Analyse 2 :

Besoins technologiques

1) Support électronique et logiciel :

Le robot doit pouvoir supporter un logiciel d’assistant vocal ainsi qu’un système électronique embarqué. Il doit donc pouvoir traiter des données pour en ressortir une action électronique ou bien une action logicielle.

Créer un assistant vocal sur un Arduino est bien trop compliqué. Il a donc fallu trouver une solution où le logiciel peut tourner et envoyer des commandes pour la partie électronique. Un Raspberry Pi est une solution idéale. Il est petit, puissant et fonctionne comme un serveur. Seulement, sa consommation est très élevée. Il n’est donc pas possible de directement l’intégrer au robot.

Le système embarqué du robot est géré par un Arduino. J’ai déjà pu réaliser de nombreux projets avec celui-ci et connait bien ses points forts et ses faiblesses.

Le choix d’avoir pris un Raspberry Pi et non un serveur est que le prix de celui est fixe contrairement à un serveur où il faudrait payer un abonnement et que le prix du Raspberry Pi serait vite atteint.

2) Communication électronique – logiciel :

La communication entre ces deux technologies se fera à l’aide du Bluetooth.

Pour cette partie, j’avais le choix entre l’utilisation du Wifi ou bien l’utilisation du Bluetooth.

Comparaison non exhaustive entre ces 2 technologies :

|  |  |  |
| --- | --- | --- |
|  | Bluetooth | Wifi |
| Prix | Bas | Élevé |
| Bande Passante | +- 800 Kbps | +- 11 Mbps |
| Configuration matérielle requise | Adaptateur Bluetooth sur tous les supports qui communiquent | Adaptateur Wifi sur tous les supports qui communiquent, un routeur Wifi et un point d’accès Wifi |
| Portée | 5 à 30 mètres | 32 à 95 mètres |
| Consommation énergétique | Basse | Élevée |

Source : [*https://www.diffen.com/difference/Bluetooth\_vs\_Wifi*](https://www.diffen.com/difference/Bluetooth_vs_Wifi)

Celui-ci étant déjà intégré dans le Raspberry Pi, il sera facile de l’utiliser au niveau de la partie logicielle. Une bibliothèque nommée « Bluetooth » est disponible en python et permet de l’utiliser facilement dans le programme.

Du côté de l’électronique, il suffit d’utiliser un adaptateur Bluetooth pour pouvoir utiliser cette méthode de communication. Il existe plusieurs modèles. Pour mon prototype, il faut un module qui possède une communication bidirectionnelle.

Je vais donc utiliser le HC-05 communiquant en RX-TX avec l’Arduino.



[*https://ram-e-shop.com/wp-content/uploads/2018/09/kit\_bluetooth\_hc05.jpg*](https://ram-e-shop.com/wp-content/uploads/2018/09/kit_bluetooth_hc05.jpg)

Il peut être utilisé en mode esclave ou bien en mode maître.

L’Arduino utilise la bibliothèque « SoftwareSerial.h » qui va permettre d’utiliser le Bluetooth dans celui-ci.

L’échange d’information se fera à l’aide de mots clés prédéfinis. Cela va permettre de faciliter la rédaction du code dans les deux parties.

3) Speech to text :

Il existe de nombreuses librairies et API sur Python qui permettent de transformer une commande vocale en texte.

J’ai utilisé l’API « speech\_recognition » de Google. Elle fonctionne mais le problème est qu’elle dépend de la qualité du réseau Internet. Si on est loin du Wifi ou même hors connexion, celle-ci ne fonctionnera pas.

Une librairie de reconnaissance vocale hors ligne peut également être une solution.

4) Synthèse vocale :

Comme le « speech to text », il existe de nombreuses librairies et API pour générer une synthèse vocale en Python.

L’API de Google « gTTS » (Google Text-to-Speech) est la plus adéquate à mon projet. Elle est très facile à utiliser et est très bien documentée.

Comme pour le « speech to text », la vitesse pour générer la synthèse vocale dépend de la qualité du Wifi.

Cette API prend une chaîne de caractères en entrée et ressort un fichier audio contenant la synthèse vocale de la chaîne de caractères.

Pour éviter une attente trop longue pour générer le fichier lorsque le Wifi est de mauvaise qualité, il est possible de charger à l’avance les fichiers audio pour pouvoir les utiliser par après sans devoir les générer avec le Wifi.

Cela ne fonctionnerait juste pas pour les fichiers audio avec une réponse personnalisée en fonction de la requête.

4) Ajouter un évènement dans un calendrier :

Le calendrier doit servir à mettre un évènement avec une date de début et une date de fin.

Une simple base de données avec des requêtes SQL auraient été suffisant. Mais utiliser l’API de Google pour le calendrier apporte bien plus de fonctionnalités. Notamment la synchronisation des données avec un compte Google pour pouvoir lire les évènements depuis un ordinateur ou bien un smartphone

L’utilisation de cette API est donc bien plus justifiée que d’utiliser un calendrier en local.

Il faut cependant que l’utilisateur puisse créer un compte ou bien se connecter à l’avance depuis un smartphone.

5) Effectuer un rappel d’une tâche :

L’utilisateur doit pouvoir mettre un rappel de x minutes et le sujet de la tâche.

Pour cette fonctionnalité, je compte utiliser la notion de « programmation parallèle » avec les Threads.

Lorsque l’utilisateur demandera le rappel d’une tâche dans x minutes, le programme lancera un Thread de durée x et qui lancera une notification à la fin avec la tâche qu’il fallait rappeler.

Le programme notifiera le rappel toutes les minutes tant que l’utilisateur n’a pas demandé l’arrêt du rappel.

6) Avoir la météo :

L’utilisateur doit pouvoir avoir la météo en précisant le lieu et le jour.

Il existe de nombreuses API pour fournir la météo. Certaines sont cependant payantes ou bien limitent le nombre de requêtes quotidiennement.

L’API « openweathermap » possède de nombreux avantages et ayant déjà travaillé avec, elle me paraît adéquate à mon projet.

Elle inclut plus de 200 000 villes et propose la météo jusqu’à 16 jours à venir.

Elle fonctionne simplement en une requête URL avec des paramètres et retourne le résultat en JSON.

7) Faire une recherche Internet :

8) Recevoir une alerte lors d’un évènement dans le calendrier :

L’utilisateur doit pouvoir recevoir une notification lorsqu’il a un évènement à venir dans le calendrier.

Il doit être notifier le matin même, puis 2H avant et 10min avant.

Contrairement aux rappels d’une tâche, la notification n’attendra pas une interaction de l’utilisateur pour s’arrêter.

Cependant, l’utilisateur peut, s’il le souhaite, désactiver la notification des évènements du calendrier. Mais il pourra toujours demander quels sont les évènements de la journée de d’un jour précis.

Comme le rappel d’une tâche, j’utiliserai les Threads. Un Thread tournera en fond et vérifiera chaque jour si un évènement est à venir prochainement à l’aide de l’API Google calendrier.

9) Se déplacer de manière autonome dans une pièce :