M3SOL033CM
Serveurs vocaux interactifs

Fanny Ducel

Projet 4 : Développement et déploiement d'un serveur vocal interactif

Master 2
« Langue et informatique »

Table des matières

Table des figures

1 Introduction

L'objectif de ce dernier projet est de modéliser, réaliser et déployer un serveur vocal interactif réaliste, proposant un service dans des conditions réelles.

Notre serveur devra donc proposer un service en énonçant diverses informations, et en récoltant des données de la part de l'utilisateur afin de maximiser son interactivité et de rendre les conversations agréables et ludiques.

Pour cela, nous utiliserons le langage VoiceXML [1] combiné avec des programmes écrits en JavaScript ainsi que des grammaires hors-contexte (CFG) stochastiques.

Nous déploierons ensuite cette application sur la plateforme Voxeo (`https://evolution.voxeo.com`), afin de pouvoir interagir avec notre serveur par téléphone.

Depuis la France, il suffit d'appeler le 01.82.88.24.99 et de composer le code d'application **9990522355** pour accéder à l'application développée et décrite dans ce rapport.

Nous avons décidé de créer un serveur vocal interactif de recommandations de recettes de cuisine végétariennes. Nous détaillerons ce choix et cette tâche dans la section 2, puis nous listerons le lexique actif de notre service, classifierons les dialogues et leurs arborescences, commenterons la synthèse utilisée, décrirons les grammaires de reconnaissance et tables de données conçues et établies avant de nous intéresser au déploiement final de l'application sur Voxeo.

L'arborescence du projet contient :

- des fichiers vxml pour les scripts des applications : *recette_app_js.vxml* et *version_anglaise.vxml*
- des fichiers vxml pour les grammaires de reconnaissance : *ouinon.vxml, yesno.vxml, recettes_grammaire_en.vxml, recettes_grammaires_js.vxml, choix_recette_grammaire_stoch.vxml, choix_vegetarisme_grammaire_stoch.vxml*
- des fichiers cfg, train, gen et pcfg utilisés pour générer les grammaires hors-contexte stochastiques
- des fichiers js contenant les programmes Javascript utilisés
- des fichiers wav avec des conversations enregistrées

1. Documentation et tutoriels : `https://voicexml.org/`, `https://www.w3.org/TR/voicexml20/`, `https://wiki.voximal.com/doku.php?id=start`

2 Choix d'une application et modélisation de la tâche

2.1 Choix et description de la tâche

Nous avons choisi de réaliser un serveur vocal interactif de recommandations de recettes de cuisine végétariennes.

L'idée centrale est qu'un utilisateur qui souhaite cuisiner mais n'a pas d'idée de recette puisse appeler et énoncer un ingrédient qu'il souhaite utiliser, et que le serveur lui propose une idée de recette végétarienne mettant en valeur l'ingrédient sélectionné.

Nous ajoutons quelques sous-tâches plus précises afin d'améliorer la qualité de l'interaction. Ainsi, après avoir annoncé l'ingrédient désiré, le système propose une recette adéquate (aléatoirement parmi 3 options, car chaque ingrédient est relié à 3 recettes indexées dans notre base de données) en indiquant les ingrédients nécessaires et le temps de préparation correspondant. L'utilisateur doit confirmer que ce choix lui convient avant que la recette ne débute. Une fois confirmé, le serveur énonce les différentes étapes de la recette une à une. Après chaque étape, l'utilisateur doit confirmer qu'il a bien compris l'instruction ou demander de répéter ou d'attendre quelques secondes. On ne passe à l'étape suivante qu'une fois que l'utilisateur a confirmé l'étape. Une fois que toutes les étapes ont été lues, la recette terminée : l'appel est interrompu.

Par ailleurs, nous avons ajouté quelques autres fonctionnalités annexes.

Au début de l'appel, l'utilisateur doit choisir s'il souhaite faire une recette (les sous-tâches décrites ci-dessus sont donc alors déclenchées), en apprendre plus sur le végétarisme, sur ce projet, ou sur les objectifs du serveur vocal. Si l'une de ces trois options est choisie, alors un prompt est lu, contenant les informations désirées, puis, l'utilisateur est renvoyé sur le menu d'accueil.

Finalement, l'utilisateur peut choisir de "parler anglais". Dans ce cas, la version anglaise de l'application est utilisée. Il s'agit de la même application, mais avec tous les prompts et données traduits.

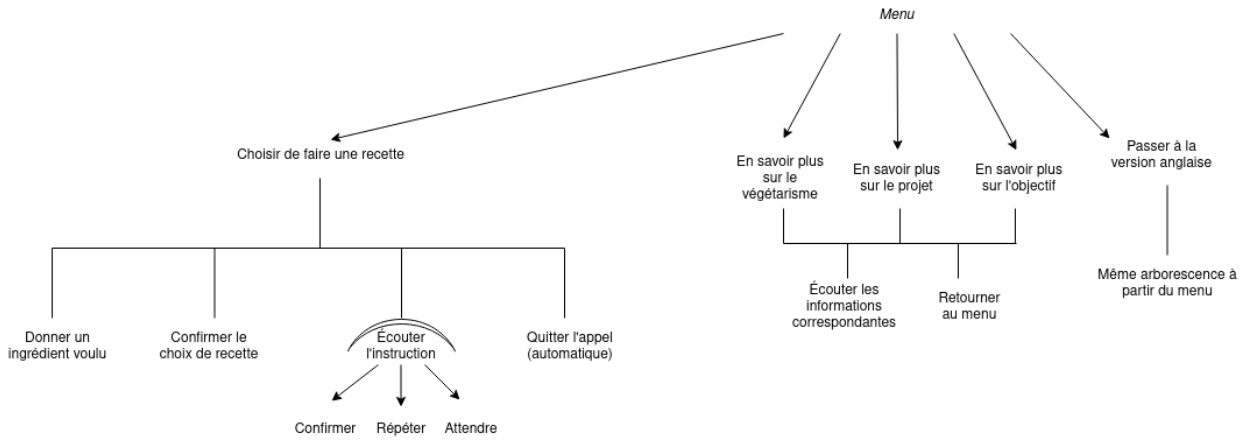

FIGURE 1 – Modélisation HTA de l'application

2.2 Modélisation de la tâche

Nous récapitulons les différentes tâches et sous-tâches décrites précédemment dans un modèle de tâche HTA (voir Figure 1).

2.3 Spécificité : recettes végétariennes

Nous avons choisi d'utiliser uniquement des recettes végétariennes par conviction personnelle, mais aussi pour créer une spécificité à notre application et la démarquer d'autres applications similaires.

De plus, cela crée un enjeu politique plus intéressant, et permet également d'atteindre un public cible plus large (les personnes avec des restrictions alimentaires concernant la viande peuvent utiliser ce service tandis que l'inverse n'est pas vrai). Par ailleurs, cela permet également à sa conceptrice d'utiliser ce service en conditions réelles, au-delà du cadre de ce projet.

Nous utilisons également cette spécificité afin de créer un aspect éducatif et préventif à notre application, qui recommande par exemple d'énoncer un ingrédient qui atteint sa prévention pour limiter le gaspillage, et qui propose également d'énoncer quelques bienfaits du végétarisme.

3 Lexique actif

Afin de réaliser notre serveur, nous devons déterminer son lexique actif.

Nous constituerons par la suite plusieurs grammaires de reconnaissance à partir de ce lexique, dont deux grammaires hors-contexte stochastiques.

Nous nous intéressons ici aux mots qui devront être prononcés par l'utilisateur et donc reconnus par le système. Nous choisissons de rassembler ces lexiques selon les thématiques.

— un lexique pour le menu : basé sur les mots clés centraux "recette", "végétarisme", "projet", "objectif"
— un lexique d'ingrédients : il correspond à une liste de 30 ingrédients (lentilles, pâtes, carottes, aubergines, ...) [2]
— un lexique de confirmation : oui, non, attendez, répétez, pause, ...

Tous ces lexiques ont été traduits en anglais pour la version correspondante. De même, tous les prompts, grammaires et scripts ont été traduits et la langue des fichiers vxml a été changée de "fr-fr" à "en-US". La traduction s'est faite de plusieurs façons : manuellement, avec l'aide d'un traducteur automatique (en l'occurrence DeepL `deepl.com`), ou avec l'aide de ChatGPT (`https://chat.openai.com/auth/login`).

4 Dialogues et arborescence des unités de dialogue VoiceXML

Nous utilisons la classification des situations de dialogue de [Loisel, 2004], en nous centrant sur son axe dialogique, pour construire un ensemble représentatif de dialogues entre l'utilisation et le serveur, et ce pour chaque sous-tâche de notre modèle HTA nécessitant une interaction avec l'utilisateur.

Cette étape nous permettra de créer des grammaires de reconnaissance, et de déterminer dans quels cas l'utilisation de grammaires hors-contexte stochastiques est pertinente. Elle nous permet également d'évaluer l'agréabilité des interactions entre l'utilisateur et notre serveur.

Tout d'abord, le système *greet* l'utilisateur : "Bonjour, bienvenue sur notre serveur vocal de recettes de cuisine végétariennes [...]".

2. La liste complète est donnée en Annexes.

Puis, il entre dans un sous-dialogue, le menu, sous forme d'une *RequestDirectif* : "Souhaitez-vous faire une recette, en savoir plus sur le végétarisme, sur ce projet ou sur ses objectifs ? Ou souhaitez-vous parler anglais ?" L'utilisateur doit alors répondre avec une phrase ou un mot-clé. Nous imaginons qu'un ou deux mot-clés sont suffisants pour les trois dernières options : prononcer "projet", "objectifs", ou "parler anglais". Toutefois, nous imaginons plusieurs formulations possibles pour l'utilisateur qui souhaite faire une recette ou en savoir plus sur le végétarisme. Par exemple, "Je veux cuisiner", "Donne moi une recette", "Je voudrais préparer une recette végétarienne". De même pour l'option "végétarisme", à laquelle on pourrait accéder en disant des phrases comme : "Je voudrais en apprendre davantage sur le végétarisme", "Donnez moi des informations sur le végétarisme" et autres dérivées sur ce même modèle. Nous décidons également de renvoyer vers cette sous-tâche un utilisateur qui demanderait de la viande. Ainsi, nous devons réaliser des grammaires hors-contexte stochastiques pour ces deux cas. Nous détaillerons plus longuement ces explications dans la partie 6.

Si ce mot n'a pas été reconnu ou que rien n'a été dit, la machine *InformIntent* : "Dites l'un des mots suivants : recette, végétarisme, projet, objectif, parler anglais."

Si le choix de l'utilisateur s'est porté sur une autre sous-tâche que "faire une recette", la machine énonce les informations désirées, dans un but d'*Inform*. Puis, l'utilisateur est redirigé vers le menu.

À l'inverse, si l'utilisateur choisit de faire une recette, la machine pose une nouvelle question (*RequestDirectif*) afin d'obtenir une donnée : "Quel ingrédient voulez-vous utiliser dans votre recette ?". Si l'ingrédient n'a pas été reconnu, la machine demande pardon ("Désolée, je n'ai pas compris"). Sinon, elle répète ce qui a été compris pour *acknowledge*, puis confirme ("D'accord !"). Elle donne ensuite des informations (*Inform*) sur la recette proposée, les ingrédients nécessaires et le temps de préparation puis demande à l'utilisateur une confirmation ("oui"). Si cette confirmation n'est pas donnée, on retourne à l'étape de choix d'ingrédient. Sinon, on *acknowledge* et annonce l'entrée dans un sous dialogue (*InformIntent*) : "C'est parti !", tout en informant des options à venir ("N'hésitez pas à me demander d'attendre ou de répéter chaque instruction.").

Ensuite, le système informe (en récitant l'étape de préparation de la recette), puis, après chaque étape, il demande confirmation de la bonne compréhension (*requestDirectif*). Si un mot-clé (type "oui", "attends", "pause", "répétez", "répète") a été reconnu, le système *acknowledge* : "Très bien", "D'accord" avant de réaliser la suite des actions. Sinon, le système *InformIntent* : "Essayez plutôt un mot-clé comme "oui", "attends", ou "répétez"".

Finalement, quand la recette touche à sa fin, le serveur clôt l'interaction

avec un acte de type *bye* : "C'est fini, bon appétit ! À bientôt !".

On remarque que dans notre système, les demandes d'informations sont uniquement réalisées par le serveur.

Nous avons essayé de rendre les interactions les plus fluides et agréables possibles en intégrant beaucoup d'actes de confirmation, d'*acknowledge*, d'*inform* et de *greet*. Ainsi, l'utilisateur peut se laisser guider par le serveur, apprécier cet échange et y trouver une plus-value, en comparaison avec une requête dans une base de données traditionnelles, où les informations sont échangées sans aucune réelle interaction ou simulation d'émotion ou de conversation.

5 Synthèse et prononciation

Afin d'améliorer la synthèse, et plus particulièrement l'expressivité des messages, nous utilisons des balises SSML :

— des balises **voice** avec des attributs *gender, age* pour personnaliser les voix des premiers prompts et qu'elles soient toujours les mêmes au début des interactions

— des balises **emphasis** pour ajouter une proéminence sur les différentes options du menu, et ainsi inciter l'utilisateur à prononcer ces mots-clés emphasés pour choisir sa sous-tâche ; de même sur le titre de la recette, autre élément important qui doit susciter l'attention de l'utilisateur

— des balises **break** pour mettre des pauses : soit à la demande de l'utilisateur (pour attendre entre deux instructions de recette), soit pour faciliter la compréhension avant de retourner au menu

— une balise **prosody rate="slow"** afin de ralentir le débit, on l'applique à l'instruction qui a demandé à être répétée afin de faciliter la compréhension de l'instruction (qui n'était sûrement pas assez claire la première fois si l'utilisateur a besoin qu'elle soit répétée)

6 Grammaires de reconnaissance

Afin que le système puisse reconnaître les données entrées par l'utilisateur lors des demandes d'informations, nous construisons plusieurs grammaires de reconnaissance. Comme indiqué précédemment (section 4), nous créons deux grammaires simples ainsi que deux grammaires hors-contexte stochastiques.

Tout d'abord, nous nous intéressons aux grammaires simples.

La première, *recettes_grammaire_js.vxml* (et son équivalente en anglais *recettes_grammaire_en.vxml*), prévoit la règle permettant à l'utilisateur d'énoncer l'ingrédient qu'il souhaite utiliser.

Elle se compose d'une première partie, optionnelle (grâce à l'attribut *repeat="0-1"*), qui permet de créer une phrase commençant éventuellement par "je veux manger des/du", ou, "je veux cuisiner des/du", "je veux utiliser des/du", suivi d'un ingrédient. Puisque le début de la phrase est optionnel, il est donc possible d'énoncer simplement l'ingrédient désiré.

La grammaire contient alors ensuite 30 ingrédients communs, notamment des fruits et des légumes. La liste complète est donnée en Annexes.

La deuxième grammaire simple, *ouinon.vxml* (*yesno.vxml*) est utilisée à la fois pour confirmer le choix de la recette, mais aussi pour confirmer la bonne compréhension de chaque instruction. Elle contient alors des mots-clés uniques, que nous estimons suffisants et évidents pour confirmer, demander au système de s'arrêter ou de répéter : *oui, non, répétez, répète, stop, pause, attends, attendez.*

Toutefois, nous estimons que les interactions ayant lieu lors du menu principal (choix de la sous-tâche : recette, végétarisme, projet, objectifs, version anglaise) sont plus complexes, notamment pour les deux premières sous-tâches. Pour les trois dernières sous-tâches, nous utilisons un ou deux mots-clés, mais en utilisant l'attribut *accept="approximate"* de la balise **choice** afin d'étendre légèrement la reconnaissance.

Pour le choix de la recette ou des informations sur le végétarisme, nous souhaitons couvrir plus de cas d'interactions, car ce sont les sous-tâches les plus riches, et auxquelles les utilisateurs pourraient avoir accès en effectuant plusieurs types de demandes.

Ainsi, nous décidons de créer des grammaires hors-contexte stochastiques (*choix_vegetarisme_grammaire_stoch.vxml, choix_recette_grammaire_stoch.vxml*) [3].

3. À noter : nous créons ces grammaires uniquement pour la version française, puis-

Pour cela, nous devons d'abord créer des grammaires hors-contexte et définir nous-mêmes les règles de reconnaissance de nos grammaires. Nous les rédigons dans les fichiers .cfg.

Nous rédigons également différents exemples de phrases à reconnaître par nos grammaires, qui sont contenus dans les fichiers .train. Ces exemples contiennent des phrases, sous-phrases et variantes de "je veux préparer une recette végétarienne", pour *choix_recette.train* et "je veux en savoir plus sur le végétarisme" pour *choix_vegetarisme.train*. Nous décidons également d'envisager les cas de personnes qui demanderaient de cuisiner de la viande dans cet ensemble d'exemples, afin que, si cela arrive, elles soient redirigées vers les informations sur le végétarisme. Nous donnons les listes exhaustives d'exemples donnés dans les fichiers train en Annexes.

Ensuite, nous utilisons ces fichiers cfg et train pour obtenir nos grammaires stochastiques, avec probabilités, grâce à l'algorithme *Inside-Outside*, dans son implémentation en Python utilisée lors de l'atelier 2. Nous obtenons alors des fichiers pcfg, grâce auxquels nous pouvons créer nos fichiers voiceXML de grammaire en ajoutant les probabilités données dans l'attribut "weight" de nos items.

qu'elle est centrale. Pour la version anglaise, ces cas sont gérés par des grammaires simples, similaires à celles présentées précédemment

7 Tables de données

Notre application nécessite l'utilisation de différentes bases de données afin de pouvoir : lier l'ingrédient énoncé à des recettes pertinentes, lier la recette choisie à tous les ingrédients nécessaires à sa réalisation, lier la recette choisie à son temps de préparation, lier la recette choisie à ses étapes de préparation.

Pour cela, nous avons rédigé 5 fonctions JavaScript. L'une sert simplement à générer un chiffre aléatoire, afin de pouvoir tirer aléatoirement une recette pertinente. Les autres fonctions permettent de renvoyer les informations désirées, comme expliqué dans le paragraphe précédent. Elles sont basées sur des dictionnaires contenant les informations nécessaires, liées aux ingrédients et aux recettes.

Nous avons envisagé plusieurs moyens de collecter ces données sur les recettes de cuisine, comme par exemple de chercher une base de données existante et librement disponible, ou d'utiliser des méthodes de *Web-Scrapping* sur des sites de recettes. Toutefois, nous avons finalement opté pour une génération avec ChatGPT, en utilisant différentes méthodes de *prompt-engineering*[4] afin d'obtenir les données voulues, avec la structure désirée.

Par exemple, nous avons utilisé le prompt suivant pour obtenir nos dictionnaires d'ingrédients :

À partir de la liste de recettes végétariennes suivante : ['nouilles sautées au tofu', 'curry de tofu', 'salade de quinoa', 'curry de quinoa', 'bowl de quinoa', "curry d'épinards", 'tarte aux épinards', 'lasagne aux épinards', 'fajitas aux poivrons', 'tarte aux poivrons', 'poivrons farcis', 'céleri-rave rôti', 'velouté de céleri-rave', 'salade de céleri-rave', 'carottes rôties au cumin', 'curry de carottes', 'tarte aux carottes', "haricots verts sautés à l'ail", 'salade de haricots verts'], crée un dictionnaire python contenant la liste d'ingrédients nécessaires pour réaliser la recette. Par exemple : "pizza au fromage" : ["pâte à pizza", "sauce tomate", "emmental", "roquefort", "basilic"]

Après avoir rédigé et exécuté différents prompts, nous obtenons alors 30 ingrédients principaux que l'utilisateur peut demander. Pour chacun de ces ingrédients, 3 recettes végétariennes sont associées, pour un total de 88 recettes (2 recettes sont utilisées pour des mêmes ingrédients, par exemple la ratatouille, qui figure dans la liste de recettes pour les courgettes et pour les au-

4. Voir par exemple : `https://generative.ink/posts/methods-of-prompt-programming/`

bergines). Chacune de ces recettes est ensuite associée à sa liste d'ingrédients, à son temps de préparation, et à sa liste d'instructions.

Nous appelons ensuite ces fonctions dans nos programmes voiceXML en temps voulu, en combinaison avec des assignations de variables globales.

8 Déploiement sur la plateforme Voxeo

Nous déposons tous les fichiers créés (voiceXML, scripts, grammaires) sur Voxeo et créons une application dédiée. Il est ainsi possible d'accéder à notre serveur vocal par téléphone.

Depuis la France, il suffit d'appeler le 01.82.88.24.99 et de composer le code d'application **9990522355**.

De nombreux tests ont été effectués afin de valider différents scénarios et d'assurer le bon fonctionnement de notre application.

Nous avons également enregistré plusieurs conversations avec notre serveur vocal interactif, que nous ajoutons à l'arborescence du projet pour illustrer divers exemples d'interactions et d'utilisations de l'application. L'enregistrement pour la version française se veut exhaustif et teste toutes les fonctionnalités, tandis que l'enregistrement en anglais est plus libre et spontané, avec un utilisateur étranger à la conception de l'application, qui l'utilise donc en conditions réelles.

9 Conclusion

Combinant interaction humain-machine, reconnaissance de la parole et synthèse de la parole, ce projet nous a permis de penser, développer et déployer un serveur vocal interactif concret, utilisable en temps et conditions réelles, avec une véritable utilité et une dimension ludique, qui permettraient son utilisation au-delà du cadre de ce projet.

Utilisant divers programmes et grammaires en voiceXML ainsi que des scripts JavaScript, nous avons pu constituer une application qui, en se basant sur un ingrédient énoncé par l'utilisateur, propose une recette présente dans sa base de données de 88 recettes végétariennes et accompagne l'utilisateur dans sa réalisation, grâce à des unités de dialogues expressives et riches.

Notre application permet aussi de sensibiliser au végétarisme et au limitage du gaspillage alimentaire, lui donnant ainsi des enjeux et dimensions écologistes.

Références

[Loisel, 2004] Loisel, A. (2004). Annotation automatique des corpus de dialogues basée sur les recherches en pragmatique et l'observation des corpus. *Mémoire de DEA, Université de Caen.*

10 Annexes

10.1 Liste complète d'ingrédients

['lentilles', 'pois chiches', 'aubergines', 'champignons', 'courgettes', 'brocoli', 'poireaux', 'chou-fleur', 'patates douces', 'tofu', 'quinoa', 'épinards', 'poivrons', 'céleri-rave', 'carottes', 'haricots verts', 'oignons', 'avocat', 'noix de cajou', 'bananes', 'pommes', 'citrons', 'chocolat', 'framboises', 'tomates', 'pommes de terre', 'riz', 'pâtes', 'oeufs', 'fromage']

10.2 Listes complètes d'exemples des .train

Pour choix_recette recette végétarienne, recette, je veux faire une recette végétarienne, faire une recette, faire une recette végétarienne, je veux cuisiner, je veux faire une recette, je voudrais cuisiner, je voudrais faire une recette, je voudrais faire une recette végétarienne, préparer une recette, préparer une recette végétarienne, je veux préparer une recette, je veux préparer une recette végétarienne, je voudrais préparer une recette, je voudrais préparer une recette végétarienne, donne moi une recette, donne moi une recette végétarienne, donnez moi une recette, donnez moi une recette végétarienne, cuisiner

Pour choix_vegetarisme je veux manger de la viande, je veux cuisiner de la viande, donne moi des informations sur le végétarisme, donnez moi des informations sur le végétarisme, végétarisme, en apprendre plus sur le végétarisme, en savoir davantage sur le végétarisme, en apprendre davantage sur le végétarisme, je veux en savoir plus sur le végétarisme, je veux en apprendre plus sur le végétarisme, je veux en savoir davantage sur le végétarisme, je veux en apprendre davantage sur le végétarisme, en savoir plus sur le végétarisme, je voudrais en savoir plus sur le végétarisme, je voudrais en apprendre plus sur le végétarisme, je voudrais en savoir davantage sur le végétarisme, je voudrais en apprendre davantage sur le végétarisme, je voudrais manger de la viande, je voudrais cuisiner de la viande