la structure A) avec l’interface créée, afin d’obtenir la communication totale entre le robot ⇔ PC ⇔ utilisateur.