Incorporació del Català en Agent Conversacional Animat per Computador

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per

Joan Soto Targa

i dirigit per

Francesc Xavier Roca

Bellaterra, 12 de Setembre de 2006

Universitat Autònoma de Barcelona

etse
Escola Tècnica Superior d'Enginyeria

El sotasignat, ...

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en

I per tal que consti firma la present.

Signat: .....

Bellaterra, ...de.....de 200.....

I’m sure you’re just as real as I am.

Max Headroom

Agraïments

Per començar voldria donar les gràcies a tota la gent que m'ha ajudat a realitzar aquest treball: A en Xavi Roca, el meu director de projecte, a en Jordi González per haver-me guiat en el camí a seguir, i animat a seguir-lo, tantes vegades, a en Carles Fernández, per la seva ajuda prestada respecte la síntesi de veu.

Als membres dels equips creadors dels softwares emprats en aquest projecte: Koray Balci, Alan W. Black i tota la gent de les llistes de correu de Festival i Festvox, per contestar les preguntes bàsiques i simplones d'un simple projectista.

Com no, a la meva família i als meus amics per soportar les meves constants queixes i donar-me tots els ànims i suport que m'han donat. Han estat tantes vegades... Moltes gràcies a tots, de debò.

I finalment als meus companys de feina per deixar-me fer servir el portàtil de l'empresa per aquest treball i també el seu suport i interès constant.

Índex

Capítol 1. Introducció

Avui en dia existeixen diverses situacions on un ordinador poden col·laborar en tasques o comunicacions humanes de manera "multimodal", és a dir, on té lloc una interacció conjunta a més d'un nivell (auditiu, visual, tàctil i/o gestual) entre l'home i la màquina**[1]**.

Caelen classifica aquestes situacions en tres tipus bàsics, segons la funció de l'ordinador**[2]**:

1. L'ordinador actua com a *mitjancer*, és a dir, permet la comunicació a llarga distància entre persones que treballen juntes o que estan col·laborant amb un objectiu comú. En aquesta situació, els documents que s'intercanvïin o manipulin haurien de ser multimedia per tal de mantenir tota la informació.
2. La màquina anima una *realitat virtual* que extén les capacitats creatives I expressives humanes. L'usuari es veu immers en un món amb el que ell/a interactua, normalment a nivell gestual.
3. L'ordinador actua com un *company*, és a dir, col·labora amb l'usuari en una certa tasca, servint-se del diàleg per entendre els objectius o fins i tot les intencions de l'usuari. Això pot incrementar l'eficiència de la sessió de treball.

En tots aquests casos, la interacció entre humans i màquines es pot concebre a nivell multisensorial. Aquestes interaccions es poden definir com a multimodals si compleixen dues condicions: primer, la interacció s'hauria d'enfocar en diverses modalitats (o *modes*) motores i sensorials humanes, com poden ser la visió, la veu, tant parlada com escoltada, i la gesticulació (moure's, senyalar, escriure, dibuixar...) **de manera simultània i cooperativa**. Segon, la màquina ha d'entendre i interpretar la informació que li arribi de diversos dispositius d'entrada i sortida (anomenats *medias*).

Per al disenyador d'Interfícies Home-Computadora, els tipus d'aplicacions descrits a dalt mostren unes certes característiques comunes, com per exemple la transmissió d'informació, però varien en el maneig i la interpretació de la informació

d'entrada i sortida. Així doncs, mentre que, per exemple, en la situació 2 (realitat virtual) es dóna més importància al comportament de l'usuari per sobre del diàleg amb la màquina, el qual es veu reduït a un simple esquema d'acció-reacció, a la situació 3 (col·laboració en tasques), passa el contrari: En aquest cas, la relació entre usuari i sistema (o, més precís, la relació entre operador i tasca) és més important, ja que aquí l'usuari està sòl, interactuant amb la màquina. Per tant, **les capacitats de comunicació d'aquesta s'haurien de modelar en base a la comunicació humana**, per tal d'augmentar l'eficiència i la fiabilitat en l'execució de la tasca.

El treball que es presenta en aquesta memòria, més concretament, forma part del projecte europeu HERMES**[3]** que té com a objectiu global oferir un **Agent Conversacional Personificat (*Embodied Conversational Agents*, ECA)** capaç d'interactuar amb una persona a nivell auditiu, gestual i emotiu. És a dir, capaç d'escoltar i parlar (en aquest cas en català), de moure's i gesticular i d'entendre emocions i expressar-ne de forma realista en resposta. Es tracta, per tant, d'una interfície multimodal del 3r tipus.

Tornant a la definició donada per Caelen: "una interfície multimodal implicarà, en qualsevol cas, tres processos: *Gestió dels modes*, la *fusió* de la informació d'entrada i la *fisió* (o separació) de la informació de sortida."

Complint aquest enunciat, la "sortida" del projecte HERMES es divideix en altres sub-projectes, on les dades de sortida es separaran i seran tractades independentment. Més concretament, la idea inicial del projecte final de carrera inicialment proposat, *Max Headroom 20 anys després*, es centra en el cap i la cara de l'agent: Partint de les eines de software lliure **XFace[4]**, les quals permeten crear **caps parlants** (*Talking Heads*, també anomenats **avatars**) en anglès i italià i que expressen emocions, l'objectiu és fer que qualsevol model de cap 3D donat pugui parlar en català també mostrant emocions, tal com feia l'avatar virtual "simulat" Max Headroom fa 20 anys a la televisió (veure Figura 1-1), d'aquí el nom original del projecte.

Figura 1-1: Max Headroom

El treball aquí presentates centra en la part de la generació de la parla. En un principi l'objectiu és fer que XFace pugui dir unes frases concretes en català, però aviat es decideix ampliar-ho i **proveïr a aquest agent conversacional d'una eina TTS** (*Text To Speech*, Text a parla) **plenament funcional amb almenys una veu catalana**. Dit d'una altra manera, fer que XFace pronuncïi en català qualsevol text escrit en aquest idioma, el qual pogués, per exemple, en un futur ser generat per un motor d'IA en resposta a una entrada concreta. Es decideix, a més, des d'un bon principi mantenir en tot moment l'ús de material de software lliure, per permetre una recerca oberta i en col·laboració amb altres centres arreu del món. D'aquesta manera, aquest treball es converteix principalment en una tasca de recerca i estudi de les diferents eines a utilitzar (Veure Figura 1-2).

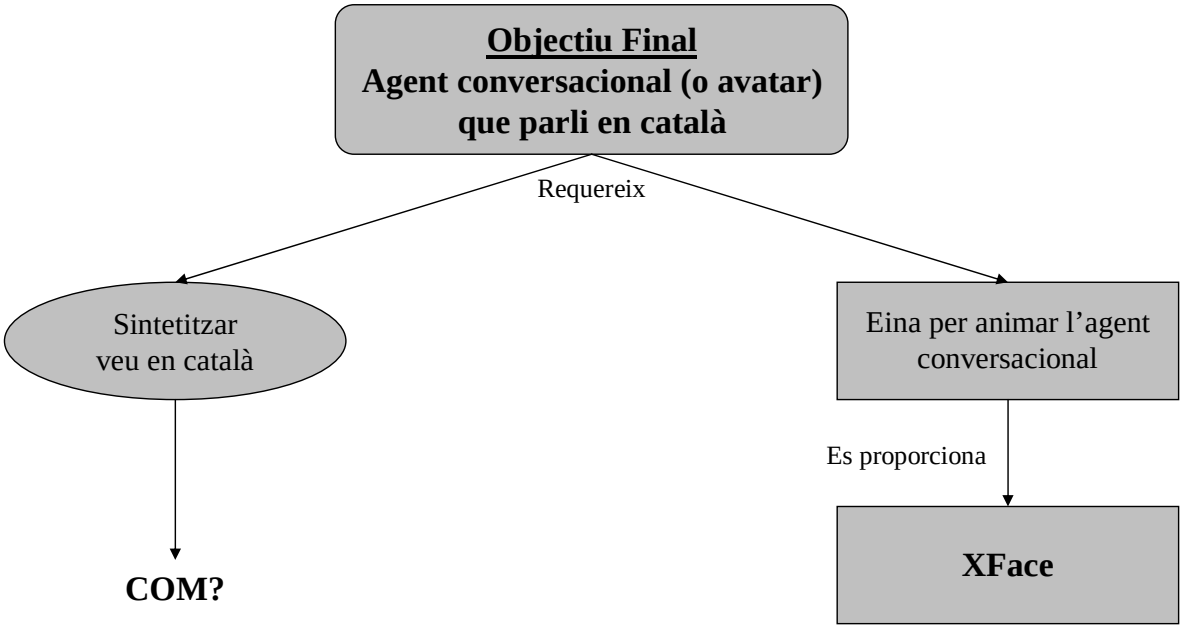

Figura 1-2: Esquema d'organització de les tasques del treball.

Així doncs, en aquest document es presenta al lector aquesta tasca, als següents capítols s'explica l'estudi realitzat de l'aplicació XFace, la comparativa entre les diferents eines TTS de software lliure localitzades, el funcionament de l'eina triada (Festival) i les modificacions realitzades a ambdues per tal de que interactuin i dotar-les de la capacitat de síntesi de veu en català. Finalment, s'explica detalladament el funcionament d'aquestes dues en conjunt mitjançant un exemple concret i al darrer capítol es donen les conclusions a les que s'han arribat.

Capítol 2. XFace

En primer lloc s'estudia el software escollit com a punt de partida d'aquest projecte, XFace, per tal d'entendre'l bé i saber on cal treballar per afegir la parla en català (veure Figura 2-1). En aquest apartat es procedeix a la seva descripció.

2.1. Introducció

Desenvolupat per la divisió de Tecnologies Cognitives i de la Comunicació (*Cognitive and Communication Technologies,* TCC)**[5]** de l'ITC-irst (*Instituto Trentino di Cultura – Il centro per la Recerca Scientifica e Tecnologica*)**[6]**, XFace és un kit d'eines "open source" que pretén cobrir la falta d'aquestes respecte els anomenats Agents Conversacionals Personificats (*Embodied Conversational Agents*, ECA), per facilitar-ne la recerca i la investigació. Concretament, vol ser una eina que permeti generar i animar fàcilment agents parlants 3D, també coneguts com avatars, "cares" o "caps parlants".

Actualment (versió 0,93, al mes de maig del 2006) aquest kit està basat en l'estàndard **FA** (*Facial Animation,* animació facial) **d'MPEG-4,** explicat en detall a l'apèndix A, de manera que pugui reproduir "streams" de **FAP**'s (*MPEG-4 Facial Animation Parameters*, Paràmetres d'Animació Facial), els quals determinen el grau de deformació i/o desplaçament de zones concretes de la cara, com per exemple el llavi superior, el cantó esquerre de l'ull dret o la parpella inferior esquerra. Alternativament, també permet fer animacions basades en "**keyframes**": models 3D estàtics per gestos concrets de la cara prèviament definits, dels quals XFace ja n'inclou alguns de bàsics, però se'n poden afegir més.

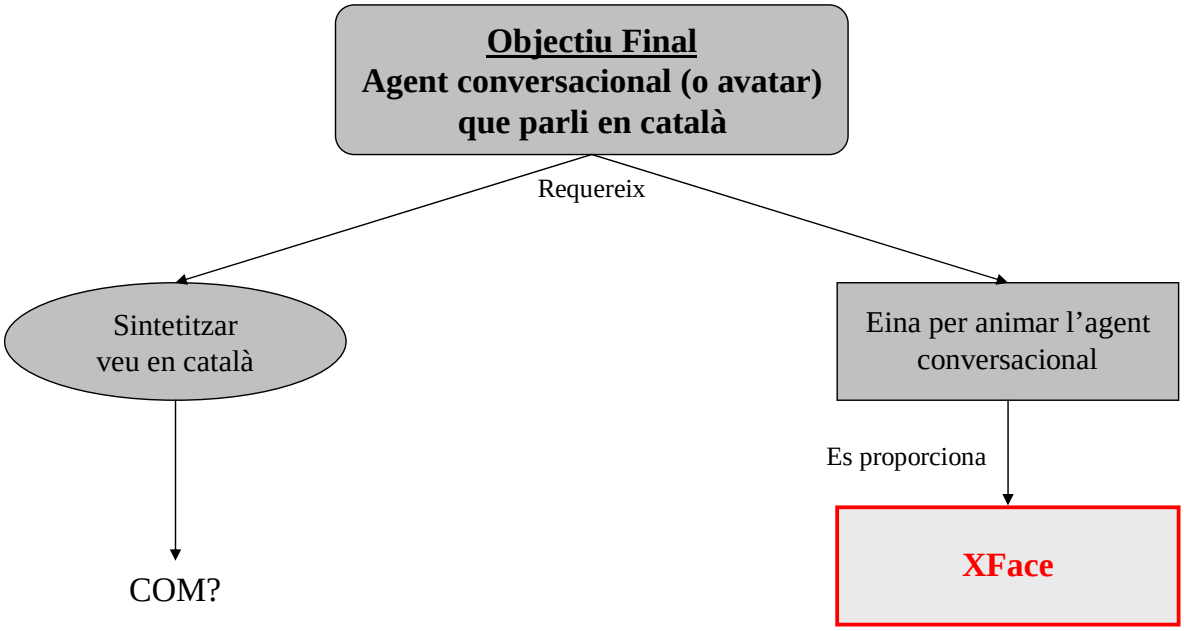

Figura 2-1: Esquema d'organització de les tasques del treball, 1a fase - Estudi d'XFace.

Degut a l'ampli colectiu de recerca al que va dirigit, l'arquitectura d'XFace està dissenyada per a que sigui configurable i fàcil d'extendre. Totes les parts d'aquest són independents del sistema operatiu i poden ser compilades amb qualsevol compilador d'ANSI C++ estàndard (si bé el codi ve preparat per a ser compilat sota Windows, en Visual C++ .Net). Per a l'animació dels objectes 3D en sí, el kit està basat en OpenGL i està prou optimitzat per aconseguir tasses de "frames" per segon satisfactòries (per a la generació de FAPs es requereixen com a mínim 25 FPS) amb models amb un nombre elevat de polígons (12.000 polígons) en un hardware modest.

Més concretament, avui en dia el projecte XFace consta de 5 programes: 2 llibreries, la llibreria central, independent de la resta (XFace) i una llibreria que interpreta el llenguatge de *markup* SMIL-Agent (XSmilAgent); i 3 aplicacions finals que les fan servir: un editor per configurar els caps per a que XFace hi pugui treballar i provar-los (XFaceEd), un reproductor d'animacions facials (FAPs) d'exemple (XFacePlayer) i un client per a treballar amb l'anterior remòtament (XFaceClient).

2.2. Funcionament

En aquest apartat s'expliquen en detall els 5 elements dels que es composa XFace, quina és la funció de cadascun i com interactuen entre ells**[7][8]** (veure Figura 2-2).

2.2.1. XFace. Llibreria central

La llibreria central d'XFace s'encarrega de carregar els models de cara i la informació MPEG-4 corresponent, de rebre els "streams" de FAP's, normalment llegides d'un arxiu de text de tipus ".fap" que conté una llista de diferents valors d'aquests, i calcular les deformacions i renderitzacions corresponents sobre el model per a crear l'animació facial i de reproduir sincronitzadament, l'arxiu d'àudio WAV amb la veu, en cas de que es proporcioni. Està dividida en quatre mòduls o "namespaces":

- **XEngine:** Classes del motor gràfic 3D que fa servir XFace. És una implementació simple, basada en OpenGL, especialitzada per XFace i implementada el més genèricament possible
- **XMath:** Classes per càlculs matemàtics complexos, com les funcions de deformació.
- **XFaceApp:** Classes per interactuar amb altres aplicacions, com funcions per XML , interfícies pel sò, per temporitzadors, etc.
- **XFace:** Resta de classes de la llibreria que tracten la cara parlant en sí.

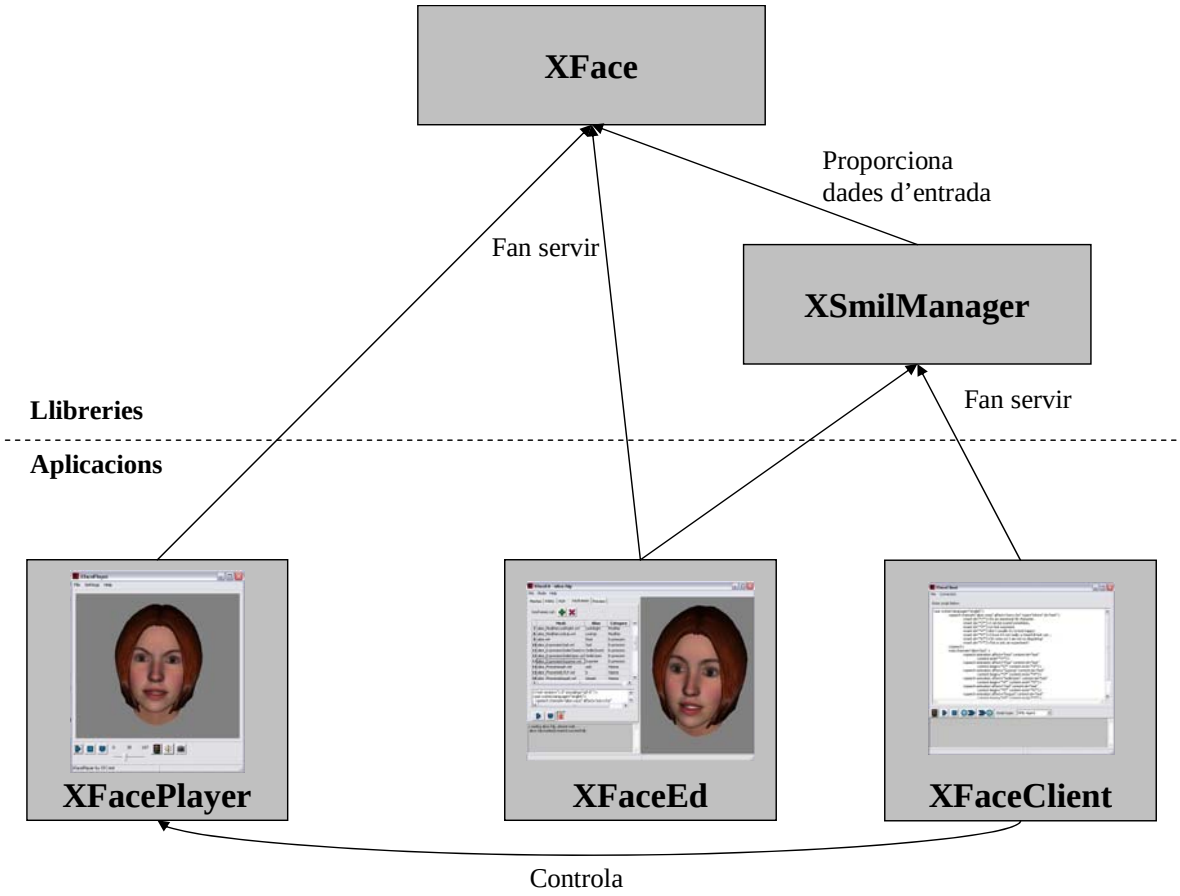

Figura 2-2: Mòduls d'XFace i relacions entre aquests.

Aquesta llibreria necessita, com a entrada bàsica per a carregar un avatar, un **arxiu de configuració** XML anomenat **FDP**, el qual es defineix per cada cap que es vulgui fer servir en XFace i que descriu els models 3D que s'han d'utilitzar per definir cada part del cap (cap, cabells, ulls, dents...), els **FPs** (*Feature Points*, Punts Característics), que serveixen per adaptar un model 3D d'un cap a l'estàndard d'MPEG-4, i les seves àrees o zones d'influència: conjunt de vèrtexs al voltant de l'FP sobre els quals s'aplicarà el càlcul de la funció de deformació. També dóna les 5 **FAPU**s (*Facial Animation Parameter Units*, Unitats del Paràmetre d'Animació Facial), que serveixen per calibrar els FAPs genèrics que arriben sobre el model en concret al que s'apliquen, i la **funció de deformació** que s'aplicarà a cada zona, la qual ha d'estar definida dins del codi de la llibreria, entre d'altres (Veure Apèndix B, exemple d'arxiu FDP). També necessita, òbviament, els models estàtics 3D requerits per l'arxiu FDP. Respecte la funció de deformació, a la implementació actual, aquesta és per defecte una funció de cosinus augmentats (*raised cosinus*), basada en la distància euclídea que un punt s'hauria de desplaçar dins la regió de l'FP en la direcció donada pel FAP corresponent.

Si bé aquesta funció dóna resultats satisfactoris, la llibreria permet que es pugui extendre fàcilment per tal d'incloure i utilitzar estrategies de deformació diferents, com poden ser *Radial Basis Functions***[9]** o *Free Form Deformations***[10]**.

També, com ja s'ha dit prèviament, es poden definir i utilitzar "keyframes" (altres models estàtics 3D del mateix cap mostrant gestos diferents) que representin certs **visemes** (posició de la cara en general i de la boca en particular en pronunciar un fonema) i certes emocions concretes, com a mètode alternatiu d'animació, els quals també vindran indicats a l'arxiu FDP. Això permet crear un avatar utilitzant simplement la combinació dels "keyframes" en temps real mitjançant alguna tècnica d'interpolació, enlloc dels FAP's d'MPEG-4. En aquest cas, l'entrada que es necessita és, a part de l'arxiu d'àudio WAV, l'anomenat **arxiu de fonemes** (de tipus ".pho") i l'**arxiu d'animacions** (de tipus ".anim"). El primer és una llista amb tots els fonemes i la seva duració acumulada, tal com mostra l'exemple següent, generat a partir de l'entrada "Digues aquest text.":

Exemple d'arxiu ".pho" (el fonema # indica silenci):

```
# 0.2500
d 0.3100
i1 0.4060
g 0.4860
e 0.5660
s 0.6760
a 0.7560
k 0.8560
e1 0.9640
s 1.0740
t 1.1590
t 1.2865
e1 1.4215
k 1.5715
s 1.7365
t 1.8640
# 2.1140
```

El segon arxiu és també una llista, en aquest cas amb el **canal**, és a dir, la part de l'avatar a la que afecten les expressions o moviments (*face* si es tracta d'una expressió facial, *eyes* si és un gest dels ulls o *head* si és un moviment del cap); els diferents gestos o expressions facials a mostrar per l'avatar durant la reproducció en sí (i que s'han de correspondre amb "keyframes" d'emocions existents en el cas de les emocions facials), l'instant en el que comencen, l'instant en el que acaben i un factor d'intensitat dins el rang [0, 1].

Exemple d'arxiu ".anim":

```
face Sad 0 3.464 0.75
face Anger 3.464 7.307 1
head nodding 0 3.464 0.75
```

A partir d'aquests dos arxius, XFace s'encarrega de calcular els visemes corresponents i, de nou, de la reproducció de l'animació. Concretament, aquest darrer càlcul es fa mitjançant uns arxius de text que defineixen la correspondència entre fonemes i visemes segons l'idioma anomenats **diccionaris**, i que es trobaran sempre al subdirectori "lang" dins del directori local d'XFace (o l'aplicació que l'estigui fent servir, com per exemple XFaceEd). A continuació es mostra un exemple d'aquests, on la primera fila indica l'idioma ("ca", català), i per la resta a la columna esquerra hi ha els diferents fonemes i a la dreta el visema que li correspon a cadascun:

Exemple de diccionari, "catalan.dic":

```
ca
# Rest
a aah
e eh
i i
o oh
u q
i0 i
u0 w
a1 aah
e1 eh
i1 i
o1 oh
u1 q
p b
t d
k k
b b
d d
g k
f f
th th
s d
x k
ch ch
m b
n n
ny n
l th
ll th
r r
rr r
```

2.2.2. XSmilAgent

El 22 de març del 2005, la divisió TCC de l'ITC-irst defineix el llenguatge de "markup" basat en XML 1.0, **SMIL-Agent** (*Synchronized Multichannel Integration Language for a synthetic Agent*, Llenguatge d'Integració Sincronitzada Multicanal per a un Agent)**[11][12]**, el qual serveix per descriure i controlar, sincronitzadament, els diferents *canals* o *modes*, tals com poden ser la veu, els gestos de la cara, els moviments del cap i del cos, etc. d'un Agent Conversacional Personalitzat (ECA). És a dir, mitjançant aquest llenguatge un autor pot, per un agent:

1- Descriure'n els diferents canals comunicatius disponibles i les seves habilitats.
2- Descriure els diferents actes a realitzar pels diferents canals de l'agent durant una presentació.
3- Descriure la sincronització temporal de la presentació.

Serveixi com a exemple un script d'entrada d'XFace, el qual indica que es processi en paral·lel (etiqueta *par*) un text a pronunciar (etiqueta *speech*) per una banda i una seqüència (etiqueta *seq*) de dues expressions facials (etiqueta *speech-animation*), tristesa i enuig, per l'altra. Per sincronitzar ambdues accions, s'usen uns marcadors (etiquetes *mark*):

Exemple d'script SMIL-Agent:

```
<?xml version="1.0" encoding="utf-8" ?>
<par system-language="catalan">
  <speech channel="alice-voice" affect="sorry-for"
      type="inform" id="exemple-smil">
    <mark id="*1*"/>Exemple de script. <mark id="*2*"/>
    Ara se m'hauria de veure trista... <mark id="*3*"/>
    i ara enfadada. </speech>
  <seq channel="alice-face">
    <speech-animation affect="Sad" content-id="exemple-smil"
      content-end="*2*" intensity="0.75"/>
    <speech-animation affect="Anger" content-id="exemple-smil"
      fill="freeze" content-begin="*2*"/>
  </seq>
</par>
```

Tornant a XFace, a les darreres implementacions s'ha reforçat la possibilitat de la reproducció de seqüències de text i animacions d'entrada donades en un script en aquest llenguatge, mitjançant la incorporació d'un intèrpret d'aquest. XSmilAgent és, doncs, una llibreria dedicada a interpretar aquest llenguatge, així com a proporcionar

una interfície entre XFace i els programes externs necessaris (generadors de FAP i eines TTS).

XSmilAgent rep dues entrades, concretament dos textos XML: l'arxiu "**scriptprocs.xml**", on s'especifiquen totes les eines externes (generadors de FAPs, com "expml2fap", i eines TTS, com "Festival" o les llibreries "Microsoft Speech Tools") amb les que aquest pot treballar i les seves característiques (idioma, si és un programa extern a XFace o una llibreria, comanda a fer servir, etc.), i l'script SMIL-Agent d'entrada mateix, a reproduir en forma de veu i expressions de la cara.

Arxiu "scriptprocs.xml" actual:

```
<?xml version="1.0" encoding="utf-8" ?>
<ScriptProcessors>
<scripter name="Italian expml2fap" type="external" lang="it">
      <executable
            exename="expml2fap.exe"
            path="..\expml2fap"
            parameters2=" -noAnim -dur 1.2 "
            tempfile="TempFile.xml"/>
      <output filename="TempFile.wav"/>
      <output filename="TempFile.fap"/>
</scripter>

<scripter name="Italian Text" type="external" lang="it">
      <executable
            exename="it_flite.exe"
            path="D:\DevLibs\it_flite\"
            parameters1=" -o TempFile.wav --sets
lab_filename=TempFile.pho --sets voice=lp -t ">
      </executable>
</scripter>

<scripter name="Catalan festival" type="external" lang="ca">
      <executable
            exename="bash"
            path="C:\cygwin\bin\"
      parameters1="C:/cygwin/home/Joan/festival/JSTP/festival_xface_ca
t"
            tempfile="TempFile.xml"
            parameters2=" -o TempFile.wav -p TempFile.pho ">
      </executable>
</scripter>

<scripter name="English expml2fap" type="external" lang="en">
      <executable
            exename="expml2fap.exe"
            path="..\expml2fap"
            parameters2=" -noAnim -dur 1.2 -lang e"
            tempfile="TempFile.xml">
      </executable>
      <output filename="TempFile.wav"/>
      <output filename="TempFile.fap"/>
</scripter>
<scripter name="SMIL-Agent" type="SMIL" lang="en_SAPI">
      <output filename="TempFile.wav"/>
```

```
      <output filename="TempFile.pho"/>
      <output filename="TempFile.anim"/>
</scripter>
</ScriptProcessors>
```

Així doncs, segons l'idioma o l'eina que es requereixi a l'script d'entrada, XSmilAgent s'encarrega de cridar a les aplicacions corresponents i genera com a sortida l'arxiu WAV amb la veu i, o bé l'arxiu FAP si s'està seguint l'estàndard d'MPEG4, o bé els arxius PHO i ANIM en el cas del mètode de "keyframes" (veure Figura 2-3).

Finalment, cal dir que ara com ara aquesta llibreria està més orientada cap al mètode de "keyframes" per evitar la necessitat d'una eina generadora de FAPs, de les quals no n'hi ha de codi lliure o gratuites i senzilles d'aconseguir avui en dia. Això, per altra banda, permet que XFace en general sigui més autosuficient, ja que només necessita l'eina TTS que generi els arxius d'àudio i de fonemes. Donats aquests dos, XSmilAgent s'encarrega d'interpretar la resta de l'script pel seu compte i, a partir dels temps donats a l'arxiu de fonemes, crear per sí sol l'arxiu d'animacions.

2.2.3. XFaceEd

L'estàndard FA d'MPEG-4 representa una manera de parametritzar el procés d'animació, però igualment s'han de definir uns certs paràmetres, com les FAPU i els FP's, manualment sobre el model 3D. XFaceEd és una aplicació que simplifica aquest procés proporcionant un mètode senzill per definir les FAPU's i les regions, pesos i paràmetres dels FP's, per poder manipular i animar els models estàtics de la cara. Els models estàtics 3D en sí es poden crear amb qualsevol paquet de creació d'elements 3D (com poden ser "3D Studio MAX", "Poser", "Blender", etc.) i importar-los a l'XFaceEd. Pel que fa a l'alternativa dels "keyframes", XFaceEd també permet fàcilment definir quins models (igualment realitzables amb qualsevol paquet 3D) seran aquests.

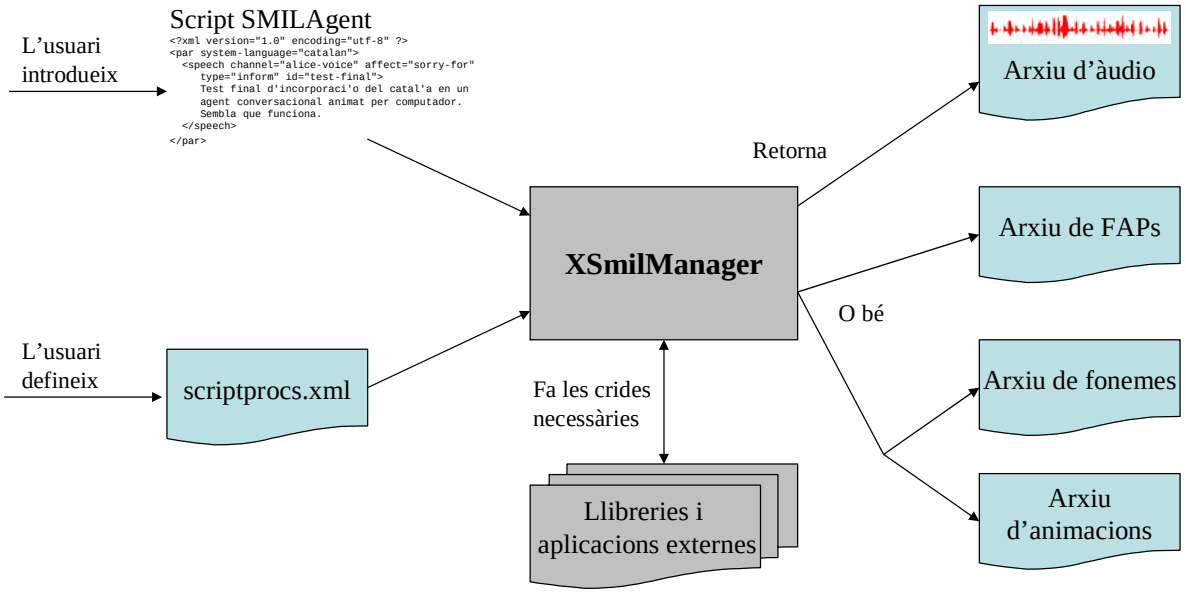

Figura 2-3: Esquema de funcionament de XSmilAgent.

En definitiva, XFaceEd és una aplicació que facilita la creació i modificació de l'arxiu FDP o de configuració de la cara. Això ajuda a la definició de regles de deformació i de paràmetres externament, de manera que no hi hagi res codificat explícitament (*hard-coded*, és a dir que si es canvia s'ha de tornar a compilar) dins la llibreria d'XFace: el model 3D es pot canviar sense gaire esforç, sense que s'hagi de programar res, l'arxiu de configuració és interpretat per la llibreria i les respectives regles de deformació es generen automàticament. A més, també permet provar qualsevol canvi realitzat mitjançant un previsualitzador de FAPs i un reproductor d'scripts SMIL-Agent que té incorporats.

Entrant més en detall, XFaceEd té 5 pestanyes les quals es descriuen a continuació:

- **Meshes (Figura 2-4):** Aquesta pestanya permet definir el conjunt d'1 o més models 3D (*submeshes*) que definiran conjuntament el cap final en **posició de descans**, que MPEG-4 defineix sempre com a punt de partida per a qualsevol seqüència de FAPs que es vulgui reproduir (veure Apèndix A).

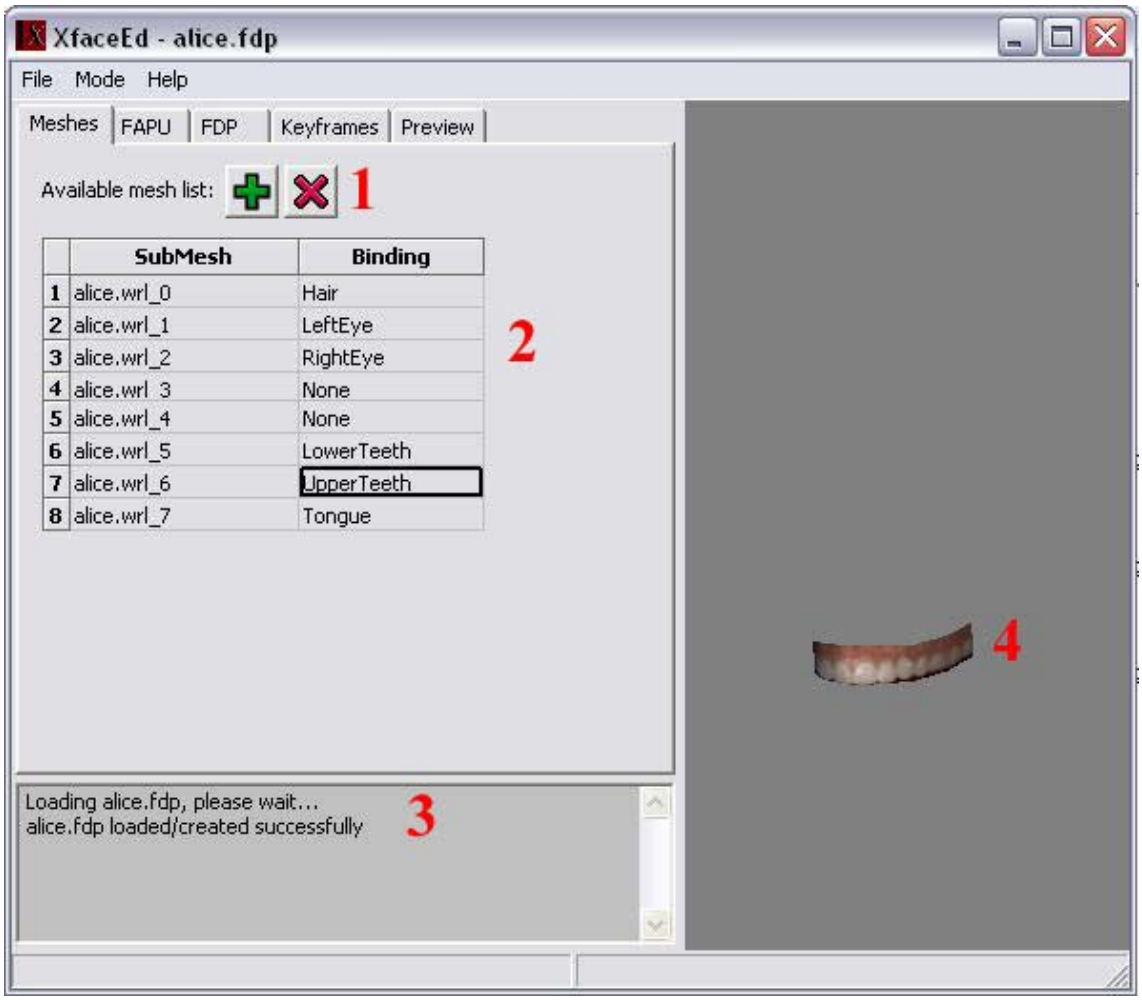

Figura 2-4: Pestanya *Meshes* d'XFaceEd. 1- Botons per afegir o treure un model (*submesh*). 2- Llista de *submeshes* del cap actual i part del cap a la que s'assignen. 3- Missatges de sortida. 4- Vista del model sel·leccionat a la llista.

- **FAPU (Figura 2-5):** Un cop definit el model del cap sencer, aquí es poden definir les FAPUs marcant sobre aquest els punts entre els quals es calculen aquestes unitats. És a dir, un cop marcats, per exemple, els centres dels dos ulls, es calcularà la FAPU *ES0* (*Eye Separation*, Separació entre ulls). Concretament per marcar els punts, XFace ofereix quatre "modes" de funcionament del ratolí, escollibles mitjançant el menú "Mode" o les tecles de funció F5 a F8:
    1. **F5 → *Rotate*:** Si es mou el ratolí mentre es manté apretat el botó esquerre, el cap rotarà sobre els 3 eixos segons el moviment.
    2. **F6 → *Zoom*:** Si es mou el ratolí cap a dalt, de nou amb el botó esquerre apretat, el model s'acosta a la camara (*Zoom In*) i si es mou cap a baix, s'allunya (*Zoom Out*).

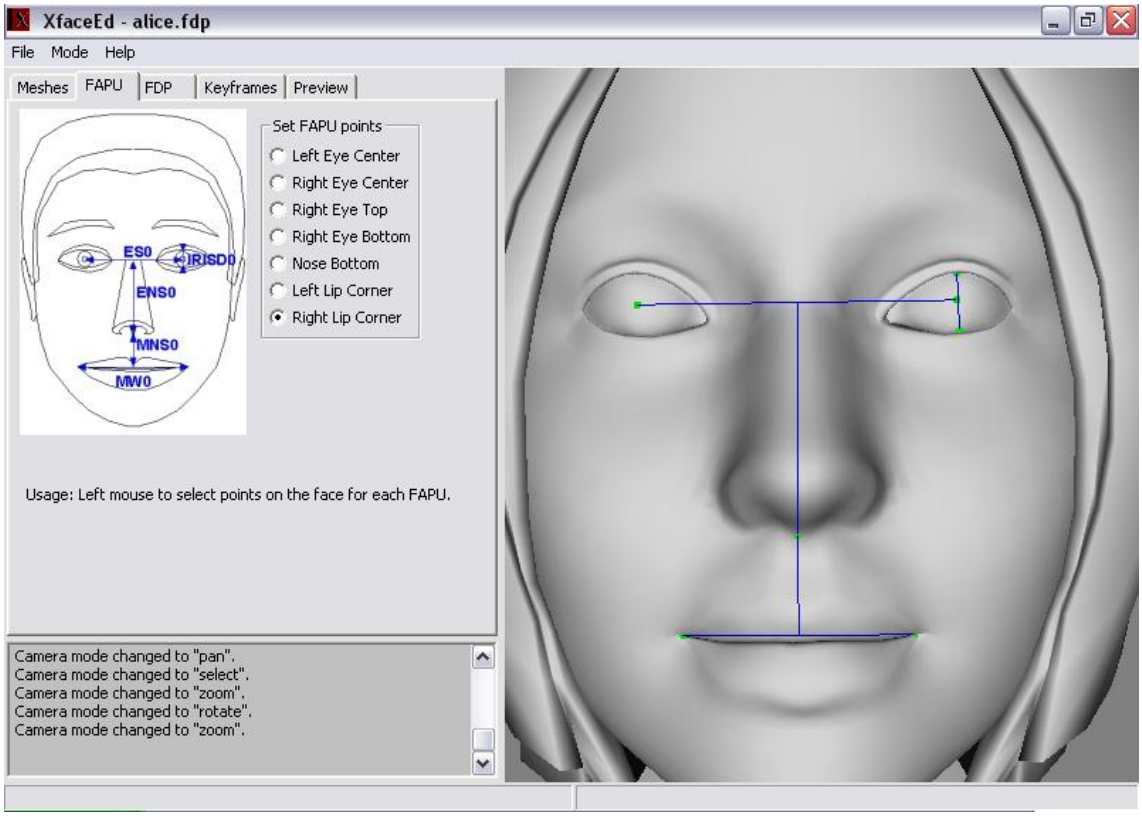

Figura 2-5: Pestanya FAPU amb FAPUs definides sobre el model 3D

3. **F7 → *Pan*:** En moure el ratolí amb el botó esquerre apretat, el cap es trasllada en la mateixa direcció (a dalt, a baix, a l'esquerra, a la dreta o en diagonal).
4. **F8 → *Select Vertex*:** Un cop col·locat el cap en la situació que vagi millor, en aquest mode simplement es marcarà (de color verd) el vèrtex sobre el qual es faci clic amb el botó esquerre, i aquest quedarà associat al punt que hi hagi sel·leccionat al menú del panell.

Cal matisar que aquests 4 modes són sel·leccionables en qualsevol de les cinc pestanyes amb el mateix efecte, excepte el 4t mode que només es fa servir en aquesta pestanya i la següent.

- **FDP (Figura 2-6):** Seguint el mateix funcionament que l'anterior, aquí es poden definir 45 dels 46 FP's afectats per FAPs (quedant fora l'únic punt que es situa darrere del cap) i les seves àrees d'influència, sobre cadascun dels models 3D en separat per una major precisió. Més específicament, primer es sel·lecciona l'FP segons la zona de la cara on es troba (cara, ulls, nas, llavis...) i el seu "nom" donat per l'estàndard (2.1, 3.6, 8.3...).

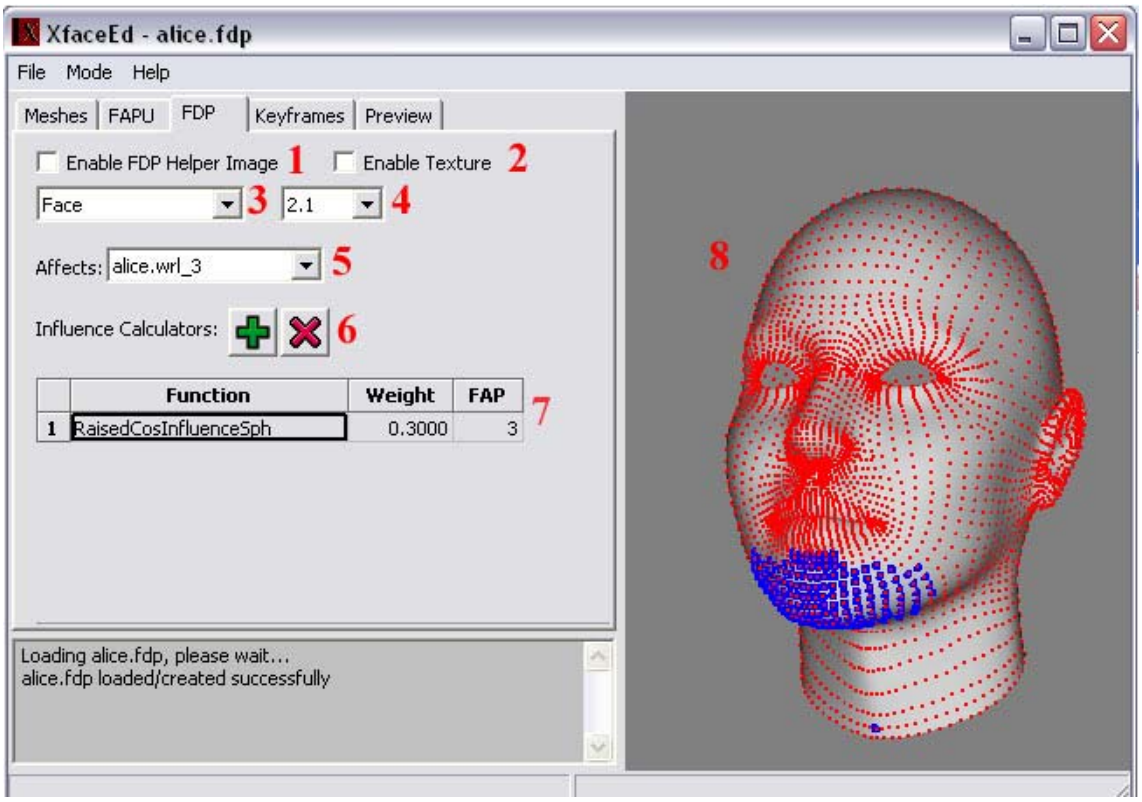

Figura 2-6: Definició dels FP i la seva zona d'influència en XFaceEd. 1- Opció per mostrar o no la imatge d'ajuda dels FPs. 2- Opció per aplicar o no la textura sobre el model 3D. 3 i 4- Selectors de la zona de la cara i de l'FP concret dins d'aquesta. 5- Selector del model 3D (*submesh*) sobre el que es sel·leccionaran els punts. 6- Botons per afegir o eliminar FAPs que hi influeixin. 7- Taula de FAPs que influeixen sobre l'FP, funció de deformació i pes. 8- Vista del model i l'FP (en verd) i els punts de la seva regió (en blau) sel·leccionats.

Per saber a quin punt concret de la zona triada es correspon cada FP, es pot fer aparèixer una imatge d'ajuda que ho mostra. Un cop sel·leccionat, doncs, es marca l'FP fent Ctrl. + clic en un vèrtex en qüestió, el qual queda marcat de color verd, i la seva zona clicant sobre els vèrtexs que hi pertanyen, que queden marcats de color blau (veure Figura 2-7, on es mostren com exemples els punts 3.1, centre de la parpella superior esquerra, i i 8.2, centre del contorn exterior del llavi inferior). També es donen, per cada FP, els FAPs que hi influeixen i, per cadascun d'aquests, la funció de deformació que s'aplica i el pès (factor en el qual influeix el FAP sobre l'FP i la seva zona).

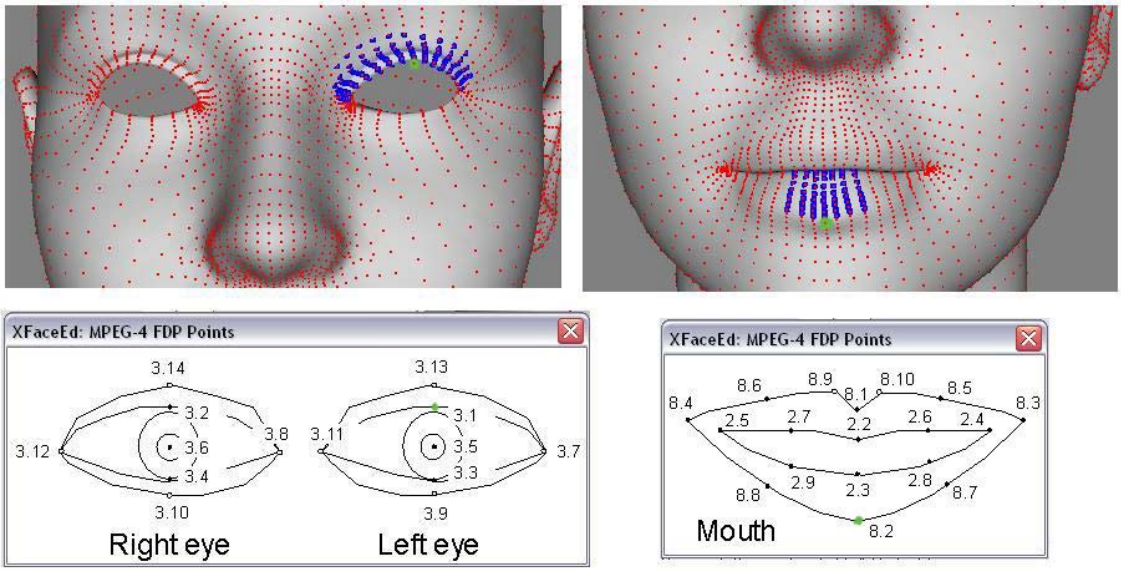

Figura 2-7: Exemples de sel·lecció d'un FP i la seva regió i de les imatges d'ajuda corresponents proporcionades per XfaceEd.

- **Keyframes (Figura 2-8):** Aquesta pestanya es dedica al mètode d'animació alternatiu. Bàsicament el que permet és afegir els diferents models 3D que serviran de "keyframes", i classificar-los segons si són visemes (*Viseme*), expressions facials (*Expression*) o petits gestos superposables a les expressions generals (parpelleig, mirar cap un costat, aixecar les celles) anomenats "modificadors" (*Modifier*) (veure Figura 2-9 per alguns exemples). És també en aquesta pestanya que s'inclou un reproductor d'scripts SMIL-Agent per a fer proves. En aquest treball s'ha fet servir aquest reproductor.

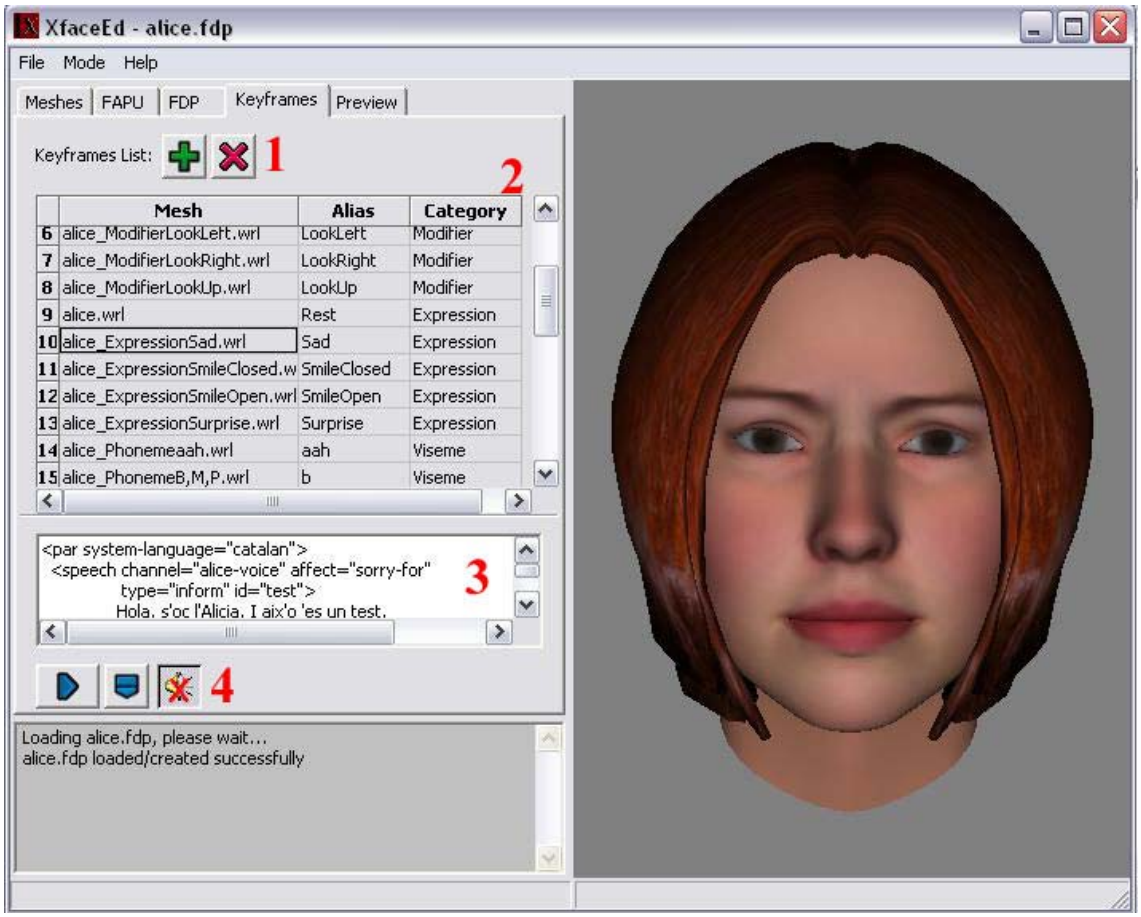

Figura 2-8: Panell de "keyframes" d'XFaceEd. 1- Botons per afegir o treure "keyframes". 2- Llista de cadascun dels "keyframes" i la seva categoria, juntament amb el model (*Mesh*) que té associat. 3- Editor de textos amb l'script d'entrada pel reproductor d' SMIL-Agent. 4- Botons del reproductor: D'esquerra a dreta són: "reproduir", "carregar un script" i "activar/desactivar la sortida d'àudio".

Exemple d'"Expressió": fàstic.

Exemple de "Modificador": Mirar cap avall.

Exemple de "Visema": Pels fonemes "j" (joc), "sh" (xec) i "tx" (cotxe).

Figura 2-9: Exemples de "Keyframes".

- **Preview (Figura 2-10):** Finalment, aquí es poden provar els diferents elements d'MPEG-4 que s'hagin configurat. Aquesta pestanya inclou un reproductor d'streams (fitxers) de FAPs i, a sota, un previsualitzador d'aquests, que permet observar com afecta cada FAP per separat al model, com aquest es deforma segons els diferents valors que prengui (a la figura 2-10, com exemple, s'ha augmentat el valor del FAP 33 que representa "aixecar el mig de la cella esquerra").

2.2.4. XFacePlayer

XFacePlayer (veure Figura 2-11) és una aplicació d'exemple que demostra com es pot implementar un reproductor de cares utilitzant la llibreria central. Permet carregar un arxiu FAP d'MPEG-4 i un arxiu d'àudio (per la parla), juntament amb l'arxiu de configuració creat mitjançant XFaceEd amb només un parell de crides a funcions i tenir, d'aquesta manera, una implementació senzilla d'un cap parlant. També permet, juntament amb un mòdul TTS, que s'utilitzi el mètode d'animació per "keyframes" com a alternativa. A més, aquest reproductor pot ser controlat remòtament a través de la xarxa mitjançant XFaceClient.

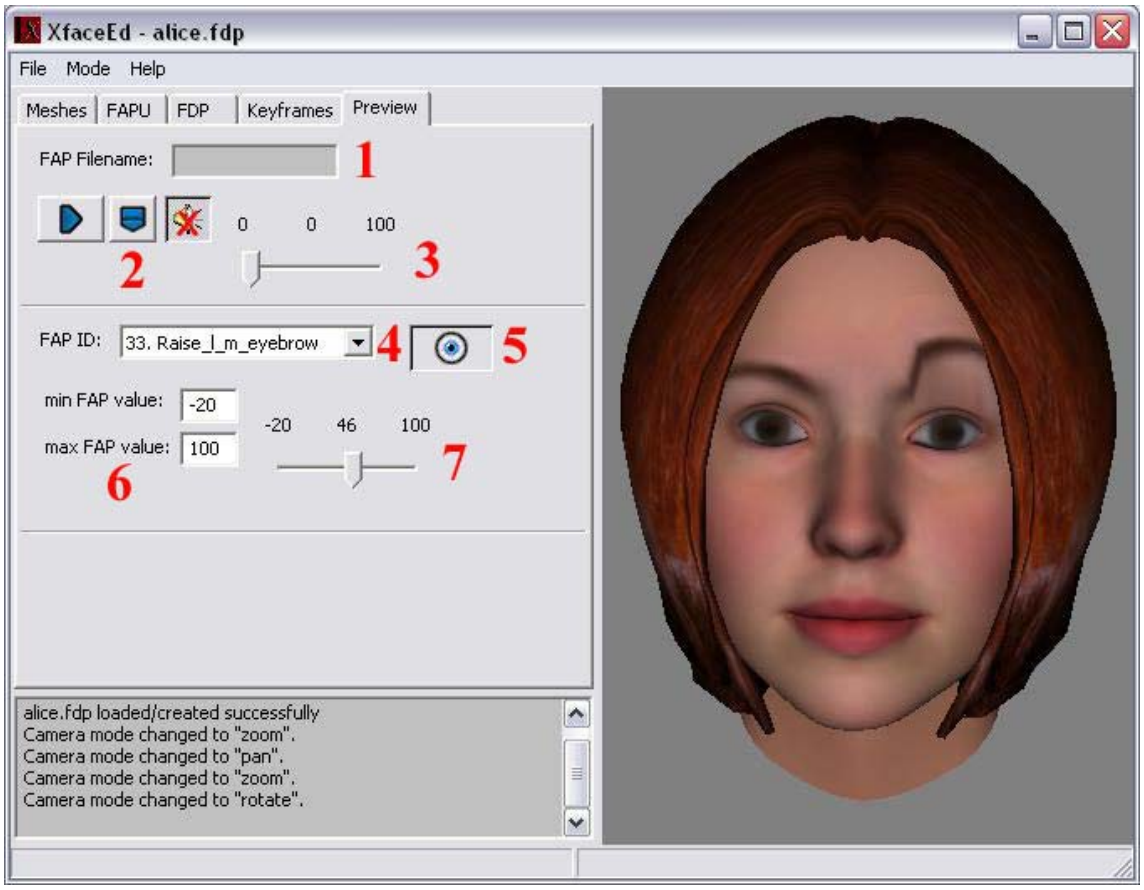

Figura 2-10: Panell de "Preview" d'XFaceEd. 1- Nom del fitxer FAP que s'ha carregat. 2- Botons del reproductor de FAP's que serveixen, respectivament, per reproduir la seqüència, carregar un arxiu FAP i un WAV i activar o desactivar la reproducció de l'àudio. 3- Barra de progrés de la reproducció. 4- Selector de FAP. 5- Botó per a que es vegi o no la deformació aplicada en el model. 6- Valors mínim i màxim que pot prendre el FAP. 7- Barra de variació del valor del FAP.

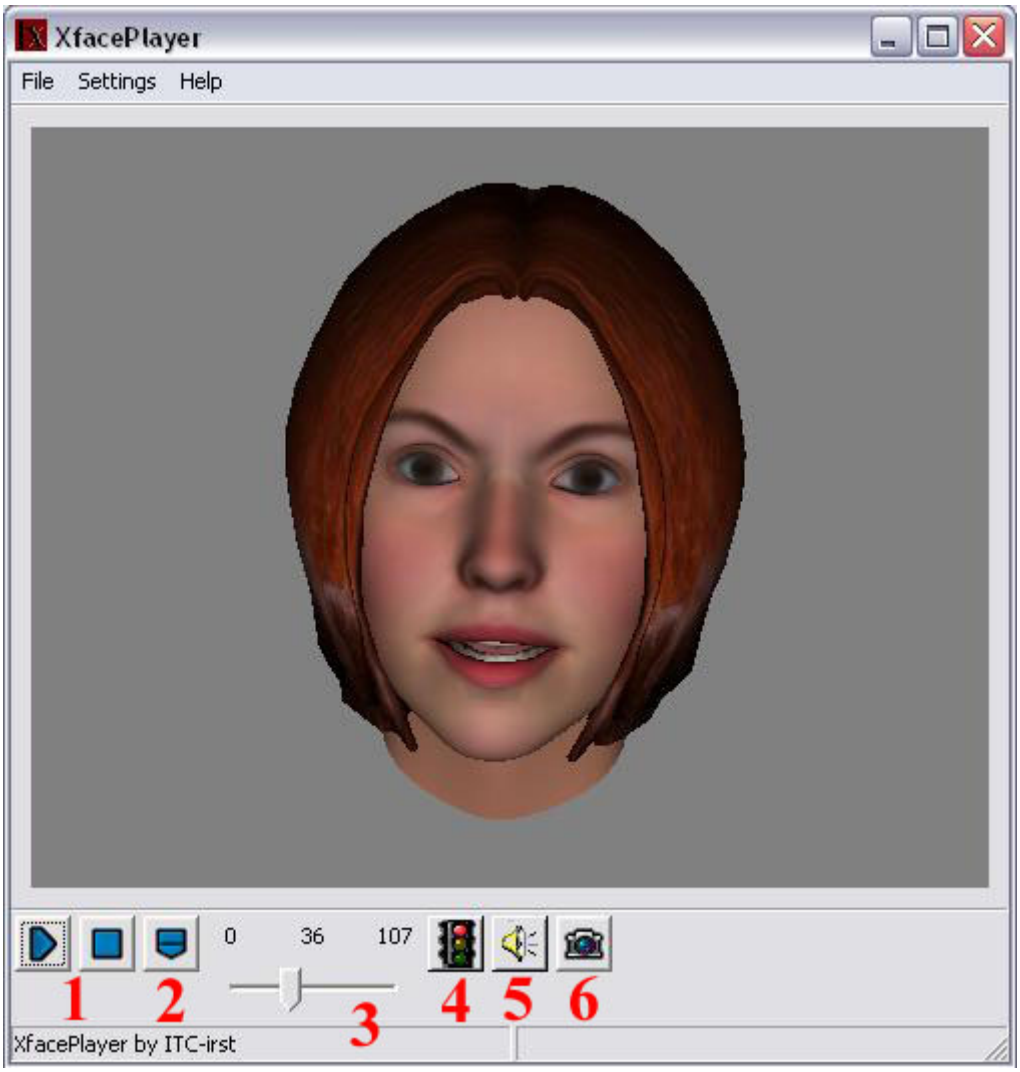

Figura 2-11: XFacePlayer. 1- Botons de reproducció i parada. 2- Botó per carregar un arxiu FAP i un arxiu WAV. 3- Barra de progrés de la reproducció. 4- Botó per activar/desactivar el mode servidor, per interactuar amb XFaceClient. 5- Botó per activar/desactivar la reproducció de l'àudio. 6- Botó per gravar la reproducció en conjunt en un arxiu de vídeo AVI.

2.2.5. XFaceClient

XFaceClient està pensat com el motor de comunicació dels ECAs. XFacePlayer necessita, com a entrada, l'àudio de la parla juntament amb els FAPs o les duracions dels fonemes amb etiquetes indicant les emocions i/o les expressions, en el cas dels "keyframes". Això es pot aconseguir utilitzant diversos llenguatges d'"scripting", sintetitzadors TTS i generadors de FAPs. Fins i tot també es pot combinar amb alguns altres "reproductors de cares". Per suportar tots aquests elements mencionats, aquests han d'estar especificats a l'arxiu "scriptprocs.xml", que XFaceClient llegeix, mitjançant XSmilAgent, juntament amb el text XML (APML, SMIL-Agent) que se li dóna

d’entrada. Després la llibreria fa totes les crides externes necessàries i XFaceClient li passa a l'XFacePlayer totes les entrades requerides, sense que de nou hi hagi res codificat explícitament. La comunicació es fa pel protocol estàndard TCP/IP i diverses tasques i dades poden ser "pujades" al reproductor mitjançant un esquema de missatges obert. Alguns exemples d’aquests missatges es poden veure a la Figura 2-12: hi ha la tasca UPLOAD_ANIM, que avisa el servidor de que s’està pujant l’arxiu d’animacions, mentre que els missatges que pujaran l’arxiu en sí són un exemple de transmissió de dades. Hi ha també les tasques RESUME_PLAYBACK i STOP_PLAYBACK, que, com el seu nom indica, inicien i aturen el procés de reproducció de la seqüència al servidor. Aquest, per la seva banda, va responent a aquestes tasques amb missatges que indiquen l’estat d’aquesta: en cua, iniciada, finalitzada correctament o erròniament, etc.

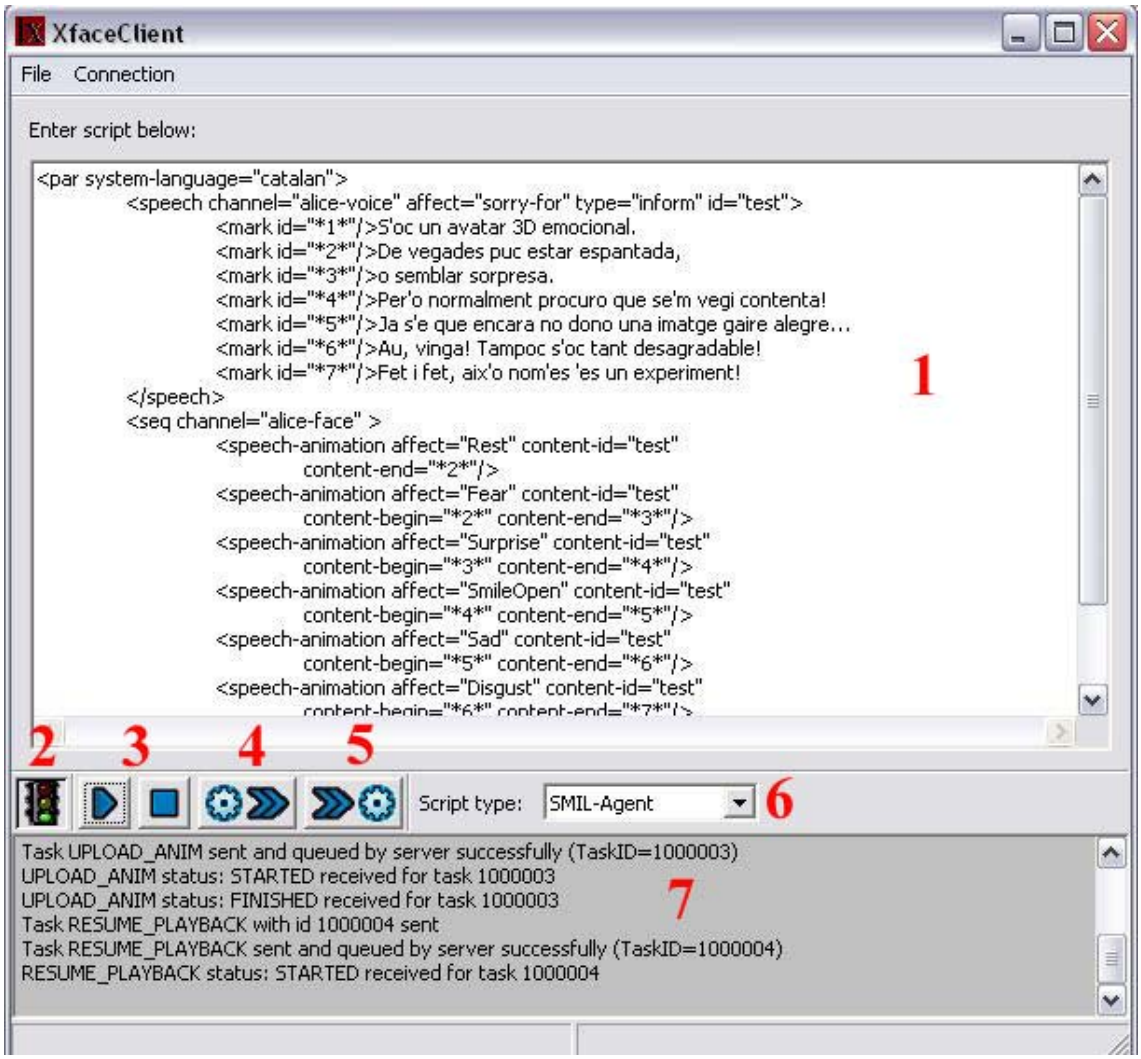

Figura 2-12: XFaceClient. 1- Editor de textos amb l'script d'entrada pel reproductor. 2- Botó per conectar amb XFacePlayer (el reproductor objectiu ha de tenir el mode servidor activat). 3- Botons de reproducció i parada. 4- Botó per processar script d'entrada al client i enviar els fitxers. 5- Botó per enviar l'script d'entrada al servidor per a que sigui processat en aquest. 6- Selector de les eines que processaran l'script (les proporcionades per "scriptprocs.xml"). 7- Missatges de sortida.

Capítol 3. Eines per a la síntesi de la parla

Un cop entès el funcionament d'XFace, la següent tasca a dur a terme és la cerca de les eines (de software lliure) necessàries per afegir la parla de la llengua catalana (veure Figura 3-1). En aquest apartat s'explica doncs el procés que es segueix en aquest estudi, una comparativa detallada entre les diferents opcions trobades i les conclusions a les que s'arriben.

3.1. Requeriments inicials

En un principi, abans de buscar eines de sintetització de veu, es procedeix a la cerca d'algun intèrpret que generi FAPs a partir d'un script d'SMIL-Agent (o algun llenguatge de "markup" similar) d'entrada, per tal de seguir l'estàndard FA d'MPEG-4, donat que XFace està programat principalment per funcionar amb aquest. Després de buscar a fons, però, no se'n troba cap. Consultat al respecte, Koray Balci, membre de la divisió TCC de l'ITC-irst**[5]** i principal dissenyador de l'eina XFace, diu que ara com ara no existeixen eines conversores a FAP a l'abast del gran públic, i que les que hi ha són molt difícils d'aconseguir. De fet, ell ara mateix està treballant en un conversor de SMIL-Agent a FAP de software lliure.

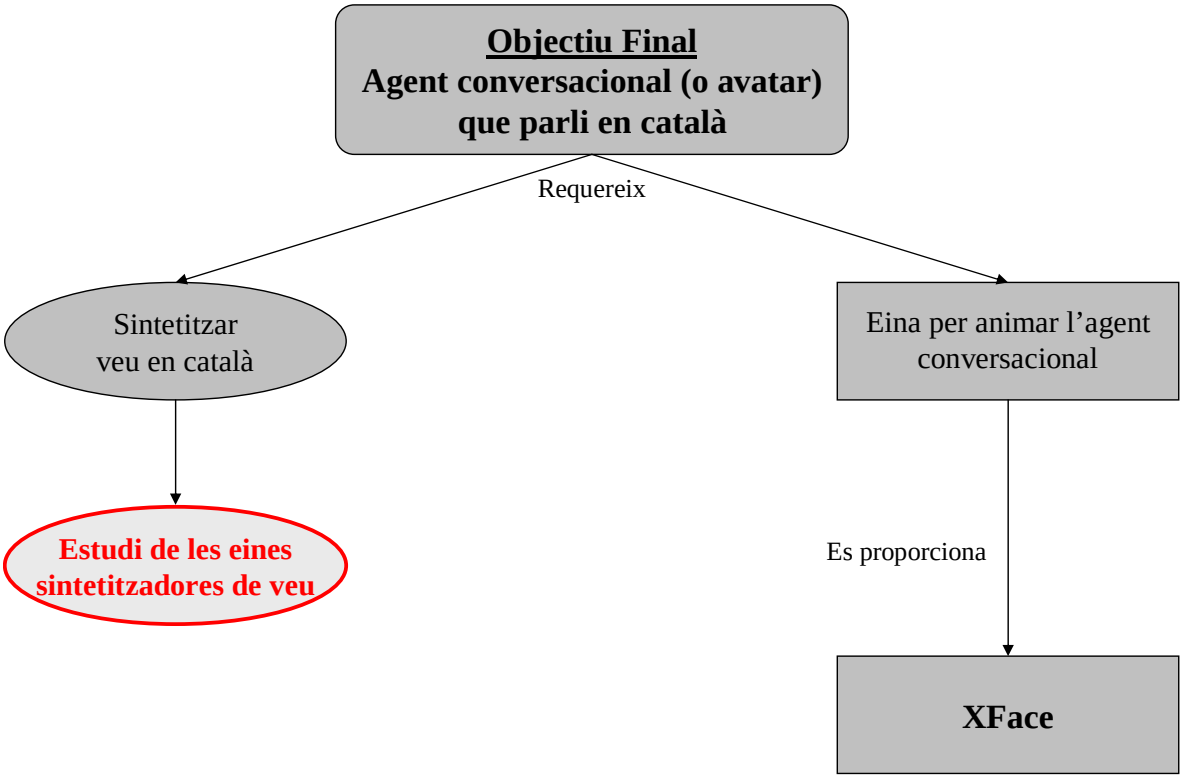

Figura 3-1: Esquema d'organització de les tasques del treball, 3a fase - Estudi de les eines sintetitzadores de veu.

Decidit doncs que s'usaria el mètode alternatiu de "keyframes", només queda centrar-se en la busca d'una eina TTS que tingués o se li pogués definir una veu en català. El resultat, però, és de nou negatiu i només es troben tres sintetitzadors que generin veu en aquest idioma, essent dos d'ells software propietari i de pagament i el tercer un sintetitzador propi, però no públic, de la UPC. A rel d'això, es procedeix a buscar i estudiar diferents eines TTS de software lliure, per tal de triar la que pugui ser la més adequada per aquest projecte. Cal remarcar que durant aquesta cerca es troba el projecte **MBROLA[13]**, projecte de síntesi de veu orientat a la recerca, gratuït però no de codi obert i que compta amb una base de dades de veus en diferents idiomes molt extensa (35 idiomes/dialectes i 67 veus a l'hora d'escriure aquest document). Si bé en un principi es decideix mirar de no fer-lo servir degut a que no és software lliure, la compatibilitat de les eines TTS amb aquest sí que es té en compte com a aspecte positiu, degut a la seva gran utilitat.

3.2. Eines i comparativa

Els diferents softwares TTS trobats són els següents:

- **Festival:** Eina TTS de codi lliure altament modular i multilingüe orientada tant a l'ús en aplicacions finals com a la recerca, creada pel CSTR (*The Centre for Speech Technology Research*) de la Universitat d'Edimburg.**[14]**
- **Flite:** Creada a la Universitat Carnegie Mellon (CMU) de Pittsburgh, EUA, per alguns dels co-creadors de Festival, es tracta d'una versió molt simplificada d'aquest (el seu nom vé de *Festival Lite*), optimitzat per a funcionar en temps real en plataformes amb recursos limitats, principalment PDAs i telèfons mòbils per una banda i servidors amb molts clients per l'altra.**[15]**
- **FreeTTS:** Adaptació al llenguatge Java de l'anterior realitzada per l'*Speech Integration Group* de *Sun Microsystems Laboratories*.**[16]**
- **Gnuspeech:** Paquet TTS de sintetització de veu en temps real basat en el model TRM (*Tube Ressonance Model*, Model de Resonància de Tub), el qual imita les propietats físiques del tracte vocal (cordes vocals, faringe, boca...) humà. Actualment funciona sobre el sistema operatiu *UNIX-like* NeXTSTEP (tant versió MacOSX**[17]** com GNUStep**[18]**) i està sent adaptat a GNU/Linux.**[19]**
- **NeXTeNS:** Adaptació de Festival i MBROLA per a un TTS específic per a l'holandès creada pel Departament de Llenguatge i Parla de la Universitat de Nijmegen i el grup d'Inducció del Coneixement Lingüístic de la Universitat de Tilburg (Holanda).**[20]**
- **EPOS:** Projecte d'eina TTS de codi lliure i multilingüe basat en regles desenvolupat actualment a Praga.**[21]**

Un cop trobats aquests es procedeix a comparar-ne els avantatges i els inconvenients de cadascun a l'hora d'emprar-los en aquest projecte. Aquests s'enumeren a la taula següent:

TTS	Avantatges	Inconvenients
Festival	+ El software lliure TTS més extès, fet servir en entorns de recerca d'arreu del món. + Molt documentat. + Existeix un projecte que facilita i	- Preparat per funcionar en entorns Linux, és compilable sota Windows, però complicat. - Sistema molt gran i complex.

	estandarditza la creació de noves veus, Festvox. + Ús d'un llenguatge d'scripting d'alt nivell per una major independència del codi central. + Inclou suport per MBROLA. + Inclou una veu castellana a partir de la qual fer senzillament la catalana.	
Flite	+ Simple, senzill + Alternativa a Festival, optimitzat per a ser més ràpid i portable.	- Pensat per ipaq, Palm OS i sistemes petits en general... o per servidors amb molts clients. Com Festival, no preparat per Windows (excepte CE). - No fa servir llenguatge d'scripting d'alt nivell, cosa que fa molt complicat canviar paràmetres de baix nivell o recompilar el codi. - Les veus s'han de fer sobre Festival i passar-les després a Flite de manera semi-automàtica.
FreeTTS	+ Escrit en Java, compilable des de qualsevol sistema amb màquina virtual per aquest. + Inclou suport per MBROLA ja implementat.	- Per crear veus s'ha de fer servir igualment Festival. - No es pot fer servir MBROLA directament per canviar d'idioma. - Menysprea difonemes improbables per millorar la velocitat, cosa que pot provocar errors puntuals.
Gnuspeech		- Només hi ha versions per MacOS/X i GNUStep. Això el fa massa complicat de coordinar amb XFace.
NeXTeNS	+ Compilable sota Windows. + Té GUI per fer-ne més fàcil l'ús, però això no serveix pel cas d'XFace.	- És una adaptació específica a l'holandès de Festival i MBROLA. - Degut a que té incorporat aquest segon, no és completament software lliure. - Poc documentat.
EPOS	+ Compilable sota Windows. + Suport per MBROLA. + Un sol arxiu de regles per definir un idioma.	- Només funciona en mode client-servidor. - Només Txec i Eslovac definits. - Massa difícil d'adaptar al català.

Taula 3-1: Comparativa entre les diferents eines TTS.

3.3. Conclusions: Per què Festival?

Finalment s'escull Festival com a software a utilitzar. Les raons per aquesta decisió són moltes: En una primera instància, pel que fa a aquest treball en concret, la presència d'una veu senzilla definida en castellà, que podria ser modificada per al català, en cas de que no es trobés la manera o el temps de fer-ne una de completa, donat que l'objectiu d'aquest és la recerca d'una eina TTS plenament funcional amb una veu catalana definida per a incorporar-la a XFace, i per tant la qualitat d'aquesta no és primordial. En aquesta mateixa línia d'obrir camins per a la definició d'una veu catalana, la possibilitat d'ús d'MBROLA és també un factor favorable, donada la seva gran base de dades; però, si bé altres eines dins de les considerades donen més facilitats al respecte (FreeTTS, NeXTeNS, EPOS), tampoc és determinant degut a la naturalesa de codi tancat d'aquest.

Pensant ja més en un futur proper, però, és on Festival es perfila més com l'eina més apropiada, donat que es tracta d'un projecte universitari obert i amb llistes de mail actives, cosa que facilita una futura cooperació en la recerca. I, si bé és cert que tots els altres projectes també compleixen aquesta condició, Festival és sens dubte el que sobresurt més, el que té més gent treballant-hi o participant-hi per millorar-lo més cada dia. O, com es diu a la introducció del NeXTeNS: "En concret, el sistema Festival i el sintetitzador MBROLA sembla que s'han convertit en els estàndards de facto per sistemes TTS de software lliure dins la comunitat de recerca i educació."**[20]**. Serveixi com a exemple l'existència del projecte **Festvox** de la Universitat Carnegie Mellon**[22]**, el qual proporciona un procés que facilita i estandarditza la creació de noves veus mitjançant Festival, però no només per aquest: les eines Flite i FreeTTS, per exemple, requereixen que qualsevol nova veu que s'incorpori s'hagi creat seguint aquest procés. A més, per acabar, Festival té el gran aventatge de que està dissenyat per a ser tant una eina TTS plenament funcional, ja avui en dia, com un entorn de desenvolupament i investigació; o, com diu Black, de la CMU: "com que el desenvolupament té lloc sobre el mateix sistema bàsic que les aplicacions finals fan servir, això permet una relació directa entre 'recerca' i 'ús'"**[23]**. És a dir, la comunicació directa amb els usuaris finals pot orientar la recerca (i de fet ja ho ha fet en més d'un cas) a resoldre problemes i situacions reals, així com a fer el software més robust.

Capítol 4. Festival

Decidida, doncs, l'eina TTS a emprar, el següent pas és, com en el cas d'XFace, l'estudi d'aquesta per tal d'entendre'n el seu funcionament (veure Figura 4-1) per posteriorment decidir com es farà interactuar amb l'avatar per una banda i la definició de la veu catalana per l'altra. En aquest apartat es procedeix a la seva descripció.

4.1. Introducció

Festival és "un conjunt d'eines que ofereix un marc general per a crear sistemes de síntesi de veu, així com diversos mòduls ja implementats com a exemple"**[24]**. En conjunt, ofereix un servei de TTS complet a traves de diferents APIs: a nivell de "shell", a nivell de consola de comandes mitjançant l'ús d'un llenguatge d'scripting d'alt nivell, a nivell de llibreria de C++ i com interfície d'Emacs. Com es pot veure, està dissenyat (i garantit) per a ser compilat i utilitzat en sistemes Linux, però, com XFace, els mòduls principals no fan servir llibreries o programes propis del sistema operatiu i és compilable en d'altres. Pel que fa als idiomes, Festival és un sistema multilingüe: permet generar veus en qualsevol idioma i incorpora, ja de per sí, veus en anglès (britànic i americà), castellà i gal·lès, si bé les veus angleses són les més avançades amb diferència.

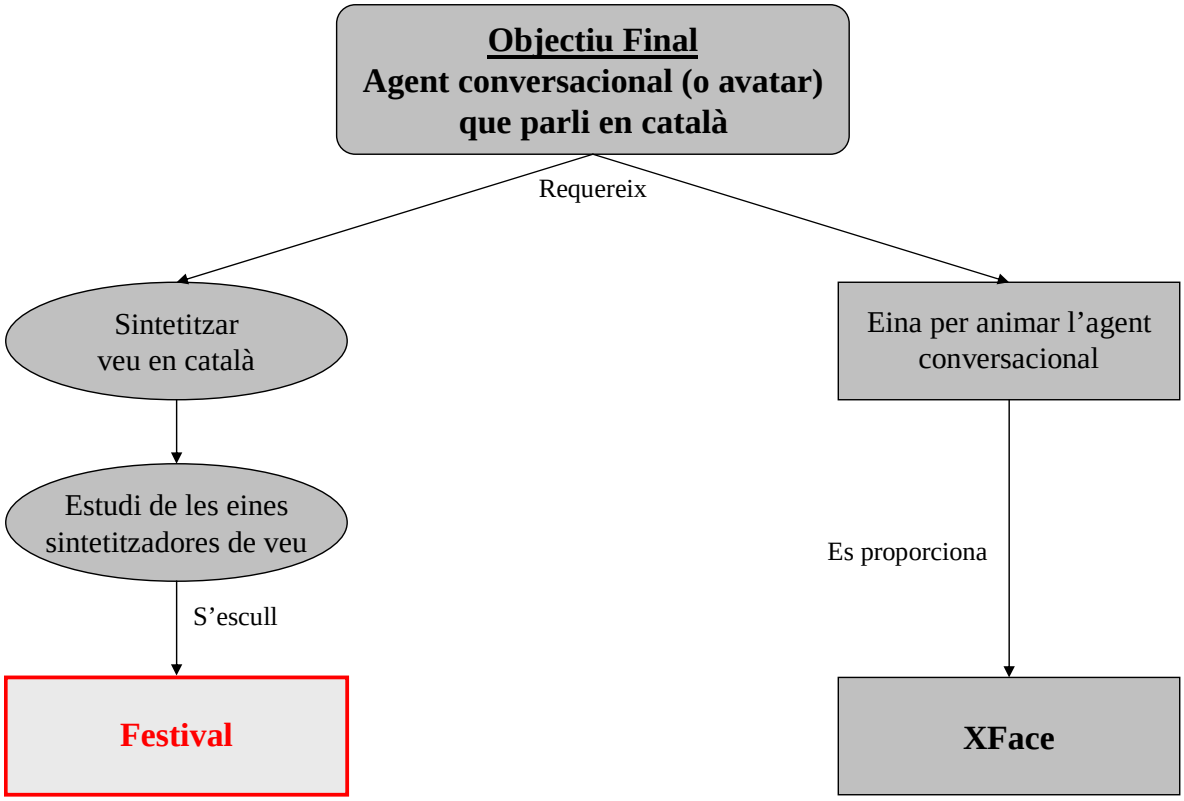

Figura 4-1: Esquema d'organització de les tasques del treball, 4a fase - Estudi de Festival.

Festival està pensat, dissenyat i programat com una eina de síntesi de veu per a, almenys, tres "nivells" o tipus d'usuari: Primer, al nivell d'Usuari, es troben aquells que simplement desitgin generar veu d'alta qualitat a partir de text arbitrari amb el mínim d'esforç. Al nivell d'Expert pertanyen aquells que estiguin desenvolupant sistemes de llenguatge i desitgin incorporar veu sintetitzada, cas en el qual es desitjarà una certa adaptabilitat del programa, com la possibilitat de definir noves veus o modificar les existents, modificar les sortides del programa, etc. I, per acabar, el nivell de Programador consisteix en la recerca, desenvolupament i prova de nous mètodes de síntesi, on ja s'ha de treballar a nivell de codi font.

Per als Usuaris, existeix la possibilitat de cridar l'executable principal amb tota una sèrie de paràmetres (veure Figura 4-2), d'entre els quals destaca el "--tts", el qual arriba a l'extrem de simplificació en que se li pot passar un arxiu d'entrada i Festival en sintetitzi el contingut i el reprodueixi immediatament.

Figura 4-2: Exemples de crida a Festival des de shell mitjançant el paràmetre "--tts". El primer és una crida normal, que sintetitza el contingut de l'arxiu "test_mode_TTS.txt", mentre que el segon i el tercer són altres mètodes d'us que permeten introduir el text a sintetitzar directament de teclat.

A un nivell una mica més avançat hi ha la consola de comandes del llenguatge d'alt nivell **Scheme**, dialecte del llenguatge Lisp**[25]**, de les quals se n'encarrega l'intèrpret SIOD (*Scheme In One Defun 3.0*) **[26]**, creat originalment per George J. Carrette i modificat i millorat lleugerament per a aquest software, i que permeten un control més directe sobre el programa (veure Figura 4-3).

Pels Programadors, Festival ofereix, per una banda, la seva arquitectura molt modular, específicament dissenyada per a poder afegir nous mòduls o modificar els existents de manera senzilla i eficient, i, per l'altra, el fet de que l'eina en sí serveixi com a entorn de test plenament funcional, el qual, a més, serà el mateix que els usuaris finals faran servir.

Finalment, als usuaris del nivell Expert, en el qual es situa aquest treball, permet usar-la entenent-la només a nivell funcional, sense entrar en els algorismes que es facin servir per implementar cada part. Com és el cas dels Usuaris, en aquest nivell només s'empra Scheme si bé aquí ja es fan servir/modifiquen els scripts propis de Festival que es criden en carregar el programa i en defineixen les variables principals, els que defineixen les veus, etc.

És, per tant, des d'aquest darrer punt de vista que s'estudia Festival, i des del qual es procedeix a explicar-ne el funcionament.

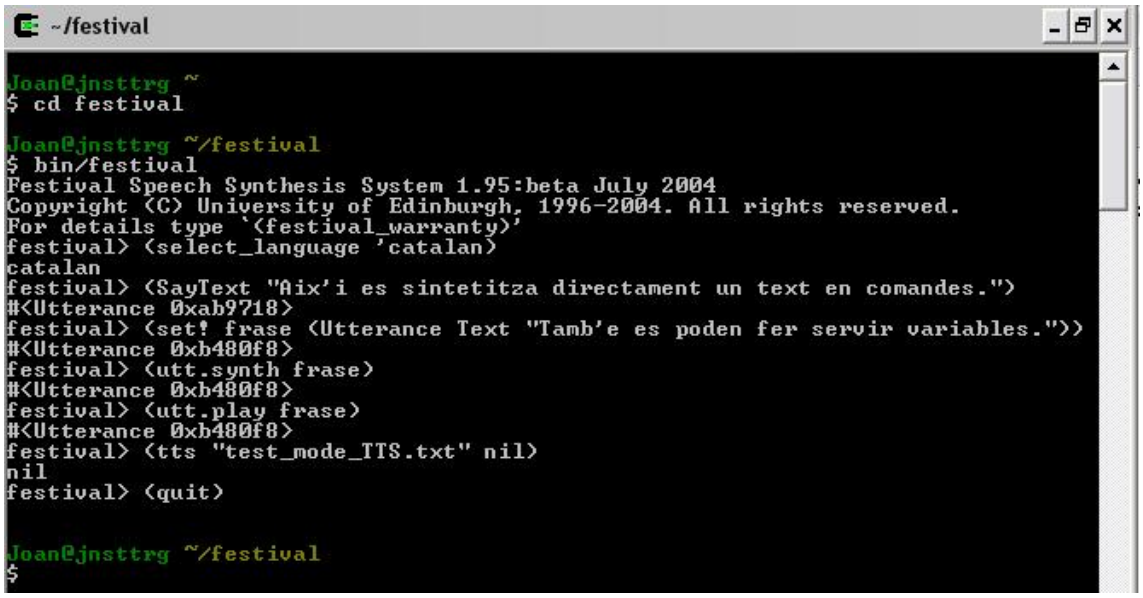

Figura 4-3: Exemple d'execució de Festival mitjançant comandes Scheme.

4.2. Funcionament

A l'hora de sintetitzar un text en veu, totes les eines TTS divideixen el problema en dues parts, segons es mostra a la Figura 4-4**[27]**: La primera, que es pot veure com la "interpretació o "lectura" del text d'entrada en sí, és produir una transcripció fonètica d'aquest amb la **prosòdia** desitjada, on s'entén per "prosòdia" certes propietats de la senyal de la veu tals com són, principalment, canvis audibles en el tò, volum i durada de la síl·laba, si bé alguns autors també n'inclouen d'altres com per exemple el "ritme", que és el temps que passa entre síl·labes tòniques.

La segona és sintetitzar els resultats del procés anterior en una forma d'ona de sortida. Per a assolir això existeixen dos mètodes: per una banda hi ha la **síntesi concatenativa**, que consisteix en la concatenació d'**unitats** (senyals) prèviaments gravades d'una veu humana real, tractades i emmagatzamades en una **base de dades** (que si bé s'anomena així pot tractar-se només d'un simple fitxer de text. Depèn de la quantitat d'unitats que requereixi el mètode emprat). Per l'altra banda hi ha la **síntesi formant** la qual consisteix en la sintetització artificial d'un model acústic. Mentre que els mètodes del primer tipus tendeixen a sonar més naturals, la interpolació entre diferents sons pot provocar petits errors o soroll; en el cas del segon tipus, en canvi, si bé la veu resultant acostuma a ser menys natural i més robòtica, pot donar molt bons resultats pel que fa a la intel·ligibilitat.

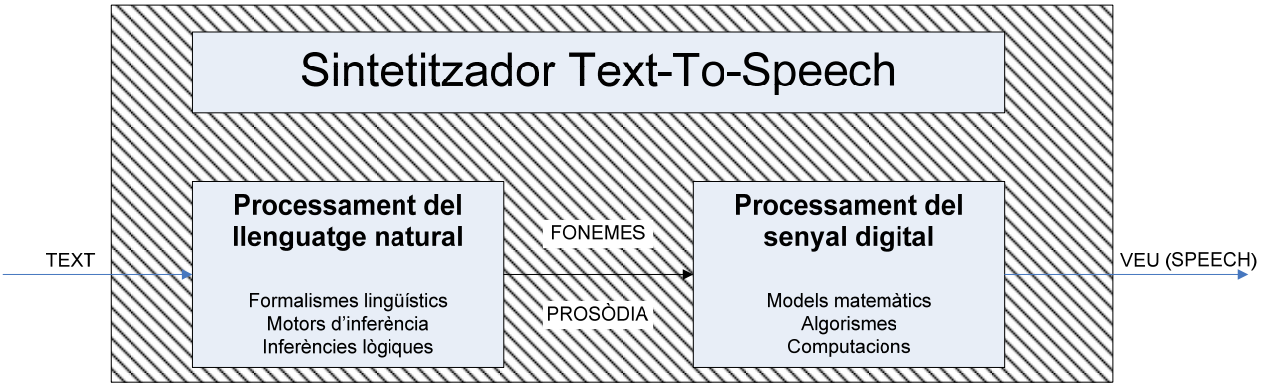

Figura 4-4: Esquema dels mòduls bàsics d'una eina TTS segons Dutoit[27].

Així doncs, entrant ja en el cas concret de Festival, aquest s'encarrega principalment de la primera part, processament del llenguatge, deixant la tasca final de la síntesi a una eina externa. Aquesta és per defecte **Edinburgh Speech Tools**, sintetitzador de veu creat també per la Universitat d'Edimburg el qual, si bé ha estat desenvolupat per separat, certes parts s'han fet segons han requerit les necessitats del projecte Festival, de manera que l'un està optimitzat per a funcionar amb l'altre. De fet, Festival també fa servir la llibreria d'Speech Tools en altres parts, com per exemple a l'hora de definir l'estructura d'una ***utterance*** (que es traduirira com a una "pronunciació"), de manera que aquesta és sempre indispensable. Per la generació de veu en sí, però, Festival també permet l'ús d'altres sintetitzadors, essent el més usat entre ells (i pel qual Festival ja està preparat per a usar-lo com a alternativa) el prèviament mencionat MBROLA.

En qualsevol cas, però, ara com ara Festival només soporta síntesi concatenativa, estant molt centrat, avui en dia, en el mètode de síntesi per **difonemes**. Aquestes unitats són totes les combinacions possibles de dos fonemes dins del conjunt dels mateixos d'un idioma, inclòs el silenci. Més concretament, comencen al mig de l'estat estable del primer i acaben al mig de l'estat estable del següent. Avui en dia és el mètode més utilitzat en general (MBROLA, per exemple, també es basa en difonemes) degut a que combina la tendència a la naturalitat de la parla resultant pròpia dels sistemes de concatenació amb requisits escassos de memòria, degut al tamany relativament petit de les bases de dades de les unitats.

Tornant al processament del llenguatge, per a poder entendre el funcionament de Festival, cal tenir en compte el concepte prèviament mencionat d'***utterance***. Aquesta és l'objecte bàsic a sintetitzar per a Festival, i representa un "tros" de text que s'ha de pronunciar i que normalment no arriba a conformar una frase sencera. En general, el procés de sintetització de veu es pot explicar com agafar una *utterance* que conté una simple cadena de caràcters i anar-la transformant, pas per pas, omplint la seva estructura amb més i més informació fins a arribar a construir la forma d'ona que "pronuncia" (*utters*) el contingut del text.

Els passos que es segueixen en aquesta conversió són, generalment, els següents:

Aquests passos s'il·lustren amb l'utterance d'exemple "Feb. 25", en anglès donat que tant la veu catalana com la castellana original són força incomplertes.

1. **Tokenització:** Convertir la cadena de caràcters en una llista de "tokens". Normalment això vol dir simplement cadascun dels elements del text separats per un espai en blanc.

"Feb. 25" es converteix en la llista (Feb, 25)

2. **Identificació de tokens:** Identificació dels tipus generals de "token" (és a dir, si un token és una paraula, unes sigles, una abreviació, un nombre, un signe de puntuació, etc.). Procés normalment trivial però que requereix un cert esforç per identificar normalment "tokens" de dígits com poden ser anys, dates, nombres, etc.

La regla

```
((member_string name '(jan january feb february mar march
                       apr april may jun june
                       jul july aug august sep sept september
                       oct october nov november dec december))
     'month)
```

identifica el token "Feb" com a *month* (mes) i la regla

```
((string-matches name "[0-9]+")
     'numeric)
```

identifica el token "25" com a *numeric*. Finalment, la regla

```
((p.lisp_token_pos_guess is month)
      ((lisp_month_range is 0) ((year)) ((ordinal)))
      ((n.lisp_token_pos_guess is month)
     ((lisp_month_range is 0) ((cardinal)) ((ordinal)))
```

identifica aquest darrer com a *ordinal*.

3. ***Token to word*:** Conversió de cada "token" en zero, una o més paraules (és a dir, no convertir "?" en cap paraula, sinó tenir-lo en compte per a la prosòdia, convertir "3" en "tres", "PFC" en "pé efa cé", "etc." en "etcètera"...). En aquest pas i l'anterior es solen fer servir les anomenades ***token to word rules*** (regles de conversió de "token" a paraula).

Les regles dins del codi font (arxiu "token.cc")

```
else if (iword < 100)
    {
      tens = iword / 10;
      units = iword % 10;
      switch (tens)
      {
        case 2: s_tens = strintern("twenty"); break;
```

i

```
else if (streq(lastword,"five"))
      CAR(last) = strintern("fifth");
```

converteixen el token "25" en les paraules "twenty" i "fifth"

4. ***Part of Speech* (morfologia):** Identificació de la morfologia per cadascuna de les paraules.

Festival assumeix la següent morfologia:
"feb" com a verb en gerundi, "twenty" com a nombre cardinal i "fifth" com a adjectiu.

5. **Frases Prosòdiques:** Divisió de l'*utterance* en **frases prosòdiques**, o, el que és el mateix, en el cas d'*utterances* llargues (sobretot si aquestes no tenen signes de puntuació) saber on posar les pauses, on acaba una afirmació i comença una pregunta, etc.

La frase de l'exemple no es divideix, per tant queda una sola frase: "feb twenty fifth".

6. **Cerca lèxica:** Per cada paraula cerca d'aquesta en les llistes de lèxic donades en la definició de la veu, i si no es troba (o si la veu no en té) en les ***letter to sound rules***, una gramàtica depenent del context consistent en una llista de normes per convertir les lletres en sons -fonemes-. D'aquesta manera es defineix tant l'estructura fonètica (fonemes dels que es composa la paraula) com sil·làbica (separació de síl·labes d'aquesta).

A partir de les següents entrades del lèxic de la veu:

```
("feb" nil (f eh1 b y ax w eh1 r iy0))
("twenty" nil (t w eh1 n t iy0))
("fifth" nil (f ih1 f th))
```

s'extreuen els fonemes d'aquesta utterance: "pau", "f", "eh1", "b", ..., "f", "th", "pau". On el fonema "pau" és el de pausa, que es posa sempre al principi i al final d'una frase prosòdica, i el nombre al final de cada vocal simplement indica si aquesta serà tònica o atona.

7. **Accents, entonació:** Assignació d'accent a les síl·labes adequades (definició de les síl·labes tòniques).

S'assignen els accents a les síl·labes "f eh b" (feb), "t w eh n" (twen) i "f ih f th" (fifth).

8. **Assignació de la durada:** Assignació a cada fonema de la seva durada. Aquesta és la informació que interessa fer arrivar a XFace, dins l'arxiu de fonemes.

Festival assigna 0,22 segons al fonema "pau", 0,12 al fonema "f", 0,09 a "eh", 0,08 a "b", 0,03 segons a "r", etc.

9. **Generació de la F0:** On **F0**, o la **freqüència fonamental**, és la freqüència bàsica d'una frase prosòdica, la qual principalment varia segons si aquesta és una pregunta, una afirmació, una exclamació... i que normalment es coneix com a "tò" (a la Figura 4-5 es poden veure diferents formes de l'F0 segons el tipus de frase). Així doncs, aquest pas consisteix en el càlcul d'aquest tò basant-se en els accents, puntuació...

En el cas de l'exemple, Festival assigna els valors d'F0 mostrats a la Figura 4-6.

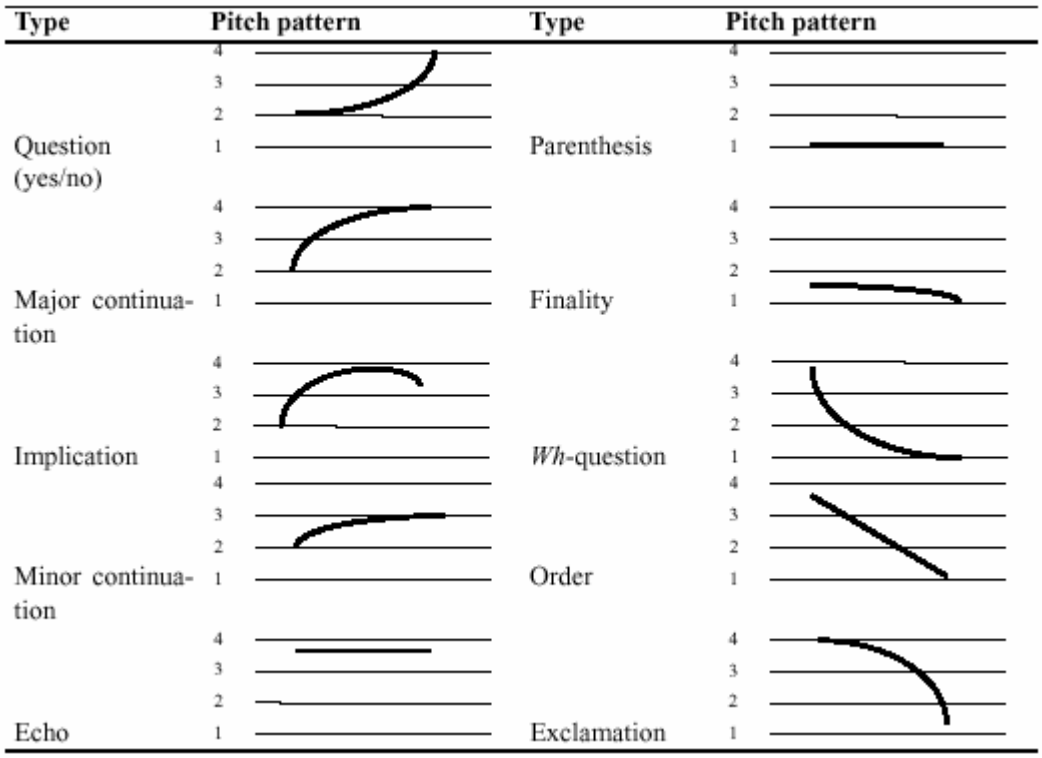

Figura 4-5: Les 10 entonacions bàsiques del Francès definides per Pierre Delattre[28]

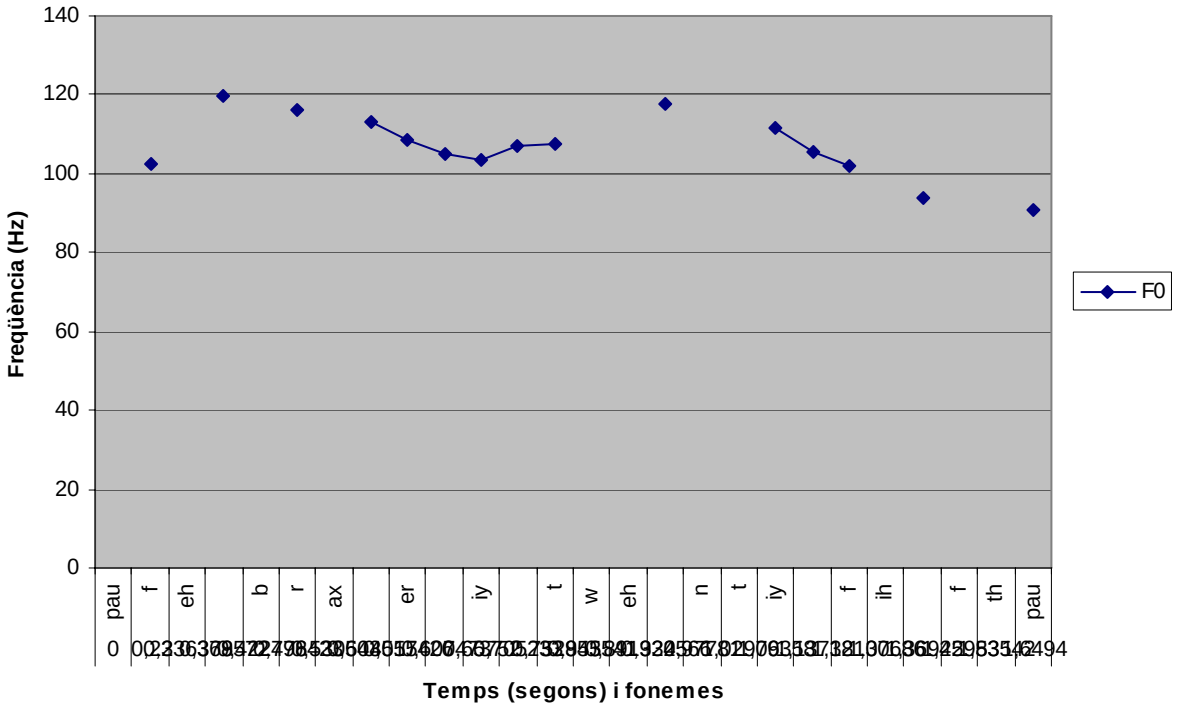

Figura 4-6: Gràfic d'assignació de valors d'F0 per a l'exemple "Feb. 25".

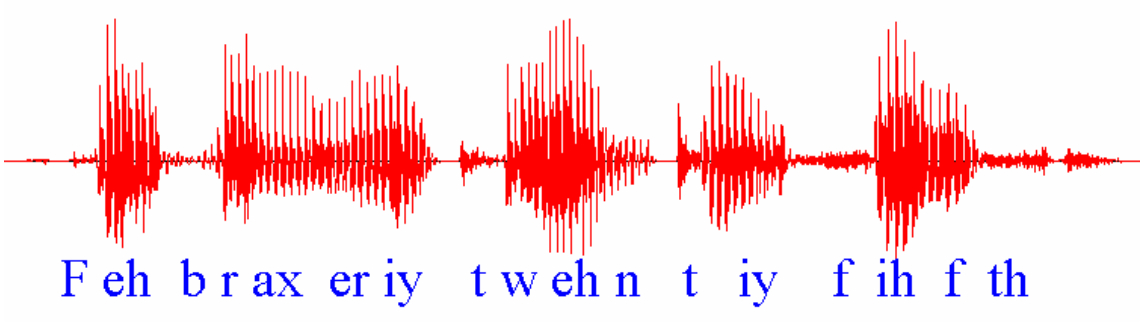

Figura 4-7: Forma d'ona de l'*utterance* "Feb. 25".

10. **Generació de la forma d'ona:** Finalment generació de la forma d'ona a partir dels fonemes, duracions i valors d'F0 respectius. Aquest procés normalment es divideix al seu temps en una sèrie de passos com poden ser sel·lecció d'unitats (difonemes o altres tipus d'unitats amb tamany donat), imposició de la prosòdia desitjada i reconstrucció de la forma d'ona (veure Figura 4-7).

Cal dir, però, que aquest procés és només una definició general i que el nombre de passos que es segueixin i, en definitiva, el que realment "passi" durant la síntesi, pot variar i dependrà de la veu sel·leccionada i com estigui definida aquesta (per exemple, les veus castellana i catalana d'aquest treball no apliquen el pas 4, càlcul de la morfologia) i del tipus d'*utterance* que s'estigui sintetitzant.

Cadascun d'aquests passos és dut a terme per un **mòdul**, el qual afegirà la nova informació a l'estructura de l'*utterance*. Aquesta consisteix en un conjunt d'**ítems** que poden formar part d'una o més **relacions**. Els ítems representen "coses" com poden ser les paraules o els fonemes, si bé també poden ser objectes menys concrets com frases nominals o nodes amb valors numèrics en arbres de decisió. Cada ítem conté un conjunt de **característiques** (normalment, com a mínim "nom" i "valor"). Les relacions, per la seva banda, són llistes o arbres d'ítems (per exemple la relació *Word* és una llista d'ítems cadascun dels quals representa una paraula dins la *utterance*), i, finalment, un ítem pot pertanyer a més d'una relació (seguint amb l'exemple d'abans, les paraules de la relació *Word* també són les arrels de cadascun dels arbres dels que es composa la relació *SylStructure*). Serveixi per il·lustrar aquesta estructura la Figura 4-8, un gràfic que mostra les relacions *Word*, *Syllable*, *Segment* i *SylStructure* (explicades més endavant) pel text prèviament processat "Feb 25":

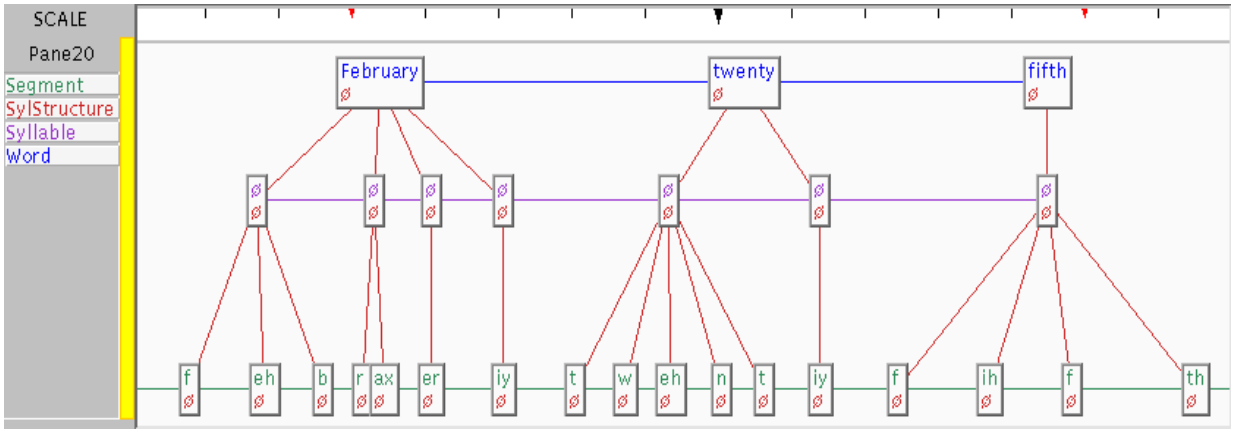

Figura 4-8: Relacions *Word, Syllable, Segment* i *SylStructure* d'una *utterance*.

L'estructura d'una *utterance* és completament oberta, de tal manera que nous mòduls que s'afegeixin puguin utilitzar els ítems i relacions que calgui o afegir-ne de nous. Les relacions més habituals, però, són les següents:

- ***Token*:** Una llista d'arbres. Inicialment és una llista dels diferents "tokens" i després, a partir de cadascun d'aquests en surten els fills que són totes aquelles paraules que formen el "token".

Seguint amb l'exemple de "Feb. 25", la Figura 4-9 en mostra aquesta relació. El primer "Feb" i el "25" en són les arrels ("tokens") i el segon "Feb" per una banda i "twenty" i "fifth" per l'altra les respectives fulles.

- ***Word*:** Una llista de les paraules de l'*utterance*.

Veure Figures 4-8 i 4-9 per al cas de l'exemple.

- ***Phrase*:** Una llista d'arbres formada per una arrel per cada frase, de les quals en surten les respectives paraules. L'arrel indica, també, el tipus de pausa que s'ha de fer en finalitzar la frase en qüestió, i els diferents valors que pren normalment són "B", de *Break* (pausa), 0,22 segons, "BB", de *Big Break* (gran pausa), 0,448 segons, i "NB", de *No Break* (sense pausa).

Veure Figura 4-9 per a l'exemple. En aquesta, la lletra "B"és l'arrel de l'arbre, i les paraules en són les fulles.

- ***Syntax*:** Un arbre binari que es branqueja sobre els membres de cada relació *Word*, es crea si es crida al "parser" (analitzador sintàctic) probabilístic.
- ***SylStructure*:** Una llista d'arbres que enllaça les relacions *Word*, *Syllable* i *Segment*. Cada paraula de la llista *Word* és l'arrel d'un arbre, els nodes fills del qual són les seves síl·labes i finalment les fulles són cadascun dels segments (fonemes i duració).

Veure Figura 4-8 per al cas de l'exemple.

- ***Syllable*:** Una llista de les síl·labes de l'*utterance*.

Veure Figura 4-8 per al cas de l'exemple.

- ***Segment*:** Una llista dels segments (fonemes) de l'*utterance*.

Veure Figura 4-9, on es mostra, per cada segment (fonema), el segon en el que comença i la duració en segons.

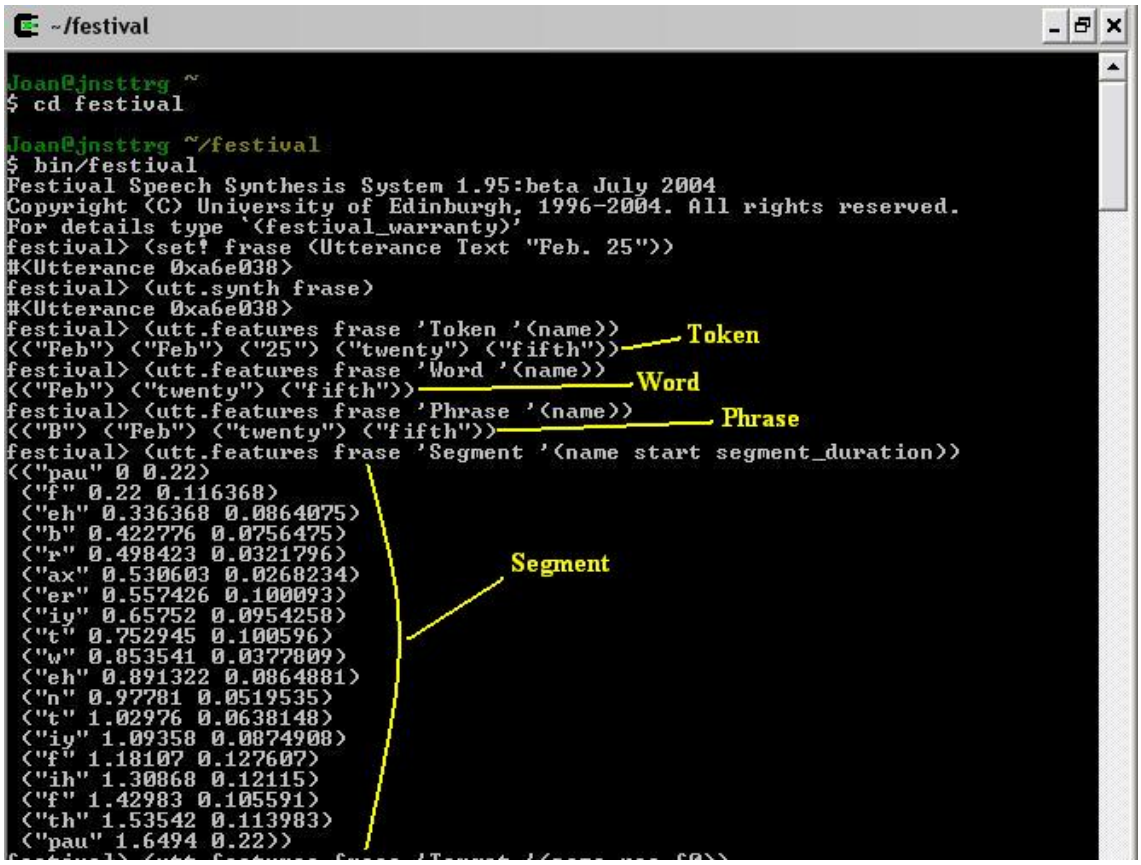

Figura 4-9: Algunes relacions de Festival per a l'*utterance* d'exemple "Feb. 25".

- ***IntEvent*:** Una llista dels "events d'entonació" de l'*utterance*. Aquests són els diferents tipus de tò que se'ls hi dóna als accents (síl·labes accentuades) i als extrems (primera i darrera síl·labes) per cada frase prosòdica, els quals poden ser més aguts ("H", *High*) o més greus ("L", *Low*) i durar més o menys temps ("H-L", per exemple, indica que dins la síl·laba es passa de més agut a més greu).

Veure Figura 4-10 per al cas de l'exemple.

- ***Intonation*:** Una llista d'arbres que relacionen les síl·labes amb els "events d'entonació". L'arrel dels arbres són cadascuna de les síl·labes i les seves fulles els "events" que li pertoquen.

Veure Figura 4-10 per al cas de l'exemple.

- ***Target*:** Una llista de llistes (parelles) que relacionen diferents valors d'F0 amb instants en el temps dins la durada total de la *utterance*, ordenada

cronològicament. En els casos que aquests valors d'F0 s'associïn a un fonema en concret, aquest també es guardat en aquesta com a pare de les parelles instant-F0 corresponents.

Veure Figura 4-10 per a l'exemple, on es mostra el contingut d'aquesta. Una llista d'instants (en segons) i el valor de F0 (en Hz) per cadascun d'aquests, així com els fonemes als que s'associen alguns dels valors, com, per exemple, la primera parella (0,22 102,618), que s'associa al fonema "f" de "feb.".

- ***Wave*:** Un sol ítem amb una sola característica anomenada també "wave", el valor de la qual és la forma d'ona ja generada.

Veure Figura 4-7 per al cas de l'exemple.

Aquesta llista, però, és el cas general, no és exhaustiva. Com ja s'ha dit, alguns mòduls poden afegir d'altres relacions, i no totes les *utterances* tenen perquè tenir-les totes.

Pel que fa als mòduls en sí, poden ser funcions definides directament en C/C++ i declarades amb un nom per a Lisp o funcions simples en Lisp que comproven algun paràmetre global abans de cridar a un altre mòdul (per exemple, per escollir entre diferents mòduls per a l'entonació). Els mòduls específics que es cridin en sintetitzar una *utterance* depenen del tipus d'aquesta que s'utilitzi (si bé també es poden afegir crides manualment): quan, per una *utterance* concreta, es crida a la funció principal de síntesi de Festival ("utt.synth"), aquesta aplicarà tots els mòduls donats pel tipus d'aquesta abans de sintetitzar-ne la forma d'ona.

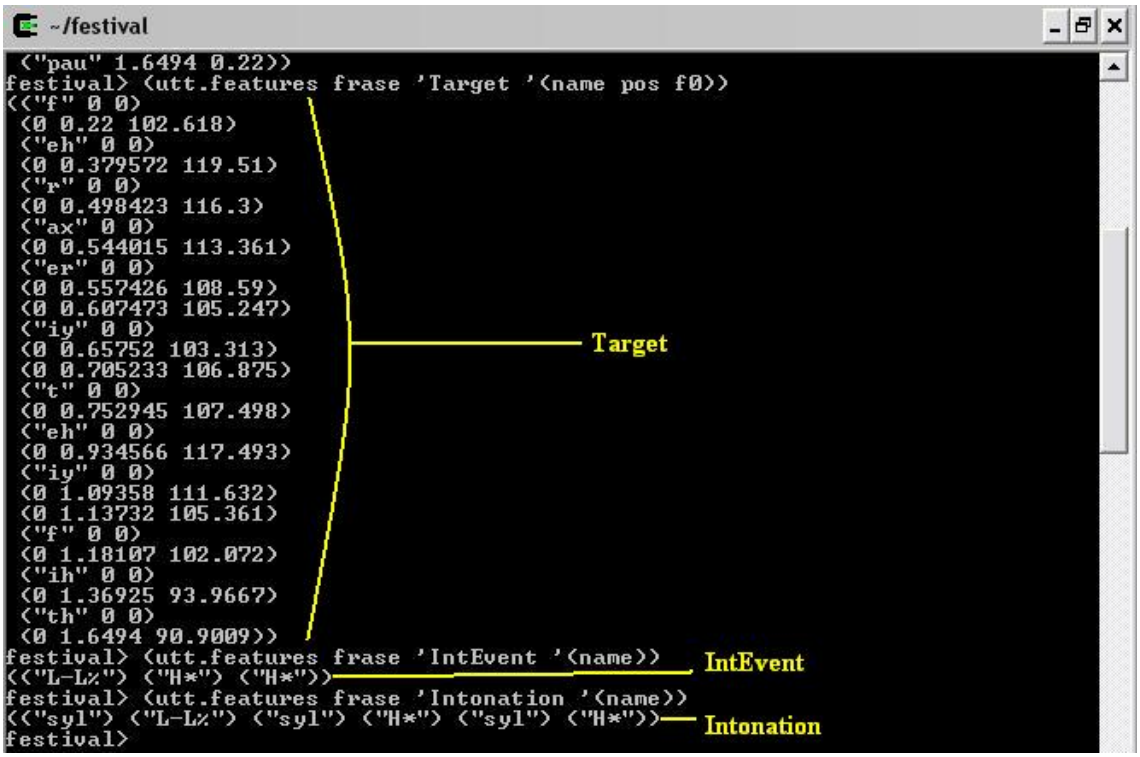

Figura 4-10: Altres relacions de Festival per a la mateixa *utterance*.

Per exemple, quan es sintetitza una *utterance* de tipus *Segment* només necessita que es carreguin els seus valors en una relació *Segment* i una relació *Target* i després ja es pot cridar al mòdul de síntesi de baix nivell (anomenat "Wave_Synth"). Aquest tipus es defineix com:

```
(defUttType Segments
  (Initialize utt)
  (Wave_Synth utt))
```

Un exemple per la paraula "hola":

```
 (Utterance
 Segments
 ((# 0.19 )
  (o1 0.037 (0.018 136))
  (l 0.064 )
  (a 0.208 (0.0 134) (0.100 135) (0.208 123))
  (# 0.19)))
```

En canvi, pel tipus més complex *Text*, es requereixen molts més mòduls abans de que es pugui sintetitzar:

```
(defUttType Text
  (Initialize utt)
  (Text utt)
  (Token utt)
```

```
  (POS utt)
  (Phrasify utt)
  (Word utt)
  (Intonation utt)
  (Duration utt)
  (Int_Targets utt)
  (Wave_Synth utt)
)
```

Exemple:

```
 (Utterance Text "Exemple de text")
```

Cal remarcar que el mòdul “Initialize” s’hauria de cridar en gairebé tots els tipus, com a norma general, ja que carrega totes les relacions necessàries a partir de l’entrada i esborra totes les altres relacions prèviament definides que puguin existir.

Finalment, cal remarcar dos conceptes més de Festival que són emprats en aquest treball per definir la interacció amb XFace: els ***hooks*** i els **modes**.

Durant l’execució de certes funcions (com són els casos d’“utt.synth” o “tts” i “tts_file”, que sintetitza una entrada donada per arxiu) s’apliquen el que s’anomenen *hooks* (ganxos). Aquests són unes llistes de funcions que es criden en el moment en que l’execució arriba al punt on així està definit i permeten un control addicional del procés de síntesi. En tots els casos, aquestes funcions han de complir el requisit de que l’únic paràmetre que se’ls hi passi i l’únic valor que retornin sigui una *utterance* (la que està sent sintetitzada). Així doncs, pels exemples donats, en el cas de les funcions “tts” i “tts_file” la llista “tts_hooks” defineix les funcions que s’aplicaran a les *utterances* que es creïn a partir de l’arxiu d’entrada i per defecte conté les funcions estàndards de síntesi i reproducció, “utt.synth” i “utt.play”. La funció de síntesi, per la seva banda, en té tres: el primer, “before_synth_hooks”, s’aplica abans que es cridi a qualsevol altre mòdul; el segon, “after_analyisis_hooks”, s’aplica abans de la crida a “Wave_Synth”, quan ja s’ha fet tot el processament de text, lingüístic i prosòdic; i finalment, “after_synth_hooks”, s’aplica al final de tot, quan ja han passat tots els mòduls.

Codi de la funció Scheme utt.synth, remarcant els punts on s’apliquen alguns dels ganxos explicats:

```
(define (utt.synth utt)

  "(utt.synth UTT)
    The main synthesis function.  Given UTT it will apply the
    functions specified for UTT's type, as defined with deffUttType
    and then those demanded by the voice.  After modules have been
    applied synth_hooks are applied to allow extra manipulation.
    [see Utterance types]"
```

```
  (apply_hooks before_synth_hooks utt)

  (let ((type (utt.type utt)))
    (let ((definition (assoc type UttTypes)))
      (if (null? definition)
        (error "Unknown utterance type" type)
        (let ((body (eval (cons 'lambda
                          (cons '(utt) (cdr definition))))))
          (body utt)))))

  (apply_hooks after_synth_hooks utt)
  utt)
```

Per altra banda, tornant a les funcions “tts” i “tts_file”, aquestes poden rebre com a entrada arxius de text de formats molt diferents, tals com correus electrònics, scripts per aplicacions, textos narratius, etc. els quals, per tant, s’hauran de “llegir” de maneres molt diferents. És a dir, en el cas del correu electrònic, per exemple, potser es desitjarà simplement que de la capçalera Festival en sintetitzi només el remitent, el destinatari i l’assumpte; o, en un script de llenguatge “markup”, pot interessar que les etiquetes no es sintetitzin, sinó que siguin ignorades o llegides per un parser, etc. Per tal de facilitar, doncs, la distinció entre aquests tipus de text diferents que requereixen síntesis diferents, Festival defineix els anomenats **modes de text**.

Un mode és un conjunt de funcions Scheme, definides en un script amb el nom “<nom del mode>-mode.scm”, que apliquen els canvis, càlculs i/o condicions extres necessaris per al procés del tipus de text que li correspongui. Un cop definit un mode, per dir-li a la funció “tts” que el faci servir se li passa el nom d’aquest com a segon paràmetre (per exemple, la comanda *(tts correu.txt “email”)* sintetitzaria el contingut de l’arxiu “correu.txt” en mode “email”, sempre que aquest existeixi).

Per a que un script Scheme pugui ser un mode, a més del nom de l’arxiu, ha de complir unes altres dues condicions: Per tal de que Festival el carregui quan s’hi fa referència, l’script s’ha de trobar en un directori de llibreries. Per defecte, aquest és “festival/lib”, però s’en poden definir més afegint-los a la variable global “lib-path” a l’arxiu de configuració “.festivalrc”

Exemple de comanda per afegir un directori a lib-path

```
(set! lib-path (cons "/home/awb/lib/festival/" lib-path))
```

La segona condició és que per tal de que el mode faci alguna cosa aquest ha de definir almenys un dels següents paràmetres:

- *filter*: el nom d'un executable extern (p. ex. una shell Unix) que processi l'arxiu de text de manera apropiada abans del procés d'aquest per part de l'eina TTS.
- *init_function*: funció Scheme que serà cridada abans de que comenci el procés. Permet canviar regles de tokenització, *token-to-words*, *hooks*, etc. que després s'aplicaran.
- *exit_function*: funció Scheme que es cridarà al final de tot del procés, permet tornar a donar els valors inicials a tots els paràmetres, regles etc. que s'hagin modificat en la funció anterior.
- *analysis_mode*: si aquest paràmetre val "xml" vol dir que l'arxiu d'entrada és un script d'aquest llenguatge i el fitxer passa pel parser inclòs a Festival, "rxp". Si val "xxml", vol dir que l'script d'entrada és en SGML. En aquest cas el paràmetre *filter* hauria de ser un parser normalitzador d'aquest llenguatge**,** i el seu resultat serà processat segons convingui. Qualsevol altre valor serà ignorat.

Capítol 5. Extensions de les eines usades

Definides i enteses les dues eines principals, la darrera tasca a realitzar és, finalment, extendre el codi d'aquestes per tal de definir-ne la interacció, que XFace pugui fer servir Festival com a eina de síntesi, per una banda i la nova veu catalana per a Festival per l'altra (veure Figura 5-1).

5.1. Interacció d'XFace amb Festival

En un principi, per a definir aquesta, es decideix seguir el model existent a XFace de l'idioma italià sota "keyframes", en el qual es fa una crida externa a "Flite", per al cas d'aquest treball. És a dir, el que es farà és una crida a Festival, com a aplicació externa a XFace, al qual se li passarà el text a sintetitzar en un arxiu d'entrada i que aquest generi els arxius de fonemes (.pho) i d'àudio (.wav) necessaris com a sortida.

El primer que es fa és preparar Festival i Speech Tools. Es decideix de compilar-los sota "cygwin" (emulador de Linux per Windows)**[29]** ja que resulta molt més senzill i funciona perfectament d'aquesta manera. Un cop compilats, es procedeix a treballar sobre els arxius Scheme de Festival per a que aquest doni la sortida desitjada per a XFace amb la veu castellana.

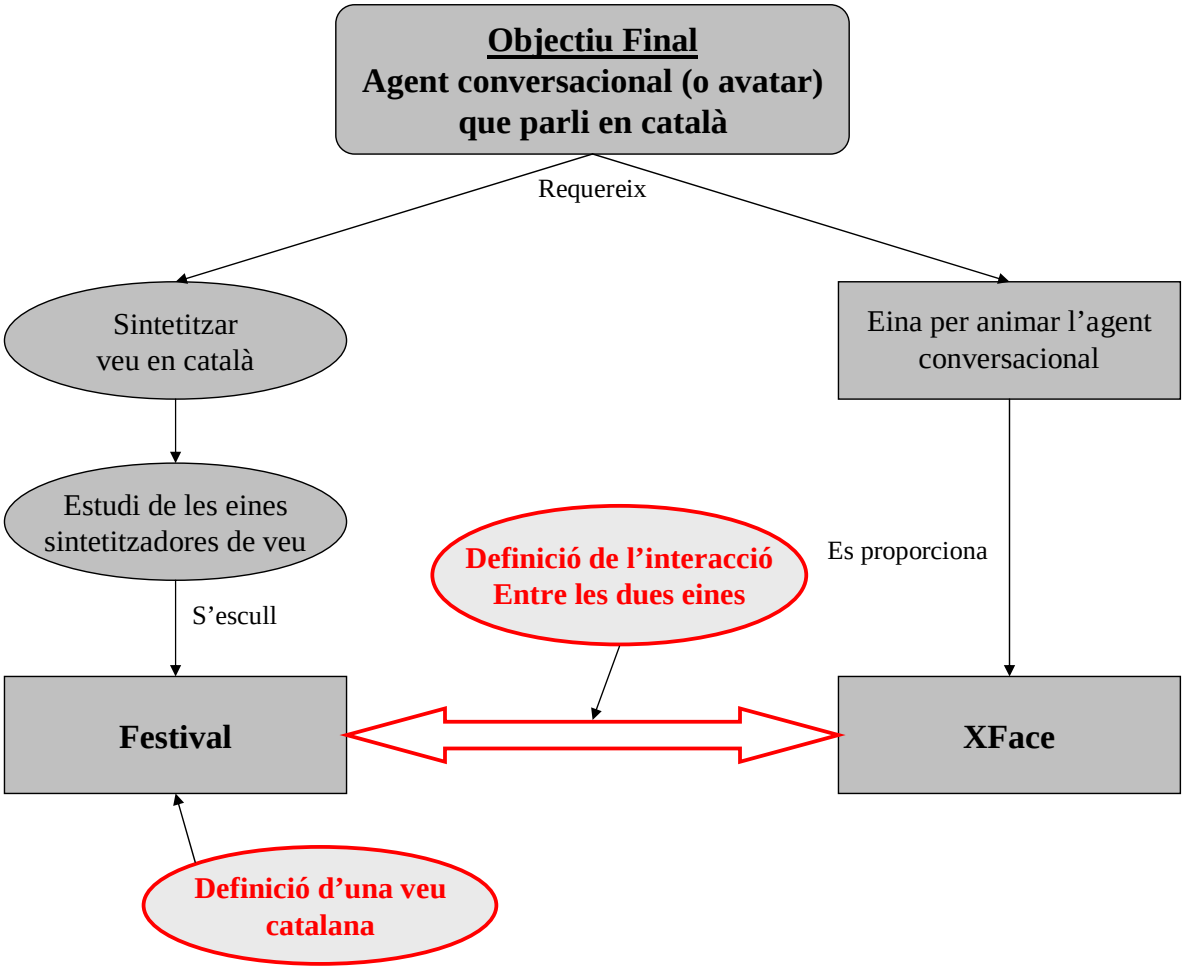

Figura 5-1: Esquema d'organització de les tasques del treball, 5a fase - Extensió de les dues eines.

Primer s'adapta l'script Scheme existent a Festival com a exemple "text2wave", el qual per una entrada donada en un arxiu de text en guarda la sortida d'àudio en un arxiu de format .wav, mitjançant la crida a la funció "tts" i modificant-ne la variable "tts_hooks" per tal de desviar l'àudio de sortida a arxius WAV diferents, que després concatena. Un cop modificat aquest script és anomenat "festival_xface_cat". Donat que a ambdues versions, per tal de simplificar la crida a Festival, les dues primeres línees són una shell de Linux que crida a l'executable principal, aquest passa a ser el programa que es crida des de la crida a sistema d'XFace.

Les dues primeres línees de "festival_xface_cat":

```
#!/bin/sh
"true" ; exec /home/Joan/festival/bin/festival --script $0 $*
```

A partir d'aquí és un script d'Scheme. Les modificacions principals realitzades estan marcades en vermell:

```
;;; Because this is a --script type file I has to explicitly
```

```
;;; load the initfiles: init.scm and user's .festivalrc
(load (path-append libdir "init.scm"))
;;; JSTP INICI
;;; - Llenguatge a catala
(language_western_catalan)
;;; - Carregar mode XFace - no cal, ja es carrega sol.
;;;(load "xface-mode.scm")
;;; JSTP FI

(...) ;; resta del codi, no modificat

;;;
;;; Redefine what happens to utterances during text to speech
;;;
(set! tts_hooks (list utt.synth save_record_wave))

(define (main)
  (get_options)

  ;; do the synthesis
  ;;; JSTP INICI
  ;;; - Crida amb mode xface
  (mapcar
   (lambda (f) (tts_file f "xface"))
   text_files)
  ;;; JSTP FI

  ;; Now put the waveforms together at again
  (combine_waves)
)

;;;  Do the work
(main)
```

Després es crea el mencionat “xface-mode.scm” a partir del també proporcionat com exemple “th-mode.scm” (on “th” vé de “Talking Head”), el qual adapta Festival per una aplicació de cap parlant genèrica. Aquest mode “xface” afegeix dues funcions dins la síntesi de veu: ignora les etiquetes XML (és a dir tot el text que estigui entre ‘<’ i ‘>’) i crea l’arxiu de fonemes amb el format que XFace requereix, cada fonema amb la seva durada acumulada. Finalment, per tal de que Festival reconegui aquest mode, es mou aquest arxiu al directori “festival/lib”, amb la resta d’scripts Scheme principals.

Arxiu “xface-mode.scm”:

```
(defvar phofile "myfile.pho")

(defvar accum_duration 0)

(define (utt.save.xface_phonedata utt filename)
"(utt.save.mydata UTT FILE)
  Saves phone and accumulated duration for XFace."
  (let ((fd (fopen filename "a")))
    (mapcar
     (lambda (seg)
```

```
       (set! accum_duration (+ accum_duration (item.feat seg
"segment_duration")))
       (format fd "%s %2.4f"
             (item.feat seg "name")
             accum_duration)
       (format fd "\n"))
     (utt.relation.items utt 'Segment))
    (fclose fd)
    utt))

(define (xface_output_info utt)
  "(xface_output_info utt)
This is called after linguistic analysis but before waveform
synthesis. It collects the phone and duration data into the file
defined in the variable phofile."
  (utt.save.xface_phonedata utt phofile)
  utt)

;;;
;;;  Define a new text mode for talking heads
;;;

(define (xface_init_func)
 "Called on starting xface mode."
 ;;; To start always a new file from 0.
 (let ((fd (fopen phofile "w")))
 (fclose fd))
 ;;; Function substitution.
 (set! xface_previous_t2w_func token_to_words)
 (set! xface_previous_after_analysis_hooks after_analysis_hooks)
 (set! after_analysis_hooks (list xface_output_info))
 (set! english_token_to_words xface_token_to_words)
 (set! token_to_words xface_token_to_words)
)

(define (xface_exit_func)
 "Called on exit xface text mode."
 (set! token_to_words xface_previous_t2w_func)
 (set! english_token_to_words xface_previous_t2w_func)
 (set! after_analysis_hooks xface_previous_after_analysis_hooks)

 ;(audio_mode 'sync) ;; so we can reset the audio
 ;(set! Parameter th_previous_Parameter)
)

(defvar SMIL nil)

(define (xface_token_to_words token name)
"(xface_token_to_words TOKEN NAME)
SMIL-Agent Markup Language specific token to word rules."
 (cond
  ;;  Symbols marked with <> are treated as commands
  ;;  and not rendered as speech
  ((string-matches name "<.+>")
   nil)
  ((string-matches name "<.+")
   (set! SMIL t) nil)
  ((string-matches name ".+>")
   (set! SMIL nil) nil)
  (SMIL nil)
  ;;  Else, they are rendered
```

```
  (t
   (xface_previous_t2w_func token name))))

(set! tts_text_modes
   (cons
    (list
      'xface        ;; mode name
      (list         ;; ogimarkup mode params
       (list 'init_func xface_init_func)
       (list 'exit_func xface_exit_func)))
    tts_text_modes))

(provide 'xface-mode)
```

El següent pas és la creació de codi d'XFace per al cas del català:

- S'afegeix l'idioma català i el processador "Festival" per a aquest a l'arxiu "scriptprocs.xml", especificant la opció de que el text a sintetitzar s'ha de guardar en un fitxer temporal, el qual serà l'entrada per a "festival_xface_cat". Més endavant això dóna el problema de que si el text té etiquetes enmig, aquest es guarda sense aquestes i sense espai (per exemple: el text "un <mark=1> exemple" es guarda com "unexemple"); el problema es localitza i es corregeix.
- Es crea el diccionari de fonema-visema pel castellà/català ("lang\catalan.dic", mostrat a l'apartat 2.2.1), usant el catàleg de visemes donat per XFace mateix, ja que resulta suficient, i es modifica la classe "Xface::FaceBase" per a que el llegeixi.
- I la modificació principal, s'afegeix la funció "speakCatalan" a la classe "XSmilAgent::SMILManager", feta a partir de la "speakItalian", la qual s'encarrega de generar la crida a Festival en sí.

Codi de la funció "speakCatalan":

```
//PROJ - INICI - Afegir funcio de parla en catala - 200603112254
void SMILManager::speakCatalan(const SpeechObj& obj)
{
      // create the text to be synthesized
      std::string script;
      std::vector<SpeechContentObj>::const_iterator it =
obj.content.begin();
      while (it != obj.content.end())
      {
            if (it->type == SpeechContentObj::kText)
                  std::copy(it->value.begin(), it->value.end(),
std::back_inserter(script));
            //PROJ - INICI - Corregir problema de falta d'espais -
200604201913
            else if (it->type == SpeechContentObj::kMark)
                  script.append(" ");
```

```
            //PROJ - FI - Corregir problema de falta d'espais -
200604201913
            ++it;
      }

      // find a suitable catalan tts engine
      std::vector<std::string>::const_iterator itTTS =
m_TTSProcessors.begin();
      boost::shared_ptr<IScriptProcessor> pProcCatalan;
      while(itTTS != m_TTSProcessors.end())
      {
            boost::shared_ptr<IScriptProcessor> pProc =
m_scriptProcLoader.getScriptProcessor(*itTTS);
            if(pProc->getLanguage() == "ca")
                  pProcCatalan = pProc;
            if(pProc->getExecutable() == "bash")     // force use of
festival under cygwin (requires cygwin1.dll)
            {
                  pProcCatalan = pProc;
                  //"-c
\"C:/cygwin/home/Joan/festival/JSTP/festival_xface_cat test.smil -o
test.wav -p test.pho\" "
                  //PROJ - INICI - Deshardcodejat del path del festival
- 200604142017
                  std::string params = "-c \"";
                  params.append(pProcCatalan->getParameters1());
//C:/cygwin/home/Joan/festival/JSTP/festival_xface_cat

                  pProcCatalan->setParameters1(params);

                  params="-o ";
                  params.append(obj.performance.id);
                  params.append(".wav -p ");
                  params.append(obj.performance.id);
                  params.append(".pho\" ");

                  pProcCatalan->setParameters2(params);
                  //PROJ - FI - Deshardcodejat del path del festival -
200604142017
                  break;
            }
            ++itTTS;
      }
      if(pProcCatalan)
      {
            pProcCatalan->process(script, obj.performance.id);
      }
}
//PROJ - FI - Afegir funcio de parla en catala - 200603112254
```

Un cop definida la interacció entre les dues eines, s'observa que a la part d'XFace hi ha molts paràmetres codificats explícitament, com per exemple la crida mateixa al Festival (que a més resulta molt complexa), el "path" absolut de l'eina TTS, així com una crida a una funció dins de "XFaceEd::KeyframePanel::OnPlaySMIL" que passa com a idioma la constant "en_SAPI", la qual indica anglès sóta SAPI (*Speech*

API) de Microsoft, que inclou funcionalitat TTS. Es fan doncs els canvis necessaris i, per provar de simplificar la crida a Festival, es prova a compilar aquest sota Windows. D'aquesta manera l'execució passaria de ser:

XFace → Crida a sistema (MSDOS) → executa "bash" sota cygwin → a aquesta shell se li passa la comanda de crida a Festival. (Veure Figura 5-2)

A ser:

XFace → Crida a sistema (MSDOS) → execució de Festival. (Veure Figura 5-3).

Fent això, però, apareix un problema: compilar Festival sota Windows no només és complex, a més fa aparèixer un path absolut codificat explícitament que provoca que si es mou Festival de lloc, aquest s'ha de tornar a compilar de 0. Amb l'altre mètode, en canvi, s'han aconseguit eliminar totes les codificacions explícites simplement modificant lleugerament l'ús de la estructura de l'interfície "IScriptProcessor". És per això que, si bé més complex, es decideix mantenir aquest sistema, ja que estiguin on estiguin situats XFace, cygwin i Festival, el sistema complet segueix funcionant, havent d'especificar només el nou "path" de Festival al camp "parameters1" i/o el de cygwin al camp "path" a l'arxiu "scriptprocs.xml", així com la ubicació de festival a "festival_xface_cat", en cas de que es mogui algun d'aquests dos.

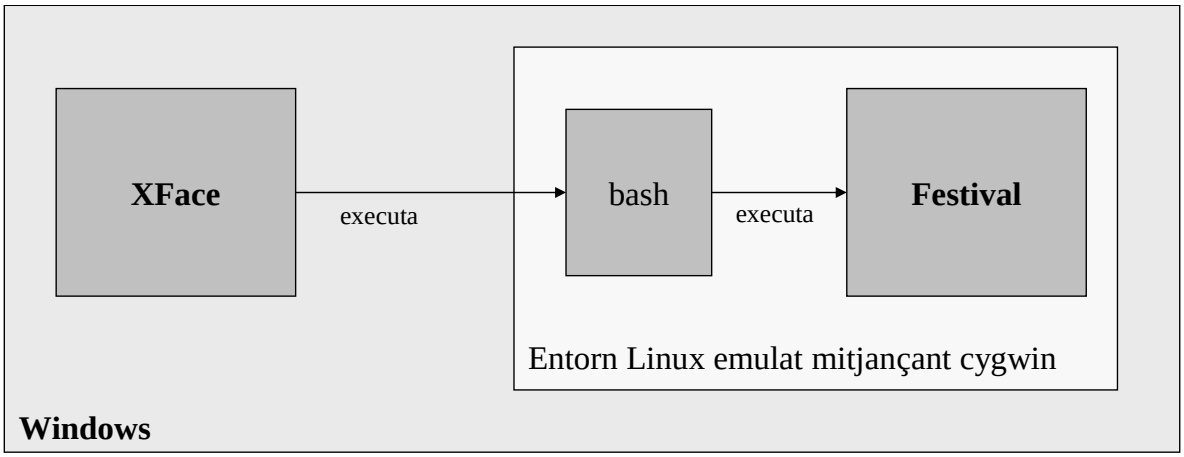

Figura 5-2: Esquema d'execució de Festival des d'XFace segons la situació actual.

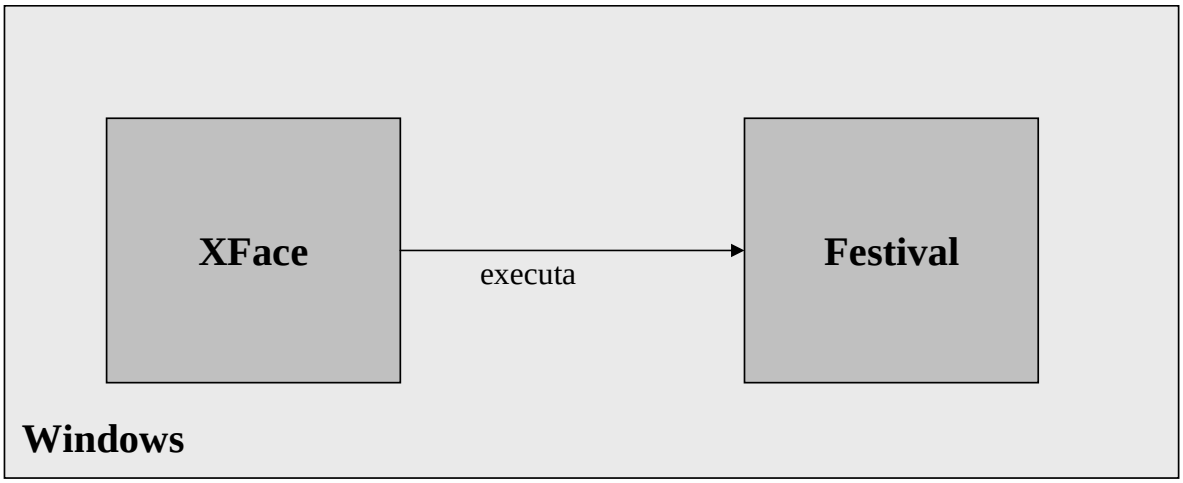

Figura 5-3: Esquema d'execució de Festival des d'XFace, en cas de que es compilés el 1r sobre Windows.

5.2. Creació de veu en català

Pel que fa a aquesta segona tasca, aquesta planteja més problemes. Al principi es consulta a la llista de correu de Festvox, destinada específicament a la definició de noves veus, respecte la possibilitat de crear una nova veu des de zero. De les respostes rebudes es dedueix que la part més crítica per tal poder assolir aquest objectiu és disposar de la base de dades del conjunt de difonemes del català. Aquesta, doncs, es busca, donat que no es disposa dels mitjans necessaris per a crear-la: de nou a través de la llista de correu del projecte Festvox es contacta amb el TALP (Centre de Tecnologies i Aplicacions del Llenguatge y la Parla) de la UPC però, degut a que aquesta forma part

d'una càtedra de la Universitat, no es pot aconseguir. Més endavant es consulta la qüestió amb Carles Fernández Tena, membre del CVC que va fer una eina TTS per a Philips com a Projecte Fí de Carrera a la UPC**[30]**, el qual també ressalta la importància de la base de dades en la definició d'una veu i, si bé ell no pot cedir la seva, ja que és privat de l'empresa, recomana l'ús d'MBROLA, degut a la gran quantitat de bases de dades que existeixen per aquesta eina.

Vist, doncs, que no resulta viable crear una veu nova, es procedeix a estudiar les dues alternatives restants: Adaptar una o més veus d'MBROLA i incloure aquesta eina en el sistema o adaptar, per similaritat, la veu ja existent per Festival en castellà al català. Examinant la primera possibilitat es troba que les veus castellanes que hi ha per aquesta resulten tenir una base de dades igual o menor que la de la veu pròpia de Festival, i que definir la veu catalana a partir d'un altre idioma (l'italià, per exemple, conté una bona part dels fonemes catalans) resulta massa complex. Això, doncs, juntament amb el fet de que afegir MBROLA encara complicaria més el funcionament global del sistema i recordant que la intenció inicial és no usar aquesta eina per no ser software completament lliure, fa que al final s'opti per la darrera opció, adaptar la veu castellana de baixa qualitat existent a Festival al català de manera provisional per, en un futur proper, definir-ne una correctament mitjançant el procés de Festvox.

A Festival les veus es guarden al directori "festival/lib/voices", cadascuna dins d'un altre directori corresponent al seu idioma. L'elaboració de la nova veu comença, doncs, amb una còpia de la veu castellana, anomenada "el_diphone", en una nova carpeta "catalan" i amb el nom "el_diphone_cat". Tant la llengua catalana com la veu específica "el_diphone_cat" s'afegeixen a l'script mitjançant el qual Festival carrega els idiomes i les veus disponibles, "languages.scm", amb la definició de la funció "language_western_catalan" i l'incorporació del català a la funció "select_language". Amb això ja es disposa d'una veu que es pot provar directament sobre Festival després de cada modificació.

Funcions "language_western_catalan" i "select_language" (amb la part afegida en vermell):

```
;; PROJ - INICI - Afegir el catala - 200604201827
(define (language_western_catalan)
"(language_western_catalan)
Set up language parameters for Western Catalan."

  (voice_el_diphone_cat)
  (set! male1 voice_el_diphone_cat)
```

```
  (Parameter.set 'Language 'catalan)
)
;; PROJ - FI - Afegir el catala - 200604201827


;; Funció select_language, usada per a canviar l'idioma de Festival i
;; la veu per defecte associada a cadascun d'aquests
(define (select_language language)
  (cond
   ((or (equal? language 'britishenglish)
      (equal? language 'english))  ;; we all know its the *real*
English
    (language_british_english))
   ((equal? language 'americanenglish)
    (language_american_english))
   ((equal? language 'scotsgaelic)
    (language_scots_gaelic))
   ((equal? language 'welsh)
    (language_welsh))
   ((equal? language 'spanish)
    (language_castillian_spanish))
   ;; PROJ - INICI - Afegir el catala - 200604201827
   ((equal? language 'catalan)
    (language_western_catalan))
   ;; PROJ - FI - Afegir el catala - 200604201827
   ((equal? language 'klingon)
    (language_klingon))
   (t
    (print "Unsupported language, using English")
    (language_british_english))))
```

A continuació es modifiquen els noms de totes les funcions i variables en cadascun dels scripts de la veu per distingir-les clarament de les seves equivalents castellanes. Després es procedeix a modificar les normes per a que la sintetització aproximi el català el màxim possible:

1- Es tradueix, a l'arxiu "el_diphone_cat.scm" la llista de paraules per a la variable "guess_pos", utilitzada per la característica "gpos" en el càlcul de l'accentuació per distingir paraules "d'enllaç" (articles, preposicions...) de paraules de "contingut".

2- Es tradueixen també les regles *token to words* (definides només pels nombres), definides a "catoken.scm".

3- Es tradueix el poc lèxic donat amb la veu "el_diphone", els noms de les lletres i els símbols, a "catlex.scm".

4- I la modificació principal, es canvien les regles *letter to sound*, donades en el mateix arxiu que l'anterior, d'acord amb les normes gramaticals del català.

5.3. Inclusió de les normes gramaticals del català

Per al darrer pas de la definició de la veu catalana, a l'hora de decidir quines regles s'eliminen, quines es canvien i quines s'afegeixen, en primer lloc s'apliquen les modificacions que ja s'han decidit prèviament (tals com les diferències de pronunciació, la introducció de la ç i dels dígrafs, etc.). Després se'n van fent més a mesura que es van trobant diferències entre la gramàtica i la fonètica catalanes**[31][32]** i les normes de la veu castellana, així com per solventar casos en que Festival no sintetitza correctament el text a l'hora de fer les proves. D'aquesta manera, al final les principals modificacions introduïdes són les següents:

- Primer una norma específica per a que s'ignorin els apòstrofs entre 'd', 'l', 'm', 's' o 't' i una vocal i es pronuncïi tot com una sola paraula. Això es fa degut a que en aquesta veu s'utilitza l'apòstrof davant d'una vocal per a simular l'accent (és a dir es pot escriure "camió" o "cami'o", "L'àliga" o "L''aliga"), per casos com és precisament el del parser XML d'XFace, que no permet l'ús de vocals accentuades, així com per teclats que no el tenen. Sense aquesta norma la vocal que seguís a l'apòstrof estaria sempre (molts cops incorrectament) accentuada.
- En general, s'adapten les pronunciacions al català, com per exemple: 'c' davant de 'e' o 'i' es pronuncia "s" (consonant fricativa alveolar sorda, com a **c**ercle, in**c**ívic...), 'g' davant de 'e' o 'i' es pronuncia "j" (fricativa prepalatal sonora, com a pa**g**ès, **g**írgola...), etc. Aquesta substitució es fa suplint la falta de fonemes aproximant de la millor manera possible: "j" i "sh" (fricativa prepalatal sorda, com a **x**oc, mar**x**a...) es substitueixen per "ch" (prepalatal africada sorda, com a fi**tx**a, **Tx**èquia... donat que en el valencià de vegades es fà aquesta substitució, sobretot a principi de mot) i es suprimeixen l'essa sonora (fent servir sempre l'essa sorda), les vocals obertes i la neutra (com al català occidental).
- S'afegeixen regles pels dígrafs:
    - Si hi ha vocal + 'ig' a final de paraula, 'ig' es pronuncia "ch" (pu**ig**, ve**ig**, go**ig**...). Així mateix, si bé no és un dígraf, és una variant d'aquest , si hi ha 'i' + 'g' al final de paraula, la 'g' sona també "ch" (mi**g**, enmi**g**...)
    - Pels casos en que es trobi el·la geminada, aquesta es converteix en una 'l' simple (lateral alveolar sonora, com a sí**l·l**aba, **l**esió...), ja que

fonèticament no hi ha diferència entre aquestes dues. Es consideren, a més, tant el cas que aquest dígraf vingui escrit amb el caracter '·' com amb el caracter '.', de nou pels casos en que el 1r no sigui un caràcter acceptat o pugui no sigui present en un teclat.

  - El dígraf 'ny' sona 'ñ' (nassal alveolar sonora, com a ca**ny**a, bo**ny**...).
  - Els dígrafs 'tg' i 'tj' (me**tg**e, pla**tj**a...) sonen com "t" i "ch" de nou a falta de sò "tj" (prepalatal africada sonora) i 'tx' com a "ch" (co**tx**e, me**tx**a...).
  - El dígraf 'ix', després de vocal, sona com "i" i "ch" (el sò "i" es fa feble més endavant en el càlcul de la síl·laba tònica), de nou a falta de sò "sh" (ba**ix**, fe**ix**, cré**ix**er...).

- S'afegeixen certes normes pel cas de la lletra 'x', que en català segons com sona "sh" (o, en el cas d'aquesta veu "ch") i segons com "ks" (sò oclusiu velar sord seguit del sò fricatiu alveolar sord). Concretament, està fet de manera que soni "sh" si és a principi de paraula (**x**ocolata, **x**urro...) o després de consonant (mar**x**ar, pan**x**a...) i per les excepcions "pi**x**" i "gui**x**" i derivades. En els altres casos sona "ks" (bàsicament després de vocal: fi**x**ar, ta**x**a, fle**x**ible...).

La següent taula 5-1 resumeix aquests canvis aplicats, mostrant-ne les regles en sí i exemples de síntesi, pels quals a l'esquerra es mostra la paraula tal com pot estar escrita al text d'entrada i a la dreta amb les lletres que s'han processat en negreta, fins arribar a la regla en qüestió, donat que les *letter to sound* s'apliquen per ordre des del primer caràcter fins l'últim.

Regles	**Funció**	**Exemples**
`(# DLMST [ "'" ] V = )` `(# DLMST [ "'" ] "'" V = )`	Eliminen l'apòstrof davant de vocal per tal de que aquest no es confongui amb un accent	L'àliga → **L**àliga L''aliga → **L'**aliga S'enyora → **S**enyora
`( [ c ] "'" EI = s )` `( [ c ] EI = s )`	Transformen 'c' en el sò "s" davant de 'e' o 'i'.	cercle → **s**ercle inc'ivic → **ins**'ivic
`( [ g ] "'" EI = ch )` `( [ g ] EI = ch )`	Transformen 'g' en el sò "ch" (a falta de sò "j") davant de 'e' o 'i'.	pagès → **pach**ès g'irgola → **ch**'irgola
`( V [ i g ] # = ch)`	Transforma el dígraf 'ig' en "ch". Aquest dígraf només apareix a final de	puig → **puch** veig → **bech**

	paraula i precedit de vocal.	goig → **goch**
`( i [ g ] # = ch)`	Quan hi ha 'ig' a final de paraula però precedit de consonant, és només la 'g' la que es transforma.	mig → **mich** enmig → **enmich**
`( [ l "\." l ] = l )` `( [ l · l ] = l )`	Transformen les el·les geminades en el·les simples, donat que sonen igual.	síl·laba → **sil**aba il·l'es → **il**'es
`( [ n y ] = ny )`	Transforma el dígraf escrit 'ny' en el sò corresponent, representat pels mateixos caracters.	canya → **kany**a bony → **bony**
`( [ t g ] = t ch )` `( [ t j ] = t ch )`	Transformen els dígrafs 'tg' i 'tj' en els sons "t ch", a falta del fonema prepalatal africat sonor.	metge → **metch**e platja → **platch**a
`( [ t x ] = ch )`	Transforma el dígraf 'tx' al seu sò corresponent, "ch".	cotxe → **koch**e metxa → **mech**a
`( V [ i x ] = i ch )`	Transforma el dígraf "ix", el qual apareix després de vocal, en els sons "i ch", a falta del fonema "sh".	baix → **baich** feix → **feich** créixer → **kreich**er

`( p i [ x ] = ch )`
`( # [ x ] = ch )`
`( C [ x ] = ch )`
`( [ x ] = k s )`
	Tracten la lletra 'x' segons pertoca, fent que soni "ch" (de nou a falta del sò "sh") si la paraula és "pix" o derivada, si comença per 'x' o si va darrera de consonant. En els altres casos sona "k s".	xocolata → **ch**ocolata
		xurro → **ch**urro
		marxar → **march**ar
		panxa → **pancha**
		pix → **pich**
		fixar → **fiks**ar

Taula 5-1: Taula-resum de les regles *letter to sound* explicades anteriorment.

Encara dins de les normes *letter to sound*, es modifiquen també les normes per a la sil·labificació (separació de síl·labes), les quals es passen després d'haver passat les anteriors. Bàsicament es mantenen iguals i el que es toca són ser les normes de diftongs i triftongs:

- S'eliminen totes les normes relatives a triftongs, ja que en català aquests són poc habituals, difícils de tractar amb normes d'aquest tipus, molt generals, i a més es comprova que l'error a l'ona resultant és gairebé inapreciable.
- Pel que fa a diftongs es defineixen les normes (per vocals normals i per les accentuades) segons les definicions da'quests en català donades per l'Institut d'Estudis Catalans**[32]**:

Tipus de diftong	Estructura	Exemples
Creixents	‘i’ (o ‘hi’) a començament de paraula + vocal	**ia**ia, **io**de, **hie**na...
	Vocal + ‘i’ + vocal	jo**ie**r, cre**ia**...
	Consonant velar (‘q’ o ‘g’) + ‘u’ + vocal	aig**ua**, q**üe**stió, lleng**üe**s...
	vocal + ‘u’ + vocal	ca**ue**n, pe**uo**t...
Decreixents	vocal + vocal dèbil (‘i’, ‘u’)	**ai**gua, alm**oi**na, b**ui**t, cr**eu**, c**iu**tat, c**ou**re...

Taula 5-2: Diftongs de la llengua catalana.

I en tercer lloc, es modifiquen les regles de tonificació (càlcul de la síl·laba tònica), que bàsicament substitueixen la vocal principal (‘a’,’e’,’i’...) per la vocal tònica (‘a1’, ’e1’, ‘i1’...) a les paraules que no tenen accent.

- En primer lloc es donen un parell de normes per fer mudes la ‘r’ al final de paraula i la ‘t’ quan la paraula acaba en ‘-nt’. Això es fa aquí enlloc de a les normes inicials perquè es necessita la presència de la ‘r’ i la ‘t’ per calcular la síl·laba tònica. Per exemple, si per la paraula “caminar” a aquesta funció li arriba “kamina”, la paraula contarà com a plana. En canvi, si es fa arribar “kaminar”, aquesta consta com aguda i només cal, finalment, eliminar la ‘r’ final.
- També és aquí on sempre que ja es determini que una ‘o’ és àtona, aquesta es transforma en ‘u’.
- S’afegeixen les normes per a que comptin com agudes les paraules acabades en ‘a’, ‘e’, ‘i’, ‘o’ + ‘i’, ‘u’ (amb i sense ‘s’, pel plural), com és el cas d’”estiu”, “dempeus”, “gripau”, etc. com a excepció a la regla següent.
- S’afegeixen les normes per a que siguin planes totes les paraules acabades en vocal, vocal + ‘s’, ‘-en’ o ‘-in’
- La resta de paraules, que no estan accentuades, ni són planes, doncs es consideren agudes.
- En un darrer procés (no modificat) totes les vocals ‘i’ i ‘u’ que estan en una mateixa síl·laba amb una vocal forta “s’afebleixen”: es substitueixen per les versions ‘i0’ i ‘u0’ dels fonemes corresponents que només es diferencien amb l’original en que duren una mica menys. Aquests fonemes es fan servir en substitució de les consonants graduals “y” (aproximant palatal sonor, com a

no**i**a, a**i**re...) i “w” (aproximant labiovelar sonor, com a **u**adi, ca**u**en), no disponibles a la base de dades.

I amb això es dóna per suficientment aproximada al català aquesta veu castellana. Si bé queda molta feina per fer de definir vocabularis, més regles i més precises, etc. aquesta es considera que seria una tasca de programació, ja no de recerca, més senzilla i que s’hauria de fer de manera conjunta amb filòlegs o experts de la llengua catalana.

Per acabar, es prova que la nova veu funciona bé des de Festival mateix i finalment es prova tot el conjunt des d’XFace. Ambdues proves surten bé, de manera que es dóna per conclosa aquesta darrera tasca.

Capítol 6. Funcionament global

Un cop completades totes les tasques, en aquest darrer apartat s'explica el funcionament global i en detall de les dues aplicacions conjuntes mitjançant un exemple de prova final. Concretament l'avatar pronunciarà el següent text:

Test final d'incorporació del català en un agent conversacional animat per computador.
Sembla que funciona.

Com ja s'ha comentat en els apartats anteriors, en el cas d'aquest projecte, per a "fer parlar" la cara de l'XFace es fa servir el mètode d'animació per keyframes i, més concretament, la pestanya "Keyframes" de l'aplicació XFaceEd, on, a més de poder definir aquests, també es poden provar fent servir el lector d'scripts SMIL-Agent que hi ha a baix a l'esquerra. Més concretament, dins d'aquest treball s'ha treballat sobre tot el que és la sel·lecció de l'idioma i de l'eina sintetitzadora i la interacció amb aquesta, que afecta només al procés de preparacions prèvies a la reproducció.

Per aquest motiu, en aquest apartat s'explicaran les classes i funcions d'XFace i els scripts de Festival que intervenen des del moment en que es prem el botó de "Play" fins que comença la reproducció en sí.

Així doncs, primer s'introdueix el text a sintetitzar en un script SMIL-Agent senzill:

```
<?xml version="1.0" encoding="utf-8" ?>
<par system-language="catalan">
  <speech channel="alice-voice" affect="sorry-for"
       type="inform" id="test-final">
      Test final d'incorporaci'o del catal'a en un
      agent conversacional animat per computador.
      Sembla que funciona.
    </speech>
</par>
```

Aquest, al seu temps es carrega a l'editor de l'XFaceEd, juntament amb l'avatar, "alice", i s'inicia la reproducció (Veure Figura 6-1).

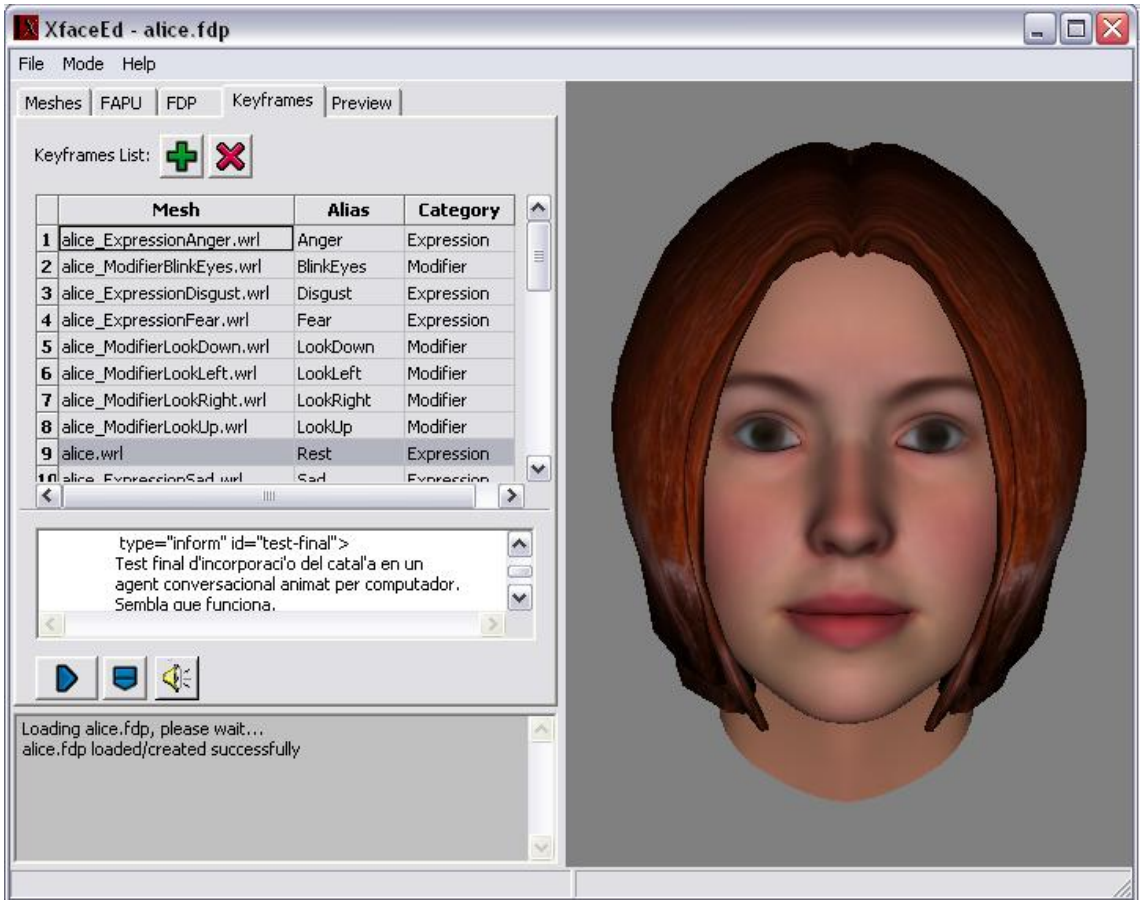

Figura 6-1: Pestanya de Keyframes d'XFaceEd amb l'avatar i l'script carregats.

A partir d'aquí el procés que segueixen les dades es pot resumir com: XFaceEd fa arribar el text d'entrada a XSMILManager, el qual fa la crida a Festival per a que el sintetitzi. Festival, llavors, sintetitza el text i en retorna els arxius d'àudio i fonemes. Finalment, XSmilAgent crea el tercer arxiu, el d'animacions, i retorna el control al reproductor d'XFaceEd, el qual carrega totes aquestes dades a la llibreria central per a que faci els càlculs d'interpolacions de keyframes i sincronització d'imatge amb àudio necessaris. Aquest procés s'explica en els següents subapartats dividit en tres fases: la primera comprèn l'execució d'XFace des de que s'inicia el procés fins que es fa la crida a Festival, la segona és la síntesi del text en aquest i la tercera la darrera part del procés de preparació, de nou sota XFace (veure Figura 6-2).

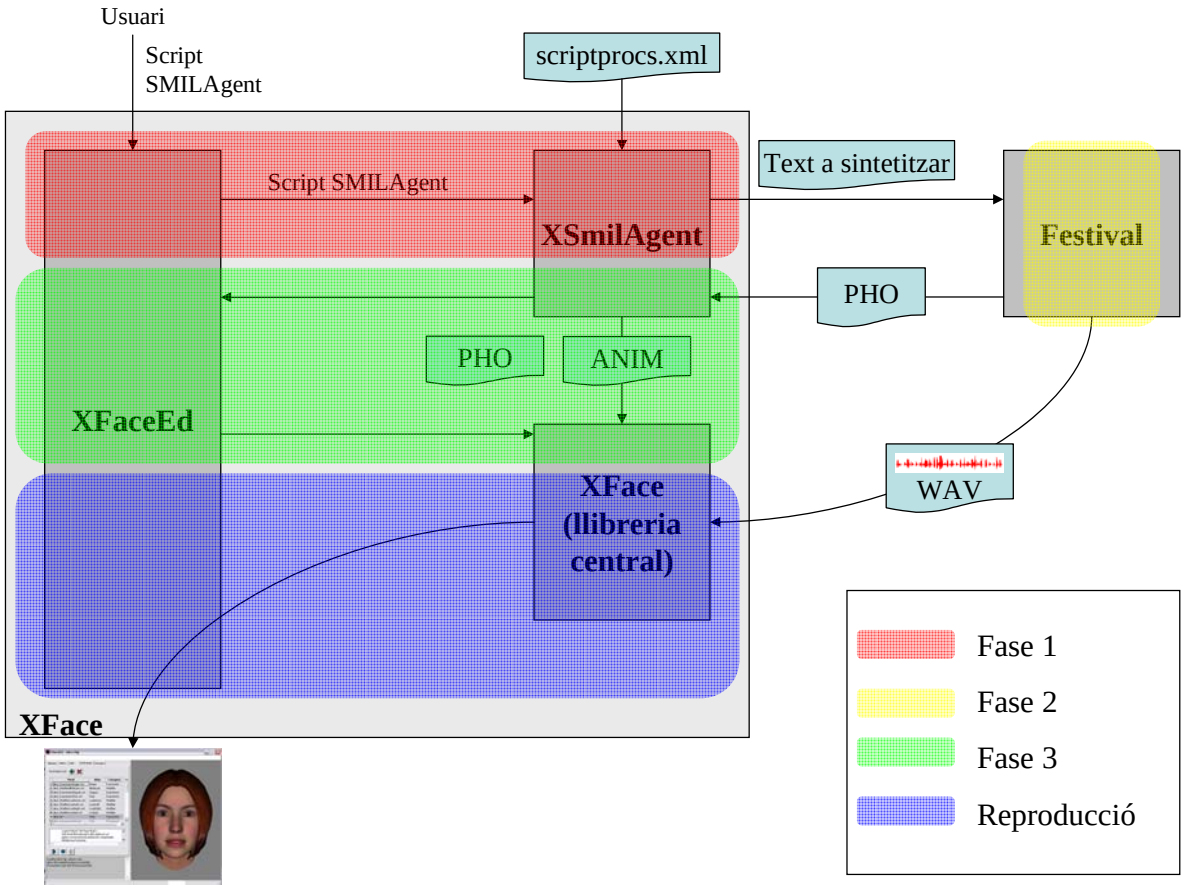

Figura 6-2: Esquema global del procés de reproducció, dividit en fases.

6.1. Primera fase: Generació del text a sintetitzar.

En aquest subapartat s'explica en detall el paper de les principals classes d'XFaceEd i XSmilAgent que intervenen en aquesta primera fase, tal com es mostren a l'esquema de la Figura 6-3.

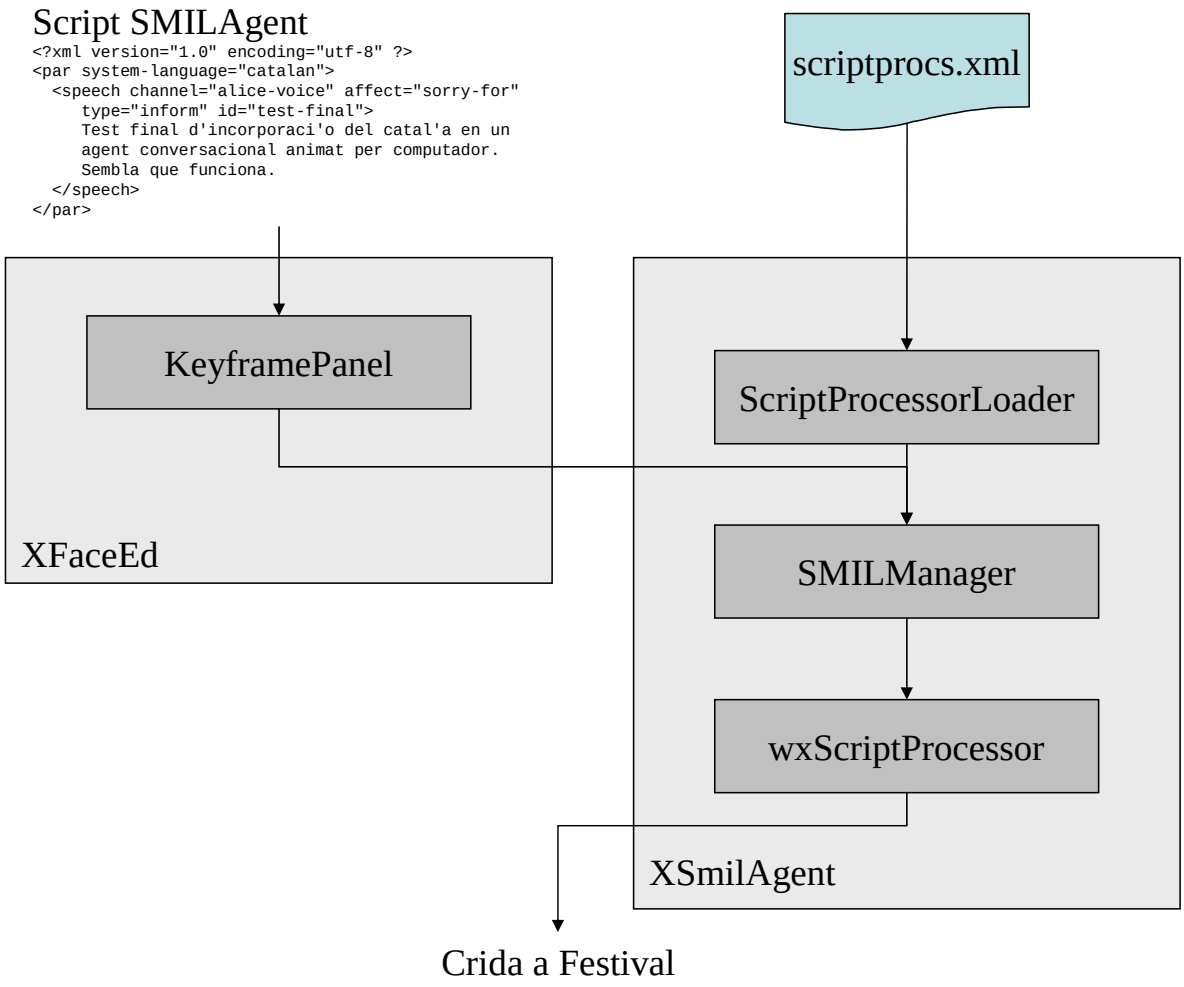

Figura 6-3: Esquema del flux de dades a la primera fase, amb les classes principals que hi intervenen.

6.1.1. Classe XFaceEd::KeyFramePanel

Des del moment en que es prem el botó de reproducció la funció que s'executa fins que s'inicia aquesta, és la funció membre d'aquesta classe "OnPlaySMIL" la que controla tot el procés. En aquesta primera fase, l'unica cosa que fa és crear l'objecte "manager", de la classe "XSmilAgent::SMILManager" i passar-li l'script d'entrada per a que el carregui i el processi.

6.1.2. Classe XSmilAgent::SMILManager

Com el seu nom indica, aquesta classe s'encarrega de controlar tot el procés d'interpretació dels scripts SMIL-Agent. En iniciar-se, el constructor carrega la primera de les dues entrades requerides d'aquesta llibreria, l'arxiu "scriptprocs.xml", i guarda en memòria els diferents processadors existents.

Després, un cop se li ha passat l'script d'entrada, s'encarrega d'interpretar-lo mitjançant la funció "parse", la qual llegeix l'script XML i el processa segons correspon. En el cas de l'exemple, quan interpreta la primera etiqueta <par> (seqüències en paral·lel, si bé en aquest cas només n'hi ha una de parla, no s'especifiquen emocions) guarda el llenguatge, "catalan" (català), ja que es passa com a atribut d'aquesta. Després, quan arriba a l'etiqueta <speech>, la funció "process_speech" s'encarrega de cridar a la funció de parla corresponent a l'idioma corresponent, en aquest cas, "speakCatalan". Aquesta funció de parla, tant en el cas de la llengua catalana com en tots els casos, té tres funcions:

1- Preparar l'script d'entrada per a que sigui compatible amb l'eina que la sintetitzarà. En aquest cas, l'únic que es fa és eliminar les tags <mark> que hi hagi, substituint-les per un espai. Com que al text d'exemple no n'hi ha, aquest queda igual:

"Test final d'incorporaci'o del catal'a en un agent conversacional animat per computador.\nSembla que funciona."

2- Llegir l'eina TTS que s'hagi especificat que sintetitzarà el text i preparar la crida a aquesta mitjançant un objecte de l'interfície "XSmilAgent::IScriptProcessor". En aquest cas només hi ha una eina (Festival) i la crida es defineix especificant-li a IScriptProcessor els paràmetres concrets amb els que haurà de cridar l'executable.

3- Cridar a la funció "process" d'IScriptProcessor per a que executi la síntesi.

6.1.3. Classe XSmilAgent::SMILProcessorLoader

Classe que carrega els diferents processadors (eines) disponibles pels diferents idiomes, o, el que és el mateix, llegeix el contingut de l'arxiu "scriptprocs.xml".

6.1.4. Classe XSmilAgent::wxScriptProcessor

Implementació de la interfície "IScriptProcessor", és la que es fa servir sempre que l'eina a cridar sigui externa i per tant requereix una crida a sistema.

En aquesta classe, la funció principal "process" prepara la que serà la crida definitiva a l'executable extern i escriu l'script d'entrada en un fitxer temporal XML, que tindrà de nom l'identificador de l'script SMIL-Agent que s'està processant (en el

cas de l'exemple "test-final"). Finalment fa la crida a sistema, que en aquest cas és la següent:

C:\cygwin\bin\bash -c "C:/cygwin/home/Joan/festival/JSTP/festival_xface_cat test-final.xml -o test-final.wav –p test-final.pho"

On l'arxiu d'entrada "test-final.xml" conté el següent:

Arxiu test-final.xml:

```
Test final d'incorporaci'o del catal'a en un agent conversacional
animat per computador. Sembla que funciona.
```

Cal remarcar que a la crida a sistema no s'especifiquen "paths" pels arxius d'entrada ni pels de sortida. D'aquesta manera tots aquests són llegits o creats per Festival a l'ubicació des d'on s'estigui executant XFace, de manera que aquest després ja disposarà dels arxius d'àudio i fonemes a la seva pròpia carpeta per treballar-hi.

A partir d'aquest punt, doncs, comença l'execució de Festival i, per tant, la segona fase.

6.2. Segona fase: Síntesi de la parla.

En aquest apartat s'expliquen els scripts de Festival que s'utilitzen des del moment en que es fa la crida a "festival_xface_cat" (veure Figura 6-4); sempre a nivell d'Scheme i funcional, sense entrar mai al codi C++/C pròpiament dit.

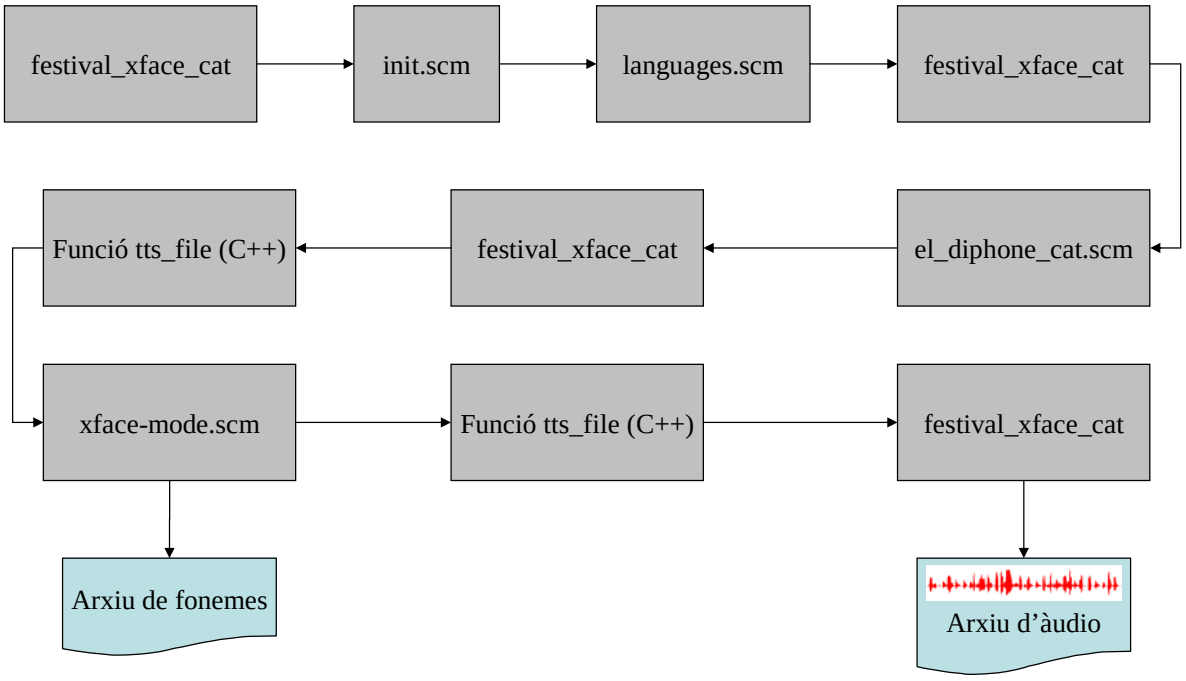

Figura 6-4: Esquema d'execució d'scripts de Festival en aquesta segona fase.

6.2.1. festival_xface_cat

Aquest arxiu comença sent una "shell" de Linux que crida a l'executable de Festival en mode "--script" (executa immediatament l'script Scheme que se li passi) i passant-se ell mateix i tots els paràmetres que se li han passat com a paràmetres. A continuació comença l'script Scheme pròpiament dit.

El primer que es fa, després de carregar manualment l'arxiu "init.scm", és seleccionar la veu catalana, ja que la que està per defecte és una anglesa, cridant a la funció "language_western_catalan". Després d'això i d'haver definit les funcions que es faran servir, comença l'execució de la funció "main", la qual primer llegeix els paràmetres que s'han passat d'entrada (l'entrada és l'arxiu test-final.xml, i les sortides són test-final.wav per l'àudio i test-final.pho pels fonemes) i després aplica la funció "tts_file" per cada arxiu d'entrada en mode "xface" (en aquest cas, només a test-final.xml).

La funció "tts_file", definida a nivell de codi intern, primer separa el text donat en diferents *utterances* i després aplica a cadascuna d'aquestes les funcions que hi hagi a la llista "tts_hooks". Aquestes funcions normalment són "utt.synth" (funció estàndard de síntesi) i "utt.play" (funció estàndard de reproducció), però en aquest cas "festival_xface_cat" ha redefinit la variable prèviament com la llista de "utt.synth", igualment, i "save_record_wave", funció pròpia que enlloc de reproduir la sortida la

guarda en un arxiu d'àudio temporal, mitjançant la crida de la funció estàndard per a aquest efecte per cada *utterance* "utt.save.wave".

En el cas de l'exemple concret d'aquesta execució, el text d'entrada es separa en dues *utterances*, corresponents a les dues frases del text. És a dir, de la següent manera:

***Utterance* 1:**

```
Test final d'incorporaci'o del catal'a en un agent conversacional
animat per computador.
```

***Utterance* 2:**

```
Sembla que funciona.
```

Al final la funció "combine_waves" ajunta tots aquests arxius d'àudio temporals en un de sol amb el resultat final (veure Figura 6-5). Després s'acaba l'script i per tant s'acaba l'execució de Festival i el control torna a XFace.

6.2.2. init.scm i languages.scm

L'arxiu "init.scm" (al directori "festival/lib"), com el seu nom indica, és l'script que serveix per inicialitzar les variables globals més importants de Festival, i per tant s'executa automàticament cada cop que aquest. Existeix, però, una excepció, el mode "--script", que és precisament el que fa servir "festival_xface_cat". Això és així perquè pot haver-hi scripts que no ho necessitin. No és el cas del d'aquest treball, però, i per això es carrega aquest manualment.

En aquest procés Scheme, doncs, es carreguen, entre molts d'altres, molts dels mòduls per fer la síntesi, funcions d'alt nivell com "tts", les configuracions personals de l'usuari i, per descomptat, totes les veus i els idiomes disponibles.

Aquesta darrera càrrega es fa mitjançant l'script "languages.scm". Aquest és el que defineix la funció usada a "festival_xface_cat" per a canviar l'idioma, "language_western_catalan", la qual dóna al paràmetre d'idioma el valor "catalan" i crida a la funció que en sí defineix la veu catalana, "voice_el_diphone_cat".

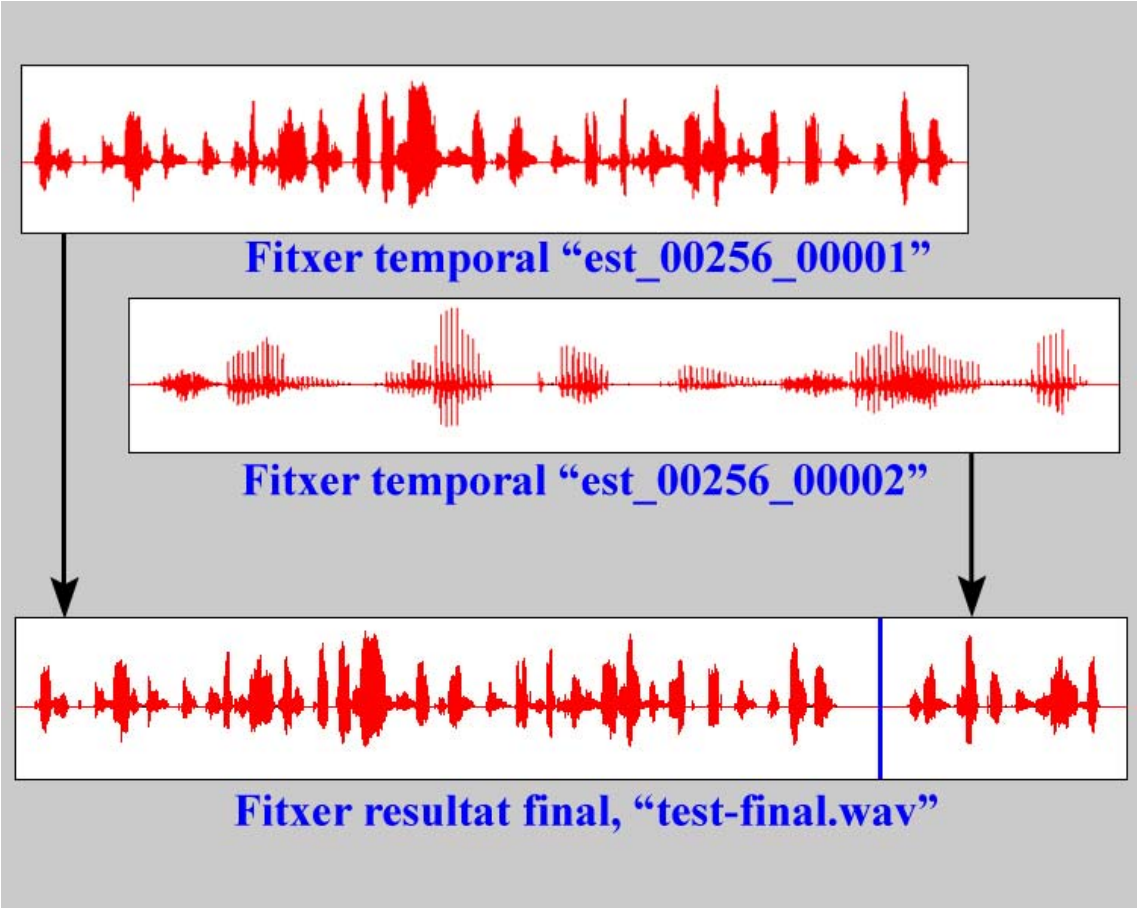

Figura 6-5: Combinació de les dues *utterances* sintetitzades en el fitxer final.

6.2.3. el_diphone_cat.scm

Aquest script (“festival/lib/voices/catalan/el_diphone_cat/festvox”) és el principal dins del conjunt d’arxius que composen la veu catalana i defineix aquesta, fent les referències apropiades als altres; és a dir, carrega els difonemes de l’arxiu “el_diphone_cat/group/ellpc11k.group”, les regles *token to words* de “catoken.scm”, les regles *letter to sound* de “catlex.scm”, etc. La funció principal d’aquest script és l’anomenada “voice_el_diphone_cat”, la qual s’encarrega de definir tots els paràmetres de síntesi de la veu. Això vol dir, el mòdul que haurà de fer la tokenització i les regles *token to words*, el lèxic i les regles *letter to sound*, el mètode de càlcul de morfologia (en aquest cas no en té), de les frases, de l’accentuació, de la duració, etc.

6.2.4. xface-mode.scm

Pel que fa al mode “xface”, definit a l’arxiu “lib/xface-mode.scm”, aquest es carrega al principi de “tts_file” i el que fa és afegir, mitjançant la funció d’inici que

tenen tots els modes “init_func” (en aquest cas es diu “xface_init_func”), unes regles *token to words* noves, així com una funció a l’”after_analysis_hooks”, anomenada “xface_output_info”:

- Noves regles *token to words*: Pels casos en que el text d’entrada arribés amb etiquetes XML, aquestes regles fan que tot text entre els símbols ‘<’ i ‘>’ s’ignori per a la síntesi. Si més endavant es modifiqués XFace per passar les etiquetes d’SMIL a Festival i processar-les en aquest, a l’hora de sintetitzar s’haurien d’ignorar, i per això aquestes normes. Donat que aquestes normes s’apliquen token per token, però, cal fer cinc casos:
  1. En el cas que un token comenci amb ‘<’ i acabi amb ‘>’, en aquest cas simplement es tracta d’una etiqueta d’una sol token i per tant s’ignora.
  2. En el cas que un token comenci per ‘<’, en aquest cas s’ignora i es posa a “true” (‘t’) la variable “SMIL”, que controla que s’està present dins d’una etiqueta d’aquest llenguatge.
  3. En el cas que un token acabi per ‘>’, s’ignora també, però el text a partir d’ara sí que s’haurà de sintetitzar, i per tant la variable “SMIL” es posa a “false” (‘nil’).
  4. Finalment, si un token no compleix la condició anterior però la variable “SMIL” és certa vol dir que aquest token pertany a un text d’etiqueta i també serà ignorat.
  5. En qualsevol altre cas és text que s’ha de sintetitzar i per tant es passa a les regles *token to words* corresponents, ara guardades a la variable “xface_previous_t2w_func”.
- Funció “xface_output_info”: Aquesta funció, que com que forma part de la llista “after_analysis_hooks” s’executarà just després que hagi acabat el procés de cada *utterance* i just abans de que comenci la generació de la forma d’ona d’aquesta, el que fa és cridar a la funció, també definida dins del mateix script, “utt.save.xface_phonedata” per a que guardi, per tota la *utterance*, cadascun dels seus fonemes i la seva duració acumulada a l’arxiu donat per la variable global “phofile”, que ja ha estat inicialitzada a “festival_xface_cat” amb el valor passat pel paràmetre corresponent (el nom que segueix a “-p” a la crida), i retorna la *utterance* mateixa al procés normal de sintetització. Així doncs, la funció “utt.save.xface_phonedata” és la que realment fa la tasca necessària per a XFace, la creació de l’arxiu de fonemes. Concretament, el que fa és obrir l’arxiu en

mode d'"append" (l'arxiu es crea net de zero a la funció "xface_init_func"), per cada fonema va afegint la duració en una variable global (inicialitzada a 0 en carregar el mode) i guarda una nova línia a l'arxiu amb el fonema i aquesta durada acumulada, el format que XFace requereix. Quan ja no queden fonemes a *l'utterance* tanca l'arxiu.

D'aquesta manera Festival ja haurà creat els arxius d'àudio i de fonemes que XFace requereix.

Per al cas de l'exemple, a continuació és mostren les primeres i les darreres files de l'arxiu de fonemes generat:

Arxiu "test-final.pho":

```
# 0.2500
t 0.3350
e1 0.4430
s 0.5530
t 0.6380
f 0.7380
i 0.8080
n 0.8880
a1 0.9960
l 1.0760
```

(...)

```
p 5.2460
e 5.3260
r 5.3560
k 5.4560
u 5.5260
m 5.5960
p 5.6960
u 5.7660
t 5.8510
a 5.9310
d 6.0210
o1 6.1560
# 6.4060
# 6.6560
s 6.7660
e1 6.8740
m 6.9440
b 7.0090
l 7.0890
a 7.1690
k 7.2690
e 7.3490
f 7.4490
u 7.5190
n 7.5990
s 7.7090
i 7.7790
o1 7.8870
```

```
n 7.9830
a 8.0790
# 8.3290
```

6.3. Tercera fase: Generació de les animacions i preparació de la reproducció

Per acabar, en aquest subapartat s'explica en detall el paper de les principals classes d'XFaceEd, XSmilAgent i XFace que intervenen en aquesta tercera fase, tal com es mostren a l'esquema de la Figura 6-6.

6.3.1. Classe XSmilAgent::SMILManager

Un cop ha acabat l'execució externa i el control torna a la funció "process" de l'objecte wxScriptProcessor, aquesta acaba i retorna el control a aquesta classe. El paper d'aquesta en aquesta darrera fase (en el cas concret del mètode de "keyframes") consisteix en llegir l'arxiu de fonemes generat pel TTS i generar a partir dels temps donats en aquest l'arxiu d'animacions.

Així doncs, quan la funció de processament de la parla acaba, la funció "parse" segueix interpretant l'script. En cas de que trobi una etiqueta <speech-animation> llegeix el contingut d'aquesta i el guarda en memòria. Finalment, abans de tornar el control a XFaceEd, crea un arxiu d'animacions bàsic, tant si se n'han especificat com no, ja que és un arxiu necessari, el contingut del qual simplement indica que es mostri expressió neutra o de descans durant tota la seqüència. Després, mitjançant la funció "post_process", el modifica amb les expressions especificades en cas de que n'hi hagués.

Per l'exemple d'aquest apartat, donat que no s'han donat animacions l'arxiu ANIM que es crea és el bàsic, el qual indica que s'ha de mantenir expressió neutra des del segon 0 fins el segon 8,329:

Arxiu "test-final.anim":

```
Rest 0 8.329 1
```

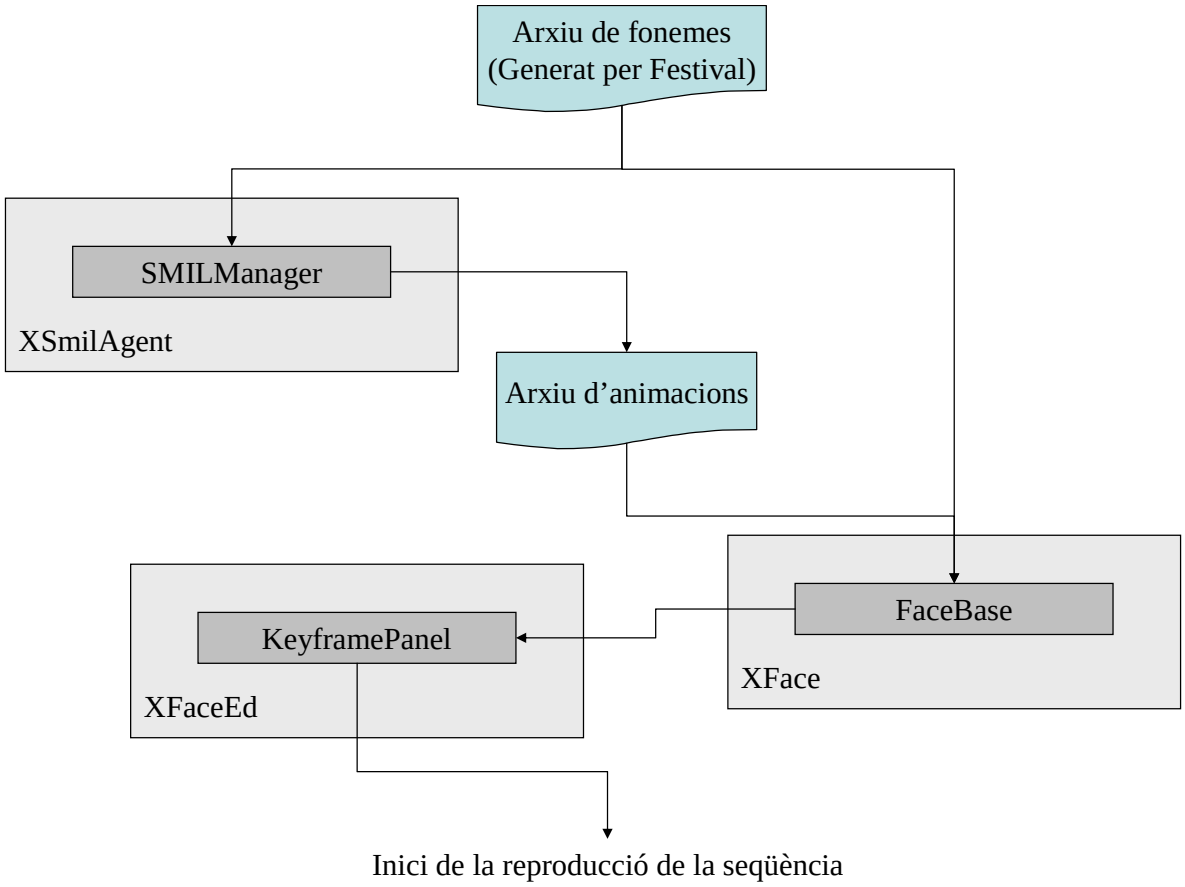

Figura 6-6: Esquema del flux de dades a la tercera fase, amb les classes principals que hi intervenen.

6.3.2. Classe XFaceEd::KeyFramePanel

Un cop creats els arxius necessaris, el control torna a la funció "OnPlaySMIL" d'aquesta classe, la qual procedeix a preparar la seqüència per a la seva reproducció. Crea l'objecte "pface" de classe "XFace::FaceBase", el qual fa referència a la cara que es mostra en pantalla, i fa que aquest carregui les llistes ordenades en el temps de fonemes (prèvia recuperació de l'idioma, guardat encara a l'objecte SMILManager) i animacions. Finalment activa la reproducció de la seqüència (veure Figura 6-7), amb sò o sense, segons si aquest està activat o no, de manera que aquí acaba la tercera fase.

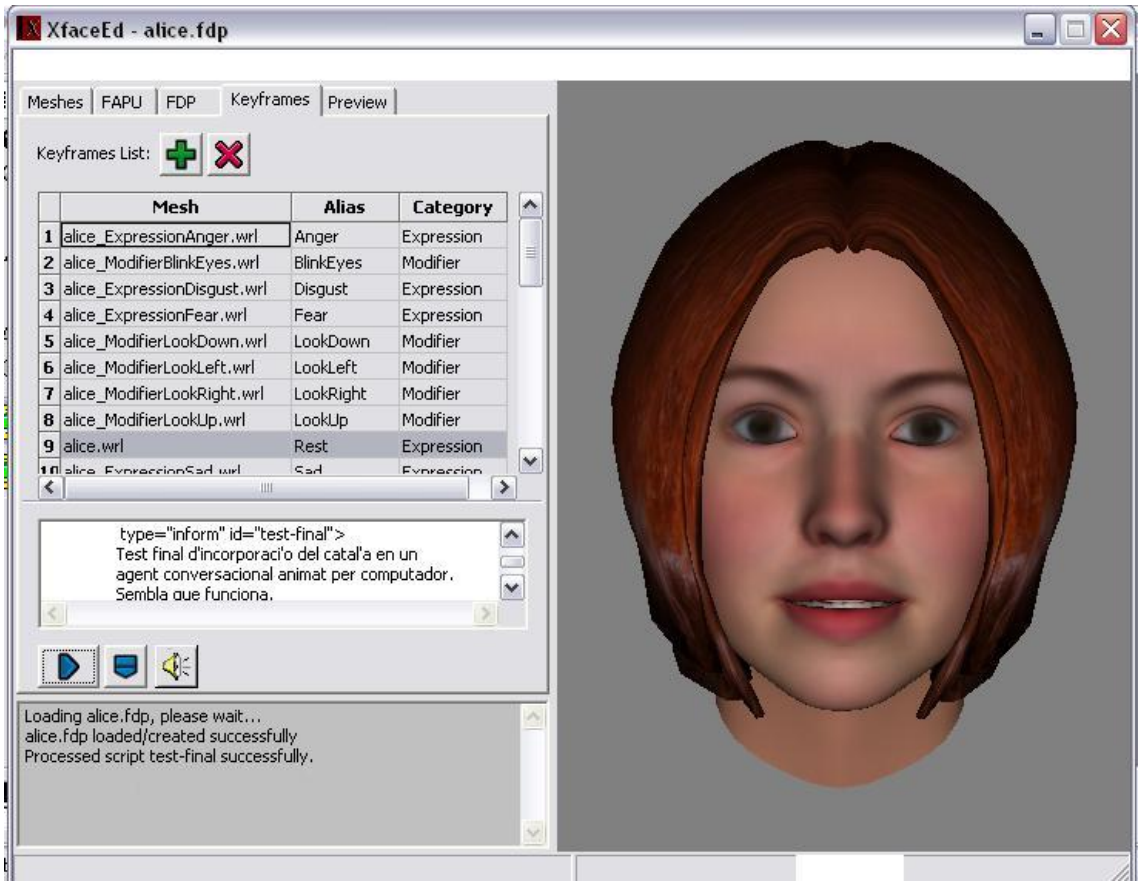

Figura 6-7: Reproducció de l'script de test a XFaceEd.

6.3.3. Classe XFace::FaceBase

Aquesta classe és bàsicament la que relaciona l'avatar carregat i la seqüència que s'aplicarà a aquest, tant pel cas dels FAPs com dels "keyframes". En aquest darrer cas, s'encarrega de guardar totes les transformacions en ordre a partir de la informació extreta dels arxius d'entrada, PHO i ANIM, mitjançant les funcions "processPhonemes" i "processAnims". La primera, més concretament, neteja i redefineix la llista de visemes i respectives duracions mitjançant subseqüents crides a la funció "addToSequence", la qual obre el diccionari de fonemes-visemes corresponent segons l'idioma i substitueix el fonema informat pel visema que li pertoca.

Capítol 7. Conclusions

Un cop finalitzades les tasques de recerca, programació i prova d'aquest treball, es dóna per complert l'objectiu principal: proveïr a XFace d'una eina TTS plenament funcional amb almenys una veu catalana. Tant durant el procés com després de finalitzar aquest, s'han anat observant una sèrie d'aspectes que es volen remarcar en aquest apartat.

Primer de tot, remarcar el fet de que **s'ha aconseguit usar en la totalitat d'aquest eines de software lliure** les quals són, a més, projectes de recerca oberts per altres universitats o centres de recerca (tant XFace com Festival), amb les quals es pot treballar en cooperació. La Universitat Autònoma podrà col·laborar en la millora d'aquestes per una banda i per l'altra es podrà beneficiar de tots els avenços i millores que hi aportin altres equips de recerca de tot el món.

Un altre aspecte important a remarcar és que, donat que durant la recerca de l'eina sintetitzadora a la Universitat Autònoma no es va trobar cap mena de software o investigació al respecte, es pot arribar a la conclusió de que **aquest treball obre**, de per sí i independentment de la seva relació amb el Projecte HERMES al qual pertany, **la recerca en eines TTS en aquesta universitat**.

En aquest darrer aspecte, l'ús de Festival també es considera un factor molt important per força raons: és el projecte de síntesi de veu a partir de text de codi lliure més extès globalment, està dissenyat amb una estructura molt modular, cosa que facilita en gran manera futures modificacions, i sobretot perquè és alhora una plataforma de recerca i una eina TTS completa i perfectament funcional. Un dels problemes més grans a l'hora de desenvolupar síntesi de veu, així com també en altres sistemes de procés de la parla i el llenguatge, és que hi ha un munt de tècniques simples i ben conegudes que poden ajudar al programador a assolir el seu objectiu en certs aspectes concrets; però per tal de treballar en alguna d'aquestes parts concretes és necessari tenir un sistema complet en el qual poder fer les proves pertinents. Festival està pensat per ser aquest "sistema complet" en el qual el programador pot treballar en una àrea concreta sense preocupar-se d'haver de fer, o adaptar, abans un entorn de test, comptant, a més, amb l'avantatge de que no és només això, un entorn de test, sinó un sistema TTS completament funcional que es pot afegir a altres projectes si és necessari, la

comunicació amb els quals pot ajudar a orientar la recerca sobre quines qüestions resoldre.

Tornant al projecte HERMES, també s'arriba a la conclusió de que **el llenguatge de markup SMIL-Agent**, estudiat en aquest treball, **pot resultar molt útil per aquest projecte**, i que per tant valdria la pena estudiar la seva incorporació per al control, ja no només d'XFace, sinó per l'agent conversacional en general, donat que de les seves especificacions**[11][12]** es dedueix que permet controlar a tots els nivells una interfície multimodal de 3r tipus, segons la definició donada per Caelen (veure introducció)**[2]**.

7.1. Línies de continuitat

Per acabar, remarcar que, tal com s'ha vist en les conclusions donades fins ara, el que ha fet aquest treball fins el moment ha estat obrir portes, noves vies de recerca. Ara queden pendents, doncs, les millores dels canvis realitzats en les eines emprades:

- Per la part de la interacció entre XFace i Festival, cal veure si aquesta es pot completar de manera que l'eina TTS retorni a l'avatar virtual els temps de les marques XML (les etiquetes <mark>), per tal de que aquest generi correctament l'arxiu d'animacions i el "cap parlant" pugui mostrar emocions també quan parli en català.
- També respecte la interacció, cal veure si es pot optimitzar el procés actual, per exemple usant Festival com a llibreria enlloc de com a executable extern.
- Pel que fa a la veu catalana, donada la baixa qualitat de la realitzada en aquest treball, queda pendent, senzillament, fer-ne una de nova. Preferiblement seguint el procediment definit pel projecte Festvox**[22]**, caldria gravar una base de dades de difonemes més completa en bones condicions, redefinir el càlcul de la prosòdia de l'actual, que és gairebé inexistent i no permet donar entonació (emoció) a les frases, definir millor la gramàtica catalana, amb l'ajuda d'experts en aquesta per canviar les regles *token to words* i *letter to sound* així com ampliar el vocabulari, etc.

I, per descomptat, finalment queda pendent integrar aquesta funcionalitat, la generació de parla en català, al projecte global HERMES.

Apèndix A. Estàndard FA d'MPEG-4

El 1999, el *Moving Pictures Experts Group* (MPEG, Grup d'Experts en Imatges en Moviment) va publicar l'estàndard MPEG-4 (ISO/IEC JTC1/WG11 N1901 i N1902), el qual abarcava una ampli espectre de temes multimèdia, com per exemple àudio i vídeo naturals i sintetitzats o gràfics en 2D i 3D. Al contrari que anteriors estàndars d'MPEG, que es centraven en la codificació eficient de diferents continguts, MPEG-4 tracta sobretot de la comunicació i la integració d'aquests. Avui en dia és l'únic estàndard que contempla animació facial, i ha estat àmpliament acceptat en l'entorn acadèmic, al mateix temps que també comença a cridar l'atenció de la indústria.

MPEG-4 *Facial Animation* (FA) descriu els passos per crear un agent parlant mitjançant la definició estandarditzada d'uns certs paràmetres necessaris. La creació d'aquest es divideix principalment en dues fases, tractades per separat en l'estàndard: la localització dels punts característics en el model 3D estàtic, cosa que defineix les regions de deformació a la cara, i la generació i la interpretació dels paràmetres que modificaran aquests punts característics per tal de generar l'animació en sí.

Per tal de crear una cara conforme amb l'estàndard, MPEG-4 defineix una sèrie de paràmetres anomenats FDP (*Facial Definition Parameters*, Paràmetres de Definició Facial), dels quals els més importants són els 84 FPs (*Feature Points*, punts característics) localitzats en un model d'un cap. Aquests s'haurien de definir per cada model 3D estàtic (indicar a quins vèrtexs concrets corresponen) que es vulgui fer conforme l'estàndard i la seva funció és descriure la forma d'un cap "normal" en "posició de descans", la qual s'asumeix sempre com la posició de partida de qualsevol animació i és definida per l'estàndard amb les següents característiques: tots els músculs relaxats, parpelles tangents a l'iris, les pupiles obertes fins a un terç del diàmetre de l'iris, els llavis i les dents en contacte i la llengua plana amb la punta tocant a les dents. Aquests punts característics, doncs, s'utilitzen per definir paràmetres de l'animació així com per calibrar els models quan s'obrin en reproductors diferents. La figura A-1**[33]** mostra el conjunt sencer dels FPs.

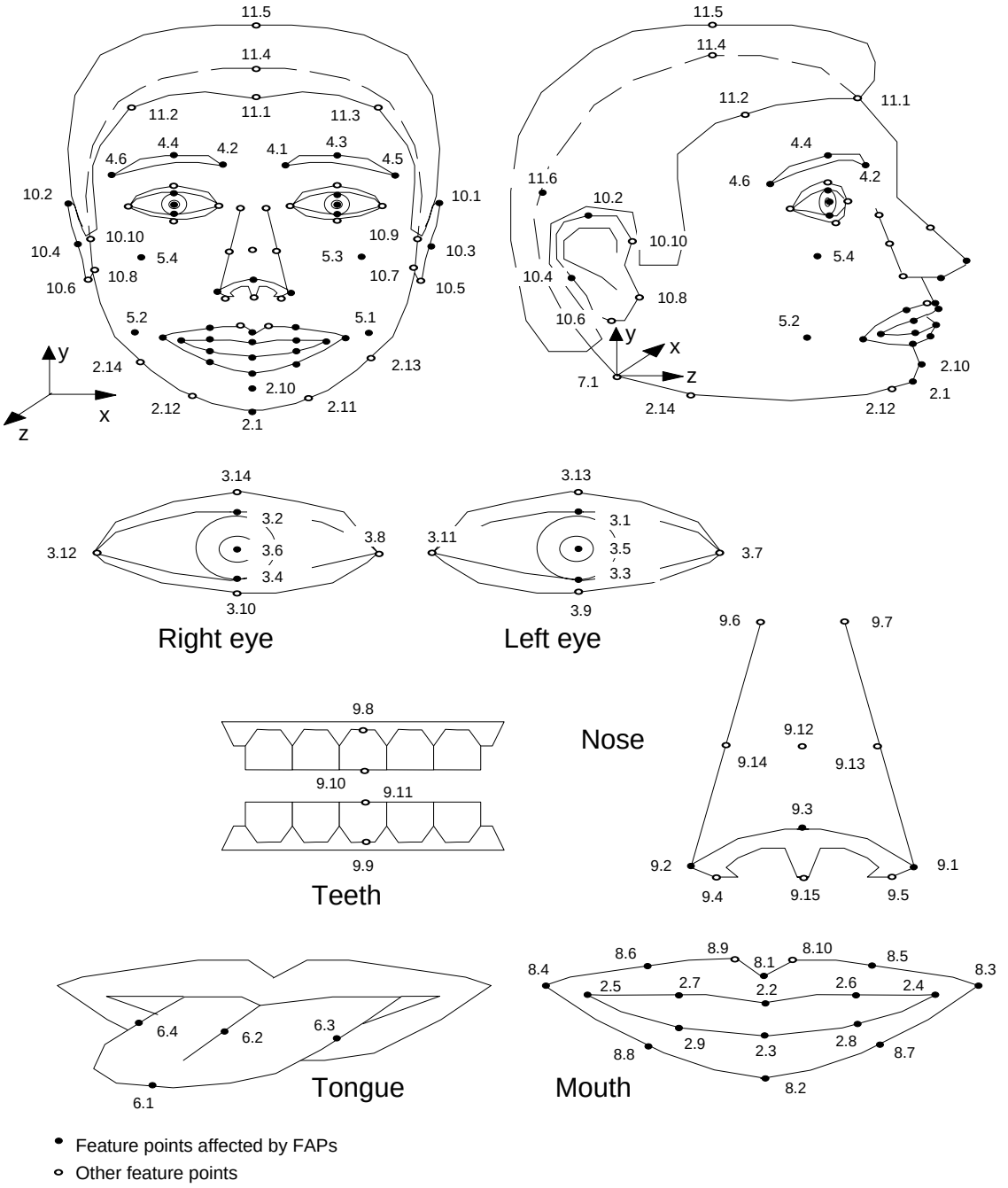

Figura A-1: FPs definits per MPEG-4 FA, agrupats per les diferents parts del cap a les que pertanyen. Diferencia els 46 FPs als que afecten els FAPs dels que no.

Pel que fa als paràmetres per a l'animació d'aquests punts, es defineixen 68 FAPs (*Facial Animation Parameters*, Paràmetres d'Animació Facial), definits amb valor 0 per a la posició de descans. Els primers dos són paràmetres d'alt nivell que representen visemes i expressions facials bàsiques respectivament. Els visemes són les posicions de la cara, més específicament les de la boca, que es corresponen a cadascun

dels fonemes, mentre que les expressions facials consisteixen en en un conjunt de sis emocions bàsiques (enuig, joia, tristesa, sorpresa, fàstic i por) que serveixen com a prototipus. Es pot controlar l'animació d'un model de cap usant només aquests 2 paràmetres i arribar a resultats satisfactoris utilitzant interpolació lineal entre cada prototipus. Per altra banda, però, per aconseguir una millor qualitat s'aconsella l'ús de la resta de paràmetres de baix nivell. Cadascun d'aquests s'encarrega d'una regió específica de la cara, com poden ser la comissura dreta dels llavis, el punt més baix de la barbeta, o el cantó esquerre de la cella esquerra. Cada FAP es correspon a un FP i defineix deformacions de baix nivell aplicables a aquest. Amb l'ajuda dels FAPs, els desenvolupadors tenen un conjunt d'entrades estàndard i al mateix temps són lliures de decidir com interpreten aquest o com aplicar les deformacions al model.

Cal notar que els FAPs són paràmetres universals, independents de la geometria del model. Per aquesta raó, abans d'utilitzar-los per a animar qualsevol model, han de ser calibrats. Això es pot fer utilitzant les Unitats de Paràmetres d'Animació Facial (Facial Animation Parameter Units, FAPU). Aquestes es defineixen com fraccions de distàncies entre característiques clau de la cara, com es mostra a la Figura A-2**[7]**, i són específiques del model 3D utilitzat. Les 5 distàncies definides són, de dalt a baix:

- **ES0:** *Eye Separation*, separació entre ulls. Concretament la distància entre el centre d'un ull i l'altre.
- **IRISD0:** *Iris Diameter*, Diàmetre de l'iris.
- **ENS0:** *Eye-Nose Separation*, separació entre ulls i nas, mesurada com la distància entre el centre del segment dibuixat per *ES0* i el punt més inferior del nas.
- **MNS0:** *Mouth-Nose Separation*, separació entre el nas i la boca. Concretament, la distància entre el punt més inferior del nas, prèviament mencionat, i el punt mig del segment definit per *MW0* (la següent distància).
- **MW0** és *Mouth Width*, amplada de la boca, que és la distància entre l'extrem de la comissura esquerra i l'extrem de la comissura dreta.

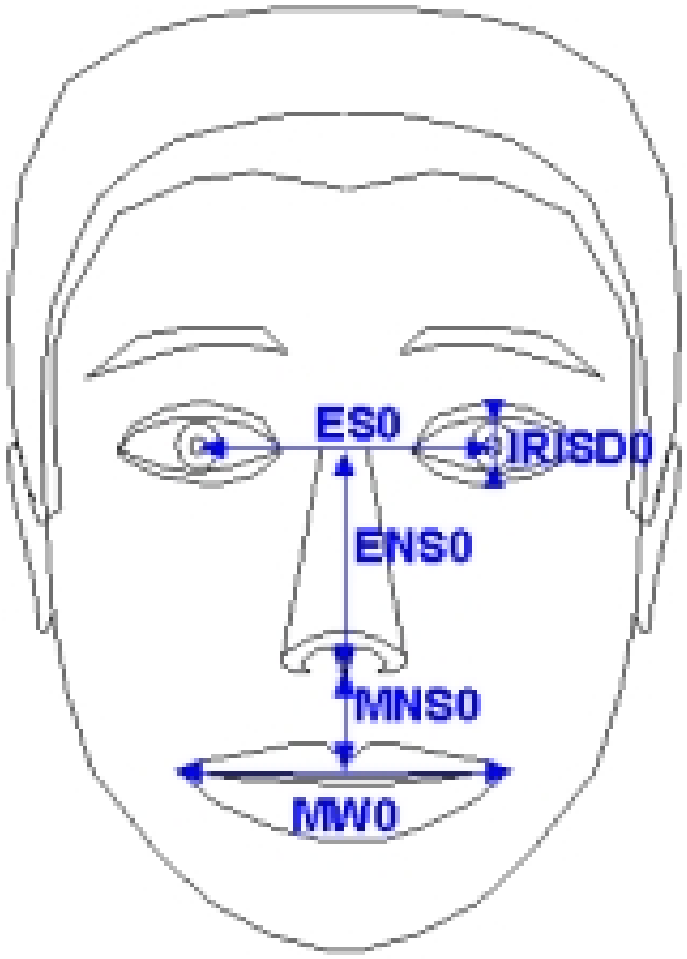

Figura A-2: FAPUs definits per MPEG-4 FA.

L'estàndard defineix, doncs, per cada FAP, la FAPU que n'ha de calibrar el valor, quan aquest arribi com a entrada de l'aplicació. Així doncs, juntament amb els FPs, les FAPUs serveixen per a arribar a assolir independència total del model final a animar per tots aquells reproductors que segueixin l'estàndard FA d'MPEG-4. Codificant doncs els models de cares en FPs i FAPUs, els desenvolupadors poden intercanviar aquests lliurement, sense haver-se de preocupar per la calibració i la parametrització de l'animació.

La següent taula**[a]** mostra com a exemple els 5 primers FAPs, donant-ne el seu número, el nom, una descripció curta, la FAPU a la que s'associa, si el FAP és unidireccional (només pot prendre valors positius) o bidireccional (pot prendre tant valors positius com negatius), i la direcció en la que es mourà per valors positius del paràmetre. Per exemple, el FAP número 3, "lower_jaw", que indica el desplaçament vertical del maxil·lar inferior, és unidireccional cap a baix ja que no té sentit (és fisiològicament impossible) pujar-lo més del punt de partida, la posició de descans, en

la qual, per definició, les dents inferiors estan tocant les superiors. El FAP número 4, “lower_t_midlip”, en canvi, sí que permet moure’s en dos direccions ja que és possible pujar més el llavi inferior des de la posició de descans. Això no vol dir, però, que no es pugui donar un valor negatiu als FAPs unidireccionals; simplement vol dir que l’expressió resultant de la cara no serà realista.

#	Nom del FAP	Descripició	Unitat	Uni- o Bidireccional	Direcció del Moviment
1	Viseme	Conjunt de valors que determina la combinació de dos visemes.	no aplica	no aplica	no aplica
2	Expression	Conjunt de valors que determina la combinació de dues expressions facials.	no aplica	no aplica	no aplica
3	Open_jaw	Desplaçament vertical del maxil·lar inferior (no afecta a l’obertura de la boca)	MNS0	Unidireccional	avall
4	Lower_t_midlip	Desplaçament vertical del punt mig del llavi superior.	MNS0	Bidireccional	avall
5	Raise_b_midlip	Desplaçament vertical del punt mig del llavi inferior.	MNS0	Bidireccional	amunt

Taula 0-1: FAPs i les seves propietats.

Apèndix B. Exemple d'arxiu FDP d'XFace

A continuació es dóna com a exemple d'arxiu de configuració d'un cap per XFace el que vé per defecte amb aquest, que defineix l'avatar femení anomenat "Alice", amb les parts importants remarcades amb text en negreta.

Exemple d'arxiu de configuració d'XFace FDP:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<xfdp>

  <!--XFaceEd generated MPEG-4 FDP specification file-->

  <!--Header information for the model-->

  <head>
    <!--Version of the FDP specification file (this file)-->
    <file version="0.2"/>
```

Definició de les FAPU:

```
    <!--FAPU (Facial Animation Parameter Units) for the model-->
    <fapu ENS0="51" ES0="69" IRISD0="16" MNS0="30" MW0="50"/>
```

```
    <!--Global Translation info for the whole face-->
    <translation x="0" y="-1" z="-659"/>
    <!--Global Rotation info for the whole face-->
    <rotation axis_angle="0.407589" axis_x="-0.999632" axis_y="-
0.0154467" axis_z="0.0223226"/>
  </head>

  <!--3D model files (mesh) configuration-->
```

Especificació dels models 3D per cada "keyframe":

```
  <source>
    <entity alias="Anger" category="Expression">
      <mesh file="alice_ExpressionAnger.wrl" format="wrl"/>
    </entity>
    <entity alias="BlinkEyes" category="Modifier">
      <mesh file="alice_ModifierBlinkEyes.wrl" format="wrl"/>
    </entity>
    <entity alias="Disgust" category="Expression">
```

```
      <mesh file="alice_ExpressionDisgust.wrl" format="wrl"/>
    </entity>
    <entity alias="Fear" category="Expression">
      <mesh file="alice_ExpressionFear.wrl" format="wrl"/>
    </entity>
    <entity alias="LookDown" category="Modifier">
      <mesh file="alice_ModifierLookDown.wrl" format="wrl"/>
    </entity>
    <entity alias="LookLeft" category="Modifier">
      <mesh file="alice_ModifierLookLeft.wrl" format="wrl"/>
    </entity>
    <entity alias="LookRight" category="Modifier">
      <mesh file="alice_ModifierLookRight.wrl" format="wrl"/>
    </entity>
    <entity alias="LookUp" category="Modifier">
      <mesh file="alice_ModifierLookUp.wrl" format="wrl"/>
    </entity>
```

Especificació dels models 3D bàsics per formar el cap en posició de descans

```
    <entity alias="Rest" category="Expression">
      <mesh file="alice.wrl" format="wrl"/>
      <bind item="Hair" submesh="alice.wrl_0"/>
      <bind item="LeftEye" submesh="alice.wrl_1"/>
      <bind item="RightEye" submesh="alice.wrl_2"/>
      <bind item="LowerTeeth" submesh="alice.wrl_5"/>
      <bind item="UpperTeeth" submesh="alice.wrl_6"/>
      <bind item="Tongue" submesh="alice.wrl_7"/>
    </entity>
```

Segueix especificant més models pels “keyframes”

```
    <entity alias="Sad" category="Expression">
      <mesh file="alice_ExpressionSad.wrl" format="wrl"/>
    </entity>
    <entity alias="SmileClosed" category="Expression">
      <mesh file="alice_ExpressionSmileClosed.wrl" format="wrl"/>
    </entity>
    <entity alias="SmileOpen" category="Expression">
      <mesh file="alice_ExpressionSmileOpen.wrl" format="wrl"/>
    </entity>
    <entity alias="Surprise" category="Expression">
      <mesh file="alice_ExpressionSurprise.wrl" format="wrl"/>
    </entity>
    <entity alias="aah" category="Viseme">
      <mesh file="alice_Phonemeaah.wrl" format="wrl"/>
    </entity>
    <entity alias="b" category="Viseme">
      <mesh file="alice_PhonemeB,M,P.wrl" format="wrl"/>
    </entity>
    <entity alias="bigaah" category="Viseme">
      <mesh file="alice_Phonemebigaah.wrl" format="wrl"/>
    </entity>
    <entity alias="ch" category="Viseme">
      <mesh file="alice_Phonemech,J,sh.wrl" format="wrl"/>
    </entity>
    <entity alias="d" category="Viseme">
      <mesh file="alice_PhonemeD,S,T.wrl" format="wrl"/>
    </entity>
```

```
    <entity alias="ee" category="Viseme">
      <mesh file="alice_Phonemeee.wrl" format="wrl"/>
    </entity>
    <entity alias="eh" category="Viseme">
      <mesh file="alice_Phonemeeh.wrl" format="wrl"/>
    </entity>
    <entity alias="f" category="Viseme">
      <mesh file="alice_PhonemeF,V.wrl" format="wrl"/>
    </entity>
    <entity alias="i" category="Viseme">
      <mesh file="alice_Phonemei.wrl" format="wrl"/>
    </entity>
    <entity alias="k" category="Viseme">
      <mesh file="alice_PhonemeK.wrl" format="wrl"/>
    </entity>
    <entity alias="n" category="Viseme">
      <mesh file="alice_PhonemeN.wrl" format="wrl"/>
    </entity>
    <entity alias="oh" category="Viseme">
      <mesh file="alice_Phonemeoh.wrl" format="wrl"/>
    </entity>
    <entity alias="q" category="Viseme">
      <mesh file="alice_Phonemeooh,Q.wrl" format="wrl"/>
    </entity>
    <entity alias="r" category="Viseme">
      <mesh file="alice_PhonemeR.wrl" format="wrl"/>
    </entity>
    <entity alias="th" category="Viseme">
      <mesh file="alice_Phonemeth.wrl" format="wrl"/>
    </entity>
    <entity alias="w" category="Viseme">
      <mesh file="alice_PhonemeW.wrl" format="wrl"/>
    </entity>
  </source>
```

Des d'aquí fins el final, definició dels FDP: Tots els FPs i per cadascun d'aquests la seva zona d'influència, el FAP que l'afecta, la funció de deformació a aplicar i el "pès" (factor que s'aplica dins la funció de deformació). Els vèrtexs del model es donen indexats.

```
  <fdp affects="alice.wrl_3" index="2395" name="2.1">
    <indices>0 1 2 78 104 176 177 178 259 282 407 408 409 410 411 412
413 414 415 416 417 442 445 446 447 449 478 499 506 782 783 788 976
977 978 979 980 981 982 983 984 1008 1009 1034 1053 1302 1303 1308
1439 1440 1441 1442 1609 1610 1611 1677 1678 1679 1835 1836 1837 1838
2024 2025 2026 2090 2091 2092 2304 2330 2331 2332 2333 2334 2335 2336
2337 2338 2339 2340 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350
2384 2386 2387 2388 2393 2394 2395 2396 2397 2398 2399 2400 2401 2402
2403 2453 2494 2496 2497 2510 2511 2512 3194 3196 3197 3326 3327 3328
3329 3330 3331 3332 3343 3344 3352 3358 3359 3360 3361 3362 3363 3396
3397 3398 3399 3406 3407 3603 3605 3612 3619 3620 3621 3647 4139 4140
4141 4142 4143 4144 4145 4146 4147 4148 4149 4150 4151 4152 4153 4154
4155 4156 4157 4158 4191 4192 4193 4199 4200 4201 4202 4203 4204 4205
4206 4207 4252 4294 4296 4297 4310 4312 4956 4957 4958 5087 5088 5089
5090 5091 5092 5093 5104 5109 5110 5112 5118 5119 5120 5121 5122 5123
5155 5156 5157 5158 5165 5360 5361 5369 5375 5376 5377 5403 </indices>
    <influence fap="2" type="RaisedCosInfluenceSph" weight="0.3"/>
  </fdp>
```

```
  <fdp affects="alice.wrl_3" index="414" name="2.10">
    <indices>413 414 499 500 506 788 982 1052 1053 1308 2339 2340 2341
2342 2398 2494 2496 2497 2498 2510 2511 2512 3358 3359 3360 3397 3398
3399 3406 3407 4148 4150 4151 4294 4295 4296 4297 4310 4312 5118 5119
5120 5156 5157 5158 5165 </indices>
    <influence fap="17" type="RaisedCosInfluenceSph" weight="0"/>
  </fdp>

  <fdp affects="alice.wrl_3" index="720" name="2.2">
    <indices>593 596 597 598 599 603 604 631 633 638 639 718 719 720
721 1143 1145 1146 1147 1151 1152 1172 1175 1178 1179 2709 2710 2712
2713 2714 2715 2716 2717 2718 2723 2724 2732 2733 2734 2735 2810 2833
2836 2837 2838 2839 2841 2851 2852 3067 3068 3069 3070 3071 3072 3073
3074 3075 3076 3077 3078 3079 3082 3084 4063 4507 4508 4510 4511 4512
4513 4514 4515 4520 4522 4529 4530 4531 4532 4600 4622 4623 4624 4625
4626 4630 4637 4639 4837 4838 4839 4840 4841 4842 4843 4844 4845 4847
5809 </indices>
    <influence fap="3" type="RaisedCosInfluenceWaveY" weight="0.25"/>
    <influence fap="16" type="RaisedCosInfluenceSph" weight="0.2"/>
  </fdp>

  <fdp affects="alice.wrl_3" index="627" name="2.3">
    <indices>413 414 498 499 505 506 611 613 614 615 616 618 619 620
621 622 623 625 626 627 636 637 982 1053 1054 1159 1160 1162 1164 1165
1166 1168 1177 2341 2494 2495 2496 2497 2510 2511 2512 2513 2753 2754
2756 2757 2758 2759 2760 2761 2762 2767 2768 2769 2770 2771 2772 2773
2774 2775 2776 2777 2778 2779 2783 2784 2785 2786 2787 2788 2789 2790
2792 2793 2794 2795 2803 2804 2805 2806 2807 2808 2820 2821 2823 2824
2826 2847 2848 2850 2854 3359 3397 3398 3406 3407 4031 4032 4033 4150
4294 4296 4297 4298 4310 4311 4312 4550 4551 4552 4554 4555 4556 4557
4558 4563 4564 4565 4566 4567 4568 4569 4570 4571 4572 4576 4577 4578
4579 4580 4581 4582 4584 4585 4593 4594 4595 4596 4597 4607 4608 4609
4611 4613 4634 4636 4640 5120 5156 5158 5165 5777 5778 5779 </indices>
    <influence fap="4" type="RaisedCosInfluenceWaveY" weight="0.33"/>
    <influence fap="15" type="RaisedCosInfluenceSph" weight="-0.1"/>
  </fdp>
```

(...)

```
    <fdp affects="alice.wrl_3" index="4383" name="3.2">
    <indices>185 187 188 260 1088 1089 1090 1091 1092 1093 1094 1095
1096 1097 1099 1109 1110 1120 1121 1122 1131 1134 1245 1246 1247 1251
1253 1254 1255 1258 1259 1260 1277 1314 1852 1855 2019 2020 2021 2022
2023 2027 2028 2029 2030 2031 2032 2033 2035 2036 2047 2049 2050 4371
4376 4378 4379 4380 4382 4385 4386 4387 4389 4394 4395 4396 4397 4398
4400 4401 4402 4403 4404 4405 4406 4419 4420 4421 4451 4456 4457 4458
4460 4478 4863 4864 4865 4866 4867 4868 4869 4870 4871 4872 4873 4887
4888 4889 4895 4896 4897 4898 4899 4900 4901 4905 4911 4912 4913 4914
4915 4916 4917 4918 4919 4920 4921 4930 4931 4976 4978 5189 5538 5539
5678 5789 </indices>
    <influence fap="19" type="RaisedCosInfluenceWaveY" weight="0.15"/>
    <influence fap="19" type="RaisedCosInfluenceWaveX" weight="0.2"/>
  </fdp>

  <fdp affects="alice.wrl_3" index="1123" name="3.4">
    <indices>186 191 197 198 201 203 220 340 357 1077 1078 1101 1111
1112 1113 1114 1115 1116 1123 1124 1125 1126 1127 1128 1129 1130 1132
```

```
1133 1137 1140 1305 1309 1865 1866 1867 1868 1869 1876 1879 1880 1881
1882 1883 2042 2217 2218 2220 4344 4345 4346 4347 4410 4412 4423 4424
4425 4426 4427 4428 4429 4430 4431 4432 4433 4434 4437 4438 4439 4440
4441 4442 4443 4444 4445 4446 4447 4448 4462 4463 4464 4465 4466 4467
4468 4469 4470 4471 4472 4473 4474 4475 4476 4484 4485 4486 4488 4497
4498 4499 4500 5037 5097 5098 5099 5101 5102 5103 5114 5124 5125 5126
5127 5128 5129 5130 5140 5160 5161 5162 5163 5164 5794 </indices>
    <influence fap="21" type="RaisedCosInfluenceSph" weight="0.1"/>
  </fdp>

  <fdp affects="alice.wrl_1" index="123" name="3.1">
    <indices>20 21 30 31 32 33 84 85 86 87 88 89 112 113 114 116 117
118 120 121 122 123 124 125 126 127 </indices>
  </fdp>

</xfdp>
```

Bibliografia

[1] World Wide Web Consortium. Oficina Española. *Guía Breve de Interacción Multimodal*. Darrera actualització: 13 de febrer del 2006. http://www.w3c.es/Divulgacion/Guiasbreves/Multimodalidad

[2] Eric Keller (editor) i altres, Universitat de Lausanne. *Fundamentals of Speech Synthesisand Speech Recognition*. John Wiley & Sons, 1995.

[3] Centre de Visió per Computador. *Human-Expressive Representations of Motion and their Evaluation in Sequences (HERMES Project)*. IST 027110. Darrera actualització: 2 de Juny del 2006. http://www.hermes-project.eu/

[4] ITC-irst. *XFace*. Darrera actualització: 7 d'octubre del 2005. http://xface.itc.it/

[5] Annalisa Guerzoni. *Web de la divisió TCC de l'ITC-irst*. http://tcc.itc.it/

[6] Instituto Trentino di Cultura. *Web de l'ITC-irst*. 2003. http://www.itc.it/

[7] Koray Balci, *Xface: MPEG4Based Open Source Toolkit for 3D Facial Animation*. ITC-irst, 2004.

[8] Koray Balci, *Xface: Open Source Toolkit for Creating 3D Faces of an Embodied Conversational Agent*. ITC-irst, 2005.

[9] N. Kojekinem V. Savchenko, M. Senin, I. Hagiwara, *Real-time 3D Deformations by Means of Compactly Supported Radial Basis Functions*, "Proc. of Eurographics02", pàgines 35 a 43, setembre del 2002.

[10] P. Faloutsos, M. van de Panne, D. Terzopoulos, *Dynamic Free-Form Deformations for Animation Synthesis*, "IEEE Transactions on Visualization and Computer Graphics", 3-3, pàgines 201 a 214, 1997.

[11] Divisió TCCde l'ITC-irst, *SMIL-AGENT Quick Reference*, 22 de març de 2005. http://tcc.itc.it/people/not/SMIL-AGENT/SMIL-AGENT-quick-reference.html

[12] Divisió TCC de l'ITC-irst, *Synchronized Multichannel Integration Language for Synthetic Agents (SMIL-AGENT) 0.1 Specification*, 22 de març de 2005. http://tcc.itc.it/people/not/SMIL-AGENT/SMIL-AGENT-01-specification.html

[13] MBROLA Team, *The MBROLA PROJECT HOMEPAGE*. Darrera actualització: 4 de gener de 2005. http://tcts.fpms.ac.be/synthesis/mbrola.html

[14] The Centre for Speech Technology Research, *The Festival Speech Synthesis System*.
http://www.cstr.ed.ac.uk/projects/festival/

[15] Alan W. Black i Kevin A. Lenzo, *Flite: a small, fast run time synthesis engine*.
http://www.speech.cs.cmu.edu/flite/

[16] Sun Microsystems Inc., *FreeTTS 1.2 - A speech synthesizer written entirely in the Java™ programming language*. Darrera actualització: 7 de febrer de 2005.
http://freetts.sourceforge.net/docs/index.php

[17] Thomas McCarthy, *Intro to NEXTSTEP*.
http://www120.pair.com/mccarthy/nextstep/intro.htmld/

[18] els webmasters de GNUStep, *Introduction to GNUstep*.
http://www.gnustep.org/information/aboutGNUstep.html

[19] David Hill, *Web del projecte gnuspeech*. Darrera actualització: 21 de març de 2005.
http://www.gnu.org/software/gnuspeech/

[20] Erwin Marsi, *NeXTeNS: Open Source Text-to-Speech for Dutch*. Darrera actualització: 30 d'octubre de 2003.
http://nextens.uvt.nl/

[21] Petr Horák, *The Epos Speech Synthesis System. A Open Text-To-Speech Synthesis Platform*. Darrera actualització: 10 d'agost de 2005.
http://epos.ure.cas.cz/

[22] Alan W. Black, Kevin J. Lenzo, *Web del projecte Festvox*, Darrera actualització: desembre del 2004.
http://festvox.org/

[23] Alan W. Black. "*Speech Synthesis in Festival. A practical course on making computers talk*". Language Technologies Institute, Carnegie Mellon University, 30 de maig del 2000.
http://festvox.org/festtut/notes/festtut_toc.html

[24] Alan W. Black, Paul Taylor i Richard Caley, *The Festival Speech Synthesis System*, edició 1.4, The Centre for Speech Technology Research, University of Edimburgh, 17 de juny de 1999.
http://www.cstr.ed.ac.uk/projects/festival/manual/

[25] Chris Hanson, *The Scheme Programming Language*, Massachussetts Institute of Technology. Darrera actualització: 23 d'octubre del 2003.
http://swiss.csail.mit.edu/projects/scheme/

[26] George J. Carrette, *SIOD: Scheme in One Defun*, 15 de juliol del 1996.
http://www.cs.indiana.edu/scheme-repository/imp/siod.html

[27] Thierry Dutoit, *An Introduction to Text-to-Speech Synthesis*, Kluwer Academic Publishers, 1997.

[28] Pierre Delattre, *Les dix intonations de base du français*, "The French Review", 40 (1), pàgines 1 a 14, 1966.

[29] els webmasters de cygwin, *Cygwin Information and Installation*. Darrera actualització: 23 de juny de 2006.
http://www.cygwin.com/

[30] Carles Fernández Tena, *Elaboración de Philips Embedded TTS*. Projecte Final de Carrera. UPC, 2005-2006.

[31] Institut d'Estudis Catalans, Secció Filològica. *Aplicació al català dels principis de transcripció de l'Associació Fonètica Internacional*. Joaquim Rafel i Fontanals, Institut d'Estudis Catalans, 1999.

[32] Institut d'Estudis Catalans. *Gramàtica de la Llengua Catalana*. Institut d'Estudis Catalans, 23 d'abril del 2002.
http://www.iecat.net/institucio/seccions/Filologica/gramatica/default.asp

[33] Gabriel Antunes Abrantes i Fernando Pereira, *MPEG-4 Facial Animation Technology: Survey, Implementation and Results*, "IEEE Transactions on Circuits and Systems for Video Technology", 9-2, pàgines 290 a 305, març de 1999.

Resum

Dins del marc del projecte europeu HERMES, al Centre de Visió per Computador de la Universitat Autònoma de Barcelona s'està desenvolupant un agent conversacional animat per ordinador el qual haurà de ser capaç d'interactuar amb l'usuari a través de diferents canals de forma simultània, o, el que és el mateix, parlar, gesticular, expressar emocions... Partint, doncs, d'un software capaç de fer que un model 3D d'un cap humà expressi emocions i parli en anglès, donat un arxiu d'àudio prèviament generat, en el treball que aquí es presenta es duu a terme la recerca d'una eina sintetitzadora de parla a partir de text que permeti fer això mateix en català. En aquest document s'explica el procés seguit per a trobar aquesta eina, la investigació realitzada sobre el funcionament d'ambdues per tal d'entendre-les i poder-hi treballar, així com, finalment, les modificacions realitzades per a fer que aquestes puguin interactuar i generar parla inteligible en català a partir de textos escrits en aquest idioma.

Resumen

Dentro del marco del proyecto europeo HERMES, en el Centre de Visió per Computador de la Universitat Autònoma de Barcelona se está desarrollando un agente conversacional animado por ordenador el cual deberá ser capaz de interactuar con el usuario a través de diferentes canales de forma simultanea, o, lo que es lo mismo, hablar, gesticular, expresar emociones... Partiendo, pues, de un software capaz de hacer que un modelo 3D de una cabeza humana exprese emociones y hable en inglés, dado un archivo de audio previamente generado, en el trabajo aquí presentado se lleva a cabo la búsqueda e investigación de una eina sintetizadora de habla a partir de texto que permita hacer eso mismo en catalán. En este documento se explica el proceso seguido para encontrar dicha herramienta, la investigación relizada sobre el funcionamiento de ambas con tal de entenderlas y poder trabajar con ellas, así como, finalmente, las modificaciones relizadas para hacer que éstas puedan interactuar y generar habla inteligible en catalán a partir de textos escritos en dicho idioma.

Abstract

Within the context of the European project HERMES, the Centre de Visió per Computador at the Universitat Autònoma de Barcelona is developing a computer animated conversational agent which will be able to interact with the user through different channels simultaneously, or, to put it another way, will be able to talk, gesticulate, express emotions... Starting from a software capable of making a human head 3D model show emotions and speak in English, given a previously generated audio file, this project's main objective is the research of a tool capable of synthesizing text to speech that would allow this avatar to speak in Catalan. This document explains the steps followed to search for this tool, the investigation done on how both of them work, in order to understand and be able to work with them, as well as, lastly, the changes and modifications done in order to make them interact and generate intelligible speech in Catalan from texts written in this language.