Bideo-sekuentziak integratzeko teknologiak. Adibide bat

Karrera Bukaerako Proiektua

Ekaitz De la Fuente Paredes

Gainbegiralea:
Iñaki Alegria Loinaz

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

Laburpena

Proiektu hau IKUSI enpresak praktika moduan proposatutako proiektu batean dago oinarrituta. Laburbilduz, beraien beharra, eta praktiken helburua, hainbat eduki multimedia izanda, bideo bat sortuz guztia elkartuko lukeen PC-rako aplikazio baten sorrera izan da (windows-pean funtzionatuko litzatekeena). Aplikazioak erabiltzaileek era errazean erabiltzeko moduko interfaze bat behar zuen izan eta emaitza moduan sortu behar zuen bideoak, formatu eta parametro konkretu batzuk. Ideia horiekin buruan, aplikazioaren sorrerarako pauso guztiak emango dira: teknologia desberdinen azterketa, analisia eta diseinua, interfazearen sorrera eta aplikazioaren garapena, eta azkenik probak.

Gaien Aurkibidea

Irudien Zerrenda

Taulen Zerrenda

1 Kapitulua

Helburuak

Gaien Aurkibidea

1.1 Helburuak

Aipatu den moduan, proiektu hau, IKUSI enpresak proposatutako praktika batzuetan oinarritzen da. Enuntziatua zera zen:

> *"Izenburua:*
> *Bideo digitaleko karrusel sortzailea, eduki multimedia erabilita.*
> *Egin beharrekoa:*
> *Praktikaren helburua, software aplikazio baten diseinu eta sorrera izango da. Hau PC ingurunean windows-pean funtzionatu beharko du, eta eduki multimedia erabilita (hainbat bideo, irudi eta audio formatuak erabilita) bideo-karrusel bat sortzeko gai izango da.*
> *Aplikazioak erabiltzaileari eduki multimedia erabilita informazio kanal bat sortzeko aukera emango dio. Emaitza moduan, aplikazioak bideo fitxategi bat sortuko du, horren formatua Transport Stream MPEG2 izango da, IKUSI-ren COFDM moduladoreek erreproduzitu ahalko dutena, karteleria elektronikoari begira. Aplikazioak edukiak era desberdinean konbinatzeko aukera eta emaitza aurreikusteko aukera ere eskainiko ditu.*
> *Praktikaren atal moduan ere hartuko dira kontuan diseinu eta garapenerako tresna egokienen aukeraketa, baita ere bideo konbertsiorako liburutegien aukeraketa ere."*

IKUSIn, multimedia produktuen artean, "MAC-Home" COFDM (Coded Orthogonal Frequency Division Multiplexing) moduladorea daukate. Hardware honek berez egiten duena, audio/bideo seinalea jaso, modulatu, eta telebista edo monitore batean ateratzen du erreproduzituz, edozein moduladoreren modura, baina hardware konkretu honek gainera, sarrera moduan *usb* portu bat dauka, honen bitartez, aparatuaren firmwarea sartu/eguneratu daiteke. Proiektua gauzatu zuen ideia, portu horri erabilpen gehiago ematea izan zen. Lortu nahi zutena, hardwarearen berezitasunak erabiliz, portu horretan *usb* bat konektatzea zen, eta bertatik edukiak irakurtzea sarrera digitaletik irakurri ordez.

Moduladorearen aldaketak, beraien barne-proiektu bat zen, falta zitzaiena, eduki horiek era errazean sortzeko tresna zen. Tresna hori, produktuaren erosle/erabiltzaileek erabiltzeko moduko aplikazioa behar zuen izan.

Bilatuz gero, aurki daitezke eduki multimedien arteko konbertsioak egiten dituzten aplikazioak, baina zerbait gehiago bilatzen da, alde batetik, beharrezkoa den formatuaren berezitasunengatik (aurrerago ikusiko den moduan) eta bestetik, eduki desberdinak konbinatzeko nahiagatik.

Beraz, helburua, aplikazio baten diseinu eta garapena izango da. Aplikazioak hainbat formatu multimediako edukiak hartuko ditu (bai bideo, audio eta irudi eran) eta interfazearen bidez eduki horiek segida batean ipiniko ditu bideo luzeago bat sortzeko. Segida horretan, bideoak eta irudiak konbinatu daitezkeenez, irudiei "soinu banda" bat jartzeko aukera ematen da, eta irudi bakoitzaren agerpen denbora ere aukeratzeko aukera emango da. Azkenik, behin emaitza-bideoa sortuta, aurreikusteko aukera emango da, kanpoko aplikazioren beharrik gabe.

Behin aplikazioa sortu ondoren, bere funtzionamendua egokia dela ikusteko probak egin ondoren, aplikazioa zabaldu ahal izateko instalatzaile bat ere sortu beharko da (aplikazioa bera, eta bere dependentziak instalatu ahal izateko).

Aipatzen den eran, produktu funtzional bat lortzeaz gain, lan metodologian trebetasuna hartzea (aukeraketa eta erabaki ia guztiak norberak hartuz) eta kasu erreal baten aurrean jartzea izan da helburu ere, produktu baten sorreraren atal desberdinak ikusita, eta neurri handiago edo txikiagoan, atal guztietan parte hartzea.

2 Kapitulua

Plangintza

Gaien Aurkibidea

2.1 Plangintza

Proiektuarekin hasi aurretik, eginbeharreko guztiak, hainbat atazetan banatu dira lana era egokian kudeatzeko asmoz. Azpi ataza hauen artean, proiektuaren kudeaketaren aldea daukagu, honetan, proiektua aurrera nola eraman erabakiko da. Beste alde batetik, proiektuaren, kasu honetan, aplikazioaren analisi eta diseinua ditugu, bertan beharrezko teknologiak, erabiliko den metodologia eta aplikazioaren beraren diseinua landuko dira. Hirugarren alde batetik, proiektuaren garapena aurkituko da, eta azkenik, proiektuaren dokumentazioarekin zerikusia duten aspektu guztiak landuko dira, memoria, eta aplikazioaren eskuliburua adibidez. Beste ataza bat bezala hartu liteke agian, erabiliko diren teknologien ikasketa, eta beharrezkoa den informazioaren bilaketa ere.

2.2 Atazak

Aipatutako atazek, bere barruan hainbat azpi-ataza izango dituzte, era honetan, betebeharrak edo “arazoak”, zati txikiagoetan banatuko dira, eta errazagoa izango da lan egiteko era planifikatzea eta lan egitea ere.

- Kudeaketa
  - Planifikazioa eta betebeharren jarraipena (denbora edo daten kontrola adibidez)
- Analisia
  - Erabilpen kasuen azterketa
  - Teknologien aukeraketa
- Diseinua
  - Klase eta sekuentzia diagramak
  - Interfazearen diseinua
- Garapena
  - Aplikazioaren interfazea
  - Aplikazioaren nukleoa

  - Erabiliko diren kanpo funtzioak garatu
- Probak
  - Kanpo funtzioen probak
  - Aplikazioaren probak
- Dokumentazioa
  - Proiektuaren memoria idatzi
  - Aplikazioaren eskuliburua idatzi
- Informazioaren bilaketa eta teknologien ikasketa
  - Qt-rekin familiarizatu
  - ffmpeg liburutegiekin familiarizazioa
  - Informazio bilaketa

2.3 Lanaren estimazioa

Proiektu hau gauzatzeko orduan, aipatu behar da enpresako praktika moduan egin direla, beraz, lana hiru hilabetetan zehar gauzatu da, baina erabilitako denbora guztia ez da proiektuarekiko erabili. Esfortzua, batez ere teknologien ikasketa eta bilaketetan, eta garapenean izan da handia, gainontzeko atazak, mota honetako proiektu batean aurki daitezkeenaren antzekoak izan dira. 2.1 taulan ikus daiteke estimatutakoa.

Taula 2.1: Esfortzua

Ataza	**Denbora**
Kudeaketa	**10h**
Planifikazioa	10h
Analisia	**10h**
Erabilpen kasuak	4h
Teknologien aukeraketa	6h
Diseinua	**20h**
Klase eta sekuentzia diagramak	10h
Interfazearen diseinua	10h
Garapena	**180h**
Aplikazioaren interfazea	10h
Aplikazioaren nukleoa	150h
Kanpo funtzioak garatzea	20h
Frogak	**40h**
Kanpo funtzioen frogak	20h
Aplikazioaren frogak	20h
Dokumentazioa	**75h**
Memoria idatzi	72h
Eskuliburua idatzi	3h
Informazio bilaketa eta teknologien ikasketa	**80h**
Qt-rekin familiarizazioa	10h
ffmpeg liburutegiekin familiarizazioa	20h
Informazioa bilaketa	50h
Guztira	**415h**

2.4 Planifikazioa eta metodologia

Proiektuaren inguruan, bertako zuzendariarekin elkartu nintzen hasieran gutxi gora behera proiektuaren iraupena eta datei buruz. Praktikak 3 hilabetekoak izan ziren, beraz lana hiru hilabete horietan burutu behar zen, lan metodologia, enpresako ordutegien araberakoa izango zen, beraz egunean 8 ordu gutxi gorabehera.

Hasiera batean, zuzendariak gidatuta, proiektuaren nondik norakoak aztertu ziren, eta hortik aurrera, ikerketa eta analisi lana hasi zen. Ondoren garapen arloa (bitartean ere, egunero ia, informazio bilaketa, ikerketa eta ka-

su batzuetan diseinu aldaketak gertatzen ziren). Garapenean zehar ere, zati desberdinen probak egiten joan nintzen. Azkenik, behin garapena bukatuta, askoz proba gehiago egin nituen, izan ere, kaleratu behar zuten proiektu bat izanda, ondo funtzionatuko zuela ziurtatu behar zen.

2.5 Baliabideak

Proiektua gauzatzeko ez da baliabide gehiegirik behar izan, aski izan da ordenagailu batekin. Garapena egiteko, Windows sistema eragilea erabili da gehien bat, zati batzuk egiteko Linux erabili da makina birtualen bitartez, eta frogak egiteko, Windows desberdinak makina birtualen bitartez.

2.6 Arriskuen analisia

- Gaixotasuna

Deskribapena: Gaixotasun egoera dela eta, lan egunen bat galduko da agian.

Probabilitatea: Ertaina

Eragina: Nahiko altua

Zer egin: Lana denborarekin eramaten saiatu, era honetan, ezbeharren bat gertatzen bada, denbora estran berreskuratu ahalko da.

- Informazioa galera

Deskribapena: Arrazoi desberdinengatik, datuen galerak gertatu daitezke, bai o ordenagailuaren funtzionamendu txarragatik, bai informazioaren gordelekua matxuratu daitekeelako.

Probabilitatea: Ertaina

Eragina: Oso altua, datuen galerak, proiektu osoa atzeratuko luke.

Zer egin: Datuak eta egindako lana gordetzerako orduan, leku batean baino gehiagotan gorde, (adibidez, ordenagailuan, usb memoria batean eta agian interneten, “hodeian”. Era honetan gainera, edozein lekutik egongo da atzigarri).

3 Kapitulua

Analisia

Gaien Aurkibidea

3.1 Betebeharrak

Aplikazioaren betebeharrak hasiera batean enuntziatuaren bitartez era orokorrean finkatu ziren. Sortu behar zen aplikazioak egin behar zuena era orokorrean esanda zera zen, hainbat bideo eta irudi hartuta, bideo-sekuentzia berri bat sortzea.

Ideia, sortutako bideo-sekuentzia hori, enpresak sortzen dituen moduladoreetan erreproduzitzeko modukoa izatea zen. Horretarako, bideoak hainbat berezitasun izan beharko lituzke, aurrerago aipatuko dira zehatzago formatu eta berezitasunen ingurukoak.

Beraz, esan bezala, hasieran aplikazioari irudiak eta bideoak sartuta, bideo-sekuentzia bat sortu behar zen, sekuentzian sartutako irudiekin, gainera, diapositiba pase moduko efektua sortu nahi zen, horretarako erabiltzaileak soinu fitxategi bat sartu behar zuen eta irudi bakoitzaren agerpen denbora.

Erabiltzaileak sartutako fitxategien izenak lista batean ikusi ahalko zituen, eta botoien bitartez lekuz aldatu, nahiago zuen ordenan jartzeko. Botoien bitartez listako elementuak ezabatu ahalko lituzke ere. Azkenik, sortuko zen bideo-sekuentzia gordetzeko lekua ere ipintzeko aukera emango zitzaion.

Proiektuaren garapenean zehar ordea, betebehar horiek aldatzea eskatu zuten, alde batetik berrikuntzak sartzeko eta bestetik, erabilgarriagoa egiteko:

- Lehenengo, erabiltzaileari begira, aplikazioa erabiltzeko erraztasuna handitzea izan zen, beraz, botoak erabiltzeaz gain (fitxategiak aukeratzeko) “Drag and Drop” motako erabilera inplementatu behar izan nuen.
- Sekuentzia osatzeko sartutako elementuak, listan izen baten moduan bakarrik agertu ordez, ondoan irudi/bideoaren *“thumbnail”* edo iruditxo bat izan behar zuen elementuak errazago identifikatzeko.
- Erabiltzaileari, bideo-sekuentziaren fitxategia gordetzeko lekua aukeratzeko ahalmena ematen zitzaion, baina fitxategiaren izena aldaezina zenez (hau inposatutako gauza bat izan zen), aplikazioari, abisuak emateko ahalmena eman behar zitzaion, hau da, gordelekuan, dagoeneko

bideo-sekuentzia bat gorde bada, abisatu egiten da, eta zer egin galdetu, fitxategia zanpatu edo lekuz aldatu.

- Bideoekin lan egiteko orduan, nahiko prozesatze ahalmen behar da, eta ondorioz, prozesua motela gerta daiteke. Hasiera batean bakarrik prozesuaren hasieraz abisatzen zen, eta bitartean, lan egiten ari zela adierazteko adierazle bat agertzen zen pantailan. Esan bezala, prozesua luzea gertatzen zenez, prozesuaren egoerari buruzko informazioa gehitzea erabaki zen. Barra baten bitartez, prozesuaren egoera azalduko zen eta, zenbat fitxategi tratatu ziren uneoro. Era honetan, erabiltzaileak ikusi ahalko zuen era arrazagoan zenbat denbora hartuko luke prozesua, eta amaitzeko zenbat falta zene

- Bezeroaren beste eskakizun bat, emaitzako bideo-sekuentziaren tamaina izan zen. Kasu honetan, erreproduzitzeko erabiliko zen hardwarearen inposaketa. Beraz fitxategiak gehienez *4GB* izan ahalko lituzke. Erabiltzaileari abisatzeko ahalmena inplementatu zen, bideosekuentziaren luzeraren arabera, bideoaren pisua egokia edo ez izango zenez, muga hori zeharkatzen bazen, abisua emango zen.

3.2 Erabilpen kasuak

Aplikazioaren ideia, nahiko sinplea da, ez ordea bere garapena, beraz, erabilpen kasuei dagokionez gutxi batzuk identifika daitezke, gehienak aplikazioaren sarrerekin erlazionatutakoak. Era berean, bi aktore desberdin identifikatu daitezke kasu hauetan partaide direnak: erabiltzailea eta sistema, edo beste era batera esanda, aplikazioa bera.

3.2.1 Erabiltzailea

Prozesua hasi aurretik aplikazioa maneiatu behar da eta beharrezko parametroak sartu, horretaz arduratuko da erabiltzailea.

Irudia(k)/bideoa(k) gehitu:

Interfazearen bitartez, erabiltzaileak nahi dituen fitxategiak sartu ahalko ditu aplikaziora. Bai funtzio horretarako sortu diren botoiak erabiliz eta

baita, fitxategiak zuzenean interfazera eramanez. Kasu honetan edozein karpetatik edukia arrastratu egin behar da. Fitxategiak sartzeak duen muga bakarra formatu batzuk bakarrik onartzen direla da, baina hau erabiltzailearentzat gardena da, aplikazioak berak maneiatzen baitu murrizketa hori.

Irudia(k)/Bideoa(k) kendu:

Interfazearen bitartez listatik elementuak kendu daitezke, horretarako, kendu nahi dena saguarekin aukeratu behar da eta ondoren, dagokion botoia sakatuz listatik kenduko da. Elementuen kontrola eraman ahal izateko, elementuak banan banan kendu beharko dira.

Eduki guztia kendu:

Listan sartutako elementu guztiak kendu nahi badira, hau da, lista berri bat hasi nahi bada, orduan dagokion botoia sakatuz interfazeko lista osoa garbituko da.

Ordena aldatu:

Behin interfazera elementu batzuk sartu direnean bere ordena aldatu daiteke lista erabiltzailearen gustura jartzeko. Horretarako elementuak igotzeko edo jaisteko botoiak erabiliko dira. Saguarekin elementu bat aukeratu behar da eta nahi adina mugitu.

Datuak sartu:

Bideoa sortzeko prozesua gauzatu ahal izateko, ez da aski fitxategiak sartzearekin, hauetaz aparte hainbat datu izango dira beharrezkoak prozesuan zehar parametro moduan erabiltzeko. Datu horietatik bi nahitaezkoak dira eta beharrezkoa ez den hirugarren bat dago. Hauek sartzeko dagokien testu kutxen ondoko botoiak erabiliko dira bi lehenengoetarako, eta hirugarrenean kutxaren ondoko geziekin sartu ahalko da balioa.

Datu hauen artean, audio fitxategi bat sartzea, irudien agerpen denbora eta bideoa gordetzeko helbidea aurkitzen dira.

Bihurketa hasi:

Behin erabiltzaileak beharrezko datu eta elementu guztiak sartu dituenean (horrela ez bada, abisatu egingo zaio), prozesua hasteko botoia sakatuko du, eta prozesua bigarren planoan hasiko da. Prozesua hastean eta honen bitartean, erabiltzaileari abisatuko zaio, baita prozesua bukatzerakoan.

Bihurketa ezeztatu:

Behin prozesua hasita, edozein momentutan erabiltzaileak dagokion botoia sakatu dezake prozesua eteteko. Behin ezeztatua, segundo batzuen ondoren etenaren baieztapena jasoko du.

Emaitza ikusi:

Behin prozesua amaitu ondoren, interfazean botoi bat aktibatuko da, eta horren bitartez, sortutako bideo-sekuentzia ikusi ahalko da kanpo programarik gabe.

3.2.2 Sistema/aplikazioa

Behin prozesua hasten denean, bihurketa hasiko da pasatako fitxategi eta parametroak erabiliz. Sistemak ere, interfazearen bitartez abisuak emango ditu.

Irudi batetik bideoa sortu:

Irudi bat aurkitzen denean, ffmpeg komando egokia exekutatuko da bigarren planoan eta erabiltzaileak sartutako iraupena duen bideo bat sortuko da.

Bideoak elkartu:

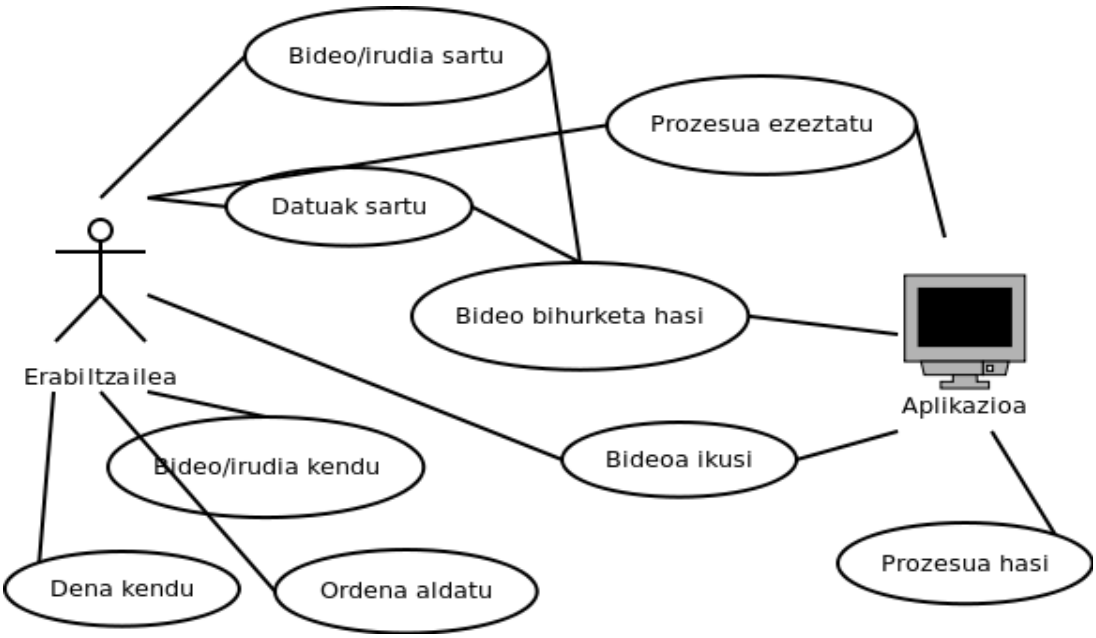
Prozesu hau, bi une desberdinetan abiatuko da, irudi sekuentzia bat sortu behar denean, hau da irudi-bideo desberdinak elkartu behar direnean eta tarteko bideo guztiak elkartu behar direnean bukaerako bideo-sekuentzia sortzeko. *ffmpeg*-ri deiak kateatuz lortzen da.

Bideo bihurketa egin:

Prozesu hau ere bi une desberdinetan abiaraziko da, alde batetik sarrera fitxategi moduan bideo bat aurkitzen duenean, bideo hori era egokian jartzeko, eta bestetik, bukaerako bideo-sekuentzia formatu egokira aldatu behar denean.

Audioa gehitu:

Irudiez osatutako sekuentzia bat daukagunean, erabiltzaileak sartutako audio fitxategia era egokian gehitu beharko zaio.



Irudia 3.1: Erabilpen kasuak

3.3 Teknologien azterketa

Proiektua gauzatzerako orduan, enpresa bitarteko praktika batzuetan egindako proiektua izan denez, erabili beharreko teknologien zati bat aurretik aukeratuta zegoen, hau da, neurri bateraino behintzat, aukeraketa batzuk eginda baitzeuden. Aukeratutako irizpideen artean, objektuetara bideratutako lengoaia erabiltzea programatzerako orduan, C++ hain zuzen ere, Windows-pean funtzionatu behar zuela, hau da, Windowserako aplikazioa izan behar zuela, eta ahal zen heinean, software librearekin sortzea. Honetaz gain, sortu behar zen bideo-sekuentziak hainbat berezitasun izan behar zituen, bai formatuagatik, baita enpresatik inposatutako parametro batzuengatik.

Erabiliko ziren gainontzeko teknologia edo programen aukeraketa, guztiz nire esku geratu ziren, beti ere, aholku batzuk jarraituz. Bi erabaki ziren nagusienak, zer erabili programatzeko eta, zer erabili bideo eta irudiekin lan-egiteko. Programatzeko, C++ ulertu eta konpilatzeko gai izango zen ingurune bat aukeratu behar nuen, era berean, interfazea era errazean sortzeko gai izango zena. Eta bideoekin lan egiteko, formatu desberdinak maneiatzeko gai den tresna beharrezkoa zen, batez eze, Transport Stream formatuarekin ondo konponduko zena.

3.3.1 Formatuak

Aplikazioak maneiatuko dituen fitxategien formatuak.

Bideo formatuak

Bideo digitala, sinplifikatuz, irudi digitalen etengabeko segida bat dela esan dezakegu, bideo digitalaren testuinguruan, hauetako irudi bakoitzari *frame* deitzen zaio, eta segundoko agertzen diren *frame* kopuruaren arabera neurtzen da bideoaren *frame-rate* delakoa. Irudi bakoitza pixelez dago osatuta, eta hauen kantitateak adierazten du irudiaren (eta ondorioz bideoaren) altuera eta zabalera, eta bien artean, bideoaren erresoluzioa sortzen dute. Bideoak duen beste ezaugarrietako bat, *bit-rate* deritzona da, honek adierazten du segundoko zenbat informazio bidaltzen den, eta bideoaren kalitatearekin dago lotuta, eta noski, bideoaren tamainarekin.

Proiektu honetan, ez gara bideo digitalaren muineraino jaitsiko, aipatu-

tako ezaugarri nagusiekin lan egingo da, horren azpian dagoenaz, aukeratuko den aplikazioa arduratuko da, aski izango da *bit-rate*, *frame-rate* eta erresoluzioarekin jolastea. Baina ideia hartzeko, kontzeptu batzuk aipatuko dira.

Codec bat (Kodifikatzaile-dekodifikatzaile), datu fluxu bat barnean duen fitxategi bat transformatzeko gai den espezifikazioa da. Kontzeptu hori bideoaren arlora eramanda, bideo digitala konprimitu eta deskonprimitzeko gai dena da. Normalean erabiltzen diren trinkotze algoritmoek informazio galera txiki bat izaten dute baina beharrezkoak dira bideoak tamaina egokiak izateko. Izan ere, bideo batek informazio kantitate oso handia du bere barnean eta hau maneiatu ahal izateko denbora errealeko trinkotze eta destrinkotze algoritmo hauek sortu ziren. Bideo bat erreproduzitzen den unean fitxategiko informazioa zabaldu egiten da, eta gordetzerakoan ordea, trinkotu egiten da.

Bideoaren munduan beste kontzeptu garrantzitsu bat, edukiontziena da, edo formatu edukiontziena zuzenago. Hauek edozein fitxategi formatu bezelakoak dira, baina barruan hainbat informazioa gordetzeko gai dira: bideoa, audioa, azpitituluak, kapituluak, meta-datuak eta formatuaren araberako sinkronizazio informazioa. Lehenago aipatu diren *codec*-ekin dute erlazio estua, izan ere audio eta bideo pistak trinkotuta gordetzen direlako. Fitxategi bat sortzen denean, pista desberdinak bakoitza bere aldetik kodifikatzen dira eta ondoren elkartu egiten dira formatu bakoitzaren multiplexazio moduari jarraituz.

Hala ere, beharrezkoa den emaitza formatua, *Transport Stream* izenekoak ezaugarri pare bat gehiago ditu kontuan hartzekoak. *MPEG-Transport Stream* formatuak, audioa, bideoa eta sistema eta programa informazio protokoloa bidaltzeko gai den formatua da, hau da, edozein bideok garraiatzen dituen audio eta bideo hariez aparte, informazio gehiago garraiatzeko gai da, izan ere, formatu hau streaming bidez telebista bidaltzeko erabiltzen den formatua da, eta bideo-audio parea eramateaz gain, kanalei buruzko informazioa ere eraman behar du.

Formatu honetan, bideoa eta audioa bakoitza bere aldetik kodifikatzen da, baita gainerako informazioa ere, ondoren guztia multiplexatzen da eta segida bitar bakar batean bidaltzen da. Helmugan, segida hori demultiplexatzen da, eta audioa eta bideoa *PCR* (*program clock reference*) balioaren arabera sinkronizatzen da hasieran bezala uzteko.

Proiektuan bideo formatu batzuk erabili dira, baina nahi izanez gero, formatu gehiago gehitu daitezke, izan ere, ffmpeg-k ia edozein formatu onartu

eta tratatzen baitu. Onartutako formatuak, ohikoenak dira: avi, mpg, mkv, flv, mp4 eta wmv.

Irudi formatuak

Bideoekin gertatzen den era berean, proiektuan ez gara sartzen irudien osaketaraino, bere ezaugarri birekin geratuko gara, altuera eta zabalera, bi hauekin sortzen baita irudiaren erresoluzioa, eta hori da beharko dugun informazioa, gainerakoaz, ffmpeg arduratuko da.

Onartutako formatuak, kasu honetan ere zabaldu daitezkeenak, erabilienak dira: jpg, png, gif eta bmp.

Audio formatuak

Audioari dagokionez, bi formatu onartu dira bakarrik, baina berriro ere, formatu gehiago onartzeko ahalmena daukagu, onartutakoak wav eta mp3 izan dira, erabilienetakoak direlako.

3.3.2 Bideo kodetzea

Bideoa kodetzeko erabiliko diren multimedialiburutegiak. Bideoa eta audioa bereiztuko dira prozesuak errazago gauzatzeko.

Ffmpeg

Ffmpeg audio eta bideoa grabatu, bihurtu (transkodetu) eta streaming egin dezakeen software-aske bilduma bat da. Bere osagai nagusienak ondokoak dira: libavcodec, codec liburutegi oso ahaltsua; libavformat, audio/bideo edukiontzi, multiplexadore eta demultiplexadore liburutegi; eta ffmpeg komando lerro programa, liburutegia maneiatzeko.

Ffmpeg, GNU/Linuxpean garatzen da, baina sistema eragile askotarako konpila daiteke, eta liberatzeko orduan, *GNU Lesser General Public License*, eta *GNU General Public License*-pean egin daiteke (erabiltzen diren liburutegien arabera).

Erabiltzeko komando-lerro bitartez egiten da, eta gauza sinpleak egiteko erraza izanda ere (adibidez, formatua jarrita, berak bakarrik *codec* egokia

aukera dezake), oso gauza konplexuak ere egin ditzake.

MEncoder

Mencoder komando-lerro bitarteko tresna askea da. Bideoa kodetu, deskodetu eta filtratzeko gai da eta *GNU General Public License*-pean argitaratzen dena. Mplayer erreproduktorearen proiektu anaia dela esan daiteke, eta ondorioz honek maneiatu ditzakeen formatu guztiekin lan egin dezake eta honek adina *codec* erabili ere.

Bere gaitasunen artean bideo baten audio edo bideoa bakoitza bere aldetik tratatzea dago, era honetan bideo bat formatuz alda dezake kalitate galerarik gabe, eta gainera hainbat filtro erabili ditzake bideoekin lan egiteko orduan, adibidez eskalatzea, tamainaz aldatzea, biratzea, kolore, saturazio eta beste aspektuen kontrola, etab...

Esan den moduan, *ffmpeg*-ren antzera ondo moldatzen da formatu askorekin, baina *MEncoder*-ekin *Transport Stream* formatuaren tratamenduari buruzko informazio gehiegirik ez da aurkitu, eta ondorioz ez da hau aukeratu.

Gstreamer

Bideo tratamenduaren munduan ezagun den beste tresna bat *Gstreamer* da, hau *pipeline*-etan oinarritzen den framework multimedia bat da, eta komando-lerro bitartez erabili daitekeen arren, aplikazioen kodeetan integratzeko gehiago erabiltzen da. *GNU Lesser Public License*-pean argitaratzen da eta multiplataforma izanda, windows-en erabili daiteke.

Hainbat formatu tratatu ditzake, besteak beste gehien interesatzen zaigun *Transport Stream* formatua hain zuzen ere, baina bere erabilera ez da oso erraza, pipelineen bitartez funtzionatzen duenez ez da oso argia, eta beraz, tresna ahaltsua izan arren, nahiko korapilatsua da erabiltzen.

3.3.3 Audio kodetzea

Audioa kodetzeko erabiliko diren tresnak

Sox

Irudi-sekuentziak lagunduko dituen audioa maneiatzeko, *sox* (SOund eXchange) liburutegiak aukeratu dira, komando-lerroan funtzionatzen duen programa erabilerraza eta ahaltsua baita. Software askea da, *GNU General License*-pean argitaratua eta hainbat plataformatan erabili daiteke, hau ere, komando-lerro bidez. Hainbat formatu tratatu ditzake eta horiekin, hainbat eragiketa burutu, guk, audioak elkartzeko, mozteko eta bolumen aldaketak egiteko erabili dugu.

3.3.4 Programazioa

Aplikazioa garatzeko erabiliko diren tresnak.

Eclipse

Eclipse kode askeko programazio ingurune bat da, berez Javan programatzeko erabiltzen da baina lengoaia desberdinetarako dituen pluginen bidez erraz molda daiteke adibidez C++-en kodea idazteko. Multiplataforma da eta interfazedun erabiltzaile aplikazioak sortzeko erabili ohi da.

Bere atzean erabiltzaile komunitate handi bat dauka, eta honi esker eskuliburu eta tutorialak aurki daitezke, baita sortu daitezkeen zalantzak argitzeko aukera ere. Hasiera batean *Common Public License*-pean argitaratu zen, baina geroago *Eclipse Public License* lizentziara aldatu zen. Edozein kasutan, *Free Software Foundation*-en hitzetan, bi lizentzia horiek ez dira GNU Public License-rekin bateragarriak.

Aplikazioarekiko nire beharrei begira, aukera on bat zen, programatu eta konpilatzeko aukera ematen baitu eta interfazeak sortzeko aukera ere agertzen da Eclipserekin.

Qt

Qt, erabiltzaile interfazedun aplikazioak sortzeko oso hedatua dagoen liburutegi multiplataforma bat da, asko erabiltzen da ere interfazerik gabeko programak garatzeko, baita komando lerrorako programak eta zerbitzarientzako kontsolak.

Qt, software libre eta kode irekiko liburutegi moduan garatzen da Qt

Project delakoaren bidez, bertan hainbat enpresek (Nokia eta Digia besteak beste) parte hartzen dute, hala nola komunitateak ere. Qt *GNU Lesser General Public* (eta honen hainbat moldaketetan) lizentziapean banatzen da.

Qt-k C++ programazio lengoaia erabiltzen du era natiboan, eta beste hainbat programazio lengoaietan erabil daiteke pluginen bitartez. Plataforma nagusietan erabil daiteke eta laguntza handiak ditu atzean. Liburutegiaren APIak (aplikazioen programazio interfazea) berezko hainbat metodo ditu funtzio asko era errazean egiteko, besteak beste: datu baseekin komunikazioa, XML-ren erabilera, sare komunikazioa, fitxategien manipulazioa, harien kudeaketa, etab. Gainera, datu mota tradizionalak ere maneiatzen ditu.

Kodea sortzeko, *IDE* (Integrated Development Enviroment) propioa dauka, Qt Creator izenekoa, eta bertan kodetu daiteke, baita konpilatu, eta interfaze grafikoa sortzeko atal bat dauka Qt Designer izenekoa. Honen bitartez era grafikoan sortu daiteke aplikazio baten interfazea. Ondoren, grafikoki sortutako elementuen kodea automatikoki txertatzen du dagokien klaseetan, eta oinarri horretatik jarraitu daiteke programatzen.

3.3.5 Instalatzailea

Aplikazioa erabilgarriagoa egiteko, bera eta behar dituen fitxategiak banatzeko gai izango zen instalatzailea prestatu behar zen, eta eginkizun horretarako programa batzuk aztertu dira.

NSIS

Nullsoft Scriptable Install System (*NSIS*) windowserako kode askeko instalatzaile sortzaile bat da. Honen funtzionamendua scripten bidez maneiatzen da, eta azkenaldian instalatzaile komertzialen alternatiba moduan asko hazi da, eta programa ezagun askoren instalatzaileak egiteko erabili da. *zlib/libpng* lizentziapena argitaratzen da *NSIS* software askea bihurtuz.

Programa erabiltzeko bi era daude, script-en bidez, eta ondoren hauek konpilatuz (script-etan kopiatu beharreko eta konfigurazio guztiak idazten dira) edo bere interfaze grafikoaren bidez. Era honetan grafikoki adierazten dira parametroak eta berak bakarrik sortzen du script-a ondoren konpilatu ahal izateko. Bere ezaugarrien artean: arintasuna, hizkuntz aniztasuna eta trinkotze ahalmena daude.

Inno Setup

Inno Setup, kode askeko eta Delphi-n idatzitako instalatzaile sortzaile bat da. Aurreko programaren antzera scriptekin lan egiten du instalatzaileak sortzeko orduan. Script hauek eskuz idatz daitezke bere sintaxi bereziarekin edo bere interfaze grafikoa erabil daiteke pausoz pauso gure instalatzailearen script-a sortzeko. Ondoren aski da script hori konpilatzearekin (*Inno Setup* berak egiten du) gure instalatzailearen exekutagarria lortzeko.

Bere ezaugarrien artean ondokoak aurki daitezke: windows desberdinen arteko konpatibilitate handia, pasahizdun eta kodetutako instalazioak, instalazio baldintzatuak, sistema eragilearen erregistro eta INI fitxategiak aldatzeko ahalmena, desinstalazio ahalmena, etab...

3.3.6 Dokumentazioa

Dokumentazioa garatzerako orduan, hainbat tresna eta programa erabili dira, bai idazketarako, baita irudiak tratatu edo formatua emateko. Ondokoak izan dira horietako batzuk.

Gimp

Gimp (GNU Image Manipulation Program) bit mapa formatuko irudien ediziorako programa bat da. GNU programaren zati da eta honen lizentzia publikopean banatzen da, software askea izanik. Hainbat plataformatan erabili daiteke eta kasu honetan, aplikazioan beran eta dokumentazioan erabilitako irudiak maneiatzeko erabili da.

Dia

Diagramak egiteko orduan, dagoeneko ezaguna nuen Dia erabili da UML diagramak egiteko, bai erabilpen kasuak, klase diagramak eta sekuentzia diagramak diseinatzeko. GNOME proiektuaren zati da, eta esan daiteke Microsfot-en Visio aplikazioaren ordezko moduan diseinaturik dagoela.

OpenOffice

OpenOffice kode askeko suite ofimatikoa da, Microsoft-en Office-ren pa-

rekoa dela esan daiteke, bere barnean dituen programak Officeren parekoak baitira. Fitxategientzako bere formatu propioak ditu, baina formatu komertzialak ere onartu eta tratatzen ditu konpatibilitate handia lortuz.

Latex

Latex dokumentazioa sortzeko zuzenduta dagoen testu sortzailea da. Bereziki liburuak eta dokumentu zientifiko edo teknikoak sortzeko erabiltzen da. Makroak erabiltzen ditu TeX errazteko eta oso erabilia da eskaintzen duen kalitate altuagatik. Software askea da LPPL lizentziapena banatua.

Texmaker

Texmaker GPL lizentziapena banatzen den Latex editore bat da, eta honekin lan-egiteko tresna asko eskaintzen ditu programa desberdinen beharrik ez izateko. Dokumentuetan aldaketak egin ondoren pdf moduan konpilatu dezake eta hauek aurrebista moduan ikusteko ahalmena ere badu.

3.4 Teknologien aukeraketa

Bideo eta irudien arloan, hainbat formatu ezagunekin lan egingo da sarrera moduan hainbat formatu onartzen baitira, eta irteera moduan, formatu konkretu bat denez beharrezkoa, Transport Stream izenekoa, irteera formatu bakarra egongo da.

Bideoak tratatu eta maneiatzeko orduan tresna ahaltsu bat behar zen, Windows-pean funtzionatuko lukeena eta komando-lerro bitartez erabiltzeko modukoa, beraz, aukera desberdinak aztertu ondoren, *ffmpeg* liburutegia aukeratu da, erabiltzen errazena delako, eta era berean ahaltsuena ere. Halere, iturburu kodetik konpilatu behar izan da, berez ez dituen formatu konpatibilitate batzuk gehitu behar izan baitira, eta formatu konkretu baten *Transport Stream* formatuak kasu honetan, trataera berezi bat behar izan duelako.

Aplikazioa programatzerako orduan, programazio ingurune bat aukeratu behar nuen, eta alde batetik, ezagun nituenen artetik bat aukeratu nuen, Eclipse hain zuzen ere, baina bestalde, aukera hau zerbait berria aztertu eta ikasteko aprobetxatzea pentsatu nuen, beraz bigarren aukera moduan eta

lankide batek gomendatua, Qt aztertzea pentsatu nuen. Biak aztertu ondoren, eta gutxiago ezagutzen banuen ere, Qt erabiltzea erabaki nuen programatzerako orduan eskaintzen dituen laguntza anitzengatik, batez ere dituen berezko klase eta funtzioengatik. Gainera atzean komunitate handia dauka, eta erraz aurkitu nituen manual eta tutorial dezente.

Instalatzailea sortzeko orduan erabiltzen errazena aukeratu da, izan ere nahiko antzekoak dira instalatzaile sortzaileak, eta berez instalatzaileak egin behar duena nahiko sinplea da, beraz Inno Setup erabili dut sinple eta arinena izateagatik.

Dokumentazioa garatzeko, openoffice erabiltzea aukeratu da, hau proiektua gauzatu ondoren egin beharko dudalako eta dagoeneko nire ordenagailuan instalatuta daukadalako, eta formatua emateko agian errazena ez bada ere, horretaz geroago arduratuko naiz Latex erabiliz editoreren baten bitartez. Kasu honetan ere Texmaker aukeratu dut eskura neukalako eta erabiltzen nahiko erraza suertatzeagatik. Diagramak egiteko, software askea den Dia erabili da, eta gainontzeko irudiak maneiatzeko, Gimp. Ikusten den moduan, ahal den moduan software askea erabiltzen saiatu naiz.

4 Kapitulua

Diseinua

Gaien Aurkibidea

4.1 Sarrera

Analisia egiterako orduan, aplikazioaren betebeharrak azaldu dira, laburbilduz, hauek direla esan dezakegu:

- Aplikazioak interfaze sinple bat behar du izan, gertatzen diren gertaerei buruz eta prozesuaren progresuari buruz informatu behar duena.
- Irudiak, bideoak, audioa eta datuak jasoko ditu sarrera moduan eta bideo bat sortuko du irteera moduan, elementuak sartzeko, bai botoien bidez baita Drag & Drop bidez egin ahalko da. Irudi/bideo horiek lista batean joan beharko dira ondoan irudi errepresentatibo bat dutela.
- Prozesua, erabiltzailearentzat gardena izan behar da, beraz, bigarren plano batean egin beharko da aplikazioaren interfazea oztopatu gabe.
- Behin emaitza bideoa sortuta, gai izan beharko da bideoa aurreikusteko kanpo programarik gabe.

Hasiera batean geneukan ideia, *ffmpeg*-ren iturburu kodea hartzea eta aplikazioaren kodean bertan erabiltzea zen, hau da, *ffmpeg*-ren barne aginduak erabiltzea konbertsioak egiteko eta fitxategiekin lan egiteko, baina aurki daitekeen dokumentazioa oso eskasa da eta ez da lagungarriegia, beraz ideiaz aldatu zen eta ffmpeg komando lerro bitartez erabiltzea erabaki zen. Era honetan, aplikazioak era ezkutuan Windows-eko *cmd* komando-lerroa irekiko du eta *ffmpeg* programari egingo dio deia beharrezko parametroekin. Era berean erabiliko dira beste eragiketa batzuk, adibidez audioarekin lan egiteko *SoX* erabiltzerakoan.

Parametro horiek, erabiltzaileak sartutako elementuak izango dira, baita sartutako datuak ere. Elementuak erabiltzeko, hauen izena gorde beharko da *ffmpeg*-ri pasatzeko, beraz, fitxategi bakoitzaren izen osoa behar da, hau da, fitxategiaren bide osoa (adibidez: *C://Program Files/...*). Baina erabiltzaileari interfazeko listan, fitxategiaren izen laburra bakarrik erakutsiko zaio irudi/bideoaren iruditxoarekin batera.

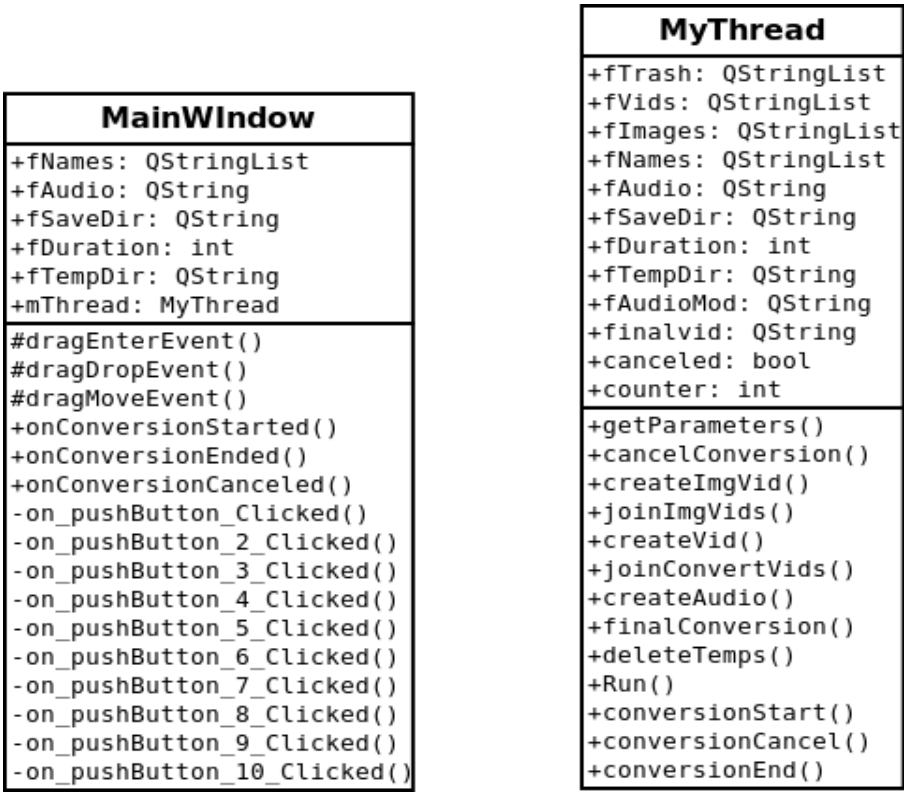
Interfazea diseinatzeko, Qt-ren IDEak duen diseinatzaile grafiko bat erabiliko da, honen bitartez, beharrezkoak diren botoiak, testu kutxak eta abar ipini daitezke, eta berak bakarrik sortuko ditu bakoitzaren klaseak dagozkien iturburu kodearen fitxategietan.

Interfazea ez oztopatzeko, egin beharreko prozesuak, bigarren plano batean egingo dira, horretarako aplikazioak hari bat jaurtiko du, eta hor egingo da benetako lana. Hari honek seinaleak bidali eta jasoz interfazearen hariarekin komunikatu daitekeenez, beharrezko mezuak bidaliko dizkio erabiltzaileak ikus ditzan.

4.2 Klase diagrama

Aplikazioaren egiturari begira, *main* bat eta bi klase nagusi aukeratu dira. Klase nagusi hauek, erabilitako liburutegiaren (Qt) berezko klaseak dira, eta bere barnean hainbat azpi-klase dituzte, batzuk Qt-ren esklusiboak eta beste batzuk, ohiko egitura klaseak. Bi klase horiek ondokoak dira: *MainWindow*, interfazearen zatiaren arduraduna eta *QmainWindow* heredatzen duena eta *MyThread*, *Qthread* heredatzen duena eta bigarren planoko prozesuez arduratzen dena.

Bi klaseak hainbat atributu eta metodo dituzte, gehienak Qt-ren berezkoak direnak. Liburutegiak datu mota eta funtzio bereziak eskaintzen ditu, eta hauek erabiltzea gauzak asko errazten ditu ohiko datu motekin konparatzerakoan. Adibidez, ohiko *String* datu motaren parekoa den *Qstring* klasea aurki dezakegu, eta honek bere metodo propioak ditu kateekin lan egiteko (eduki gehitzeko, konparatzeko, etab...)



Irudia 4.1: Klase diagrama

MainWindow

MainWindow klasearen barruan, fitxategi eta datuei buruzko informazioa gordetzen duten azpi-klaseak aurki daitezke, eta hauen segidan, hari klasearen instantzia. Ondoren, klaseak bidali eta jasotzen dituen seinaleak aurki daitezke (seinale hauek harien arteko komunikazioa ahalbidetzeko sortu dira). Azkenik, erabiltzaileak sartzen dituen datuak maneiatzen dituzten funtzioak aurki daitezke, eta prozesua jaurtitzen duen funtzioa ere.

MyThread

MyThread klasea, aipatu den moduan, prozesuak bigarren planoan gertatzeaz arduratzen den hariari dagokio. Bere barnean prozesuan erabiliko diren datuak gordetzeko atributuak ditu, *Mainwindows*-ek pasatzen dizkionak. Atributuez aparte, prozesuan erabiltzen diren metodoak ere sortu dira, honakoak dira:

- **+getParameters:** interfazean hasteko botoia zapaltzen denean, datuak jasotzeaz arduratzen da. Fnames listan, fitxategi guztien izenak gordetzen ditu; fAudio-n audio fitxategia; *fSaveDir*-en emaitza fitxategia gordeko den helbidea eta *fDuration*-en irudiek izango duten iraupena.

- **+cancelConversion:** edozein unetan interfazeko ezeztatzeko botoia sakatzen bada, prozesua guztiak gelditzeaz arduratzen da, baita memoria libratzeaz eta sortzen diren tarteko fitxategiak ezabatzeaz.

- **+createImgVid:** irudi bat hartzen du eta erabiltzaileak sartutako iraupena hartuz *ffmpeg*-ri egiten dio deia eta iraupen horretako bideo estatiko bat sortzen du. Sortutako bideo hori lista batean (*fImages*) gordetzen du geroago erabiliko baita, eta baita *fTrash* listan ere, bukaeran ezabatu ahal izateko. *Ffmpeg*-ri deitzen zaio hemen.

- **+joinImgVids:** Irudien sekuentzia bat amaitzen denean, metodo honi deitzen zaio eta *fImages*-en gordetako bideoak elkartzen ditu sekuentzia horretako irudi-sekuentziaren bideoa sortuz. Erabiltzaileak audio fitxategi bat sartu badu gainera, audioa gehitzen zaio sekuentziari soinu banda moduan. Bukatzean bideo hau *fVids* listan gordetzen du geroago erabiltzeko eta *fTrash*-en bukaeran ezabatzeko. *Ffmpeg*-ri deia egiten zaio funtzio honetatik.

- **+createAudio:** Erabiltzaileak audio fitxategi bat sartu badu, metodo honek, audio hori dagokion irudi-sekuentziari egokitzen dio. Adibidez, irudi-sekuentziaren iraupena audioarena baino luzeagoa bada, audioa errepikatuko du behar haina aldiz, edo alderantziz, audioa luzeagoa bada, iraupen egokira moztuko du.

- **+createVid:** Bideo bat tratatu behar denean metodo hau deitzen da, eta bideoa formatuz aldatzen da geroago erabili ahal izateko, izan ere bideoak bata bestearekin elkartu ahal izateko, edo beste era batean esanda, konkatenatzeko, mpg formatuan egon behar dute. Kasu honetan ere *fVids* eta *fTrash*-en gordeko da sortutako bideoa. *Ffmpeg*-ri dei egiten zaio.

- **+joinConvertVids:** hau, egiten den azken aurreko prozesua da, izan ere, sortu diren bideo guztiak (*fVids*-en gordeta ditugunak) elkartzeaz arduratzen da. Bukatzean *finalVid*-en eta *fTrash*-en gordetzen da. *Ffmpeg*-ri egiten zaio dei metodo honetatik ere.

- **+finalConversion:** Metodo honek, sortutako bideo guztiak elkartzean sortutako bideoari formatua aldatzen dio *Transport-Stream* formatuan uzteko. Bukatzean, seinale bat bidaltzen dio interfazeari bukatu duela abisatuz. *Ffmpeg:*-ri deitzen zaio azkeneko aldiz metodo honetatik.

- **+deleteTemps:** Metodo hau bi egoera desberdinetan deitu daiteke: lehenengoa, prozesua ezeztatzen bada, une horretan, ordurarte sortutako fitxategi guztiak ezabatu behar dira. Bigarrena, prozesua era egokian bukatu denean, eta kasu honetan ere, sortutako tarte-fitxategi guztiak ezabatu egingo ditu.

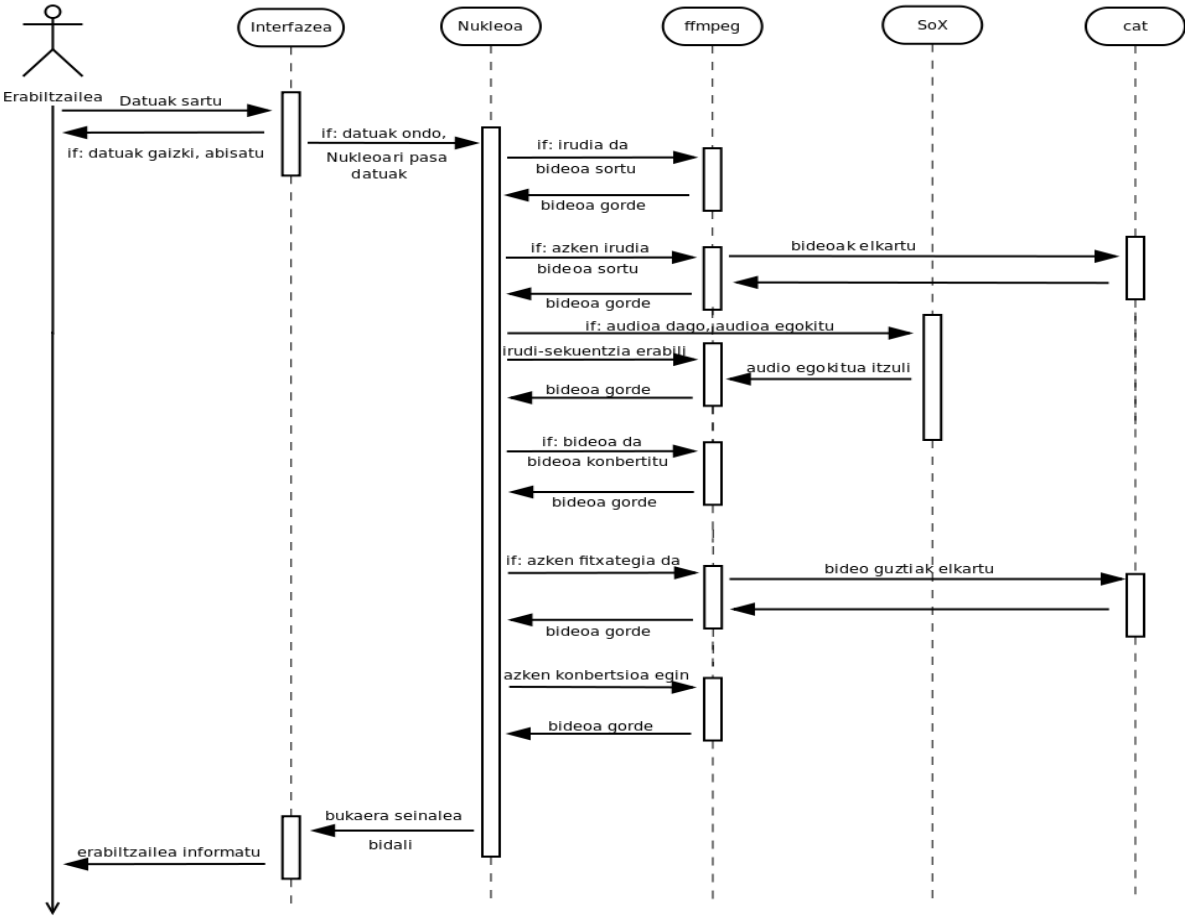
- **+Run:** Prozesu nagusia da, eta haria jaurtitzeaz arduratzen dena. Bere lana, sarrerako fitxategiak banan bana tratatzea da, eta fitxategi motaren arabera metodo bat edo bestea deitzea.

Esan den moduan, bi klase nagusiak *Signals & Slots* (seinaleak eta portuak) bidez komunikatzen dira, Interfazeak hasteko eta ezeztatzeko seinaleak bidaltzen ditu eta hariak bukaera egokiaren edo ezeztatzearen baieztapenaren seinaleak bidaltzen ditu.

Ffmpeg-ri egiten zaizkion dei guztiak, komando lerro bitartez egiten dira, eta dei zehatzak eranskinetan aurki daitezke.

4.3 Sekuentzia diagrama

Aplikazioa irekitzerakoan, erabiltzaileak datu kopuru minimo bat sartu beharko du prozesua aurrera eraman ahal izateko. Datuak ondokoan dira gutxienez: sarrera fitxategi bat, emaitza non gorde nahi duen eta irudien iraupena pantailan (baldin eta irudiak badaude). Behin nahi dituen fitxategiak sartu dituenean, nahi duen ordenan ipini eta datu guztiak sartu dituenean, prozesua hasteko botoia sakatuko du.



Irudia 4.2: Sekuentzia diagrama

Kasu honetan bi gauza gerta daitezke, dena ondo badago, prozesua hasiko da, bestela, dagokion abisua emango zaio erabiltzaileari. Abisua hiru arrazoi hauetako batengatik izango da: 1) beharrezko daturen bat falta delako, 2)

dagoeneko aukeratu duen helbidean emaitza fitxategi bat dagoelako edo 3) sortuko den bideoa handiegia delako emango zaio.

Dena ondo joan bada, interfazeak hari bat sortuko du, eta bertan, bigarren planoan hasiko da prozesua. Erabiltzaileari interfazearen bidez honen hasieraz abisatuko zaio, eta fitxategiak tratatzen joan ahala, abisu gehiago emango zaizkio prozesuaren berri izan dezan.

Harian, eta erabiltzailearekiko era gardenean, prozesua hasiko da. Sartutako fitxategi guztiak tratatuko dira *fNames* lista korrituz, eta mota bakoitzari dagokion ekintza egingo da.

Irudi bat bada, erabiltzaileak sartutako iraupeneko bideo estatikoa sortuko da, *fImages* listan gordeko da eta hurrengo fitxategia aztertuko da, beste irudi bat bada, prozesu bera egingo da, baina hurrengo elementua bideo bat bada, edo tratatutakoa listako azken elementua bada (bi kasuetan irudisekuentzia amaitutzat ematen da), orduan *fImages*-en dauden bideo guztiak elkartu behar dira bideo berri bat sortuz. Hau egiterakoan *fImages*-en edukia ezabatzen da hurrengo sekuentzia baterako prest egon dadin.

Bideoak elkartzeko prozesu horretan, erabiltzaileak audioa sartu badu, honen iraupena lortzen da azpi-prozesu baten bitartez, eta iraupen hori, bideoarenarekin konparatzen da. Audioa motzagoa bada, audio-loop bat egin beharko da behar adina luze egiteko. Audioa luzeagoa bada, orduan moztu egiten da bideora egokitzeko. Bi kasuetan, sortutako bideoa *fVids* listan gordeko da geroago tratatzeko.

Fitxategia bideo bat bada, tarteko formatu batera bihurtzen da geroago besteekin elkartu ahal izateko eta *fVids* listan gordetzen da.

Behin *fNames* lista osoa korritu denean, sortu diren tarteko bideoak elkartu behar dira, horretarako fVids lista korritzen da izen guztiak lortuz, izen hauek ffmpeg-ri deia egiteko parametroak izango dira. Elkartzerakoan, tarteko formatuan egongo dira (mpg, formatu honek zuzeneko elkarketa ahalbidetzen baitu), beraz, azken bihurketa bat egin behar da bideoa *TransportStream* formatuan uzteko.

Hau bukatutakoan, tarteko bideo eta fitxategi guztiak ezabatzen dira (horretarako beren helbideak gordetzen joan dira *fTrash* listan) eta hariak seinale bat bidaltzen dio interfazeari, honela abisua emango dio erabiltzaileari. Azken pauso moduan, orain arte desaktibatuta zegoen botoi bat aktibatzen da interfazean, eta hori sakatuz, sortutako bideoa ikusi daiteke.

Prozesuan zehar abisu gehiago emango zaizkio erabiltzaileari ere. Hasieran hasi dela esaten zaio, eta zenbat fitxategi tratatuko diren ere esaten zaio. Fitxategi horiek bihurtzen joan ahala, grafikoki eta idatziz erakusten zaio prozesuaren aurrerapena.

Salbuespen moduan, beste trataera bat dago, izan ere, erabiltzaileak edozein momentuan prozesua ezezta dezake, beraz hori gertatzen den ikusteko, harian egiten den azpi-prozesu bakoitzaren aurretik atributu bat kontrolatzen da aurrera jarraitu behar den edo ez jakiteko. Atributu hori aktibatuta badago, ez da aurrera jarraituko, eta ordurarte sortutako fitxategi guztiak ezabatuko dira, azkenik ezeztatze zuzenaren abisua emateko erabiltzaileari.

4.4 Erabiltzailearen interfazea

Atal honetan, lehenago analisian ikusitako prozesu eta elementuak interfazean nola antolatu diren ikusiko da (ikus 4.3 irudia). Hiru gauza nagusi beharko dira, alde batetik erabiltzailearentzako kontrolak, hau da, botoiak. Beste alde batetik elementuak sartu eta ikusteko lekua beharko da, eta azkenik erabiltzailea informatzeko leku bat beharko da.

Erabiltzaileak erabiliko dituen elementuak errepresentatzeko lista bat erabiliko da, bertan fitxategien izenak ager daitezen (eta hauen irudi txiki bat ondoan). Hau guztia testu kutxa batean ikusi ahalko da. Bere ondoan fitxategiei dagozkien botoiak agertuko dira. Botoi hauekin fitxategiak gehitu, kendu, lekuz aldatu edo guztiz ezabatu ahalko dira.

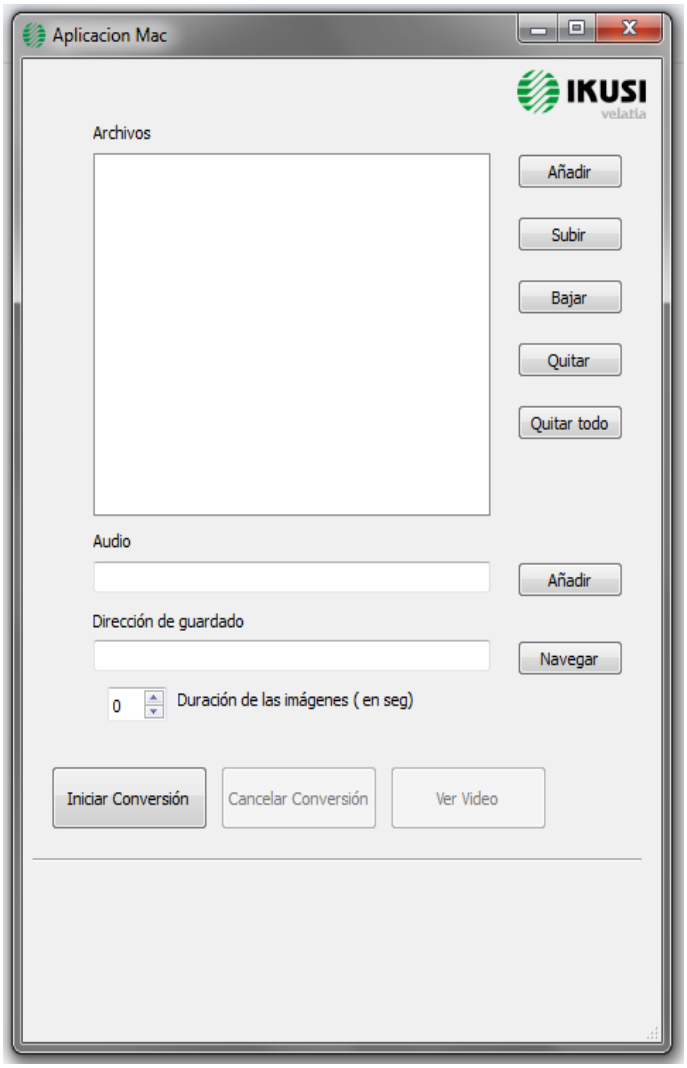
Fitxategientzako kutxaren azpian, eta beste testu kutxa batean, audio fitxategi bat sartzeko lekua egongo da, eta ondoan fitxategia aukeratzeko botoia. Honen jarraian, emaitza fitxategia gordetzeko helbidea aukeratzeko atala daukagu, aurreko kasuan bezala, testu kutxa bat eta helbidea aukeratzeko botoiak osatuta.

Beherago, irudien iraupen denbora aukeratzeko kutxa daukagu, kutxa berezia da, balioak aukeratzeko geziak baititu ondoan, eta bere azpian prozesuari dagozkion botoiak aurkitzen dira. Hiru botoi dira eta, hasiera batean, aplikazioa irekitzerakoan prozesua hasteko botoia bakarrik egongo da aktibaturik. Behin prozesua hasten denean, bigarren botoia aktibatuko da, prozesua ezeztatzeari dagokiona. Azkenik, behin prozesua era egokian amaitu denean, hirugarren botoia aktibatuko da, eta honek sortutako bideoa erre-

produzituko du leiho berri batean.

Guzti honen azpian beste atal batean, abisuetarako gunea aurkitzen da. Bertan interfazeak prozesuaren egoeraz informatuko du, bai hasten denean, baita momenturo tratatzen ari den fitxategiaz, eta baita prozesuaren bukaeraz ere.

Gainera, abisu garrantzitsuak pop-up leihoetan emango zaizkio erabiltzaileari.



Irudia 4.3: Erabiltzaile interfazea

5 Kapitulua

Garapena

Gaien Aurkibidea

5.1 Azalpena

Atal honetan, aplikazioaren garapenari buruzko hainbat xehetasun azalduko dira. Lehenago aipatu den moduan, garapena aurrera eramateko Qt liburutegiak eta bere IDE-a, Qt Creator, erabili da. Hori aplikazioari dagokionez, baina garapen arloan, beste bi atal bereizi daitezke. Alde batetik bideo eta audio bihurketak egiteko komando lerroak, eta bestetik *ffmpeg* liburutegietan egin behar izan ziren moldaketak ere.

5.2 Interfazea garatzen

Interfazea garatzerakoan, Qt Creator-ek duen editore grafikoa erabili zen interfazearen elementuak era errazean sortu eta kokatzeko. Editore honekin, adibidez, botoi bat ipini daiteke interfazean, eta berak bakarrik sortzen ditu botoiaren kode-lerroak dagokion iturburu kodeko klasearen fitxategian.

Beraz, interfazeko elementu guztiak era honetan ipini dira beharrezkoak diren kode-lerro guztiak automatikoki idatziz. Ondoren, Qt Creator barruan sortutako elementuak programatu (behar) dira. Botoi bakoitzak sakatzean izango duen portaera idazteko, lehenago sortu diren instantziak bete beharko dira. Hau interfazeko elementu desberdinentzat automatikoki sortu diren izenak erabiliz eta erreferentziatzen egin da, eta gehienbat Qt-ren berezkoak diren funtzioak erabiliz, beraien artean komunikatzeko edo funtzio bereziak egiteko.

5.2.1 Interfazeko botoiak

Fitxategiak gehitzeko botoiak, Qt-ren metodo bat erabiltzen du fitxategiak kargatzeko, *"QfileDialog::getOpenFileNames"* erabiliz, Windows-en exploratzailearen leihoa irekitzen du, eta bertan nahi adina fitxategi aukeratu daitezke. Honi mugak jartzeko, hau da, bakarrik guk nahi ditugun fitxategiak onartzeko, aski da metodoari parametro moduan fitxategien luzapenak pasatzea (adib: *"Images (*.jpg *.png);; Videos (*.wmv *.mkv *.mpg *.mp4 *.flv *.ts)"*). Era honetan, aukeratzen diren fitxategi guztien izenak (helbide absolutuarekin) gordeko dira *fName* aldagaian. Ondoren, egitura errepikakor baten bidez, lista hori korritu egiten da, eta elementu bakoitza tratatzen

da. Alde batetik, helbide absolutua *fNames* listan gorde da, eta bestetik fitxategiaren izen laburra lortzen da interfazean ipintzeko.

Interfazeko listako elementuak kudeatzeko botoiak, *fNames* listako elementuak lekuz aldatuko dira, eta aldaketa horiek interfazean ikusten dira, lista guztiak momenturo eguneratuta eta koherente mantentzen direlako. Botoi hauen efektua, interfazeko listan aukeratutako elementuarengan dute eragina, eta esan den moduan, listan gora (atzera), behera (aurrera) edo listatik kanporatzen ditu elementuak. Guztia ezabatzeko botoiak, interfazea garbitzen du eta baita fitxategien izen absolutuak gordetzen dituen lista ere.

Audioa eta gordetzeko helbidea aukeratzeko botoiak, lehenago deskribatu den "fitxategiak gehitzeko botoia"-ren antzekoak dira. Windows-eko esploratzailea irekitzen dute, lehenak filtroa erabiliz, audio fitxategiak bakarrik aukeratzen uzten du, eta bigarrenak *"QfileDialog::getExistingDirectory"* metodoaren bitartez, gordetzeko lekua aukeratzen edo sortzen uzten du.

Prozesua hasteko botoiak egoera egokia denean bigarren planoan hari bat jaurtiko du konbertsioa hasiz. Baina horren aurretik hainbat gauza aztertu behar ditu:

1. Dagoeneko emaitza fitxategia existitzen den aztertu behar du, horretarako erabiltzaileak sartutako helbidea erabiliko du, eta bertan "video-final.ts" fitxategia bilatuko du *"Qfile::exists ("../video-final.ts")"* metodoa erabiliz. Existitzen bada, abisua emango zaio eta nahi badu, fitxategi hori zapaltzeko aukera.

2. Prozesua gauzatzeko beharrezkoak diren elementuak aztertuko dira, hau da, listan elementuak daudela, gordetzeko helbidea daukagula eta, irudiak baldin badaude, hauen iraupena sartu dela. Baten bat falta bada, pop-up leiho baten bitartez abisatuko da eta ez da prozesua hasiko.

3. Sortuko den emaitza-bideoaren luzera izango da, lehenago aipatu den moduan, gehienezko pisua 4GB izan daitezke, eta proba askoren ondoren pisua eta luzeraren arteko erlaziora iritsi nintzen. Bideo segundo batek 1,6 MB-ko pisua duenez, gehienez 42 minutuko bideoak sortu ahalko dira. Beraz sortuko den bideo luzera kalkulatzeko, irudi kopurua iraupenarekin biderkatzen da, eta honi, sartu diren bideoen luzera gehitzen zaio. Lortutako luzera 42 minutu baino motzagoa bada, dena ondo doa, bestela pop-up bidez abisua ematen zaio. Bideoen iraupena lortzeko, *ffmpeg*-ren *ffprobe* funtzioa erabiliko da.

Aurreko guztia ondo joan bada, haria jaurtiko da beharrezkoak diren parametroak pasatuz. Aldi berean, prozesua ezeztatzeko botoia aktibatu egingo da.

Prozesua ezeztatu ahal izateko, edozein momentutan sakatu daitekeen botoia daukagu interfazean. Prozesuan zehar hariaren barruan, edozein azpiprozesu hasi aurretik atributu boolear bat erabiltzen da, eta horren arabera aurrera jarraitzen da (ezeztatu ez bada) edo prozesua eteten da. Interfazeko botoia sakatzean, hariari mezu bat bidaltzen zaio eta boolear horri balio aldatu.

Prozesua bukatzen denean, era egokian amaitu bada, interfazeko azken botoia aktibatzen da. Honen bitartez sortutako bideoa erreproduzitu daiteke. Hau lortzeko *ffmpeg*-ren *ffplay* komandoari deitzen zaio, eta honek leiho bat irekitzen du, leiho horretan bideoa ikusi ahalko da.

5.2.2 *Drag & Drop*

Interfazearekin lan egiteko, botoiez aparte, drag & drop bidezko sarrerak ere garatu dira. Era honetan, erabiltzaileak fitxategiak zuzenean hartu ditzake ordenagailuko edozein karpetatik eta interfazera arrastratu ditzake. Baina aipatu den moduan, bakarrik fitxategi formatu batzuk onartzen dira, beraz, era horretan sartutako fitxategientzako ere filtro bat ipini behar da.

Interfazeak elementuak *drag & drop* bidez onartu ahal izateko, alde batetik modu hori aktibatu behar da honako lerroekin:

```
ui->listWidget->setAcceptDrops(true);
ui->listWidget->setDragEnabled(true);
ui->listWidget->setDefaultDropAction (Qt::MoveAction);
```

Beste aldetik, era horretan sartzen diren fitxategiak nola tratatu definitu behar da. Horretarako, bi funtzio sortu behar dira, bata elementuak sartzerakoan zer egingo den erabakitzeko, eta bestea elementuak interfaze gainean uzterakoan zer egingo den jakiteko.

Lehenengo funtzioa, *"void MainWindow:: dragEnterEvent (QDragEnterEvent *event)"* sartzen diren fitxategiez arduratu behar da, eta ondorioz hemen jarri behar da formatuen filtroa. Onartzen diren formatu guztiak definitzen dira parametro batean (*"QRegExp formats(".jpg,.png,.bmp,.gif,.mpg")"*).

Interfazera fitxategiak arrastratzean, hauen informazioa lortzen da, eta bere izenean definitutako formaturik ez badaukate, ez dira onartuko. Informazioa lortzeko *"QList<QUrl>urls = event->mimeData()->urls()"* funtzioa erabiltzen da, eta ondoren lista hori korritzen da eta konparaketa egiten da:

```
QString fName = urls.at(i).toLocalFile();
if (fName.contains(formatos))
event->acceptProposedAction();
```

Bigarren funtzioa *"void MainWindow::dropEvent(QDropEvent *event)"*, fitxategiak interfaze gainean uztean egingo denaz arduratzen da. Kasu honetan fitxategi motaren arabera bi gauza desberdin egin behar dira. Irudia bada, denbora kontagailurako erabiltzen den parametro bat inkrementatzen da, irudiaren miniatura sortzen da eta interfazeko listan ipintzen da izenarekin batera:

```
QPixmap image(itemInfo.filePath());
image.scaled(52,52);
QIcon icono(image);
QlistWidgetItem*listItem= new QListWidgetItem(icono, file-
Name, ui->listWidget);")
```

Fitxategia bideo bat bada, aurreko kasuan bezala *thumbnail* bat sortzen da, kasu honetan irudi generiko bat (thumbnail moduan irudi adierazgarri bat lortzea ezinezkoa litzatekeelako) eta listan jartzen da izenarekin batera. Kasu honetan gainera bideoaren iraupena lortzen da *ffmpeg* erabiliz denbora kontagailuarentzat.

5.3 Fitxategiekin lan egiten

Aplikazioak emaitza bideoa sortzeko prozesuan fitxategiekin lan egiten du, eta ez bakarrik existitzen direnekin, lan egiten duen bitartean fitxategi berriak ere sortzen baititu. Hauekin lan egiteko, aplikazioak fitxategien bide absolutuak hartzen ditu izen moduan, eta izen hauek, bai fitxategiak aztertzeko edo baita berriak sortzeko *ffmpeg, sox* eta *cat* programei pasatzen dizkie.

Prozesuan zehar sortzen diren tarteko fitxategiak gordetzeko karpeta bat behar da, beraz aplikazioa lehenengo aldiz irekitzerakoan karpeta hori sortzen du *"QDir().mkdir(tempDir);"* ordenagailuaren disko-gogorraren erroan (irekitzen den aldi bakoitzean karpeta hori existitzen den edo ez aztertzen du). Esan den bezala, hor gordeko dira tarteko fitxategi guztiak. Fitxategiak sortzeaz *ffmpeg* arduratzen da, bere komando-lerroko deien emaitza modura, beraz, aski da parametro moduan karpeta hori pasatzea.

5.4 Interfaze eta nukleoaren arteko komunikazioa

Interfazea, edo aplikazioaren hari nagusia eta bigarren planoko haria, edo nukleoaren hariaren artean komunikazioa izan ahal izateko, seinaleak eta portuak erabiltzen dira. Horretarako seinaleak definitu behar dira eta jasoko dituzten portuekin konektatu:

connect(mThread, SIGNAL(ConversionStart(bool)), this, SLOT(onConversionStarted(bool)));

connect(mThread, SIGNAL(ConversionEnd(bool)), this, SLOT(onConversionEnded(bool)));

connect(mThread, SIGNAL(ConversionCancel(bool)), this, SLOT(onConversionCanceled(bool)));

connect(mThread, SIGNAL(sendMessage(bool)), this, SLOT(onMessageReceived(bool)));

Prozesuaz arduratzen den hariak seinaleak bidaliko ditu beharrezkoa denean, eta interfazeak jasotzerakoan funtzio desberdinak gauzatuko ditu, aski da funtzio horiek definitzea. Funtzioa hauek, prozesuaren hasieraz, bukaeraz, ezeztatzeaz eta eguneraketaz informatzen arduratuko dira.

Adibidez, prozesuaren egoeraz informatzen duen funtzioak interfazean testua idazten du fitxategiak tratatu ahala, eta aurreratze barra bat eguneratzen du (*"ui->progressBar->setValue(valor);"*) grafikoki lagungarriagoa izateko.

5.5 Azpiprozesuak

Hasiera batean, *ffmpeg*-ren funtzioak aplikazioaren nukleoaren kodean bertan sartzea pentsatu zen, era honetan, bideoak sortzeko eta tratatzeko funtzioak era programatikoan erabili ahalko lirateke. Baina funtzio horiek integratzeko informazioa bilatzerakoan, ez zela gauza erraza izango ikusi nuen. Alde batetik dokumentazioa oso exkasa eta laguntza gutxikoa zen eta bestetik, ez zen oso era eraginkorra. Ondorioz, ffmpeg-ren funtzioak beste era batean erabiltzea erabaki nuen.

Ffmpeg, lehenago aipatu den moduan, komando-lerro bitartez erabiltzen den funtzioa da, beraz, moduren batean, aplikazioak era horretan erabili beharko luke funtzioa. C++ lengoaian, kanpo funtzioak erabiltzeko aukera aurki daiteke, horretarako *system* sistema deia aurki daiteke, beraz alde horretatik begiratu ondoren, *Qt*-n antzeko funtzio batekin eman nuen. Kasu honetan, *Qt*-ren berezkoa den deia da, eta beste funtzio baten modura erabiltzen da. *Qprocess* du izena, eta lehenago aipatutako *system* deiaren parekoa da. Honi kanpo funtzioaren izena (hau da, bide absolutua, edo funtzio hori aurki daitekeen lekua) pasatzearekin nahikoa da hori exekutatzeko. Deitu nahi den funtzioak parametroak behar baditu, *Qprocess*-i pasa behar zaizkio funtzioaren izenarekin batera.

```
QString program53 = "cmd.exe";
QStringList arguments53;
arguments53 <<"/C";
arguments53 <<"C:/aplicacion-mac/sox-14-4-0/sox";
arguments53 <<audioIn;
QString wDir5 = audioInfo.path();
QProcess * myProcess53 = new QProcess(this);
myProcess53->setWorkingDirectory(wDir5);
myProcess53->start(program53, arguments53);
```

Nire kasuan, deia *ffmpeg*-ri da (aurretik aplikazioaren instalazio karpetan kopiatu dena) eta parametro moduan, aldi bakoitzean beharrezkoak izango liratekeenak, adibidez: sarrera eta irteera fitxategiak, funtzioaren parametroak eta hauen balioak, etab...

5.6 *ffmpeg*

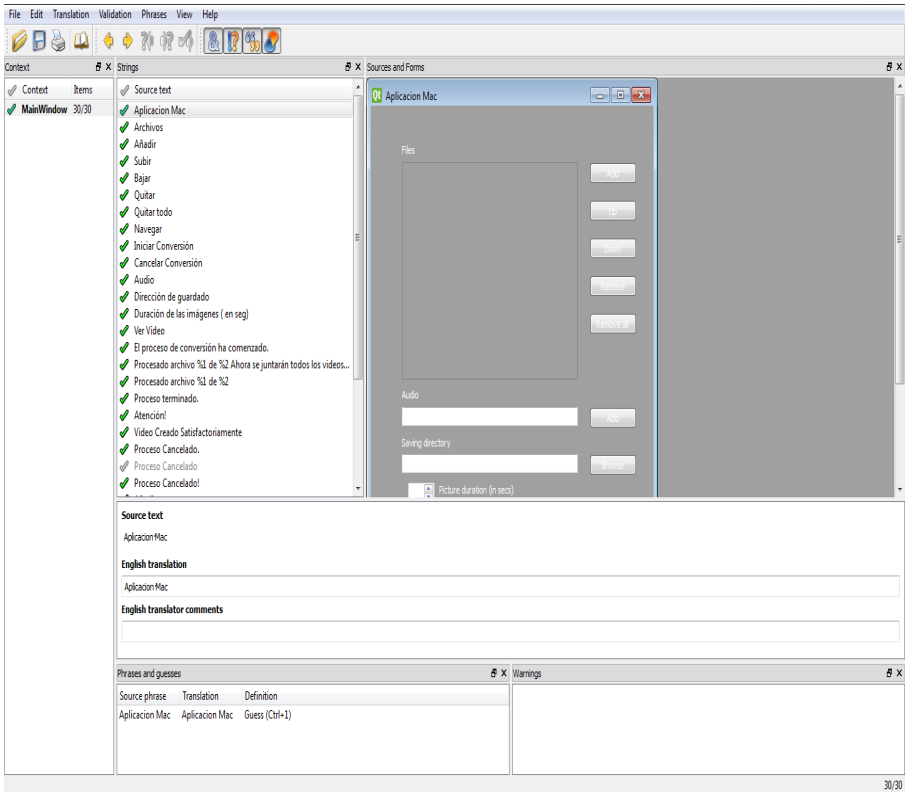
Aurreko atalean aipatu den moduan, *ffmpeg* erabiltzerako orduan, komando-lerroko dei baten bidez egiten da, beraz egin beharrekoa, erabiltzen den aldi bakoitzean komando egokia prestatzea da. Komandoen egitura, gutxi gorabehera antzekoa da beti, aldatzen dena erabilitako parametroak eta hauen balioa da. Aplikazioan zehar lau komando nagusi erabiltzen dira: irudi batetik bideo bat sortzea, edozein bideo formatu hartuta *mpg* formatura konbertsioa egitea, *mpg* formatuko bideoak elkartzea eta *mpg* formatutik *Transport-Stream* formatura bihurketa egitea. Eranskinean sortutako komando-lerroak aurki daitezke.

5.7 Hizkuntza aniztasuna

Azken momentuko ideia moduan, aplikazioa hainbat hizkuntzatan eskuragai izatea bururatu zitzaigun. Hasiera batean ez nuen ideia gehiegirik hau nola egin nezakeen. Bilatuz, aplikazio auto-itzulien funtzionamendua eta atzean zuten ideiaz ikasi nuen. Hizkuntzaz aldatzen diren aplikazioek (aldaketa hori automatikoki egiten denean, ez eskuz hizkuntza aukeratzen denean) egiten dutena, sistema eragilearen hizkuntza detektatzea da, eta horren arabera hizkuntza aldatu. Beraz interfazearen testu elementu guztiak nahi diren hizkuntzetara itzuli behar dira eta erabakitze mekanismo hori garatu.

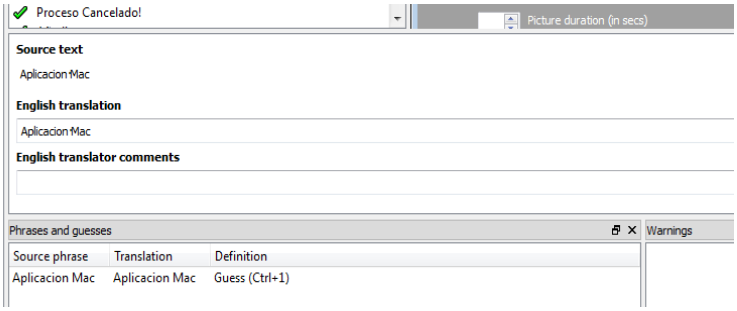
Qt-ek horretarako ere berezko tresna bat dauka, *QtLinguist* izenekoa eta Qt-ren SDK-rekin batera instalatzen den tresna da. Horren bitartez kudeatzen dira hizkuntza desberdinetara moldatu nahi diren testuak. Interfazeko testuak iturburu kodean definitzen dira, eta itzulgarriak izateko era konkretu batean idatzi behar dira: *"tr("testua.")"*. Ondoren behin testu guztiak era egokian ipini ondoren QtLinguist erabiliz testu guzti horiek iturburutik ateratzen dira eta fitxategi berezi batean gordetzen dira (*"aplikazio-testuak.ts"* fitxategia).

Behin fitxategi berezi hori daugakula, QtLinguistekin irekitzerakoan, lehenago definitutako testu guztiak banatuta agertzen dira jatorrizko hizkuntzan. Itzuli nahiko den hizkuntza bakoitzarentzat erregistro bat sortu behar da, eta hitz eta testu bakoitzarentzat automatikoki sortuko da sarrera berri bat. Ondoren aski izango da hitzez hitz (edo esaldiz esaldi) aztertzen joatea eta hizkuntza bakoitzarentzat itzulpen bat idaztea. Hau bukatzean, pro-



Irudia 5.1: Qt Linguist-en interfazea

gramak, exekutatzerakoan erabiliko diren *"aplikazio-testuak.qm"* fitxategia sortuko du.



Irudia 5.2: Qt Linguist-en xehetasuna

Hau egin ondoren, nahi diren hizkuntzetara itzulita izango dugu aplikazioa eta orain erabakitze mekanismoa programatu behar da. Hau aplikazioaren *main*-ean idatzi beharko da. Sistema dei baten bitartez ordenagailuaren sistema eragilearen informazioa jasoko du, eta honen hizkuntzaren kodea jasoko da. Kode hauek hizkuntzen laburdura dira, eta hori izango da erabakitzearen muina. Gure hizkuntza fitxategiak (*".qm"* formatuan) hizkuntza kodearekin izendatuko ditugu, eta sistema deian jasotako hizkuntzaren arabera fitxategi bat edo bestea kargatuko da interfazeko testuak ipintzeko.

```
QApplication a(argc, argv);
QString locale = QLocale::system().name();
QTranslator translator;
translator.load(QString("conversor-mac_") + locale, ":/trans-
lations");
a.installTranslator(&translator);
```

Garapenean zehar, aplikazioa gaztelerara, ingelerara, frantsesera eta italierara moldatu da, geroago erabiliko den hardwarea hizkuntza horietan aurki daitekeelako, baina aipatu den moduan, erraz gehitu daiteke edozein hizkuntza.

5.8 Aplikazioa zabaltzen

Qt-ren *IDE*-a erabiliz aplikazioa programatzen den bitartean *debug* moduan egiten da exekuzio probak eta honen trazak aztertu ahal izateko, eta ondorioz, sortzen diren exekutagarriak IDEarekiko dependenteak dira, hau da, behar hau dute funtzionatzeko. Adibidez, sortutako exekutagarria hartzen badugu eta *IDE*-tik kanpo irekitzen saiatzen bagara ez da irekiko, hainbat liburutegi (*dll* fitxategiak) behar baititu. Beraz, behin aplikazioa sortu denean, *standalone* moduan, hau da, era independentean exekutatzeko gai, bihurtu behar da.

Qt-rekin garatu dugunez, honen liburutegi konkretu batzuk beharko ditu eta horietaz gainera, aplikazioak berak erabiltzen dituen kanpo programak ere beharko ditu funtzionatu ahal izateko, ondorioz, bai liburutegia eta baita erabiltzen dituen programak, aplikazioaren exekutagarriarekin batera banatu beharko dira bere funtzionamendua bermatzeko.

Honetaz, sortuko den instalatzailea arduratuko da, izan ere, instalatzaile batek egiten duena, exekutagarri baten funtzionamendurako beharrezkoak diren fitxategi guztiak leku egokian kopiatzea da, eta nire kasuan hori bera egin dut. Gainera, kopiatzen diren fitxategiak leku konkretu batean egon behar dute, iturburu kodean fitxategi horiei egiten baitzaie erreferentzia eta ondorioz ezin dira lekuz aldatu. Beharrezkoak diren liburutegiak ondokoak dira:

- *QtGui4.dll, QtCore4.dll, mingw10.dll, libgcc_s_dw2-1.dll, ffmpeg.exe, ffprobe.exe, ffplay.exe, sox.exe eta cat.exe*

5.9 Instalatzailea sortzen

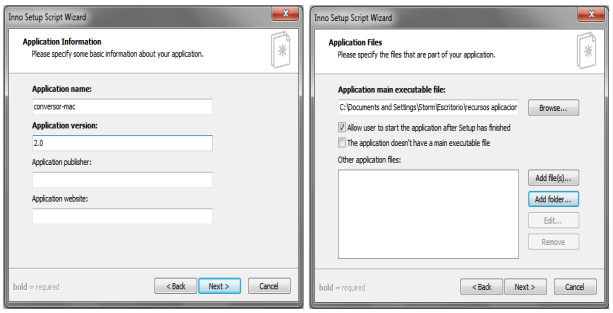
Aplikazioaren instalatzailea sortzeko, hau da, aplikazioaren exekutagarria eta behar dituen dependentziak ordenagailuan kopiatuko dituen instalatzailea sortzeko *Inno Setup* erabili da. Programa honen bitartez, aplikazioa erabili ahal izateko banatuko den exekutagarria sortuko da.

Inno Setup programak pausoz pausoko gida bat eskaintzen du instalatzailea sortzeko, eta hauek jarraituz, bukaeran konpilatuko den script bat sortzen da. Irekitzerakoan instalatzaile berri bat sortzeko botoia sakatu on-

doren, aipatutako pauso-pausokoa agertuko da.

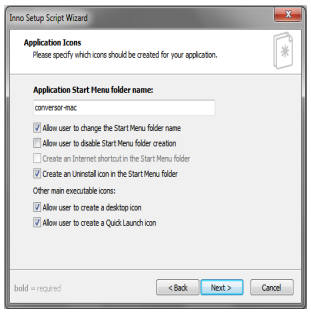
Pauso hauetan, lehenengo aplikazioari buruzko informazioa sar daiteke (gero instalazio wizard-ean agertuko dena), ondoren instalazio helbidea eskatuko da, eta hau aldagarria egiteko aukera emango da.

Hurrengo pausoan exekutagarria bera sartu behar da eta honek behar dituen fitxategiak. Aurretik beharrezkoa eta lehenago aipatutako dependentzia guztiak karpetatan sartu dira prozesua errazteko asmoz.



Irudia 5.3: Inno Setup-en pausoak

Ondoren aplikazioarentzako lasterbideak sortzeko aukera ematen da, eta instalatzerako orduan agertuko diren onartu beharreko lizentziak ipintzeko lekua dago, baita, nahi izanez gero, pasahitz bat jartzeko ere. Modu honetan instalatzaile pribatua sor daiteke.



Irudia 5.4: Inno Setup-en xehetasunak

Azkeneko pausoan sortuko den instalatzaileari ikono bat jartzeko aukera ematen du, eta non gordeko den galdetu. Pauso guztiak jarraitu ondoren, instalatzailearen script-a sortuko da eta konpilatzeko botoia sakatuz, gure exekutagarri lortuko dugu.

6 Kapitulua

Probak eta Hobekuntzak

Gaien Aurkibidea

6.1 Probak

Garapen prozesuan zehar hainbat proba desberdin egiten joan dira atal desberdinak bukatu ahala horien funtzionamendua egokia zela ikusteko eta aplikazioaren beste atalekin batera ondo funtzionatzen zuten ikusteko, adibidez interfazeko elementuen lista. Hau garatu garatzen lehenetariko gauza izan zen eta eraman behar duen kontrola probatzeaz gain, sartutako elementu desberdinen *thumbnail*-en sorrera egokia ere probatzen joan da. Tarteko proba horietaz aparte, bi gauza nagusiki probatu dira ahalegin handiekin, erabilitako *ffmpeg* komandoak eta behin garapena bukatu ondoren, aplikazio osoaren portaera.

Ffmpeg-ri dagokionez, erabili diren komando-lerro desberdinak hainbat bideo eta irudi formatuekin probatu dira beraien zuzentasuna ikusi ahal izateko, gainera, era horretan probatutz, ez zen aplikazio osoa jaurti behar. Proba hauek egiteko, komando-lerroak zuzenean idatzi dira Windows-en *cmd* komando interpretatzailean eta horrela eginda gainera pantailan komandoaren *feedback*-a erakusten du eta hutsegite baten aurrean, zergatia aurkitzen saiatu daiteke.

Aplikazioari dagokionez, mota guztietako probak egin dira akats posibleak aurkitzeko ideiarekin, beraz irudi eta bideo mota desberdinak erabili dira probetarako, eta hauen artean adibidez tamaina oso handiko irudiak, kalitate eta erresoluzioa handiko bideoak eta abar erabili dira. Bestalde, luzeera desberdinetako bideoak sortzeko probak ere egin dira, adibidez bideo motzak, edo irudi askoko bideoak edo irudi gutxikoak baina luzeera handikoak. Beraz esan daiteke proben aldetik lan handia egin dela eta ondo pasa dituela kasu guztiak.

6.2 Oinarrizko hobekuntzak

Aplikazioa, sortu den moduan guztiz erabilgarria da, eta egin behar duena egiten du era egokian, baina helburu zehatz batekin sortu denez, xede hori betetzen du. Beste era batera esanda, formatuen multzo zehatz bateko formatuak sarreratzat hartuta, irteera bideo konkretu bat sortzen du. Hobekuntza moduan, helburua zabaltzea izango litzateke, eta bideoak elkartzeko aplikazio orokorrean bihurtzea litzateke. Atal honetan, hobekuntza edo aldaketa horietako batzuk azalduko dira.

6.2.1 Formatu berriak irteeran

Lehenago aipatu den moduan, aplikazioa helburu oso zehatz baterako sortu da, eta ondorioz honen irteera oso mugatuta dago. Sortzen den irteera bideoak, alde batetik izen finko bat dauka (hau hardwarearen inposaketa izan zen) eta bestetik *Transport Stream* formatuan sortzen da, erresoluzio eta bitrate konkretu batekin. Hobekuntza moduan, irteerako bideoarentzako aukera gehiago eskaintzea litzateke. Interfazea egokituz, formatua, *codec*-a, erresoluzioa eta izena hautatzeko aukerak sar litezke, eta prozesua hasterakoan, parametro finko batzuk erabili ordez, erabiltzaileak sartutakoak erabiliko lirateke. Aski izango litzateke interfazeko balioak irakurtzea eta ffmpeg-ren komando lerroa horiekin betetzea.

Era honetan aplikazioa orokortuko litzateke eta edozeinek erabili ahalko luke irudiak eta bideoak erabiliz era oso errazean bideo-sekuentzia bat sortzeko, nolabait, *ffmpeg*-ren interfaze grafiko baten modukoa lortuz, baina bideoak elkartzeko gehigarriarekin.

6.2.2 Formatu berriak sarreran

Gaur egun mundu digitalean eduki multimedia gordetzeko formatu asko aurki daitezke, aplikazioari begira, onartuko ziren sarrera formatuei begira gutxi batzuk erabiltzea aukeratu zen, erabilienak edo zabalduenak hain zuzen eta gauza bera gertatu zen irudi eta audio formatuekin. Printzipioz formatuak mugatu ziren, hauek onartzeko iturburu kodean definitu behar zirelako, baina berez edozein formatu hartu liteke sarrera moduan, beti ere ffmpeg-ek onartzen dituenen artean (asko da esatea agian, baina ia existitzen den edozein

formatu kudeatzeko gai da).

Ondorioz, hobekuntza sinpleena litekeena, sarrera formatu gehiago gehitzea litzateke. Horretarako aski litzateke formatu berri horik erazagutzea iturburu kodean interfazeak onartu ahal izateko, gainerakoaz ffmpeg arduratuko bailitz.

6.3 Hobekuntza aurreratuak

Lehenago aipatutako hobekuntzen funtzioa aplikazioari funtzionalitate gehiago emateko egin litezkeenak izan dira, baina horretaz aparte, aplikazioa erabilgarriagoa egiteko ideiak ere hartu daitezke eta hobekuntza bihurtu.

6.3.1 Elementuen aurrebista

Aplikazioak emaitza bideoa sortzen duenean, interfazean botoi bat aktibatzen da, eta honen bitartez (*ffmpeg*-ren laguntzaz berriro ere) bideoa ikusi daiteke leiho berri batean. Ideia hori hartuta, gauza bera egin liteke sarrerako elementuentzat.

Erabiltzaileak berak nahi dituen fitxategiak sartuko ditu interfazera, eta irudien kasuan, listako izenarekin batera *thumbnail* txiki bat agertuko da irudia errazago identifikatzeko, baina hala ere ongi legoke agian hori zabaltzea, eta nahi izanez gero elementu bat aukeratzea eta irudia bada, irekitzea, bideoa bada aurreikusi ahal izatea eta audioaren kasuan gauza bera.

Hori egiteko aski litzateke dagoeneko irteera bideoarentzat sortu den funtzio berdina erabiltzea, eta era horretan lagungarriagoa egingo litzateke aplikazioa.

6.3.2 Bideoaren iraupena

Aplikazioaren murrizketetako bat hartuko dut berriro hobekuntza batentzako ideia moduan. Berez murrizketa da, lehenago aipatu den moduan, erreproduzitzeko erabiliko den hardwareak gehienez tamaina zehatz bateko fitxategiak onartzen dituelako, baina berez, edozein tamaina edo iraupeneko bideoak

sortu daitezke. Aski izango litzateke iraupenaren murrizketa kentzea edozein bideo sortzeko (beti ere hardwarearen menpe, hau da, disko gogorra betetzen bazaigu, ezingo dugu bideo gehiagorik sortu).

Horretaz aparte, lagungarriagoa egiteko gainera, elementuak sartu ahala, sortuko litzatekeen bideoaren iraupena denbora errealean erakutsi litzaioke erabiltzaileari ardura handiagoa izateko bideoaren sorreran.

Beste hobekuntza bat, sarrera moduan pasatako bideoak aurreikusteko aukera jartzea izango litzateke, eta baita irudi eta audioarentzat, era honetan, sartutako edozein elementuren aurrebista bat izango genuke.

7 Kapitulua

Ondorioak

Gaien Aurkibidea

7.1 Ondorioak

Karrera amaierako proiektu hau, enpresan egindako praktika batzuetan egindako lanari dagokio eta ondorioz, helburua han sortutako enuntziatua zen. Proiektua beraz, bideo eta irudiak hartuta, guztiak elkartzea eta beraien hardware berezi batean erreproduzitzeko moduko bideo bat sortuko zuen aplikazio bat egitea zen.

Hasieran ipinitako helburu guztiak bete egin dira, aplikazioak hainbat formatuko edukia onartzen du, eta irteeran hauek guztiak elkartzen dituen bideoa sortzen du. Gainera, garapenean zehar sortutako ideia berrien azpihelburuak ere betetzen joan dira aplikazioa hobetuz, beraz proiektua arrakastatsua izan dela esan daiteke.

Edozein aplikazio hobetzeko aukera beti existitzen da, beraz ideia horrekin buruan eta etorkizunari begira, aplikazioaren gainean aldaketak egiteko aukera ere sortu nuen. Horretarako, atal desberdinak lantzeko ingurune egokiak sortu nituen, alde batetik aplikazioa programatzeko ingurunea windowspean sortutako alegiazko makina batean (bertan *Qt*-ren *SDK*-a (*Software Development Kit*) instalatuz eta beharrezko kanpo programak ere prestatuta utziz), eta bestetik, *ffmpeg* era errazean konpilatzeko ingurunea Ubuntupean lan egiten duen beste alegiazko makina batean (kasu honetan ere, konpilaziorako toolchain-a eta dependentzia guztiak prest utzita). Azken honetan gainera, konpilazioaren prozesua era automatizatuan, sortutako script baten bidez.

Proiektuaren garapenaren helburuetatik kanpoko helburuak ere bete direla esan daiteke, aplikazioaren sorrerak hainbat alorretako ezaguerak erabiltzea eskatu baitu. Era honetan karreran ikasitako kontzeptu eta gauza asko erabili dira; eta era berean berriak ere ikasi, beraz alde horretatik ere arrakasta izan dela uste dut. Gainera, proiektua lanpostu batean garatzeak, lan munduaren ezagutza ere eman dit, eta nolabait, "mundu errealean" nola lan egiten den ikastea prozesu desberdinetan murgilduz.

Planifikazioari dagokionez, proiektua gauzatu ondoren ikusi ahal izan da, erabilitako orduak gutxi gora-behera bete direla, zaila da zehazki esatea non desbideratu diren gehiago egindako kalkuluak, baina batez ere, garapenean eta informazio bilaketan izan da, izan ere, bilaketa eta ikasketa, konstantea izan da proiektuaren garapenean zehar, uneren batean zerbait nola egin ez nekienean, horri buruzko informazioa bilatu behar bainuen eta izandako arazoa konpontzen ikasi ere.

Bukatzeko esan behar dut, egindakoarekin pozik geratu naizela, zerotik hasita aplikazio funtzional bat sortzea, eta etorkizun hurbil batean aplikazioari erabilera emango zaiola jakitea, hau da, sortutakoa erabilgarri izango dela sentsazio ona sortzen du.

I Eranskina

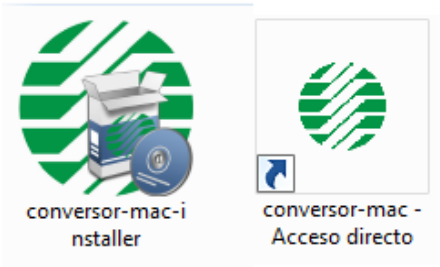
Erabiltzailearen eskuliburua

Gaien Aurkibidea

Hau lehenengo apendizea da.

I.a Aplikazioa instalatzen

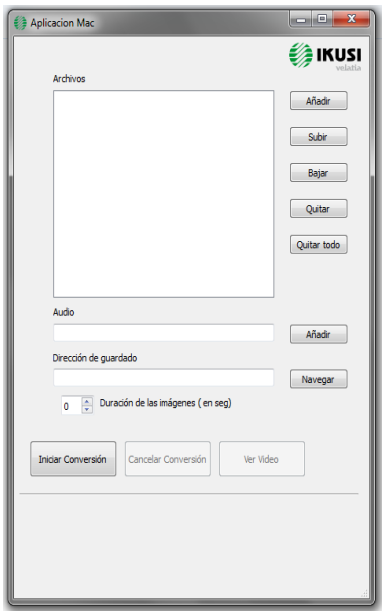
Aplikazioa erabili ahal izateko, lehenengo instalatu behar da. Horretarako, Aplikazioaren instalatzailea erabili behar da. Programa honek instalazio laguntzailea irekiko du, eta eskaintzen diren pausoak jarraitu beharko ditu erabiltzaileak. Pauso hauetako batean, erabiltzaileari mahaigainan aplikazioaren lasterbidea sortzeko aukera ematen zaio (sortzen ez bada, *"C:/Aplicacion-mac/conversor-mac.exe"*-n aurkitu ahalko da) eta behin instalazioa bukatuta, aplikazioa irekitzeko aukera ematen da ere.



Irudia I.1: Instalatzailea eta lasterbidea

I.b Aplikazioa erabiltzen

Erabiltzaileak aplikazioa erabili nahi duen bakoitzean lasterbidearen ikonoa sakatu beharko du eta aplikazioa ireki egingo da:



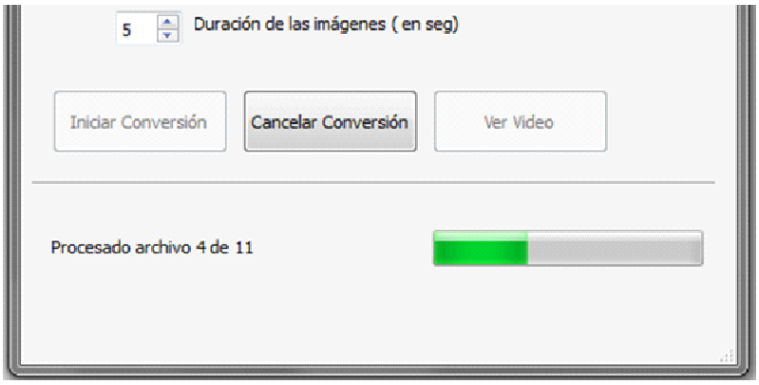
Irudia I.2: Aplikazioa

Behin aplikazioa irekita erabiltzen hasi daiteke. Edukia gehitzeko bi era daude, lehenengoa, zuzenean ordenagailuko edozein karpetatik fitxategiak arrastatzea da, (kasu honetan, onartzen ez diren fitxategiak ez dira gehituko interfazeko listara. Bigarren aukera, Gehitzeko botoia erabiltzea da.

Behin nahi adina irudi sartu ondoren, hauen ordena aldatu ahalko du eskuinaldeko igotzeko eta jaisteko botoiak erabiliz, aski izango da listako elementu bat aukeratzea bere posizioa aldatzeko. Nahi izanez gero elementuak kendu daitezke, listatik aukeratu eta kentzeko botoia sakatuz, edo lista guztia ezabatu nahiko balitz, dena ezabatzeko botoia sakatuta lortu daiteke.

Erabiltzaileak gainera audio fitxategi bat sartu ahalko du irudiekin sortutako bideoari eransteko. Horretarako dagokion testu kutxaren ondoan dagoen Gehitzeko botoia sakatu beharko du eta windows-en exploratzailearen bitartez nahi duen fitxategia aukeratuko du. Honetaz gain, irudi bakoitzaren agerpen denbora ere adierazi beharko du. Hau egiteko, zenbaki kutxaren ondoan agertzen diren geziez baliatu daiteke.

Azkenik, sortuko den emaitza bideoa non gorde aukeratu beharko du erabiltzaileak. Testu kutxaren ondoko botoia sakatuz karpeta aukeratzeko leihoa irekiko da eta honen bitartez aukeratu ahalko da. Ondoren konbertsioa hasteko botoia sakatu behar da eta zerbait gaizki dagoen abisurik jasotzen ez bada prozesua hasiko da eta interfazearen azpialdean prozesuari buruzko informazioa agertzen joango da.



Irudia I.3: Aplikazioaren xehetasuna

Prozesua bukatzerakoan pop-up baten bitartez honen amaieraz informatuko da, eta honekin batera interfazean botoi berri bat aktibatuko da. Honen bitartez sortu berri den bideo ikusi ahalko da horretarako irekiko den leiho berri batean eta kanpo programarik erabili gabe. Bideoaren fitxategia lehenago sartutako helbidean aurkitu ahalko da *"video-final.ts"* izenarekin.

Prozesuaren edozein momentuan, nahi izanez gero ezeztatu ahalko da eta prozesua eten. Ezeztatzeko botoia sakatzearekin aski izango da eta segundo batzuen ondoren ezeztatzearen baieztapena jasoko du erabiltzaileak.

Prozesua hasteko botoia sakatzerakoan abisuren bat jasotzen bada, pop-up batean azalduko da gaizki joan dena erabiltzaileak aldatu dezan. Sortuko den bideoa luzeegia den kasuan gainera, zenbat denboran den luzeegia azalduko da.

II Eranskina

Erabilitako komando-lerroak eta konpilazioa

Gaien Aurkibidea

II.a *ffmpeg*

ffmpeg-ri deiak egiteko komandoen egitura nahiko sinplea da, lehenengo sarrerako fitxategia (edo fitxategiak) jartzen da eta ondoren irteerako bideoa sortzeko kontuan hartu beharreko parametroak eta hauen balioak ipini behar dira. Azkenik irteerako bideoaren izena jarri beharko da.

Komandoak ikusita gauza erraza dela iruditu daiteke, baina parametro egokiekin emateko proba ugari egin dira konbinaketa zuzenekin eman arte, batez ere azkeneko komandoan, Transport Stream fitxategia sortzen duena.

Irudi batetik bideoa sortu

Komando hau, aplikaziora sartzen den irudi bakoitzarekin exekutatuko da, eta egiten duena, erabiltzaileak aukeratutako iraupeneko bideo estatiko bat sortzea da. Ondoren, irudien sekuentzia bukatzean, sortutako irudi-bideo guztiak elkartuko dira tarteko bideo bat lortuz.

ffmpeg -loop 1 -i jatorria -t iraupena -s zabaleraXaltuera -aspect 4:3 -sameq -y helburua

-loop 1: jatorrizko irudia behin eta berriro erabiltzen du
-t iraupena: bideoaren iraupena, erabiltzaileak emanda.
-s zabaleraXaltuera: bideoaren erresoluzioa
-aspect 4:3: bideoaren proportzioa
-sameq: kalitatea mantendu
-y: gainidatzi

Bideoak elkartu

Komando hau bideoak elkartu behar direnean erabiliko da. Bai irudiekin sortu diren bideoak elkartzeko, eta baita azken aurreko pauso moduan tarteko bideo guztiak elkartzeko.

cat jatorria1 jatorria1 — ffmpeg -f mpeg -i - -vcodec copy -acodec copy -y helburua

-f mpeg: helburuaren formatua definitzeko
-vcodec copy: jatorrizko bideo-codec berdina erabiltzeko
-acodec copy: jatorrizko audio-codec berdina erabiltzeko

Audioa gehitu

Komando hau, irudiekin sortutako bideo-sekuentzia bati audio gehitzeko erabiliko da. Hau, erabiltzaileak audio fitxategi bat sartu duenean bakarrik erabiliko da, bestelako kasuetan sekuentzia mutuak izango dira.

ffmpeg -i jatorrizko-bideoa -i jatorrizko-audioa -shortest -acodec mp2 -ac 2 -ab 192k -ar 44100 -vcodec copy -y helburua

-shortest: iraupen moduan, bi jatorrietako motzena hartzeko
-acodec mp2: mp2 audio codec-a erabiltzeko
-ac 2: 2 audio kanal erabiltzeko
-ab 192k: audioaren bitrate-a
-ar 44100: audioaren frekuentzia

Bideoa bihurtu

Komando hau, erabiltzaileak sartutako bideoekin erabiliko da, izan ere, bideoak elkartu ahal izateko ezin dira edozein formatuan egon. Era egokian konkatenatu ahal izateko mpg formatuan egon behar dute bien bit-segidak ondo elkartzeko. Kasu honetan formatua helburuaren izenarekin batera jartzen da.

ffmpeg -i jatorria -r 25 -s zabaleraXaltuera -aspect 4:3 -sameq -y helburua.mpg

-r 25: bideoaren bitrate-a

Bideoa erreproduzitu

Komando hau behin bideoa sortu denean erabiliko da eta bideoa leiho berri batean aurreikusteko balio du.

ffplay jatorria –autoexit

-autoexit: bideoa bukatzean leihoa bakarrik ixteko

Azken bihurketa

Komando hau behin erabiltzen da konbertsio prozesuan zehar eta garrantzitsuena da. Honekin Transport Stream bideo bat lortuko dugu behar dituen doikuntza guztiekin.

ffmpeg -i jatorria -r 25 -vcodec mpeg2video -b 6000k –minrate 6000k -maxrate 6000k -bufsize 1835k -muxrate 13500k -mpegts_pmt_start_pid xxxx -mpegts_user_pcr_pid yyyy -streamid zzzz -streamid wwww -acodec mp2 -ab 192k -s zabaleraXaltuera -sameq -trellis 2 -g 45 helburua

-vcodec mpeg2video: mpeg2 bideo codec-a
-b 6000k: bitrate-a
-minrate 6000k: bitrate minimoa
-maxrate 6000k: bitrate maximoa
-muxrate 13500k: multiplexazio ratioa
-bufsize 1835: buffer tamaina

Parametro guzti hauen konbinaketarekin, *Transport Stream*-entzat beharrezkoa den *CBR* (Constant Bitrate) Bitrate konstantea lortzen da. r

-mpegts_pmt_start_pid xxxx: pmt taularen pid-a (hari desberdinen edukia gordetzeko)
-mpegts_user_pcr_pid yyyy: pcr-ren pid-a (audio eta bideoaren sinkronizaziorako)
-streamid zzzz: bideo hariaren pid-a
-streamid wwww: audio hariaren pid-a
-trellis 2 –g 45: irudiko “soinua” kentzeko parametroa

II.b *SoX*

SoX erabiltzeko komandoak ffmpeg-ren oso era antzekoan erabiltzen dira, eta azken honek audioa bakarrik erabiltzeko oso egokia ez delako erabili dira.

Audio leuntzea

Komando hau audioa leuntzeko erabiliko da. Kasu batzuetan, erabiltzaileak sartutako audioa motza bada, hau behin eta berriz errepikatu beharko da bideoaren luzerara moldatzeko. Kasu horietan audioaren mozketa gogorra ez izateko, hasieran eta bukaeran bolumena jaisten da leuntzeko.

sox jatorria helburua fade h 2 0 2

-fade h 2 0 2: leuntzea kurba sinusoidala erabiliz, hasieratik 2 segundo eta beste 2 bukaeran

Audio leuntzea eta mozketa

Audio fitxategia moztu behar denean erabiliko da komando hau.

sox jatorria helburua trim 0 iraupena fade h 2 0 2

-trim 0 iraupena: audioa moztu hasieratik iraupena parametroak duen balioraino

II.c Konpilazioa

Bideoekin lan egiterako orduan *ffmpeg* liburutegiak izan ziren aukeratuak gure behar guztiak asetzeko gai zirelako, edo hori pentsatu genuen hasiera batean. Baina garapenean zehar irteera formatuak inposatutako murrizketa baten ondorioz, *ffmpeg* motz geratu ziren.

Izan ere, lehenago aipatu den moduan, *Transport Stream* formatuak bere ezaugarriengatik nahiko berezia da. Bere barnean audio eta bideo hari parea eramateaz gain hainbat informazio eraman dezake ere, eta hari desberdinak sinkronizatu ahal izateko hainbat identifikatzaile daramatza hauekin batera.

Transport Stream formatuak identifikatzaile (pid) bat behar du bideoarentzat, bat audioarentzat, bat programarentzat (informazioa bidaltzeko) eta azken identifikatzaile bat behar du PCR-arentzat (Program-Clock-Reference).

Ffmpeg-rekin *Transport Stream* formatuko bideo bat sortzen dugunean ondo sortzen ditu beharrezkoak diren multiplexazioak eta harien banaketa, baina hauei pid-ak ezartzeko orduan, pid berdina ematen die bideo hariari eta PCR-ari. Era honetan sortzen den bideoa probatzerakoan ondo ikusi daiteke, eta era egokian erreproduzitzen da ordenagailuan, baina ez da ahaztu behar sortuko diren bideoak hardware berezi baten bidez ikusiko zirela, eta honek pid desberdinak behar zituen bideoarentzat eta pcr-arentzat.

Ondorioz, zerbait egin behar zen hau konpontzeko. Aukera desberdinak aztertu ondoren, errazena, *ffmpeg*-ren iturburu kodea hartzea eta aztertu ondoren gure nahietara moldatzea izan zen. Alde batetik, arazoa *Transport Stream* formatuarekin geneukan, beraz formatu horri zegokien iturburu fitxategiak aztertu behar ziren. Eta bestalde liburutegiak konpilatzeko "arazoa" ere geneukan aurrean.

Iturburu kodeari dagokionez, hainbat fitxategi desberdin ditu formatu eta *codec* desberdinentzat, beraz, *Transport Stream* formatuaren kodeketari dagokiona aztertu behar zen, *mpegtsenc.c* fitxategia hain zuzen ere. Berez, *pcr*-aren *pid*-ari, bideoaren pid bera asignatzen zion kodeak, beraz asignazio hori egiten zen leku guztietan balio hori aldatzea nahikoa izan zen gure arazoa konpontzeko.

Beharrezko aldaketak burutu ondoren, iturburu kodea konpilatzeko lana zegoen oraindik. Hau egiteko dependentzia askorekin lehiatu behar da, eta eginkizun hori Windows-en egitea oso gauza zaila suertatzen zen, beraz konpilazioa Linux bidez egitea erabaki zen *Cross-compiling* teknika erabiliz. Linux erabiliz dependentzien arazoa apur bat errazten zen, baina hala ere ez zen gauza tribiala. Lehenengo ingurunea prestatu behar da konpilazioan zehar beharko diren fitxategi guztiak lortuz, eta ondoren konpilazioa jaurti behar da parametro egokiekin. Asko bilatuz, dependentzia guztiak ekiditeko *script* bat aurkitu nuen (Bibliografian aurki daiteke honi buruzko informazioa). Honek ingurunea prestatzen du beharrezko fitxategi eta dependentzia guztiak lortuz. Prozesu hau amaitzean (ordu batzuen ondoren) dena prest zegoen konpilatu ahal izateko. Hau Linux-en edozein instalazio bezala gauzatzen da, baina berriro ere prozesu luzea da. Amaitzerakoan, ffmpeg.exe exekutagarria lortzen da aplikazioan erabiltzeko prest.

III Eranskina

Bibliografia

Gaien Aurkibidea

III.a Bibliografia

- ffmpeg dokumentazioa eta iturburu kodea - *http://www.ffmpeg.org*
- ffmpeg-ren konpilazioa - *http://ffmpeg.zeranoe.com/blog/*
- SoX dokumentazioa - *http://sox.sourceforge.net/*
- Qt dokumentazioa - *http://qt-project.org/doc/*
- Qt tutorialak, Bryan Cairns - *http://www.voidrealms.com/*
- Inno Setup dokumentazioa - *http://www.jrsoftware.org/isinfo.php*
- Latex - *http://www.latex-project.org/*
- Texmaker - *http://www.xm1math.net/texmaker/*
- Dia - *http://projects.gnome.org/dia/*
- Oppenoffice - *http://www.openoffice.org/es/*
- gimp - *http://www.gimp.org/*