PROJET INTEGRATEUR

XI 2021-2022

15/06/2022

City Modeling

by IngéniCar

1 PRESENTATION DU ROBOT 3

2 INTERFACE 2D TEMPS REEL 4

3 APPLICATION PRINCIPALE (OPTIONNEL) 6

4 MODELISATION ET ARCHITECTURE (OPTIONNEL) 7

5 INTEGRATION DES DONNEES (OPTIONNEL) 9

6 MODELE COLLABORATIF (OPTIONNEL) 10

7 STATISTIQUES (OPTIONNEL) 11

8 REPLICATION DU MODELE ET EXPANSION (OPTIONNEL) 12

1
PRESENTATION DE CITY SMART ROAD

Vous avez désormais un robot fonctionnel capable d'envoyer les informations collectées.

Il est maintenant temps de trouver un acquéreur (ou plusieurs) pour le système City Smart Road.

Le problème est que vous n'avez que le robot et pas l'interface.

Quoi qu'il en soit, vous avez décrochez un rendez-vous avec le maire (et son équipe technique) de la plus grande ville la plus proche et celui-ci est parfaitement disposé à entendre vos arguments vantant les mérites de votre robot et du système City Smart Road.

En plus de la démonstration de l'ensemble des fonctionnalités de votre robot, vous devez construire une présentation de l'interface City Smart Road qui permettra de voir votre ou vos robot(s) en action.

Plusieurs options sont envisageables selon votre état d'avancement. Elles sont présentées dans les parties suivantes.

Check 5

Vous allez réaliser une présentation (jeu de rôle) de l'interface de City Smart Road ainsi que des fonctionnalités de votre robot. L'objectif est de convaincre le maire que votre solution est la meilleure.

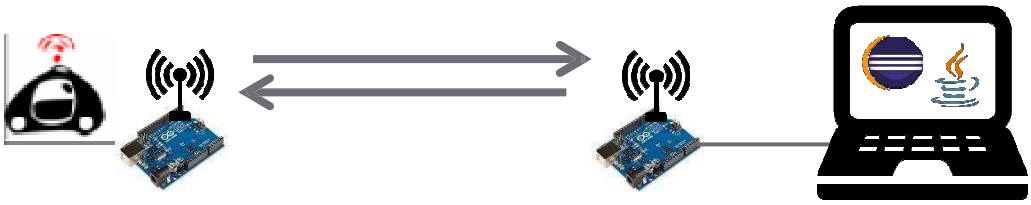
Il vous faudra faire preuve de conviction et de créativité pour décrocher ce contrat.

Prenez soin de bien structurer votre présentation et de faire preuve d'originalité !

2
INTERFACE 2D TEMPS REEL

Le point suivant, consiste à faire communiquer l'Arduino avec un programme (java ?) qui pilotera l'ensemble de l'application (et pourra servir d'interface avec une base de données).

Voici l'architecture que nous devrions obtenir.



Il existe bon nombre de pages web qui traitent de ce sujet, vous devriez rapidement trouver un exemple de code qui vous permettra de le mettre en œuvre.

Communiquer avec un programme Java

- https://cs.iupui.edu/~xiaozhon/course_tutorials/Arduino_and_Java_Serial.pdf
- https://vincentbarousse.wordpress.com/2017/03/30/ trashed/#more-610

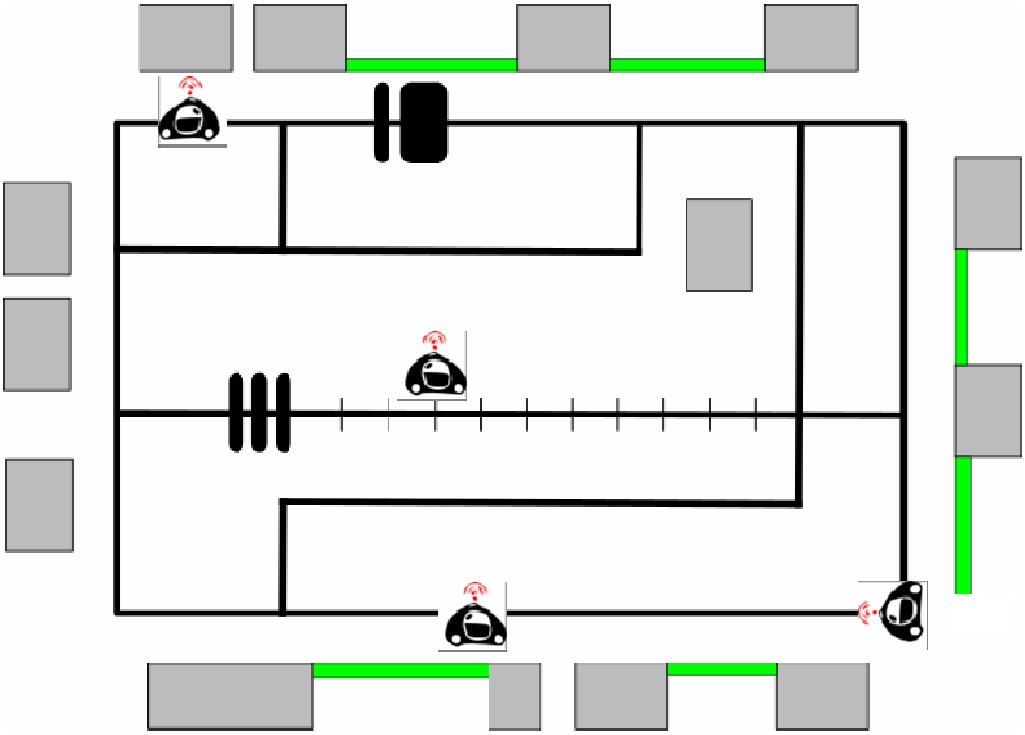
Le serveur doit être capable de modéliser es données pour qu'un opérateur soit capable de les lire en un instant. Vous allez pour cela construire une interface qui permet de visualiser la position du véhicule.

Cette interface sera construite grâce aux données de la ville et devra correspondre au plan au sol. Le robot peut toujours prendre des directions aléatoires, ce qui simulera un environnement urbain. Pour le localiser on peut utiliser plusieurs moyens :

- Soit sur les segments
- Soit sur les distances

Chaque méthode a ses avantages et ses inconvénients, à vous de choisir celle qui vous parait la plus pertinente.

Il est important de prendre en compte, qu'à terme, votre véhicule ne sera pas seul sur le plan, il faut donc à tout prix éviter les collisions. Vous pouvez mettre en place des moyens restrictifs pour empêcher les prototypes de s'engager un segment occupé.



On devrait donc avoir quelque chose de ce type sur l'écran de contrôle. Encore une fois, libre à vous d'adapter les informations des véhicules à l'écran (id, vitesse, …). Toute idée est la bienvenue.

Quel est le nombre maximal de voiture que l'on peut faire fonctionner sur cette carte ?

3
APPLICATION PRINCIPALE (OPTIONNEL)

Il est maintenant temps de mettre tout en œuvre pour que le prototype puisse collaborer avec d'autre et partager sa position en temps réel.

Chaque prototype démarre « nu », c'est-à-dire qu'il n'a pas les informations nécessaires à son fonctionnement. De plus, le pc principal ne le connait pas. Il y a donc une phase d'initialisation importante dans le projet.

Voici quelques étapes qui pourront vous aider à réaliser ce point :

- Le programme principal écoute via l'Arduino les trames qui arrivent sur sa fréquence, et qui
  +concerne l'application.

- Le nouvel ACAR, envoi des trames pour dire « je suis là, que dois-je faire ? »
- Le programme principal prend en compte cette demande, identifie le prototype et valide sa demande.

Lorsqu'un nouveau prototype est identifié, il doit apparaitre sur le coin gauche de l'écran et indiquer son statut. Lors de cette phase il doit indiquer « Connexion en cours... »

- Une fois ACAR et le serveur connectés, il faut réaliser l'envoi des données, ville et coordonnées afin de remplir les tableaux initialement crées.
- Les données ne pourront pas être transmises en une fois, il faudra donc un certain nombre de trames.

« Chargement des données... » Semble tout indiqué comme statut. Les plus avancés d'entre vous pourront même créer une barre de chargement.

- En possession de ses données le prototype peut maintenant se déplacer aléatoirement sur la carte.
- Lorsque le module de mesure enregistre une place disponible, il envoie une trame pour indiquer celle-ci. Et l'enregistrer dans la base de données.

Cette étape sera nommée, « En fonctionnement ». Elle permettra d'avoir un fonctionnement de base et de données aux utilisateurs les informations, sur les places de stationnement disponibles.

4
MODELISATION ET ARCHITECTURE (OPTIONNEL)

Afin que le système ACAR fonctionne de manière optimale, il faut désormais analyser et traiter cette masse d'informations qui vous arrive.

Afin d'avoir une vision claire des tâches à réaliser il est préconisé de découper l'application en modules, qui auront tous une fonction bien précise. Par exemple, le module communication, qui s'occupe de transmettre à l'Arduino émetteur la trame à envoyer.

De plus, un certain nombre de ces informations vont devoir être stockées. Pour effectuer les traitements nécessaires, le mieux est de l'insérer dans une base de données. Cela nous permettra, d'utiliser les requêtes ou les procédures. Le choix du SGBD a été arrêté sur Mysql.

En ce qui concerne la base de données, elle sera divisée en x parties :

- Ville
  - Structure
    Stockage des informations sur la structure de la ville. On déporte les données stockées en dur sur le prototype.
  - Description
    Données générales sur la ville
  - Stationnement
    Référencement des places disponibles dans la ville.
- Utilisateurs
  Données utilisateurs
- Véhicules
  - Informations générales sur les véhicules
    Marque
    Dimensions
    Constructeurs
- Déplacements

    - Référencement des déplacements
- Pool de demande
    - Liste des demandes de stationnement
- Stationnement
    - Coordonnées des places
    - Dimensions
- Historique
    - Historique de tous les mouvements d'information dans les tables.

Bien sûr, vous êtes libre d'ajouter des bloc et tables à votre guise. Afin d'éviter toute erreur il est fortement recommandé de transmettre à l'équipe ACAR, un MCD.

D'autres traitement purement liées à la base de données viendrons se greffer après selon les besoins de l'application, comme par exemple des Trigger et des procédures.

Parlons maintenant de l'application Java qui va être en charge du pc de contrôle. Le code aura surement, de nombreuses modifications ou amélioration, il aura donc besoin d'être modulaire et très flexible.

Le modèle MVC sera de rigueur, pour gérer de manière les différentes entités liées à la base de données. De plus, il serait vraiment pertinent de réaliser des packages par domaines fonctionnel.

Il convient donc de réaliser différents diagrammes afin de s'assurer que l'ensemble du programme sera cohérent. Vous fournirez à l'équipe projet un diagramme de classe par package et un diagramme de composant.

N'hésitez pas à réutiliser d'anciens projets qui comportent une structure fiable et organisée, pour ne pas perdre de temps. Les outils de travail collaboratif ne sont pas non plus à exclure.

Rappel important, les diagrammes rendus ne verrouillent en aucun cas l'application, vous pouvez à tout moment modifier ceux-ci. Ils feront l'objet d'une base d'échanges avec l'équipe qui supervise le projet.

5
INTEGRATION DES DONNEES (OPTIONNEL)

Maintenant que le pc de contrôle, peut faire l'interface entre la base de données et l'Arduino. Il faut traiter celle-ci et les intégrer.

Pour cela, vous allez d'abord supprimer toutes les données stockées de manière statique dans l'Arduino, comme par exemple le plan et les coordonnées des intersections. Ensuite, vous allez ajouter toutes les données nécessaires au bon fonctionnement de l'application telles que l'identification des véhicules, leurs marques et constructeurs. (Attention quelques exemples suffiront).

Une fois ces données intégrées, nous allons faire fonctionner ACAR, pour alimenter la base en temps réel, ainsi nous pourrons développer le modèle collaboratif.

6
MODELE COLLABORATIF (OPTIONNEL)

Cette étape est relativement simple, il faut ajouter un bouton de réservation pour chaque véhicule sur l'interface de contrôle. Ainsi lorsqu'un utilisateur veut trouver une place de stationnement il lui suffit de presser celui-ci.

Ensuite la démarche est simple, le système met en opposition les places disponibles et la taille de la voiture. S'il y a une correspondance, le prototype est alors emmené en face de l'emplacement et patiente une minute avant de repartir.

Au prochain passage le véhicule détectera cet emplacement comme vide et de nouveau disponible.

7
STATISTIQUES (OPTIONNEL)

A ce stade le prototype et le système sont fonctionnels. Mais il manque encore un point important pour l'utilisateur du système. Rien n'est plus frustrant que ne pas savoir, il faut donc lui indiquer un certain nombre d'informations.

- Le nombre de places disponibles pour son véhicule
- Le temps d'attente moyen
- Le temps d'attente « normal »
- Le pourcentage de chance d'obtenir une place en cas e surcharge.

8
REPLICATION DU MODELE ET EXPANSION (OPTIONNEL)

Aujourd'hui vous avez travaillé sur une petite ville. Nul doute que le modèle collaboratif à l'echelle de Paris ou New York sera beaucoup plus grand.

Afin de voir l'impact d'une ville beaucoup plus grande vous allez répliquer le modèle en mode miroir, vertical et horizontal à un instant T donné.

Il sera intéressant d'avoir l'impact la capacité de circulation et sur le

modèle statistique. Exemple de plan résultant :