

# Catalogue Augmenté

Projet : Interfaces Multimodales

Réalisé par

Nicolas Martenet  
Patrick Terreaux  
Timothée Maret

Semestre d'été 2006

# Table des matières

1 Introduction.....	3
2 Principe.....	3
2.1 Principe Global.....	3
2.2 Entrées.....	3
2.3 Sortie.....	4
3 Fonctionnalités.....	4
3.1 Affichage.....	4
3.2 Changement de taille.....	5
3.3 Changement de couleur.....	5
3.4 Gestion des achats.....	6
4 Matériel nécessaire.....	6
5 Implémentation.....	6
5.1 Architecture logicielle.....	7
5.2 Diagrammes des classes.....	7
5.3 Grammaire Sphinx 4.....	9
6 Conclusions.....	9
7 Annexes.....	9
A Références.....	9
B CD du projet.....	10

## 1 Introduction

---

L'objectif de ce projet est de réaliser une application utilisant plusieurs modalités. Nous avons choisi de compléter l'application réalisée lors de l'étude de la librairie ARToolKit [docARTK] et d'en augmenter les possibilités grâce à la librairie Sphinx 4 ([docSphinx], [webSphinx]). Cette librairie permet de réaliser des applications utilisant la reconnaissance vocale.

Le catalogue augmenté, permettant d'afficher les meubles en trois dimensions à l'écran, se voit donc amélioré et offre ainsi un nouveau niveau d'interaction.

Nous n'avons pas utilisé de phidgets dans notre application car nous désirions garder une interface simple et sobre. De plus aucune des fonctions que nous avons prévues ne correspondait à un phidget. Nous nous sommes donc concentrés sur les deux librairies ARToolKit et Sphinx 4 pour réaliser notre application.

## 2 Principe

---

### 2.1 Principe Global

Avec un catalogue classique, on ne peut observer les meubles qu'en deux dimensions, sous l'angle choisi par le fabriquant. Nous avons donc décidé de créer un nouveau type de catalogue qui permettrait à l'utilisateur de pouvoir interagir avec le catalogue et ainsi d'avoir une nouvelle vision des meubles. Cette application permet de mettre en évidence les possibilités de la librairie ARToolkit.

Pour offrir d'avantages de souplesse lors de l'utilisation du catalogue, nous avons rendu certaines fonctionnalités accessibles grâce à des ordres vocaux. Pour cette partie, nous avons utilisé la librairie Sphinx 4.

### 2.2 Entrées

L'intérêt principal de notre application réside dans la multiplicité des modalités d'entrées. La vidéo, la voix et le clavier pourront être utilisés par l'utilisateur pour interagir avec l'application.

La caméra est l'outil principal et indispensable puisqu'elle permet d'acquérir les images qui seront ensuite traitées par ARToolKit. Mais nous avons décidé de donner encore plus d'importance à la caméra puisque celle-ci va permettre d'activer certaines fonctions de l'application. Dans notre application, elle est directement manipulée par l'utilisateur.

En incluant un micro et la voix, nous fournissons à l'utilisateur une nouvelle possibilité d'utiliser l'application. Grâce à cette redondance, l'utilisateur peut choisir le moyen qui lui semble le plus adapté. Dans le but de garantir la simplicité d'utilisation, les commandes vocales doivent être très simple. Chaque fonction est ainsi associée à un mot clé. La voix est une alternative très intéressante dans notre application car une main de l'utilisateur est utilisée pour la caméra, la seconde reste donc libre.

La dernière source d'entrée est plus classique puisqu'il s'agit du clavier, périphérique habituellement utilisé dans les applications informatiques. Il fournit une 3ème façon d'entrer les ordres destinés au système.

Notre application doit donc réaliser une fusion de ces différentes modalités. On peut donc

situer notre application dans la catégorie suivante.

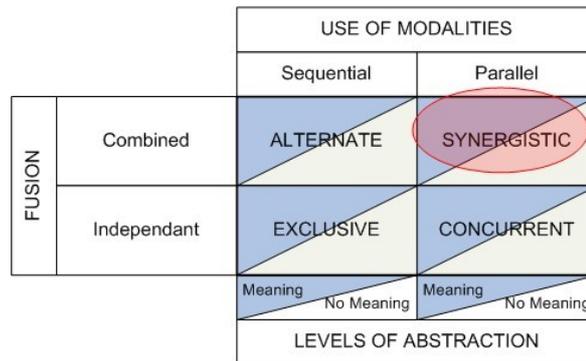


Illustration 1: CASE Diagram

Les différentes entrées sont utilisées de manières combinées et en parallèle puisque l'aquisition des images est effectuée en même temps que la reconnaissance de la voix. Enfin les différents signaux sont interprétés par la librairie à différents niveaux, le signal de la voix est ainsi interprété pour reconnaître les mots prononcés.

### 2.3 Sortie

Il n'y a qu'une seule sortie dans notre application, il s'agit de l'écran sur lequel est affiché les images résultants de notre application. L'utilisateur peut ainsi voir ce qu'il filme ainsi que les images superposées générées par la librairie ARToolKit. L'écran sert donc de référence à l'utilisateur pour diriger la caméra.

## 3 Fonctionnalités

### 3.1 Affichage

La fonction principale de ce catalogue est de permettre à l'utilisateur de visualiser les images habituellement en deux dimensions par des modèles en trois dimensions. Ce mode d'affichage permet ainsi de visualiser sous tous les angles le meuble puisque c'est l'utilisateur qui choisit l'orientation en manipulant la caméra autour du pattern.



Illustration 2: Affichage 3D

La caméra ne sert alors plus seulement à acquérir les images mais à choisir les images que l'on désire acquérir, de manière dynamique. L'utilisateur n'est plus spectateur mais acteur ou plus exactement réalisateur de son film.

Seule la caméra permet cette navigation autour de l'objet, et puisque c'est l'utilisateur qui dirige et actionne la caméra il n'y a pas besoin d'infrastructure particulière.

### 3.2 Changement de taille

Tous les objets représentés en trois dimensions peuvent prendre différentes tailles et c'est l'utilisateur qui la choisit.

Une petite taille permet une navigation plus facile autour de l'objet alors qu'en agrandissant l'objet il est plus facile de se rendre compte des différents détails.

Le changement de taille peut être obtenu de plusieurs manières. Tout d'abord en utilisant la caméra associée à une télécommande.



Illustration 3: Télécommande

Cette télécommande est pourvue de trois boutons, lorsque les trois sont visibles aucune action n'est effectuée. C'est en cachant le bouton + ou le bouton – que l'on obtient la fonction désirée.

L'agrandissement et la réduction de la taille sont également associées à deux commandes vocales, *increase* et *reduce*.

Enfin le clavier peut également servir pour réaliser ses actions. Deux touches permettent de modifier la taille de l'objet.

### 3.3 Changement de couleur

Les meubles proposés sont disponibles dans plusieurs couleurs. Au lieu de présenter chaque possibilité par une image, notre catalogue permet de modifier la texture des objets visibles en temps réel. Les couleurs changent une après l'autre circulairement.

Pour changer la couleur trois modalités sont à nouveau possible. La caméra est à nouveau mise à contribution pour cette fonction mais elle est cette fois utilisée de manière tout à fait original puisque c'est un mouvement de « clic » qui va permettre de changer la couleur. L'application analyse continuellement le mouvement de la caméra et lorsque un mouvement rapide en avant puis en arrière est détecté la couleur change.

La voix est la seconde alternative pour cette fonction. En prononçant *change color*, la couleur sera modifiée sur le modèle 3D.

Pour terminer, le clavier fournit une dernière solution pour changer la couleur à nouveau avec des touches spéciales.

### 3.4 Gestion des achats

En plus de la visualisation, l'utilisateur peut effectuer ses achats à partir du catalogue. Un caddie est automatiquement créé et l'utilisateur peut le remplir lorsqu'il se décide pour un meuble.

Un article peut être ajouté ou enlevé de la liste des achats par les commandes vocale *add* et *remove*. La commande doit être appelée lorsque l'objet tel que le client le désire est visualisé (objet et couleur), de même pour l'enlever.

Enfin la commande vocale *bill* permet d'afficher la liste courante des achats et permet ainsi de visualiser à l'écran la facture et son détail.

Encore une fois, le clavier sert d'alternative à la voix et plusieurs touches sont associées aux mêmes fonctions.

## 4 Matériel nécessaire

---

Nous avons conçu l'application de manière à ce que l'utilisateur ait besoin du moins de matériel possible. Il n'a ainsi pas besoin de s'encombrer d'un grand nombre de périphérique pour utiliser l'application. Mais pour pouvoir utiliser toutes les possibilités un minimum d'équipements est tout de même nécessaire.

- Un PC classique avec clavier et écran et Windows XP installé.
- Une webcam ou un casque virtuel
- Un micro
- Un catalogue augmenté

Il est important que la caméra puisse être manipulée, elle ne doit donc pas être intégrée à l'ordinateur comme c'est le cas sur certains portables.

## 5 Implémentation

---

Bien que nous avons déjà testé la version écrite en C de la librairie ARToolkit, nous avons tenté d'utiliser la version portée en Java pour ce projet, afin de simplifier le couplage avec la librairie Sphinx qui n'est disponible qu'en Java. Cependant, le portage de ARToolkit en Java n'est pas complet, les fonctionnalités VRML que nous utilisons ne sont pas disponibles.

Nous utilisons donc la librairie ARToolkit écrite en C et nous l'interfaçons avec notre code Java en utilisant la technologie JNI.

### 5.1 Architecture logicielle

L'architecture complète de notre application est représentée sur la figure suivante. Nous avons confiné la partie écrite en C dans une librairie Dynamique. La DLL est chargée par l'instruction suivante dans le code Java. Cette opération est effectuée dans la classe CatalogueListener.

```
System.loadLibrary(String)
```

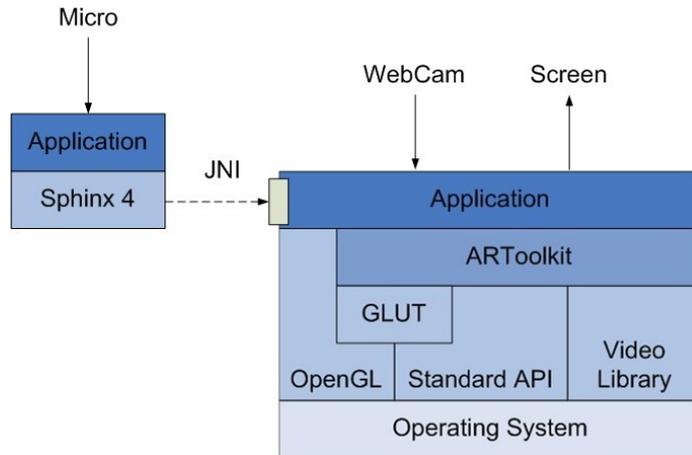


Illustration 4: Architecture logicielle

L'intégration de la librairie Sphinx 4 s'est effectuée rapidement grâce au programme de démonstration documenté dans [docSphinx]. Lorsqu'un mot de la grammaire est reconnu, la méthode correspondante dans la classe CatalogueListener est appelée.

### 5.2 Diagrammes des classes

Les illustrations suivantes montrent l'ensemble des classes utilisée pour notre catalogue, tout d'abord en Java puis les fichiers C.

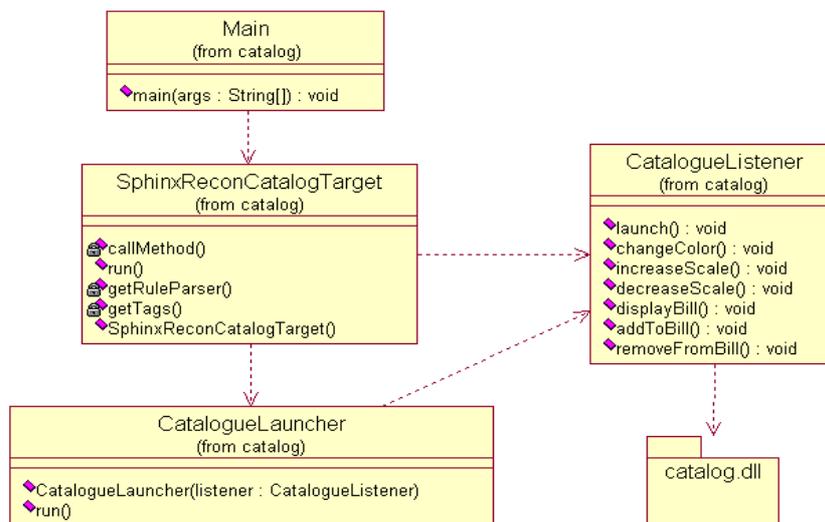


Illustration 5: Diagramme des classes Java

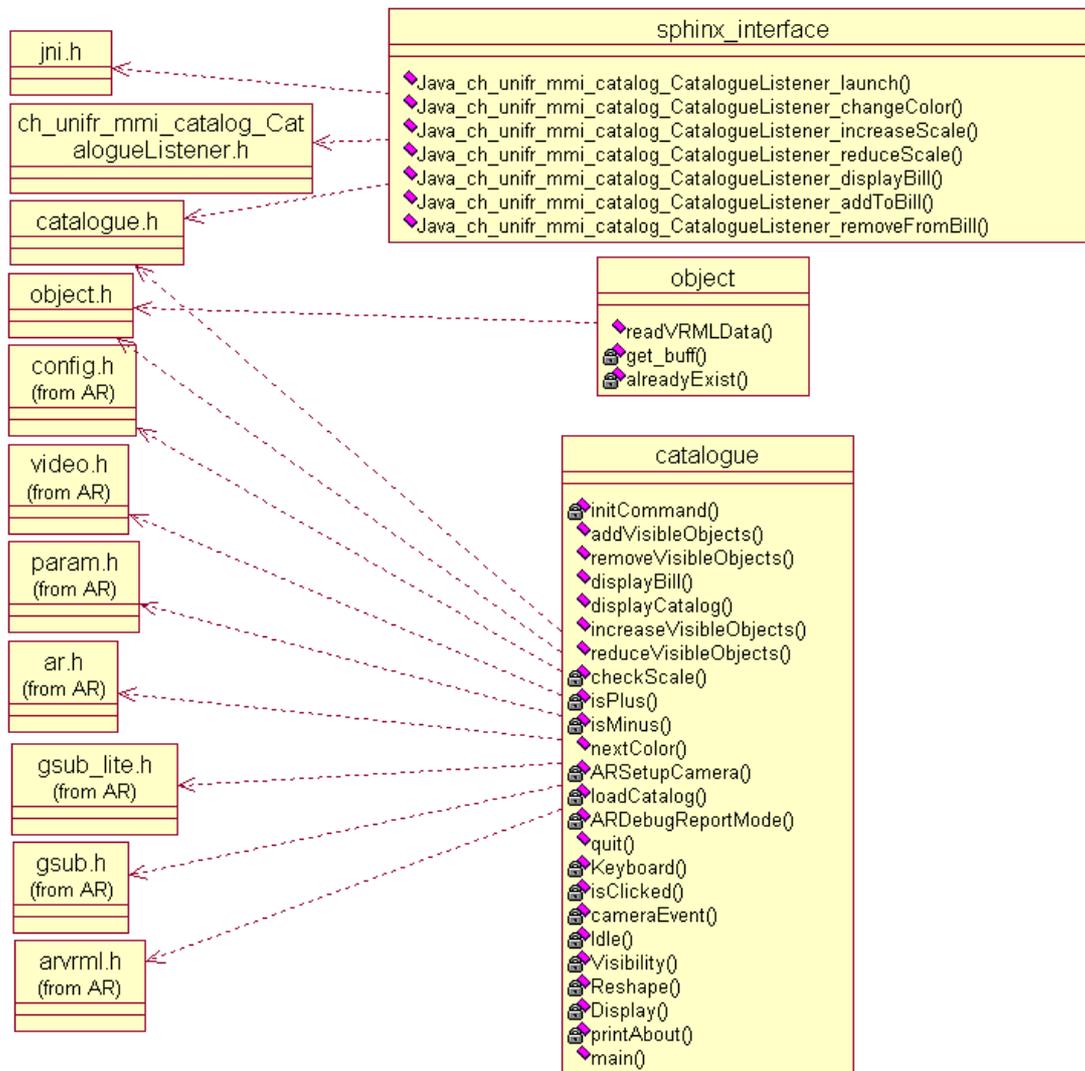


Illustration 6: Diagrammes des classes C

Les classes `SphinxReconCatalogTarget` et `CatalogueLauncher` sont utilisées par Sphinx pour opérer la reconnaissance vocale.

La classe `CatalogueListener` contient uniquement la définition de méthodes natives qui seront implémentées dans la DLL.

Le fichier `ch_unifr_mmi_catalog_CatalogueListener.h` est un fichier d'en-tête qui définit les méthodes à implémenter pour réaliser l'interfaçage avec le code Java. Nous avons utilisé l'utilitaire `javah` fourni avec le SDK de Java qui permet de générer automatiquement ce fichier.

Le fichier `sphinx_interface.c` contient l'implémentation de ces méthodes. Ces méthodes ne contiennent pas de traitement modifiant le comportement de l'application, elles servent uniquement à appeler des méthodes contenues dans le fichier `catalogue.c`.

Les méthodes contenues dans le fichier `catalogue.c` implémentent les fonctionnalités de l'application.

### 5.3 Grammaire Sphinx 4

La grammaire utilisée pour réaliser notre application est volontairement très simple. Le but est de fournir un moyen intuitif à l'utilisateur de commander l'application. Il suffit de prononcer l'un des mots clés pour activer la fonction correspondante.

De plus, nous avons veillé à ce que les mots soient suffisamment différent phonétiquement parlant afin de limiter les problèmes de reconnaissance vocale.

Voici la grammaire complète.

```
[
    increase
    reduce
    change color
    add
    remove
    bill
]
```

## 6 Conclusions

---

Les tests effectués ont montrés que notre application fonctionne correctement, les différents types d'entrées peuvent être utilisés en parallèle pour commander l'application.

La manipulation de la caméra autour des pattern demande un temps d'adaptation car ce n'est pas une opération habituelle mais en utilisant les modifications de taille il est tout à fait possible de bien visualiser l'objet.

La librairie Sphinx ne fournit pas encore un taux de reconnaissance suffisant, les mots doivent être prononcés avec un accent particulier et il faut apprendre à les dire de manière à ce que le système les reconnaisse correctement.

Malgré ces quelques points à améliorer, notre application est tout à fait utilisable par un utilisateur classique. Elle met surtout en évidence l'utilisation particulière de la caméra qui peut être effectuée.

## 7 Annexes

---

### A Références

- [artDoc] Documentation de la librairie ARToolKit  
<http://www.hitl.washington.edu/artoolkit/documentation/index.html>
- [GDC] Projet PatternMaker du groupe GDC  
<http://www.cs.utah.edu/gdc/projects/augmentedreality/index.html>

- [**glut**] Description de la librairie GLUT  
<http://raphaello.univ-fcomte.fr/IG/OpenGL/Glut.htm>
- [**openGL**] Description de la spécification OpenGL  
<http://fr.wikipedia.org/wiki/OpenGL>
- [**webSphinx**] Site de la librairie Sphinx 4  
<http://cmusphinx.sourceforge.net/sphinx4/>
- [**docSphinx**] Rapport sur l'étude de Sphinx 4  
P. Galley, B. Grand, S. Rossier, 2006
- [**docARTK**] Rapport sur l'étude de ARTToolKit  
T. Maret, N. Martenet, P. Terreaux, 2006

## **B CD du projet**

Le CD du projet contient tous les documents relatifs à ce projet.

A la racine du CD se trouve le présent rapport et le fichier Readme indiquant comment exécuter le programme.

Le dossier « project » contient l'ensemble des fichiers pour exécuter le projet ainsi que les sources java et C.

Le dossier « Documentation » contient les rapports des librairies Sphinx 4 et ARTToolKit qui ont été réalisés lors de la première partie du cours.

Enfin le dossier « Vidéo » contient la vidéo de démonstration de notre application.



*Illustration 7: Arborescence du CD*