UPSSITECH
ÉCOLE D'INGÉNIEURS

Interface Humain-Machine
Fusion Multimodale

Auteurs	Constant ROUX Larissa XAVIER VITOR
Année	SRI 2024 - années 2023-2024
Date	12 novembre 2023
Ecole	UPSSITECH
Encadrant	Philippe TRUILLET

Table des matières

1 Introduction

Le présent bureau d'étude se consacre à la spécification, la conception et l'implémentation d'un moteur de fusion multimodal innovant destiné à interagir avec une palette de dessin dépourvue de tout bouton physique. L'objectif central de ce projet est de permettre la création, le déplacement, et la suppression de formes sur cette palette par le biais de modalités diverses, notamment la reconnaissance de la parole via l'agent Ivy SRA5, la reconnaissance de gestes grâce à l'agent Ivy 1 Recognizer, ainsi que le pointage à l'aide de la souris sur la palette.

Le problème à résoudre consiste en la création d'un moteur de fusion robuste qui harmonise ces modalités disparates pour offrir une expérience utilisateur fluide et cohérente. Ce moteur de fusion multimodal sera au cœur de l'architecture développée, exploitant le bus Ivy pour assurer une communication efficace entre les différentes modalités.

Le cahier des charges délimite clairement les fonctionnalités attendues, telles que la capacité de créer, déplacer, supprimer des formes, et enfin, quitter l'application. L'ensemble de ces actions sera réalisé en utilisant le langage de programmation Python.

Ce projet s'inscrit dans une démarche novatrice, explorant les frontières de l'interaction homme-machine en tirant parti de la complémentarité des modalités choisies. La réussite de ce bureau d'étude reposera sur la qualité de la conception de l'architecture multimodale et la mise en œuvre efficace du moteur de fusion, promettant ainsi une expérience utilisateur immersive et intuitive.

2 Fonctionnement général

2.1 Principe

Le principe général du logiciel est décrit sur la figure 1. Nous exploitons trois interfaces principales (bloc **Interface**) : la souris, le microphone avec le module SRA5 pour la reconnaissance de la parole, et une caméra avec le module 1-Recognizer pour la reconnaissance de gestes. Chaque interface génère différents types de données, associées à un niveau de confiance spécifique :

- **Souris** : L'utilisateur utilise la souris pour envoyer des clics, qui sont convertis en données de type POSITION (bloc **Click to Data**) avec un score de confiance de 100 % ;
- **Microphone (SRA5)** : Le microphone, via le module SRA5 et un modèle de dialogue pour la reconnaissance de la parole, permet à l'utilisateur d'envoyer des données de types variés telles que FORME, COULEUR, ou ACTION (bloc **Voice to Data**). Le score de confiance par donnée extraite dans la phrase enregistré est celui donné par SRA5 pour la phrase entière ;
- **Caméra (1-Recognizer)** : La caméra est utilisée pour capturer des gestes de l'utilisateur, qui sont ensuite interprétés comme des données de type FORME en utilisant 1-recognizer (bloc **Gesture to Data**). Le score de confiance associé est celui renvoyé par 1-recognizer.

Une fois les données extraites, elles sont organisées dans une File d'Attente (bloc **FIFO**). La FIFO agit comme un lien entre les données utilisateur, arrivant de manière aléatoire et imprévisible, et le **Process Thread**.

Le Process Thread (détaillé dans la section 2.2) intervient pour :

- filtrer les données en fonction de leur taux de confiance par rapport à un seuil de confiance (bloc **confidence filter**) ;
- ranger les données filtrées dans un tableau (bloc **SlotArray**) ;
- gérer un compte à rebours pour réinitialiser le tableau de données ;
- détecter une commande cohérente à partir du tableau de données ;
- tester et exécuter une commande en communiquant avec le **App Thread**.

.

Le App Thread (détaillé en section 2.3), quant à lui, gère l'interaction avec la palette en créant, modifiant, ou supprimant des formes. Lorsqu'une commande potentielle est proposée par le Process Thread, le App Thread vérifie sa cohérence. Si la commande est réalisable, elle est exécutée ; sinon, le choix est renvoyé au Process Thread. En cas d'impossibilité de la commande, le Process Thread réinitialise le tableau de valeurs et le compte à rebours. En cas de multiples possibilités, le système attend d'autres données de l'utilisateur, maintenant ainsi une interaction fluide et réactive avec la palette de dessin.

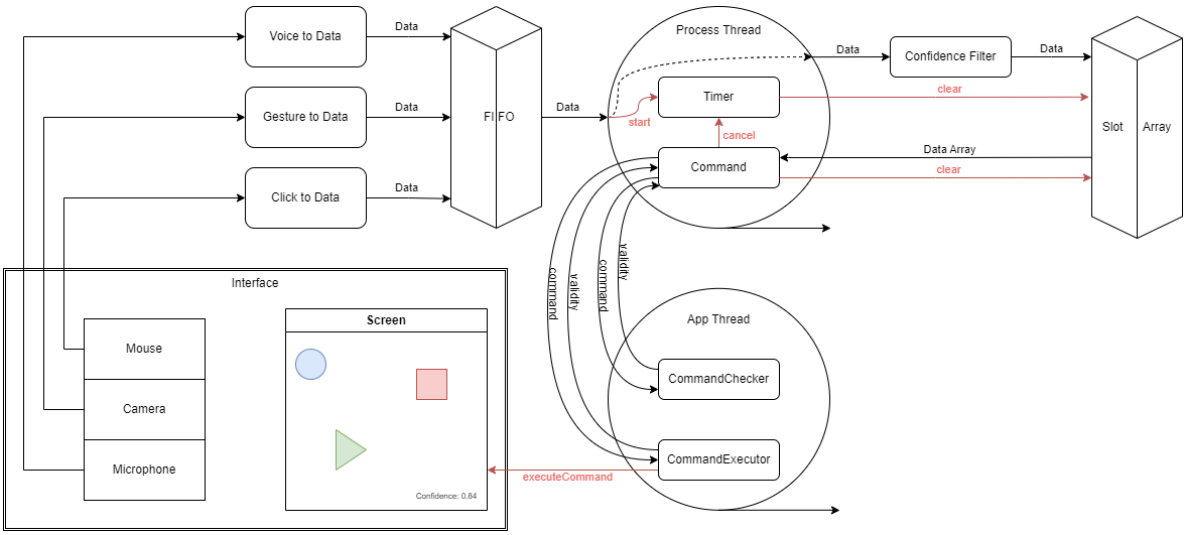

FIGURE 1 – Architecture principale du système

La mise en oeuvre logicielle suit le diagramme présenté, tout en utilisant le bus Ivy. Une autre subtilité pour des raisons techniques et que l'objet FIFO se trouve dans la classe SlotArray. Un diagramme de classe (figure 2) est présenté pour résumer l'architecture et les liens entre les différentes classes tout en mettant en avant le bus Ivy sous forme de classes héritantes.

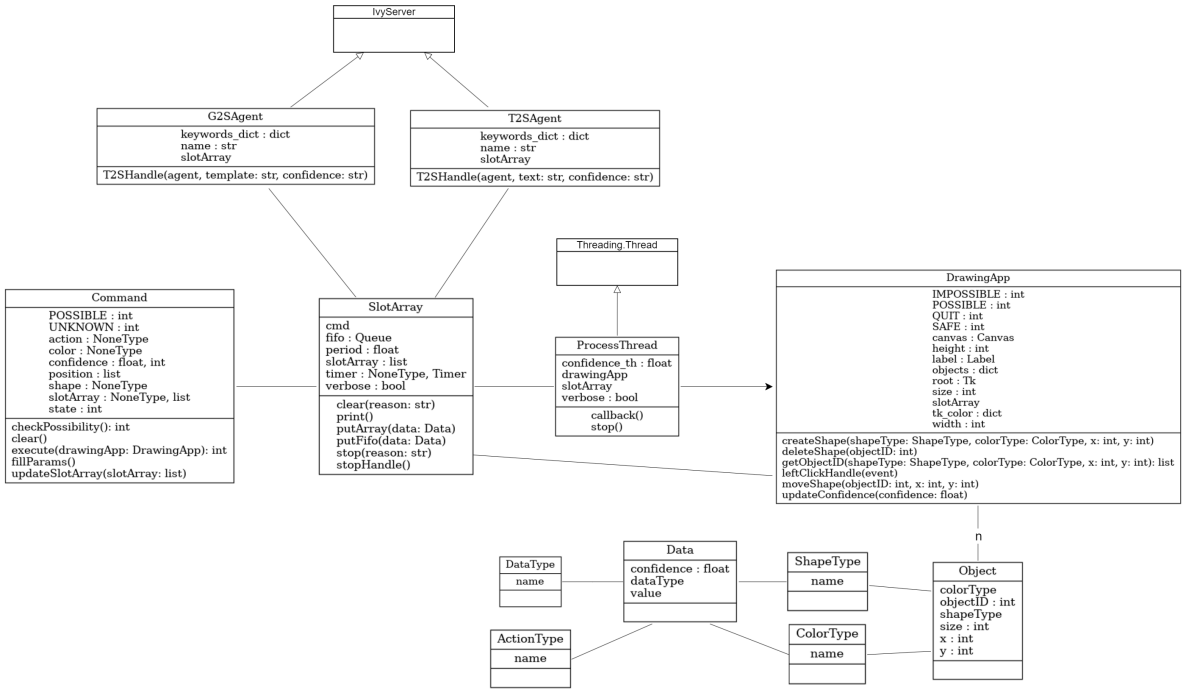

FIGURE 2 – Diagramme de classe du logiciel

2.2 Process Thread

Nous proposons ici de détailler le fonctionnement du Process Thread étape par étape selon les machines à états de la figure 3 :

— **Etat 0** : Dans cet état, le thread lit une donnée présente dans la FIFO et se rend à l'état 1. Si rien n'est présent, le thread attend ;

— **Etat 1** : Ici, le thread applique le filtre à la donnée lue. Si le taux de confiance de la donnée est supérieure au seuil, alors la donnée est acceptée et la machine passe à l'état 2. Sinon, le donnée est rejetée et la machine repasse à l'état 0 ;

— **Etat 2** : La donnée est ajoutée au tableau de données et ls sytème se rend à l'état 3. Si le tableau de données est vide, alors la donnée entrante est la première ce qui lance le compte à rebours. Ce compteur permet de réinitialiser le tableau si l'utilisateur ne fournit pas de données complémentaires en un certain temps ou si les données sont trop nombreuses et ne permettent pas de choisir la bonne commande à exécuter. Ce mécanisme est géré par les états 5 et 6 ;

— **Etat 3** : Avec les données actuelles, le thread essaie de voir s'il est possible de générer une commande possible théoriquement (sans tenir compte de l'état de la palette). Si cela est possible, nous passons alors à l'état 4, sinon à l'état 0 pour obtenir de nouvelles informations ;

— **Etat 4** : Maintenant que nous avons une commande possible, il faut l'envoyer au App Thread pour vérifier que celle-ci est bien exécutable. Si celle-ci est exécutable, elle est alors réalisée, le compteur et le tableau de données se réinitialisent. Si elle n'est pas exécutable, le compteur et le tableau de données se réinitialisent. Sinon, nous repassons à l'état 0 pour obtenir de nouvelles informations et supprimer les ambiguités éventuelles empêchant de savoir si la commande est oui ou non exécutable.

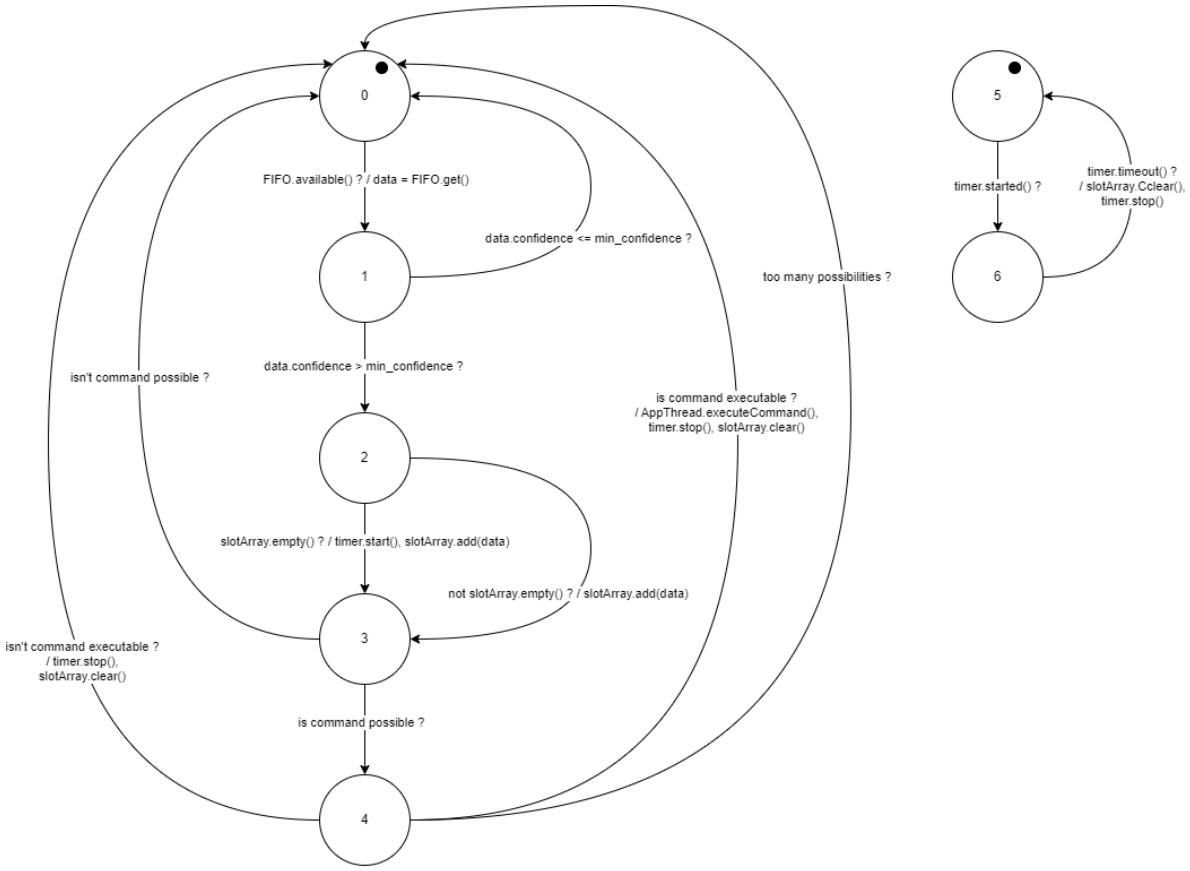

FIGURE 3 – Machine à états décrivant le fonctionnement du Process Thread

2.3 App Thread

Le App Thread est le thread principal du projet sur lequel l'interface graphique est mise à jour. Ce dernier implémente les fonctions suivantes :

— **createShape** : permet de créer une forme. Elle prend en entrée une forme, une couleur et une position et crée sur l'interface la forme correspondante avec un ID associé afin de l'identifier ;

— **deleteShape** : permet de supprimer une forme. Elle prend en entrée l'ID de la forme ;

— **moveShape** : déplace une forme. Elle prend en entrée l'ID de la forme ainsi que la position de destination ;

— **getObjectID** : trouve la liste des ID des formes possibles en fonction des données en entrée. Les données d'entrée sont comparées avec les autres objets uniquement dans les cas proposés par la table de vérité 1. Plusieurs objets peuvent correspondre aux données d'entrée, ce qui demande alors au Process Thread d'autres informations pour conclure sur la commande à réaliser.

TABLE 1 – Table de vérité permettant de choisir la combinaison de données à vérifier

forme	couleur	position	à vérifier
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

3 Fonctionnalités

3.1 Données

Cette section détaille les différentes entrées et informations manipulées par notre système de fusion multimodale.

3.1.1 Actions

Les actions possibles dans notre système de fusion multimodale sont déterminées par les commandes vocales. Les actions autorisées comprennent :

— **Ajouter** : ajouter une certaine forme d'une certaine couleur à un certain endroit ;

— **Déplacer** : déplacer une certaine forme existante à un certain endroit ;

— **Supprimer** : supprimer une certaine forme existante ;

— **Quitter** : quitter l'application (interface graphique et Ivy).

3.1.2 Formes

Les formes manipulables dans notre application sont le **rectangle**, le **cercle** et le **triangle**. L'utilisateur peut spécifier la forme souhaitée lors de l'utilisation de la voix ou des gestes.

3.1.3 Couleurs

Les couleurs **rouge**, **vert** et **bleu** associées aux formes peuvent être définies par des commandes vocales.

3.1.4 Position

La position des commandes est un vecteur (x, y) déterminé par les entrées de la souris relativement à la fenêtre de l'application. Cela permet notamment d'indiquer où créer/déplacer une forme ou en sélectionner une.

3.2 Commandes

Cette section détaille les différentes commandes que le système est capable d'exécuter.

3.2.1 Création d'un objet

La commande de création d'objet nécessite que l'utilisateur donne une forme, une couleur et une position dans n'importe quel ordre avant que le compte à rebours ne se termine (figure 4). La reconnaissance vocale n'étant pas "atomique", il est difficile d'associer temporellement par exemple le clic souris au mot "ici". Aucune valeur par défaut n'est assignée et la commande ne sera pas réalisée s'il manque un des types de données.

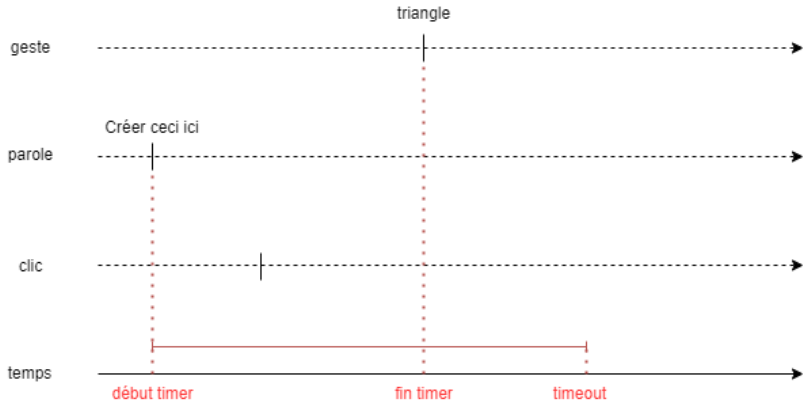

FIGURE 4 – Chronogramme pour la création d'un objet (commande valide)

3.2.2 Déplacement d'un objet

La commande de déplacement d'objet nécessite que l'utilisateur donne assez d'informations pour identifier l'objet à déplacer (voir section 2.3) ainsi qu'un clic souris pour déterminer la destination de l'objet. Si un clic souris est aussi utilisé pour déterminer l'objet à déplacer, le premier clic correspondra à l'identification de l'objet et le second à la destination (figure 5). Si plusieurs objets sont identifiés, la commande ne sera pas exécutée tant que l'utilisateur n'aura pas donné de nouvelles informations permettant de n'en avoir plus qu'un.

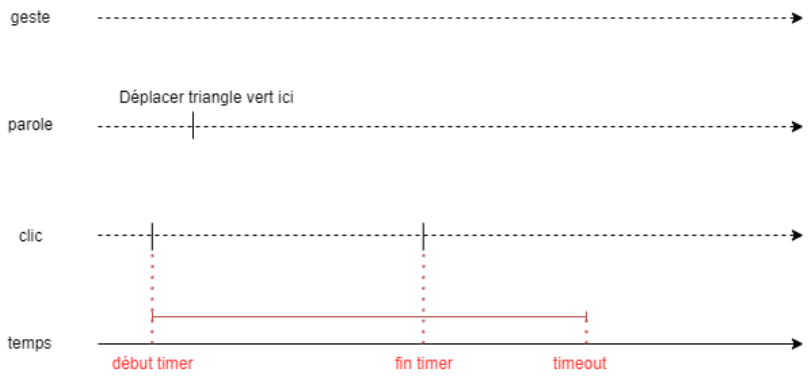

FIGURE 5 – Chronogramme pour le déplacement d'un objet (commande valide)

3.2.3 Suppression d'un objet

A l'instar du cas précédent, la commande de suppression d'objet nécessite que l'utilisateur spécifie assez d'informations pour identifier l'objet à déplacer. Seul un objet doit être identifié et non une liste.

3.2.4 Sortie du logiciel

L'utilisateur doit simplement dire un des mots clefs spécifiés dans la grammaire pour quitter l'application.

4 Limites

Cette section présente les limites de la conception et de l'implémentation de notre système de fusion multimodale tout en soulignant les possibles axes d'amélioration.

4.1 Traitement de la parole

Le logiciel traitant la parole renvoie les informations phrase par phrase. Ce mécanisme limite fortement le système car il est alors impossible d'associer temporellement un mot à une donnée (exemple du mot "ici" avec le clic souris). L'axe d'amélioration le plus simple serait d'envoyer les données brutes de la reocnnaissance vocale mot par mot et de traiter ces données à la volée.

4.2 Types de données

Les types de données sont pauvres (seulement 3 couleurs et 3 formes) et pourraient donc être enrichis. Il est aussi possible d'ajouter de nouveaux types de données pour localiser les objets relativement entre eux ("à gauche de", "au-dessus de", etc) ou détecter certains mots clefs pour réaliser des commandes différentes (comme "ici", "là", etc).

4.3 Mécanisme du compteur de temps

Le compteur de temps tel qu'il est implémenté limite fortement le système :

— si l'utilisateur commet une erreur, il est obligé d'attendre la fin du compteur ;

— si plusieurs possibilités de commande sont détectées, aucune n'est retenue à la fin du compteur (système trop rigide) ;

— le temps alloué peut ne pas être convenable pour tous les utilisateurs (système n'englobant donc pas tous les utilisateurs possibles).

Il est possible de remplacer ce compte à rebours par un modèle de langage sophistiqué traitant les informations de manière atomique pour détecter les erreurs, les temps de réflexion et pour s'adapter au débit de mots de l'utilisateur.

5 Résultats

Une vidéo de démonstration de toutes les fonctionnalités du projet est présente dans le code source du projet à cette adresse.