



**BILBOKO INDUSTRIA INGENIARITZA TEKNIKOKO  
UNIBERTSITATE ESKOLA**



**KUDEAKETAREN ETA INFORMAZIO SISTEMEN INFORMATIKAREN  
INGENIARITZAKO GRADUA**

**GRADU AMAIERAKO LANA**

2013 / 2014

*BABELIUM PROJECT: ANDROID GAILUETARAKO ADOBE AIR  
PROTOTIPOA*

**PROIEKTUAREN MEMORIA**

**IKASLEAREN DATUAK**

IZENA: JON KOLDO

ABIZENAK: TERA DEL ARCO

NAN zk.: 45753630-Y

SIN.:

DATA:

**ZUZENDARIAREN DATUAK**

IZENA: JUAN ANTONIO

ABIZENAK: PEREIRA VARELA

SAILA: HIZKUNTZA ETA SISTEMA INFORMATIKOAK

SIN.:

DATA:

Eskerrik asko Juanan Pereirari proiektu hau gauzatzeko jarri duen esfortzu, pazientzia eta gidaritzagatik. Eskerrik asko nire familiari nire hezkuntzarako (eta denetarako) eman didaten laguntza eta eskuzabaltasun guztiagatik, esker guztiak ez dira sekula nahiko izango.

Eta azkenik, eskerrik asko nire egoeragatik eta proiektu honen burutzeaz interesatu diren guztioi. Ez da erraza izan, baina azkenean, amaitzea lortu dut.

# Aurkibidea

<b>AURKIBIDEA.....</b>	<b>2</b>
<b>IRUDIEN AURKIBIDEA.....</b>	<b>5</b>
<b>PROIEKTUAREN HELBURU DOKUMENTUA .....</b>	<b>7</b>
SARRERA .....	7
ZER DA BABELIUM PROJECT? .....	7
ZERGATIK AUKERATU DUT PROIEKTU HAU?.....	8
PROIEKTUAREN HELBURUAK .....	9
ARKITEKTURA .....	10
SOFTWARE BALIABIDEAK .....	11
HARDWARE BALIABIDEAK .....	12
PROIEKTUAREN NORAINOKOA.....	13
<i>Proiektuaren fase desberdinak.....</i>	<i>13</i>
<i>Bizi zikloa (prototipoak).....</i>	<i>15</i>
<i>LDE diagrama .....</i>	<i>17</i>
DENBORA-PLANGINTZA.....	18
GANTT DIAGRAMA .....	21
ARRISKUAK .....	24
BIDERAGARRITASUN EKONOMIKOA .....	27
<i>Eskulana .....</i>	<i>27</i>
<i>Softwarea .....</i>	<i>27</i>
<i>Hardwarea.....</i>	<i>28</i>
<i>Beste gastu batzuk .....</i>	<i>29</i>
<i>Kostuak guztira.....</i>	<i>29</i>
<i>ROI (Return On Investment) .....</i>	<i>30</i>
<b>ZENBAIT KONTZEPTU GARRANTZITSU .....</b>	<b>31</b>
PLATAFORMA ANITZEKO GARAPENA .....	31
ANDROID SISTEMA ERAGILEA .....	31
RIA, EDO RICH INTERNET APPLICATIONS .....	34
ARM ARKITEKTURA .....	36
RTMP PROTOKOLOA .....	37

ACTIONSCRIPT.....	38
ADOBE FLEX.....	39
ADOBE AIR .....	40
CAIRNGORM MIKROARKITEKTURA.....	42
<b>BETEKIZUNEN BILKETA .....</b>	<b>45</b>
ERABILPEN KASUAK .....	45
KASU GUZTIETARAKO HASIERAKO ERABILPEN KASUA .....	45
ERABILPEN KASUEN EREDUA .....	48
1. EBALUAZIO ZERREDA NAGUSIA .....	49
2. EBALUATZEKE DAUDEN ERANTZUNAK ZERRENDATU .....	50
3. ARIKETA BATI EGINDAKO ERANTZUN BAT BISTARATU .....	51
4. EBALUAZIO BAT IGORRI .....	52
4.1. Erabiltzailearen hizkuntza-maila balioztatu.....	53
4.2. Ebaluazioa balioztatu.....	55
5. ERABILTZAILEARI EBALUATUTAKO ERANTZUNAK ZERRENDATU .....	59
6. ERABILTZAILEAREN ERANTZUNEN EBALUAZIOAK ZERRENDATU .....	60
7. BIDEO BATEN EBALUAZIO GUZTIAK BISTARATU.....	62
8. APLIKAZIOAREN HIZKUNTZA ALDATU .....	64
<b>ANALISIA ETA DISEINUA.....</b>	<b>67</b>
KLASE DIAGRAMA .....	67
SEKUENTZIA DIAGRAMAK.....	68
<b>INPLEMENTAZIOA .....</b>	<b>74</b>
BABELIUM MOBILE PROIEKTUAREN EGITURA .....	74
<i>Egitura SDK-aren barne.....</i>	<i>74</i>
<i>Proiektuaren azpi-atalak.....</i>	<i>75</i>
APLIKAZIOAREN BARNEKO BISTAK .....	79
<i>Ebaluaketa aukeren CalloutButton osagaia.....</i>	<i>79</i>
<i>PendingEvaluations.mxml, EvaluationsOnMe.mxml eta MyEvaluations.mxml .....</i>	<i>81</i>
<i>Evaluate.mxml.....</i>	<i>84</i>
<i>ViewEvaluations.mxml.....</i>	<i>88</i>

<i>SelectLanguage.mxml</i> .....	92
<b>PROIEKTUAREN KUDEAKETA .....</b>	<b>95</b>
<i>Proiektuaren aurrerapenak ebaluatzeko bilerak</i> .....	95
<i>Denbora-plangintzarekiko desbiderapena</i> .....	95
<i>Kodearen kudeaketa</i> .....	96
<i>Bug edo programazio-akatsak</i> .....	97
<b>ONDORIOAK ETA KONKLUSIOAK .....</b>	<b>98</b>
ZER IKASI DUT? .....	98
ZER HOBETU DEZAKET? .....	99
ZER GERTATUKO DA PROIEKTU HONEKIN ETORKIZUNEAN? .....	100
<b>BIBLIOGRAFIA .....</b>	<b>101</b>
LIBURUAK .....	101
WEBGUNEAK .....	101
<b>ERANSKINAK .....</b>	<b>102</b>
1. ERANSKINA: LAN-INGURUNEAREN HASIERAKETA ETA SOFTWAREA .....	102
<i>Adobe Flash Builder</i> .....	102
<i>Adobe AIR</i> .....	105
<i>Git + GitHub</i> .....	105
<i>Red5 Media Server</i> .....	106
<i>OflaDemo aplikazioa</i> .....	108
2. ERANSKINA: RTMP PROTOKOLOA TESTEATZEKO APLIKAZIO BAT .....	111
<i>Behar diren elementuak</i> .....	111
<i>Proiektua sortu eta konfiguratu</i> .....	111
<i>Proiektuaren</i> .....	112
<i>Gailuaren konfigurazioa</i> .....	113
<i>Aplikazioa eraikitzeke elementuak</i> .....	115
3. ERANSKINA: RTMP FROGA APLIKAZIORAKO KODEA .....	123
4. ERANSKINA: MYRESOURCES.PROPERTIES FITXATEGIETAN GEHITUTAKO SARRERAK ..	127

## Irudien aurkibidea

1. irudia: LDE diagrama .....	17
2. irudia: GANTT diagrama (1.zatia).....	21
3. irudia: GANTT diagrama (2.zatia).....	22
4. irudia: GANTT diagrama (3.zatia).....	23
5. irudia: Android arkitektura .....	33
6. irudia: Cairngorm mikroarkitektura .....	44
7. irudia: Kautotze-pantaila .....	46
8. irudia: Igotako azken bideoak.....	47
9. irudia: Erabilpen-kasuen eredua.....	48
10. irudia: Ebaluazio zerrenda nagusia .....	49
11. irudia: "To evaluate" aukera .....	50
12. irudia: Ebaluatu ez diren bideoak .....	51
13. irudia: Ebaluatzekeo pantaila .....	52
14. irudia: Hizkuntza-maila mezua.....	54
15. irudia: Ebaluazio botoia desgaituta .....	55
16. irudia: Amaitzeke dagoen ebaluazioa .....	56
17. irudia: Ebaluazio desegokiaren mezua .....	57
18. irudia: Ebaluazioa ondo gorde da .....	58
19. irudia: "Assessments to me" aukera.....	59
20. irudia: Jasotako ebaluaketak .....	59
21. irudia: Assessments by me aukera.....	60
22. irudia: Egindako ebaluaketak.....	61
23. irudia: Ebaluazioak bistaratzeko pantaila .....	62
24. irudia: Ebaluatzaileen zerrenda .....	63
25. irudia: Ebaluazio baten informazioa .....	63
26. irudia: Hizkuntza aldatzeko botoia .....	64
27. irudia: Hizkuntza aldatzeko pantaila .....	65
28. irudia: Hizkuntzen zerrenda.....	65
29. irudia: Hizkuntza aukeratu eta gero.....	66
30. irudia: Hasierako pantaila hizkuntza aldatuta .....	66
31. irudia: Klase diagrama .....	67
32. irudia: 1. Sekuentzia diagrama .....	68
33. irudia: 2. Sekuentzia diagrama .....	69
34. irudia: 3. Sekuentzia diagrama .....	70
35. irudia: 4. Sekuentzia diagrama .....	71
36. irudia: 5. Sekuentzia diagrama .....	72
37. irudia: 6. Sekuentzia diagrama .....	73
38. irudia: SDK-aren barneko egitura .....	74
39. irudia: Ebaluazio CalloutButton-a .....	79
40. irudia: Egindako ebaluaketen zerrenda .....	81
41. irudia: Ebaluatzekeo pantaila .....	84
42. irudia: Ebaluazioak ikusteko pantaila .....	88
43. irudia: Hizkuntza aldatzeko pantaila .....	92
44. irudia: Adobe Flash Builder-ren lizentzia onartua.....	104
45. irudia Red5 zerbitzari lokalaren pantaila nagusia .....	107

# Babelium Project: Android gailuetarako Adobe AIR prototipoa

## PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

46. irudia: OflaDemo aplikazioa .....	108
47. irudia: OflaDemo aplikazioa martxan .....	109
48. irudia: Proiektua sortzeko lehen leihoa .....	112
49. irudia: Proiektua sortzeko bigarren leihoa .....	113
50. irudia: Proiektua sortzeko hirugarren leihoa .....	115
51. irudia: Aplikazioa Sony Ericsson Xperia Neo V telefonoan .....	121
52. irudia: Aplikazioa Samsung Galaxy S5 telefonoan .....	122

## **Proiektuaren Helburu Dokumentua**

### **Sarrera**

Gaur egungo gizartean, kulturen artean, fluxu ugariko trukaketa bat gertatzen ari da. Fenomeno hau globalizazio izenarekin ezagutzen dugu. Munduko nazioen tradizio, kultura, hizkuntza eta beste korrante askok beste lurraldeekin zituzten mugak zeharkatu eta atzerriratu dira.

Hori dela eta, nahiz eta gizakien funtsa berdina izan mundu guztian, aipatutako kultura-desberdintasunek oinarritzeko eginkizun bat jokatu dute, eta desberdintasun nabarmenetako bat hizkuntzetan ikusten dugu. Badira munduan orokorrak diren hizkuntzak, eta gutxienak diren beste batzuk, eta ezinbestekoa da ahalik eta gehien ezagutzea bizitzen ari garen mundu modernora era egokian moldatzeko.

Globalizazioak, gehienetan, teknologia inplikatu du. Teknologia munduan milaka kilometroko distantziak gerturatu ditu eta puntu hauen arteko komunikazioak bideratu. Hizkuntza eta teknologia berriekin hibrido bat sortu nahiko bagenu, Babelium Project-en antzeko zerbait lortuko genuke: hizkuntzen elkartrukatzerako ingurune birtual bat.

### **Zer da Babelium Project?**

“Babelium hizkuntzak ikastea eta irakastea gustoko duten pertsonen komunitate bat da.” Definizio honekin egingo dugu topo Babelium Project-en web orrialde nagusira jotzen badugu: <http://babeliumproject.com>



Hizkuntzak ikasi eta irakasteko ingurune bat da, erabiltzaileen ekarpenaz elikadura gehiena jasoz. Erabiltzaileek praktikan jar ditzakete beraien ezagutzak, ahozko trebetasunak landuz, edo beste erabiltzaile batzuen saiakerak ebaluatuz. Webgunean aurkitzen diren ariketa desberdinei erantzunez, eta bideo edota audio formatuan erantzun berriak eskegiz, erabiltzaileen arteko interakzioa sustatuko da, eta erabiltzaileek haien hizkuntza-gaitasunak landu ahal izango dituzte.

Partaideek, bideo bat igo izan den ondoren, bideoa azpigitulatu beharko dute, behin azpigitulu hauek izanda beste kide batzuek hauekin praktikatu ahal izateko. Praktika hauek ebaluatzeko, erabiltzaileek hizkuntza-maila jakin bat menperatu behar dute, logikoa denez ebaluazioak egin ahal izateko hizkuntza horren ezagutza beharrezkoa baita.

Hizkuntzak ikasteko era sozial honek prozesua entretenigarriago bilakatzen du, interneteko beste sare sozialek duten osagarri hori besarkatuz eta teknologia berrietan integratzeko aukera emanez. Hori dela eta, ez da batere arraroa pentsatzea horrelako webgune batek bere bertsio mugikorra izan beharko lukeela.

## **Zergatik aukeratu dut proiektu hau?**

Gailu mugikorren garrantzia era esponentzian hazten da gaur egun teknologia arloan. Orain dela urte eta hamarkada batzutako makina erraldoiak fenomeno arkaikotzat hartzen dira, eta askotan teknologiaren tamaina edota eramangarritasuna baloratzen da, memoria, potentzia eta prozesatzeko ahalmenaren aurretik.

Arrazoi honengatik, telefono mugikor eta tabletentzako software garapenak hazkunde handia jasan du azken urteetan, eta inbertsio gehienak esparru honetan egiten dituzte enpresa handiek. Kaletik joaten garenean, oso pertsona gutxi izango dira eskuetan gailu eramangarri bat izango ez dutenak. Bizitza askoz errazagoa egiten dute arlo praktikora eramaten direnean, eta zeregin askotarako ezinbesteko laguntzaile bilakatu dira.

Software mota honen garapenak lanpostu asko sortu ditu eta aplikazioak diseinatu eta programatzeko pertsonen eskaria egunero areagotzen da. Nire iritziz, arlo interesgarria iruditzen zait gainera eta gai honi buruz ikastea aukera ontzat begiesten dut.

Aldi berean, gailu mugikorrekin kontaktu jarraitua izan dut azken 2-3 urteetan eta honi esker lortutako ezagutzak eraginkortasunez aplikatzeko aukera izatea espero dut etorkizunean.

## Proiektuaren helburuak

Proiektu honetan Adobe Flash teknologiak aprobetxatuz Babelium proiektuaren bertsio mugikorrerako funtzionalitate berriak implementatzea izango da helburu nagusia. Hau gauzatzen doan heinean, Actionscript, Flex eta AIR teknologiei buruzko ezagutzak lortuko dira, Android ingurunearen barruan moldatzeko erabiliko direnak.

Android gailu mugikorren ahalmena aprobetxatuz, kamara, mikrofono eta sareko konexioaz baliatuko gara, beste batzuen artean, Babelium Project-ek eskaintzen dituen ezaugarriak ingurune eramangarri batera moldatzeko.

Erabiltzen den gailuaren arabera, aplikazioa ahalik eta era egokienean moldatzea izango da beste helburu bat. Gaur egun aurkitu ditzakegun telefono eta tabletek ezaugarri tekniko oso desberdinak eskaintzen dituzte. Desberdintasun handiena pantailaren tamainan ikusten da, beraz, 5 hazbeteko pantaila baten, 10 hazbetekoan bezain funtzionala izan behar du aplikazioak.

Aplikazioak aurretik inplementaturik zituen API eta metodoen berrerabilera egokia izango da beste puntu garrantzitsu bat. Honek lana erraztea du helburu, bertsio mugikorra ordenagailuetan exekutatzen den bertsioaren ahalik eta antzekoen izatea lortzeko. Arlo honetan aurretik parte hartu duten guztien lana oso garrantzitsua izanen da.

## Arkitektura

Cairngorm mikroarkitekturan oinarritzen da aplikazioa. Sakonago azalduko dira arkitekturaren ezaugarriak aurrerago memoria honetan, garrantzitsua baita bere funtzionamendua ulertzea kodean murgildu baino lehen.

Bezero eta zerbitzari arteko komunikazioan enfasi handia jartzen du, errekurtsio gehienak aplikaziotik kanpo mantendu behar baitira. Aplikaziotik ValueObject (VO) izeneko datu-motako unitateak bidaltzen dira gertakarien (Event) barruan, zerbitzarian kudeatu eta prozesatzeko.

Horrela, lokalki soilik beharrezkoak diren elementuak gordetzen dira: interfaze grafikoak, datu-moten eraketa, komandoak... bilaketa eta prozesu pisutsuagoak zerbitzariari eskuordetuz. Ondorioz, internet konexioa beharrezkoa da Babelium Project erabiltzen denean.

## Software baliabideak

Hurrengoak dira proiektu honen burutzean erabili diren software baliabideak:

- **Adobe Flash Builder:** aplikazioaren kodeketa eta arazketarako erabili den IDE-a (Integrated Development Environment). Emulatzaile bat barneraturik dakar programatutakoa testatzeko.
- **Adobe AIR:** Adobe AIR (Adobe Integrated Runtime) aplikazioak garatzeko erabiltzen den plataforma anitzeko exekuzio-ingurumena da.
- **Red5 Media Server:** RTMP protokoloarekin eskariak zerbitzatzeko erabili den tresna.
- **Microsoft Office:** memoriaren elaboraziorako erabili den ofimatika software multzoa, Microsoft Word software tresna hain zuzen.
- **Google Drive:** proiektuaren plangintza, baliabideak eta hainbat informazio eguneratuta mantentzeko erabilgarria suertatu den lanabesa, oso egokia online datuak gordetzeko eta editatzeko eskaintzen duen aukera dela eta.
- **Ganttproject:** plangintza grafikoki errepresentatzeko erabili den doako erreminta, Gantt diagrama baten bidez erabilitako denboraren informazioa irudikatzeko.
- **Gliffy:** diagramak eta gertaera-fluxuak osatzeko software grafikoko tresna. Online bertsioa hurrengo helbidean aurkitu daiteke (froga aldi bat eskaintzen da, gero ordaindu behar da): <http://www.gliffy.com>

- **Git:** errepositorioaren kodea eguneratuta mantentzeko erabili da software hau, **GitHub** web orrialdean aldaketak gordez. Orrialde hau erabiltzeko doako erabiltzaile-kontu bat sortzea beharrezkoa izan da.

## Hardware baliabideak

Aplikazioa garatzeko eta probatzeko ezinbestekoa da hardware oinarri sendo bat izatea. Hauek izan dira proiektuan erabilitako hardware baliabideak:

- **Ordenagailu eramangarria:** Acer Aspire 5750G

### Ezaugarri teknikoak:

- Intel Core i7, 2Ghz
- Geforce GT540M txartel grafikoa (2GB)
- 4GB DDR3 RAM memoria
- 500GB-ko disko gogorra

- **Android telefono mugikorak:**

- Sony Ericsson Xperia Neo V telefono mugikorra
- Samsung Galaxy Young telefono mugikorra
- Samsung Galaxy S5 telefono mugikorra

- **Android tablet-ak:**

- Samsung Galaxy Tab tablet-a

## Proiektuaren norainokoa

### Proiektuaren fase desberdinak

- **Ikerketa:** proiektu berri baten hasieran, normalena da gaiari buruzko ezagutza murrizta izatea, edo landuko diren gaiaren aspektuekin oraindik tratatu ez izana, nahiz eta beste batzuk ezagutu.

Kasu honetan, Android gailuetarako aplikazioei buruz, edo Adobek eskaintzen dituen programazio-lengoaiei buruzko informazioa biltzea beharrezkoa izan da, hauek ez baitira normalean ikasketa-zentro edota unibertsitateetan eskaintzen. Izan ere, programazio metodologiaren aspektuak hobeto bereganatzen dituzte ikasleek erabilpen orokorrakoak dituzten programazio-lengoaia batzuekin: Ada, Java, ...

- **Ikaskuntza:** aurreko fasean aztertu diren ezagutzak asimilatzea da hurrengo pausoa. Behin kontzeptuak ulertuta, hauek praktikara eramatea beharrezkoa suertatzen da ondoren, geroago erraztasunez berrerabili ahal daitezen.

Honen adibide izan daiteke memoria honen eranskinean aurkitzen den froga aplikazioa, non liburu eta beste iturri batzuetatik jasotako informazioa aplikatzen den, proiektuaren gai nagusiarekin zerikusia duten kontzeptu batzuei hauteman daitekeen forma bat emateko.

- **Analisia:** analisi-fasean, aplikazioa sortzeko zer egingo dugun azalduko da, eta zergatik bide hori aukeratu den argudiatu. Hau da, aurkeztuko diren erronkei zein motatako metodo aplikatuko diogun jakin behar dugu, aurreko faseetan lortu dugun ezagutzan oinarrituz. Oztoporen bat gainditzeko gai ez bagara, ziurrenik aurreko faseetan ez da egin beharreko lan guztia gauzatu.

Lehenengo bi faseak burutu eta gero ideia berriak agertuko balira, analisisian erantzun egoki bat ezin emanda, hirugarren fasetik lehenengora bueltatuko ginateke, kontuan harturik momentu horretan ikertuko dena aurrekoarekin integragarria izan behar duela.

- **Diseinua:** behin funtzionalitateen analisia aztertu dugula, hauek aplikazioaren barruan nola integratu aztertuko dugu, baliabideak ahalik eta gutxien erabiliz eta funtzionalitateen arteko erlazio egokiak ezarriz. Sekuentzia eta klase-diagramak eratzea oso garrantzitsua da, metodoak eta klaseen ezaugarriak oso argi izateak denbora asko aurreztuko digu inplementazioa egiteko orduan, batez ere akatsak aurreikusi eta, ematen badira, lehenbailehen zuzentzeko.

Horrez gain, interfaze grafikoa inplementatuko duten aplikazioetan, aurreko antolaketa errespetatzen duen interfaze ahalik eta praktikoena diseinatzea izango da atal garrantzitsuenetariko bat. Zirriborroak laguntza handikoak dira helburu honetarako: botoiak, irudiak, zerrendak, leihoak, ... nola kokatu erabakitzerako orduan aplikazioaren irudikapen argiago bat lortzea erraztuko dute.

- **Inplementazioa:** behin aplikazioaren diseinua (bai funtzionalitateena, bai interfaze grafikoarena) landu eta erabaki dugula, aurreko ideia eta diagrama guztiak koderaz bihurtzea da helburu nagusia. Aurretik egindako lanaren arabera finkatuko da inplementazioan eman beharko dugun denbora: zenbat eta diseinu hobeagoa, are eta inplementazio errazagoa.

Nahiz eta gauza batzuk diseinu fasean finkatu ezin izango ditugun, edo denborarekin oso azkar aldatzen diren elementu batzuen arabera lan egin beharko dugun (API publikoak, esate baterako), inplementazioaren

aspektu guztiak diseinu fasean hartutako erabakiekin azaltzeko gai izan beharko genuke.

- **Probak:** aplikazioaren inplementazioa amaitzen denean, saihestu ezin izan diren eta espero ez ziren erroreak zuzentzeko ordua da. Nahiz eta aplikazioaren garapenaren fase guztietan probak egiten diren, fase honetan testatzen dira egindako lanaren funtzionalitate guztiak, ahalik eta erabilpen kasu gehienak probatzen saiatuz. Lehen aipatu bezala, gero eta diseinu hobeagoa, are eta gutxiago akats suertatuko dira azkenean.

## Bizi zikloa (prototipoak)

Aurretik partzialki eraikita zegoen aplikazio batekin egin da lan proiektu honetan. Beraz, heredatutako aplikazioa lehen prototipotzat hartuko dugu.

- **Lehen prototipoa:** Babelium Project-en mugikorrentzako lehen bertsioa. Bertsio honetan erabiltzaileak aurreko kamera eta mikrofonoa erabiliz, bideo-erantzunak grabatu eta zerbitzarira igo ditzake.

- **Bigarren prototipoa:** bertsio honetan, ebaluazioekin erlazionatutako audio eta bideo-erantzunak zerrendatzen dira pantaila desberdinetan. Pantaila hauetara heltzeko, botoi-zerrenda bat inplementatu da aplikazioan, klik egiten denean zerrenden artean bat aukeratzea ahalbidetuz.

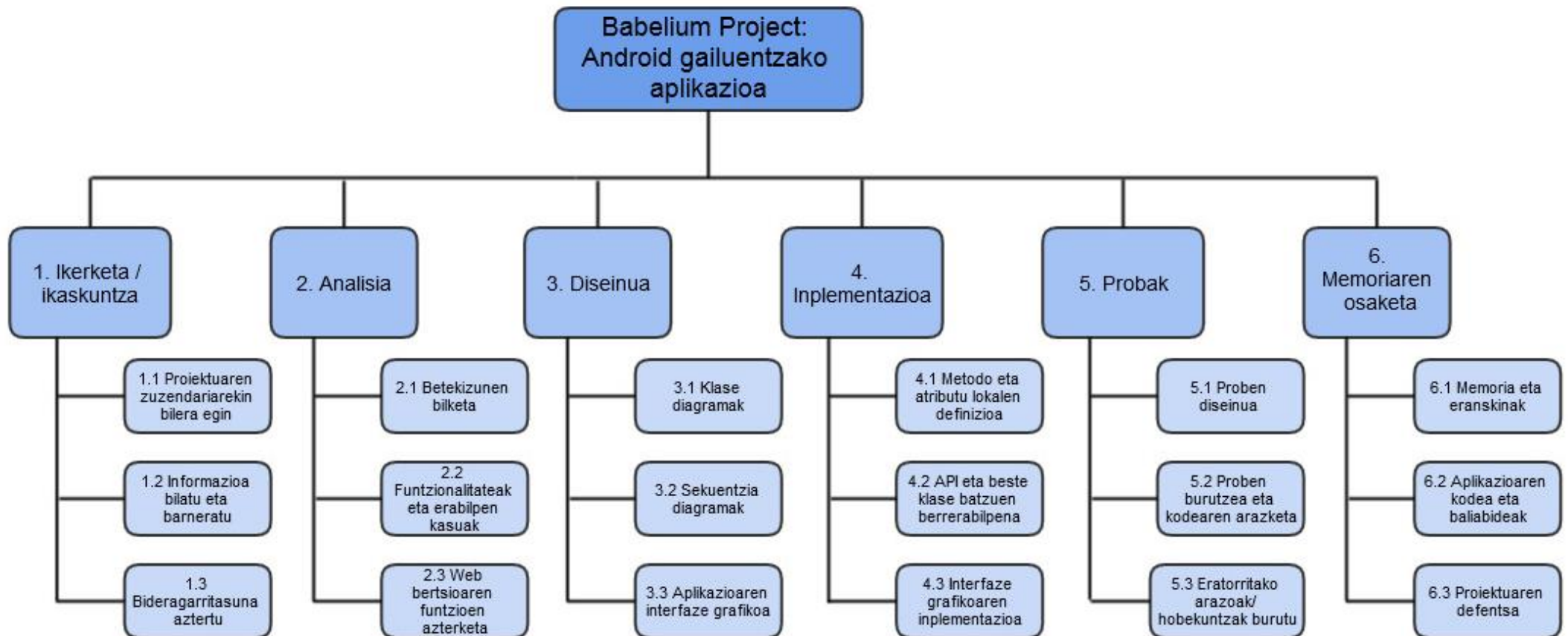
- **Hirugarren prototipoa:** hirugarren prototipoan, aurreko zerrendetatik bideo bat aukeratzen pantaila berri batera heltzen gara, aukeratutako ariketaren metadatuak (izenburua, data, erabiltzailea, ...) honekin batera pantaila berrira eramanez.



· **Laugarren prototipoa:** prototipo honetan, ebaluazio pantailetak funtzioak inplementatuta daude: erabiltzaileen audio eta bideo-erantzunak ikusi, ebaluazioak bete eta bidali, beste erabiltzaileen ebaluazioak bistaratu, ... Horrez gain, aplikazioaren hizkuntza aldatu dezakegu.

· **Bosgarren prototipoa:** hauxe da azken prototipoa. Prototipo honetan, erabiltzaileen ebaluazioak bistaratzean, pertsonaien rolak era egokian kudeatzean dira, eta horren ondorioz, bideoen arteko audioaren sinkronizazioa modu zuzenean txandakatzen da.

### LDE diagrama



1. irudia: LDE diagrama

## Denbora-plangintza

LDE diagramaren arabera, 6 fasetan banatuko da denboraren plangintza, eta beste 3 azpi-fasetan zatituko da fase bakoitza:

---

FASEA	Iraupena
<b>Ikerketa eta ikaskuntza</b>	<b>135 ordu</b>
Proiektuaren zuzendariarekin bilerak finkatu eta elkartu	<b>10 ordu</b>
Informazioa bilatu eta barneratu	<b>120 ordu</b>
Bideragarritasuna aztertu	<b>5 ordu</b>

---

---

FASEA	Iraupena
<b>Analisia</b>	<b>40 ordu</b>
Betekizunen bilketa garatu	<b>15 ordu</b>
Funtzionalitateak eta erabilpen kasuak planteatu	<b>15 ordu</b>
Web bertsioaren funtzioen azterketa egin	<b>10 ordu</b>

---

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

---

FASEA	Iraupena
<b>Diseinua</b>	<b>45 ordu</b>
Klase diagramak garatu	<b>15 ordu</b>
Sekuentzia diagramak garatu	<b>20 ordu</b>
Aplikazioaren interfaze grafikoaren zirriborroak landu	<b>10 ordu</b>

---

---

FASEA	Iraupena
<b>Inplementazioa</b>	<b>150 ordu</b>
Metodo eta atributu lokalen definizioa	<b>60 ordu</b>
API eta beste klase batzuen berrerabilpena	<b>40 ordu</b>
Interfaze grafikoaren inplementazioa	<b>50 ordu</b>

---

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

FASEA	Iraupena
<b>Probak</b>	<b>30 ordu</b>
Proben diseinua	5 ordu
Proben burutzea eta kodearen arazketa	10 ordu
Eratorritako arazoak zuzendu eta hobekuntzak burutu	15 ordu
FASEA	Iraupena
<b>Memoriaren osaketa</b>	<b>100 ordu</b>
Memoria eta eranskinen osaketa	80 ordu
Aplikazioaren kodea eta baliabideak paketatu	5 ordu
Proiektuaren defentsa prestatu	15 ordu
<b>Guztirako iraupena:</b>	<b>500 ordu</b>

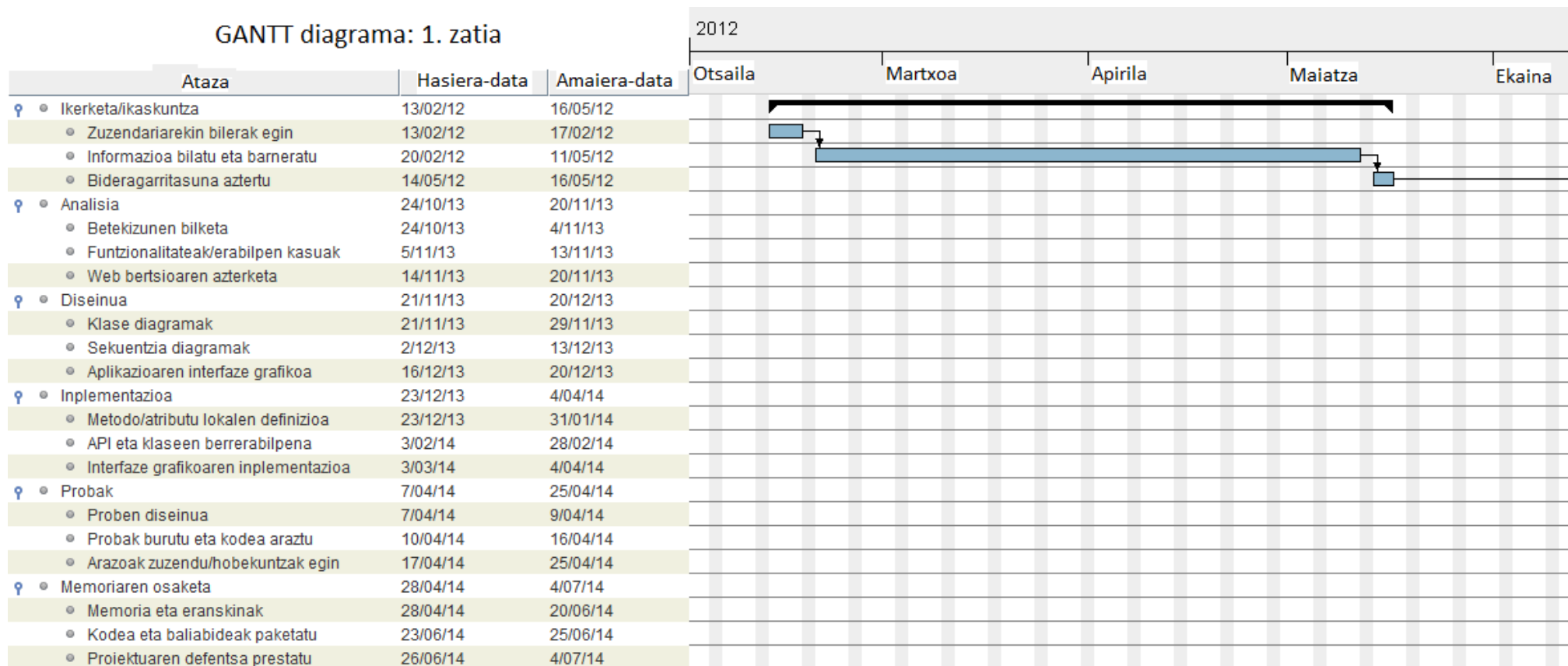
## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

## GANTT diagrama

GANTT diagrama: 1. zatia



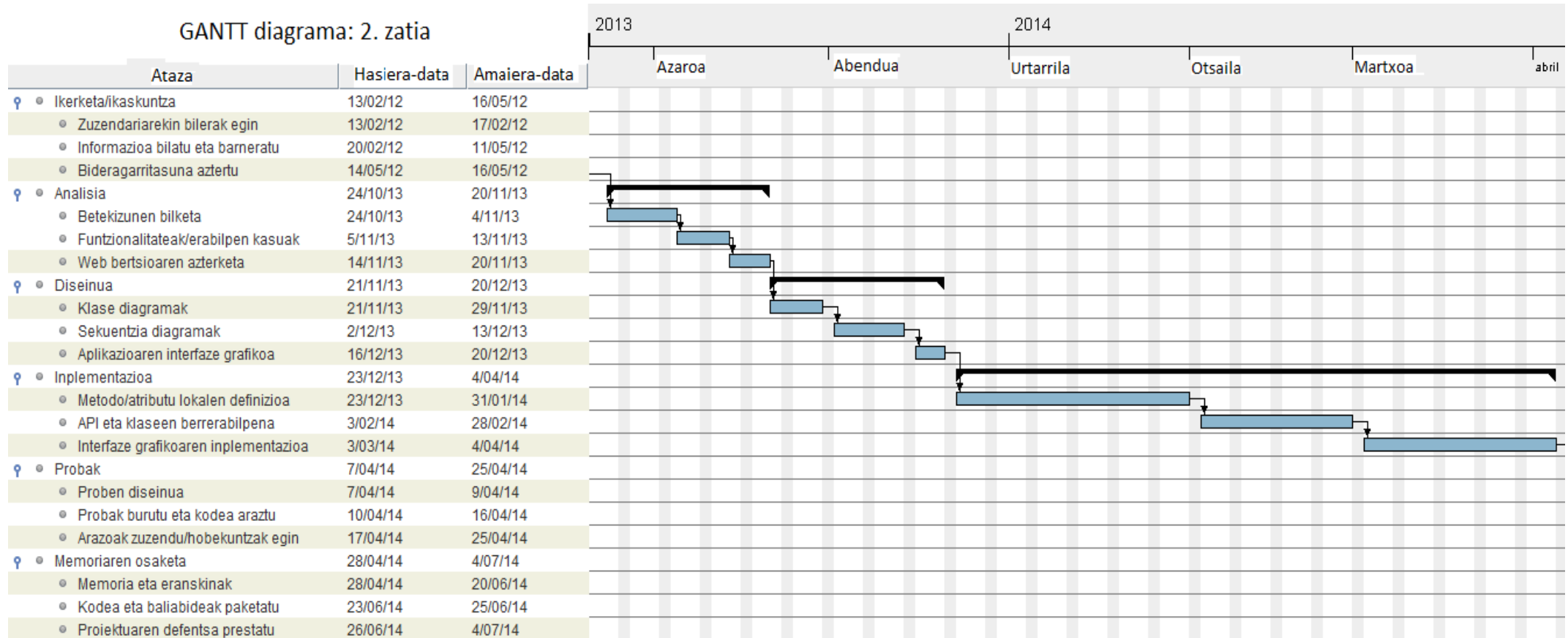
2. irudia: GANTT diagrama (1.zatia)

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

GANTT diagrama: 2. zatia



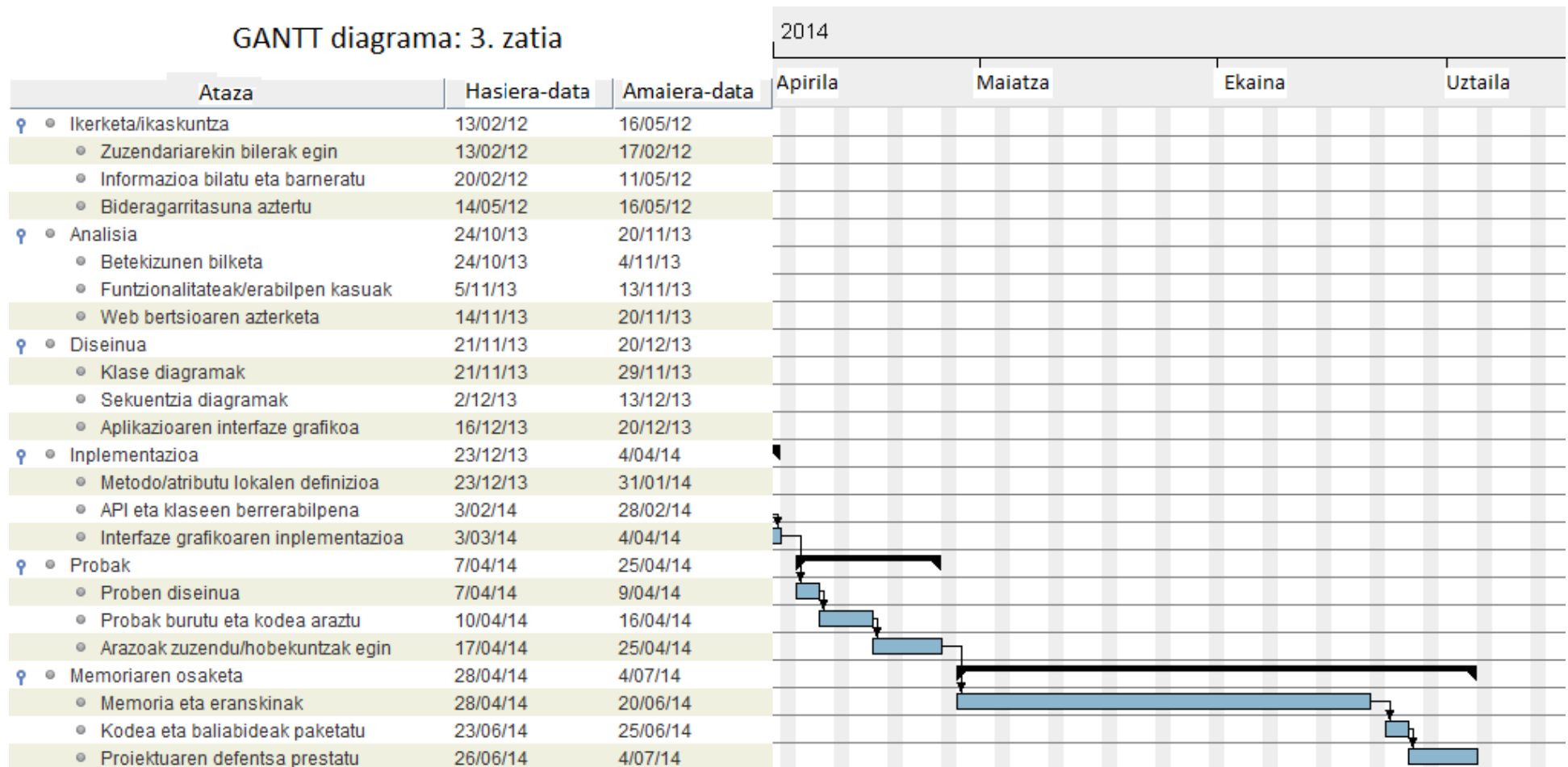
3. irudia: GANTT diagrama (2.zatia)

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

GANTT diagrama: 3. zatia



4. irudia: GANTT diagrama (3.zatia)



## Arriskuak

Edozein proiektu espero zein eran amaitu baino lehen, arriskuak suertatzea da normalena. Kasu honetan ere, proiektuaren hasieran arriskuen jatorria eta hauek gertatzeko probabilitatea kontrolatu beharko dugu, eta gainera, arriskuak suertatzen direnerako kontingentzia-plan minimo batzuk diseinatu eta aplikatu. Proiektu honetan eman diren eta eman zitekeen arriskuak analizatuko dira orain.

### • Planifikazioaren birplanteamendua

Aplikazioa garatzen ari garen bitartean, egindako plangintza egokia ez dela antzematea gertatu daiteke. Aurretik edozein atazarako kalkulaturako denbora nahikoa ez dela sarritan suertatzen den arazo bat da, eta hau kontuan izanda ere zaila da plangintza egitean epeak zehaztasunez esleitzea, nahiz eta askotan denbora gehiago estimatzen den badaezpada.

Hau gertatzeko probabilitatea oso altua da, baina aurretik ezustekoak direla eta malgutasunez egin badugu planifikazioa, ez ditu eragin larriak izango.

### • Informazioa galtzea (hardwarea apurtu, hondamendi naturalak, etab.)

Iraupen luzeko proiektuetan, sarri gerta daiteke informazioaren iraunkortasunarekin arazoak izatea, zenbat eta denbora gehiago eramaten badu proiektu batek, are eta arrisku handiagoa baitugu garatutako lanaren integraltasuna mantentzeko.

Arazo hau ekiditeko erarik ezagunena babes-kopien erabilera da, informazio digitalari dagokionean. Garatutako lanaren asteroko, edo

egindako lana oso pisutsua ez bada, eguneroko kopia batek lasaitasuna emango digu informazioaren osotasuna galzorian jartzen denean. Aldaketa garrantzitsu edota kritikoak egiten ditugunean, babes-kopia eskuz hasieratu daiteke ere.

Informazio-galera bat izateko probabilitatea normalean baxua da sistema egonkor batean egiten badugu lan, baina gertatzen denean, ondorioak oso larriak dira, gehiago edo gutxiago galdutako informazio-kopuruaren arabera.

#### • Ezagutza falta dela eta lana aurrera ezin eramatea

Ikerketa eta ikaskuntza faseetan egindako lana hurrengo faseetan hedatzeko probabilitatea oso altua da. Gai berri bati buruz informazioa biltzean, informazio guztia asimilatzea prozesu konplexua da, eta ez dago zehaztasunez eskura eduki beharko den informazioa jakiteko modurik, nahiz eta badakigun zertan oinarritzen den proiektua.

Ikaskuntza prozesua hasieran oinarritzkoa da soilik, nondik hasi jakin ahal izateko. Ikaskuntza prozesu hau proiektuan zehar luzatuko da gehienetan, eta kanpoko laguntza beharrezkoa izan daiteke. Liburu, internet eta proiektuaren zuzendariaren erreferentzia izatea laguntzazkoa suertatuko da noizbehinka, zalantzak eta oztopoak gainditzeko.

Egoera hau emateko aukera nahiko handia da, baina ez luke buruhauste handirik eman behar ikerketa eta ikaskuntza faseetan denbora nahikoa inbertitu bada, are gehiago gaiari buruz aurretiko ezagutza izan bada.

#### • Babelium-en egituran ezusteko aldaketak

Babelium Project garatzen duten lankideek maila desberdinetan egiten dute lan, ikasleak, irakasleak edo ikertzaileak izanik. Nahiz eta Android aplikazioa eta web bertsioa desberdinak diren, zerbitzari berbera erabiltzen dute informazioa eskuratzeko. Beraz, aldaketaren bat egongo balitz zerbitzarian, bai web bertsioan eta bai Android aplikazioan islatu beharko litzateke.

Aplikazioaren proben fasean, login egiteko funtzioaren parametroetan aldaketa bat egin zuen zerbitzarian lankide batek, eta komunikazio faltagatik, aplikazioan kautotzeko arazoak zirela eta, ezin izan ziren probak egin egun batzuetan.

Lankideen arteko komunikazio on batekin, eta horrelako aldaketak beste kideei jakinarazteko sistema bat erabiliz, arazo hau erraztasunez saihestu daiteke. Hau gertatzeko probabilitatea ez da oso altua, horrelako aldaketak normalean ez direlako egiten, eta egiten badira, arrazoi garrantzitsu batengatik egiten direlako.

#### • Arazo pertsonalak, istripuak edota lesioak

Garapen prozesu osoan, teknikoak ez diren arazoak eman daitezke, lana aurrera eramateko zailtasunak suposatu ditzaketenak, larritasunaren arabera. Lesio fisikoak, beste konpromezu batzuk, eta abar. Hauen ondorioz, lanarekin jarraitzeko zailtasunekin topatu gaitezke.

Eragozpen hauei aurka egin behar izateko probabilitatea ez da oso altua, baina gertatzen denean, ezaugarrien arabera, proiektuan lan egiteko ahalmena partzialki edo osorik gal daiteke, planifikazioan dakarren eraginarekin.

## Bideragarritasun ekonomikoa

Orain, proiektuaren bideragarritasun ekonomikoa aztertuko da aplikazioa garatzeko baldintzak onargarriak izan daitezen, denbora eta lan asko inbertituko denez ekonomikoki aprobetxagarria izan behar baitu prozesua.

## Eskulana

Eskulanari dagokionez, aurreko denbora-plangintzan 500 orduko estimazioa egin da. Interneteko foroetan eta lankide edota lagunei galdetuz, orduan 18€ko prezioa estimatu da junior programatzaile batentzat.

Proiektuaren iraupena (500 ordu) eta orduko eskulana (18 €/orduko) biderkatuz, eskulanaren kostu osoa lortu dezakegu:

$$500 \text{ ordu} * 18 \text{ €/orduko} = \text{Eskulana: } 9000\text{€}$$

## Softwarea

Ahal izan den neurrian, software guztia erabilpenagatik dohainekoa izatean jarri da esfortzua. Erabili den software guztitik, Microsoft Office, Windows sistema-eragilearen lizentzia eta Adobe Flash Builder izan dira ordaindu beharreko programa bakarrik.

Lehenengo biak aktibatzeko ordenagailu eramangarria erosterakoan erantsitako lizentziak erabili dira. Flash Builder-ren kasuan, Adobe enpresak aspaldi eskaintzen zuen ikasle eta ikertzaileentzako lizentzia bat erabiliz aktibatu zen. Honi buruz gehiago azaltzen da memoria honen

lehenengo eranskinean, lan-ingurunearen hasieraketari dagokion lekuan, hain zuzen.

Lehen aipatutako beste software-tresna guztiak doakoak edo erabilera librekoak dira. Gliffy-ren kasuan, dena den, eskaintzen zen 15 eguneko proba-sasoia erabili izan da.

## Hardwarea

Hardwarearen kasuan, proiektuaren luzapena dela eta hardware desberdinak erabili dira aplikazioa garatu eta probatzeko. Hona hemen erabilitako hardwarea:

- Ordenagailu eramangarria: 690€
- Android gailuak guztira: 450€

Hardwarearen erabilera proiektuan ordenagailuaren kasuan %40koa izan da, eta Android gailuena %10koa. Gaur arte, hau izan da urtero amortizatu den kantitate bakoitza:

Hardwarea	Prezioa	Denbora	Amort./urtean	Amort. totala
Ordenagailua	690€	3 urte	230€	$230 \cdot 0,4 = 92€$
Android gailuak	450€	2 urte	225€	$225 \cdot 0,1 = 22,5€$

Gauzak horrela, hardwarearen amortizazioaren kostu totala 114,5 eurokoa izan da.

## Beste gastu batzuk

Azkenik, bestelako gastuak egotea ere espero da. Hurrengoak izan dira proiektu honetan zehar bestelako kostuak gutxi gorabehera:

- Garraioa/bazkariak: 150€
- Sare konexioa: 75€
- Materiala, inprimaketak: 30€
- Argi-korrontea: 40€

Guztira, 295 euroko bestelako gastuak gehitu beharko dizkiogu aurrekontu finalari.

## Kostuak guztira

Aurreko atalak gehituz, guztirako kostuak kalkulatu dira hurrengo taulan:

Kontzeptua	Kostua
Eskulana	9000€
Software	0€
Hardware	114,5€
Beste gastu batzuk	295€
<b>Totala:</b>	<b>9409,5€</b>

## ROI (Return On Investment)

Babelium Project aplikazioaren kodea publikoa da eta GitHub-en aurkitu daiteke edonork berriro konpilatu dezan, izan ere, ez da dirua irabazteko asmoarekin sortu den eginkizun bat. Dena den, mantentze-kostuak leuntzeko edota beste edozein inbertsiorako, publizitatea sartzea izango litzateke soluzioetako bat.

Publizitatea irtenbide dotorea ez denez ia kasu guztietan, eta hezkuntza-sistema bat izateko xedearekin garatu den aplikazio bat dugularik, gobernu- eta kultura-erakundeei eskatu diezazkiekeen laguntza ekonomikoak edo subentzioak, lehen aipatutako mantenu-kostuentzako soluzioa aurkitu ahal izateko.

Gauzak horrela, ez da beharrezkoa inbertsioaren errekupeziarako modurik bilatzea, eta horrenbestez, ez da honi buruz analisi sakonik egingo memoria honetan.

## Zenbait kontzeptu garrantzitsu

Babelium Project neurri handiko proiektu bat izanik, ulertu beharrezko kontzeptu asko daudela nabarmen suertatzen da. Ez dira guztiak azalduko, baina proiektu honetan sarri aipatuko direnak aztertzea beharrezkoa da, garrantziaren arabera gehiago edo gutxiago sakonduz.

### Plataforma anitzeko garapena

Plataforma anitzeko garapena gaur egungo telefono mugikorrei zuzendua dagoen kontzeptua da. Gehien erabiltzen diren sistema eragileetan (Android, iOS eta Blackberry) funtzionatzeko gai diren aplikazioak sortzea da helburua, kode berdina konpilatu eta sistema eragile desberdinentzat paketatuz.

Adobe Flex eta Adobe AIR-i esker, aipatutako sistema eragileetan funtzionatu dezaketen aplikazioak kodetu eta garatu ditzakegu, eta gure kasuan, Adobe Flash Builder garapen-ingurune integratua (IDE, edo *Integrated Development Environment*) erabiliz eramango dugu aurrera gure zeregina.

### Android sistema eragilea

Android sistema eragilea Linux-en oinarritutako eta batez ere telefono mugikorretan erabiltzen den sistema da, ukipen-pantailetan erabiltzeko diseinatua batez ere. Android, Inc enpresak hasi zuen sistema eragile honen garapena, Googleren babes ekonomikoa jasoz hasieran, eta honek enpresa erosiz 2005. urtean.

2007. urtean izan zen aurkeztua Android, eta gailu eramangarrien kode irekiko estandarren normalizazioan garrantzi izugarria izan du.



Android erabili zuen lehenengo telefono mugikorra HTC Dream modeloa izan zen, 2008ko Urrian saltzen hasi zena.

Android sistema eragilearen arkitektura maila desberdinetan deskonposatu daiteke:

- **Aplikazio-maila:** erabiltzaileak era grafikoan ikus ditzakeen aplikazioak. Beste batzuen artean, hurrengoak ditugu: home, mezuak, kamera, nabigatzailea, argazki-albumak, ...

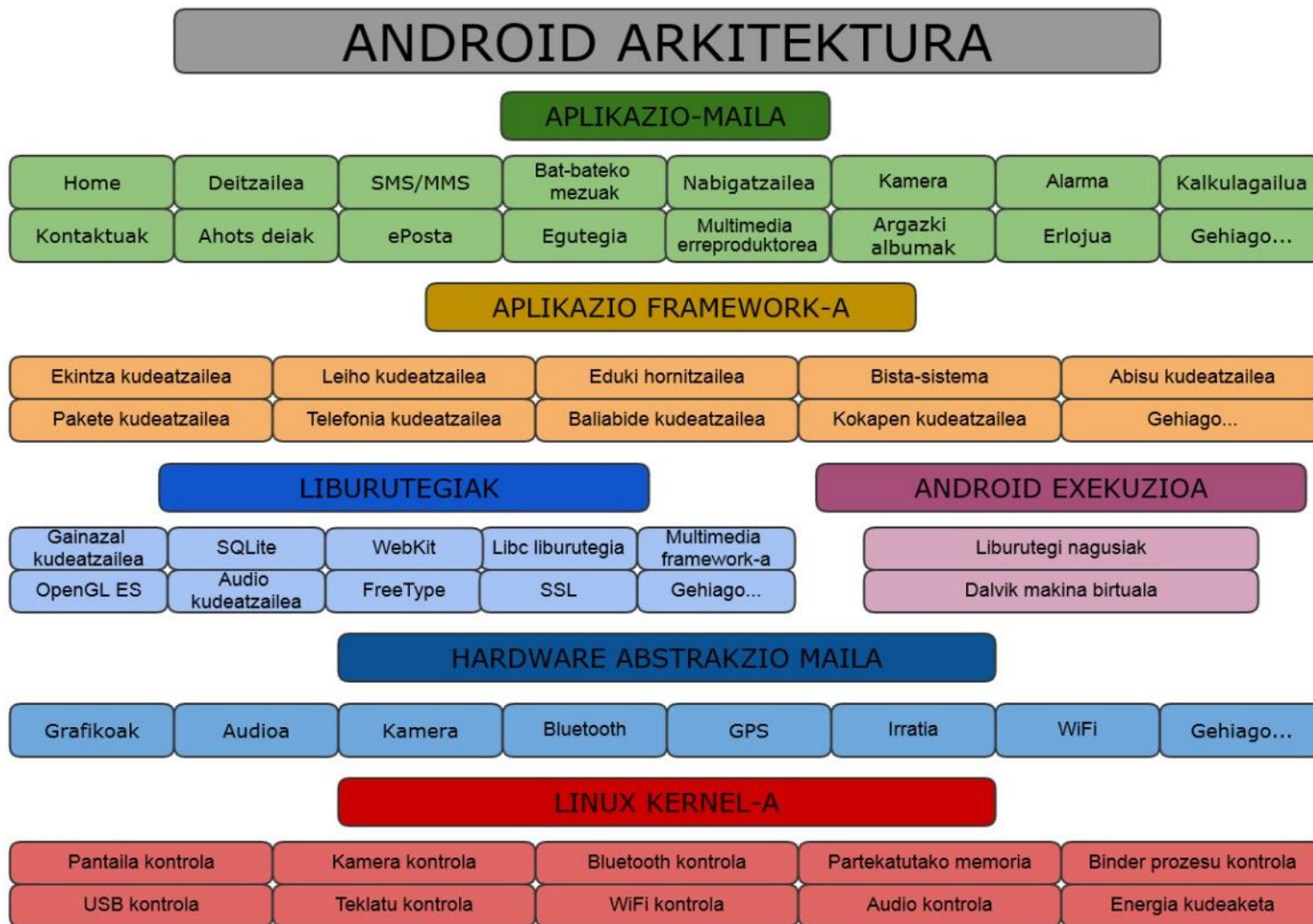
- **Aplikazio framework-a:** garatzaileek berrerabil ditzaketen APIak hemen sartzen dira, framework honen segurtasun arauak errespetatu ezker. Berrerabilpen hori bermatzeko dago prestatuta Android-en arkitektura, aplikazio baten ezaugarriak beste aplikazioekin partekatzea ahalbidetuz.

- **Liburutegiak:** sistemaren hainbat osagaik erabiltzen duten C/C++ liburutegi-sorta bat du Android, software garatzaileek hauek erabil ditzaten: System C library, 3D, SQLite, eta abar.

- **Android exekuzioa:** maila honetan Dalvik makina birtuala erabiltzen duten prozesu guztiak daude, bakoitza Dalvik-en instantzia berri bat sortuz. Makina birtual mota hau hainbat makina desberdin prozesatu ahal izateko programatu da, memoria minimoa erabiliz.

- **Hardware abstrakzio maila eta Linux kernel-a:** maila baxuenean, Android Linuxekiko oso independentea da, oinarrizko zerbitzu asko hortik baitatoz: memoria eta prozesuen kudeaketa, sarearen pila, kontrolatzaile-eredua, segurtasuna, eta abar. Horrez gain, nukleoak beste rol bat jokatzen du: software eta hardwarearen arteko abstrakzio-geruza gisa funtzionatzen du.

Maila guzti hauen irudikapen grafiko bat ikus dezakegu jarraian, 5. irudian:



5. irudia: Android arkitektura

## RIA, edo Rich Internet Applications

*Rich Internet Applications* izenekoak, edo RIAk (euskaraz Internet Aplikazio Aberastuak), mahaigaineko aplikazio tradizionalak dituzten ezaugarri gehienak mantentzen dituzten aplikazioak dira. Web nabigatzaile estandar bat erabiltzen da hauek exekutatzeko, eta osagarri edo makina birtualen bitartez gehitzen dira bestelako ezaugarri asko. Izan ere, web eta mahaigaineko aplikazioen onurak konbinatzeko asmoarekin sortu ziren, eta erabiltzailearen esperientzia eta produktibitatea indartzeko helburua dute.

Normalean, web aplikazioetan, orrialdeak era jarraituan kargatzen dira erabiltzaileak esteka bat klikatzen duenean, sarri gertatzen den ekintza bat izanik. Honen ondorioz, bezero eta zerbitzariaren arteko sekulako trafikoa ematen da, eta gehienetan aldaketa oso txikiak jasaten dituen orri bat behin eta berriro kargatzen da.

RIA inguruneetan, aldiz, ez dira karga jarraitu hauek etengabe gertatzen, baizik eta aplikazio osoa hasieran kargatzen da, eta zerbitzariarengako komunikazioa momentu puntualetan ematen da, adibidez, kanpoko datu base edo fitxategietatik informazioa atzitu behar denean.

## Arkitektura

Arkitekturari dagokionez, orokorrean, *stateful* deritzon bezero-aplikazio bat dugu, eta bereizitako zerbitzu-maila bat. Bestalde, aplikazio mota honetan zerbitzu hauek integratzen dira eta bezeroak badaki bere egiturari buruz, eta zein datu eskatu behar dituen. Horrela, datuak bakarrik transmititzen dira eta ez da bestelako informaziorik behar.

## Bezeroa

Erabiltzaile-interfaze eta erabiltzailearen arteko interakzioa kontrolatzen da. Erabiltzaileak komando-deiak egin, bistak eguneratu eta datuak kargatzen ditu. Aplikazioaren egoera monitorizatzen da eta zerbitzariarenganako eskaerak kudeatzen dira, aldi berean datuen aurkezpena ere kontrolatuz.

## Zerbitzaria

Hemen bezeroaren eskera guztiak jaso eta prozesatzen dira, eta zerbitzariari pasatzen dio eginbeharrekoen erantzukizuna. Hauek, mota honako eginkizunak izan daitezke: datu basean idazketak eta irakurketak egitea, sistemaren fitxategiak eguneratzea, prozesu analitikoak burutzea... Azkenik, bezeroari bueltatzen zaizkion datuen formatua aukeratu eta gauzatzen du ere.

## Zertarakoak

Erabilgarritasunaren muina erabiltzailearen mahaigaineko aplikazioen aurretiko ezagutzak aprobetxatzea da: kopiatu, moztu eta txertatzea, ordenatzea, ilarak eta errenkadak txikitu eta handitzea... multimedia eta aplikazioen onurekin nahasiz, aplikazio ikusgarri eta erraz erabiltzekoa dena lortuz.

Ikus-nabarmentasunari dagokion hobekuntza honekin batera, aplikazioaren erabilera neurri handian erraztuz, ingurumen desberdinekin konektatu eta elkarri eragiteko ahalmena dator, atzipen guztiak azkartuz,

eta aurkezpen-mailarekiko independentzia emanez. Horrela, munduko edozein gailutik aplikazioa erabiltzeko ahalmena du erabiltzaileak.

## Abantailak

Hauexek dira, azken finean, RIAk eskaintzen dituzten abantaila batzuk:

- Ez dute instalaziorik behar, web nabigatzailea eguneratua mantentzeaz nahikoa da.
- Aplikazioen bertsio-eguneratzea automatikoa da.
- Sistema eragilearekiko independenteak dira, internet konexioa behar da soilik.
- Orokorrean birusekin infektatzeko aukera txikiagoa da, fitxategi exekutagarriekin ez bezala.
- Erantzuteko ahalmen handiagoa, erabiltzaileak zerbitzariarekin zuzeneko erlazioa baitu, orrialdea berriro kargatzeko beharra deuseztatuz.
- HTML soilik erabiliz garatu ezin diren aplikazio interaktiboak eskaintzen ditu, eta zerbitzariari informazioa bidali gabe orria eraldatzeko aukera emanez.
- Nabigatzaile desberdinen erabilpenaren ondorio diren arazoak alde batera uzten ditu, framework bat erabiliz.

## ARM arkitektura

Gaur egun, telefono mugikor gehienek barneko egitura oso antzekoa dute haien artean. Egitura hau gaur egun gehien erabiltzen den arkitekturari esker da, ARM arkitektura, hain zuzen. ARM, *Advanced RISC*

*Machine*-tik hartzen du bere izena, eta RISC akronimo hau *Reduced Instruction Set Computer*-etik dator, PUZ diseinu mota bati dagokiona.

ARM prozesatzaileen sinpletasun erlatiboa oso egokia da potentzia txikiko sistema edo aplikazioetan erabiltzeko. Honi esker, lehen aipatu bezala, elektronika mugikor eta integratuaren merkatuan teknologia nagusi bilakatu da, mikroprozesatzaile eta mikrokontrolagailu txikiak diren gailuekin, orokorrean kontsumoa eta kostuari dagokionez oso ezaugarri lehiakorak erakutsiz.

ARM arkitektura lizentziagarria da, eta gaur egun, ARM lizentziak dituzten enpresa batzuk hauek dira: Apple, Microsoft, Nintendo, Sony, Samsung, Yamaha, Nokia, LG, Intel, Broadcom... Horrez gain, ARM prozesatzaileak, ARM enpresa beraz gain, lizentzia hauen jabeek ere garatzen dituzte.

Unix (Mac OS X eta Solaris), Linux (Android, Ubuntu, Debian, Fedora), BSD (FreeBSD, NetBSD, OpenBSD), eta Windows 8-rekin, beste batzuk barne, bateragarria da. ARM arkitekturan oinarritutako lehen ordenagailu pertsonala Acorn Archimedes-a izan zen, 1987ko Ekainean kaleratua.

## RTMP protokoloa

RTMP protokoloa (Real Time Messaging Protocol) audioa, bideoa eta beste datu-formatu batzuk interneten bidez igorri eta jasotzeko erabiltzen da, Flash aplikazio eta zerbitzari baten arteko komunikazioa ezarriz. Macromedia enpresak garatu zuen protokoloa hasieran, baina Adobek Macromedia erosi eta gero amaitu gabeko bertsio bat kaleratu zuen, protokoloak erabilera publikoa jaso zezan.

RTMP protokoloak zenbait antzeko bertsio dauzka: RTMP bertsio soila maila altuenekoa da eta 1935 zenbakiko TCP portua erabiltzen du. Beste hainbat daude, adibidez RTMPS, TLS/SSL konexio baten bidezko RTMParen aldakuntza bat dena. Gainera, RTMPE eta RTMPT ere erabil daitezke.

Protokolo hau hainbat motako zerbitzari eta proiektutan inplementatuta ikus daiteke: nginx zerbitzarirako RTMP modulu bat, Adobe Flash Media Server, FFmpeg proiektuaren barruan... kasu honetan, Red5 Media Server-aren barruan ikusiko dugu bere funtzionamendua.

## ActionScript

Adobe ActionScript 1997. urtean kaleratutako programazio-lengoaia da, Adobe Flash oinarriarena hain zuzen. Era interaktiboagoan programak garatzeko helburuarekin izan zen sortua hasiera batean. Mota askotako animazioak, bai sinpleak, bai konplexuak, eta datu eta interfaze aberatsak, beste ezaugarri batzuen artean, onartzen ditu. Gainera, eraginkortasuna indartzen du alor hauetan.

Gaur egun gehien hedatu den bertsioa ActionScript 3.0 da, objektuetara zuzendutako programazioan aurrerakuntza nabarmena suposatu zuena bere egunean. ECMA-262 (programaziorako arau-multzo espezifiko bat) estandarraren onarpenean doikuntza hobeagoa lortu zuen. Adobe Flash eta Flex-en azken bertsioetan erabiltzen da, eta Flex-en bertsio lehenagokoetan ere, Flex-ek bere 2. bertsiotik aurrera barne hartzen du izan ere. Honek errendimendua hobetu du aurrekoekin alderatuta, klaseak paketatze modu berriak eta testua aldatu eta tratatzeko adierazpen erregularren erabilera eskaintzen dute.

Flash objektuz osatzen da, bakoitza bere *path* edo bide-izen propioa izanik SWF (Shockwave Flash) objektuaren barruan. Hauetariko bakoitza klase baten barne sailkatzen da. Hauek dira klase batzuk: MovieClip, botoiak, bektoreak edo zerrendak, eta abar. Klase bakoitzak, horrez gain, hainbat propietate eta metodo edo funtzio ditu.

- Propietateak: klasearen barnean aldagai moduan deklaratzeko dira. Adibidez: `length`, botoi baten neurriak (`width`, `height`), bideo baten iraupena...

- Metodo edo funtzioak: klasearen barnean deklaraturiko instrukzio-sekuentziak: `stop()`, `play()`, `setVolume(value)`...

## Adobe Flex

Adobe Flex, edo 2005. urteraino Macromedia Flex izenaz ezagutzen zena, RIA (Rich Internet Applications) edo Internet Aplikazio Aberastuen hedapen eta garapenerako teknologia multzo bat da, Flash plataformaren funtsa gisa hartuz. Programatzaileek ez dute animazioa programaziolengoaia batera moldatzearen erronka gauza erraztat hartzen, baina Flex-i esker arazo hau era dotorean saihesten da, lan-fluxu eta programazio-eredu lagunkoiago bat eskainiz. Gaur egun, Apache Flex izenarekin ezagutzen dugu, Adobek Apache Software Foundation-i eman baitzion 2011. urtean.

Hasiera batean, Javarekin erabiltzeko aplikazio moduan izan zen kaleratua, JSP (JavaServer Pages) liburutegien bidez Flex-en marka lengoaia (MXML) konpilatzen zuena, ondoren Flash aplikazioak ActionScript erabiliz exekutatzeko. Flex-en azkeneko bertsioek



konpilatutako fitxategi estatikoen sortzea onartzen dute, eta zerbitzari lizentzia ez da beharrezkoa hauek sarean libreki banatzeko.

Lehen aipatutako RIA hauek sortzea da Flex-en helburu nagusia, aplikazio garatzaileei haien xedera iristeko erremintak eskainiz. Geruzanitz eredu batean, Flex aplikazioak aurkezpen-maila osatzen dute. XML-ren antzeko MXML lengoaia erabiltzen da, GUI (Graphical User Interface) edo irudi-erabiltzaile interfazeen garapena bideratuz.

Are gehiago, Flex-ek osagai eta ezaugarri gehiago eskaintzen ditu: web zerbitzuak, urruneko objektuak, zutabe ordenagarriak, grafikoak eta animazio-efektuak... eta beste interakzio sinple asko. Bezeroak aplikazioa behin bakarrik kargatzen du, datuen fluxua hobetuz HTML aplikazioekin konparatzerako orduan. Flex-en lengoaia eta fitxategien antolamenduak logika eta diseinuaren arteko independentzia bilatzen du.

Flex zerbitzariak *gateway* edo pasabidearen eginkizuna ere burutzen du, bezeroari XML web zerbitzu eta urruneko objektuekin komunikazioa ahalbidetuz (Java klaseak, Coldfusion...). Adobe Flex-ek dituen alternatibak hurrengoak dira: Google Web Toolkit, JavaFX, OpenLaszlo, Microsoft Silverlight...

## Adobe AIR

Adobe AIR (Adobe Integrated Runtime), gako izentzat *Apollo*, RIA aplikazioak garatzeko erabiltzen den plataforma anitzeko exekuzio-ingurumena da. Aplikazio hauek Adobe Flash, Adobe Flex, HTML eta AJAX erabiltzen dute mahaigaineko aplikazio gisa funtzionatzeko.

2007ko martxoaren 19an Adobe enpresak AIR teknologiaren aurretiazko bertsio bat kaleratu zuen, SDK (Software Development Kit) bat eta Flex-ekin erabiltzeko Apollo aplikazioen garapenerako gehigarri bat. Gero AIR (Adobe AIR) izena eman zitzaion eta exekuzio-ingurumenaren beta bertsio bat merkaturatu zen. Azkenik, 1.0 bertsioa 2008ko Otsailaren 25ean kaleratu zen. Gainera, hurrengo hilean Linux-entzat alfa bertsio bat ere kaleratu zen.

Nabigatzailean implementatzen diren RIAk ez dute instalaziorik behar, baina Adobe AIRen implementatutako aplikazioak erabiltzailearen sisteman desentzaketatu eta instalatzen dira. Honek fitxategi-sistema atzitzea posible egiten du, ez nabigatzaileetan bezala, non datuak gordetzeko aukerak askoz murriztagoak diren. Gehienetan haien zerbitzarietan gorde behar dituzte datuak, aldiz, fitxategi-sistema lokala eskuragarri izateak malgutasun handiagoa ematen dio erabiltzaileari.

Adobe AIR-en egindako aplikazioak JavaScript-en programatu daitezke haien osotasunean. Nabigatzaileen arabera eraldatu zen lengoia, kodearen exekuzio dinamikoarekin bateragarria izateko, izan ere, ezin baitzen hau egin nabigatzailean egiten zen exekuzioaren barne. Horrela, urrutiko kode maltzurak erabiltzailearen sistemari eraso egitea saihesten da, eta hasieran JavaScript-ek erabiltzen dituen funtzio batzuk, adibidez, *eval()* funtzioa, ez ziren AIR-ekin bateragarriak.

## Cairngorm mikroarkitektura

**Zer da Cairngorm?** Cairngorm MVC ereduan (Model View Controller) oinarritzen den mikro-arkitektura arina da, AIR edo Flex-en eraikitako aplikazioetan erabiltzen dena. Hiru zatitan banatzen da: eredua, bista, eta kontrolatzailea (*model*, *view* eta *controller*, hain zuzen). Hurrengo ezaugarriak dira esanguratsuenak Cairngorm mikro-arkitekturari dagokionez:

- RIA (Rich Internet Applications) motako aplikazioetara zuzenduta dago, Flex framework-a erabiliz aplikazioen garapen azkarra bideratuz.
- Talde-garapena errazten du.
- Kodea banatu eta antolatze beharra duten proiektu ertain eta handietan erabiltzeko dago gehien bat prestatua.
- Cairngorm framework-ak berak bermatu eta bultzatzen ditu lan metodo onenak lehen aipatutako Flex oinarria duten RIAk garatzeko.

Nahiz eta hasiera batean Cairngorm framework-a Adobe-ren sorkuntza bat ez izan, gaur egun, Cairngorm kode irekiko proiektu moduan aurkitu daiteke Adobe Open Source webgunean, hurrengo URLa erabiliz: <http://opensource.adobe.com/wiki/display/cairngorm/Cairngorm>. Izan ere, *iteration::two* izeneko konpainia batek sortu zuen Edinburgh-eko hirian, eta geroago Macromedia enpresak erosi zuen, gaur egun Adobe Consulting moduan ezagutzen dugun horretan bilakatuz.

Zein da Cairngorm-en osakera? Lehen esan bezala, Cairngorm MVC ereduan oinarritzen da. Bezero eta zerbitzari artean egoera eta informazio sinkronizazio konplexua errazteko zehazki diseinaturik dago. Bitartean,

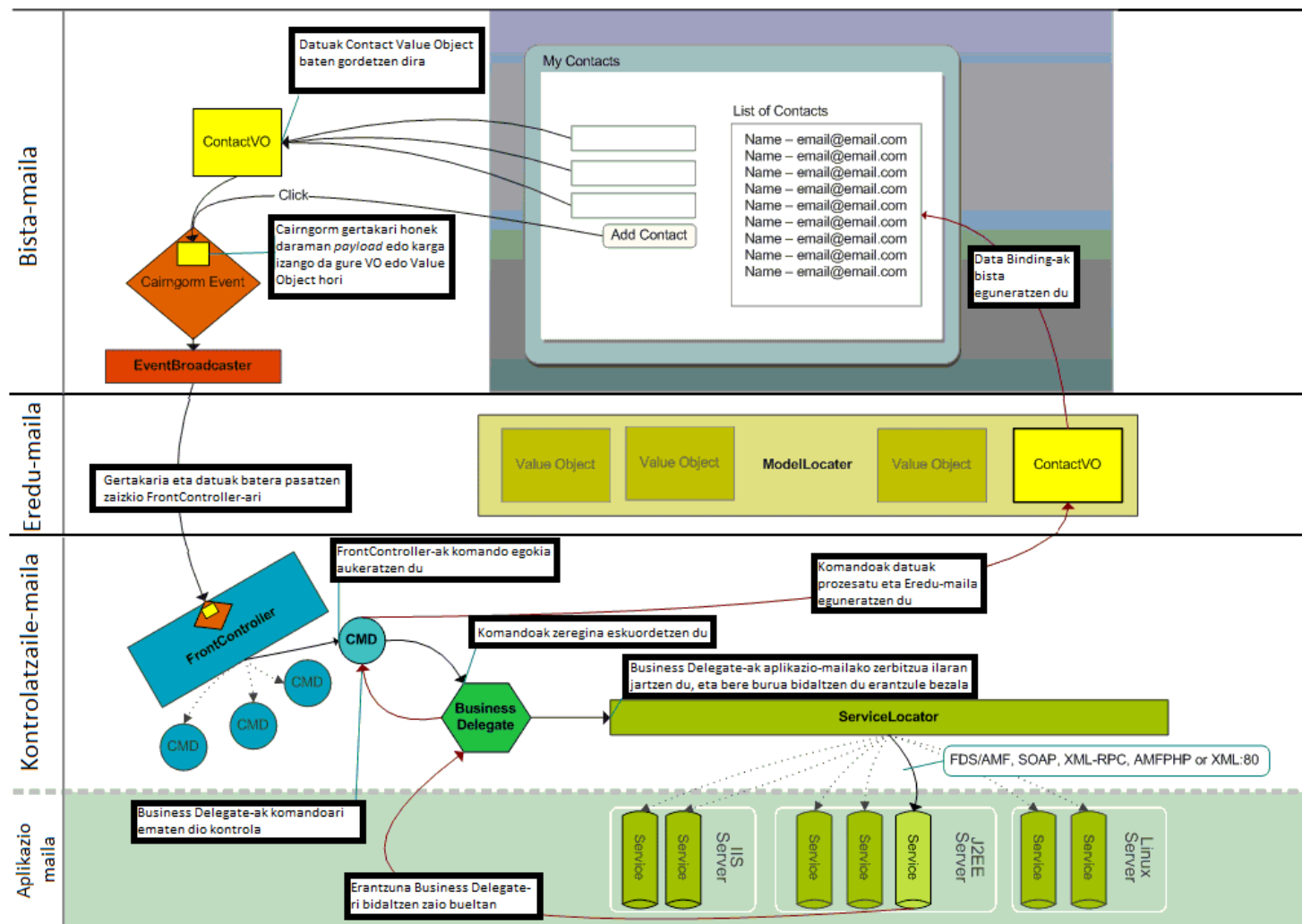
datu-egituren implementazioa erabiltzaileak ikusiko dituen bista-geruzatik at mantentzen du.

Bista geruza edo maila honen zeregina hurrengoa da: eredu mailan gordetzen den informazioaren sinkronizazioa bermatzen du, eta aldi berean ematen diren gertakizunak, *listener* baten antzera, entzun eta haien araberako erantzunak emango ditu.

Informazioa eta datuak VOetan (Value Objects) gordetzen dira, eta aplikazio-mailako aldagai sinpleak ModelLocator izeneko klase batean gordetzen dira, honen propietate zuzenak balira bezala. Gauzak horrela, estatikoki erreferentziatzen da ModelLocator hau, singleton instantzia bat erabiliz, eta bista-mailak instantzia hau berrerabiliko du datuak atzitu eta eskuratzeko.

Azkenik, kontrolatzailea dugu, Cairngorm arkitekturaren zatirik konplexuena, eta singleton patroiarekin ere implementatzen da, FrontController izena jasoz. Kontrolatzaile honek bista-mailak sortzen dituen gertakari guztietaz arduratuko da, eta ezarritako Command klase bati pasatuko dizkio gertakari honen izaeran oinarrituz.

### CAIRNGORM ARKITEKTURA



6. irudia: Cairngorm mikroarkitektura

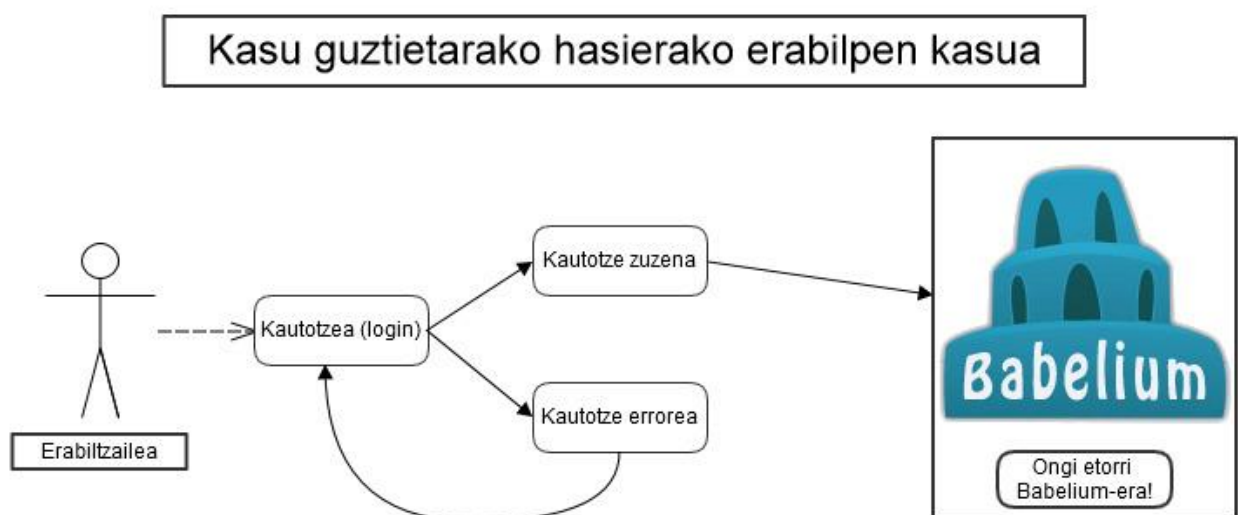
## Betekizunen bilketa

### Erabilpen kasuak

Erabiltzaile batek Babelium Project-en barruan hainbat ekintza izango ditu aukeran, ikaskuntza-prozesu globalaren barne. Hemen azalduko dira zeintzuk dira gehitu diren funtzionalitateak, eta hauetara heldu ahal izateko jarraitu behar den ordena.

### Kasu guztietarako hasierako erabilpen kasua

Proiektu hau ezerezetik hasi ez denez, hau da, hasita zegoen lan batetik jarraitu baitenez, era sinpleenean azalduko da garatutako funtzionalitateetara heltzeko bidea. Egia esan, ez da gehiegi egin behar, *login* edo kautotze soil batekin ebaluazioei dagokion aukera gehienak gure eskumenean izango ditugu. Hona hemen kautotze-prozesuaren erabilpen kasuaren ereduak:



Erabiltzaileak, aplikazioa hasieratzen badu, kautotze-pantaila ikusiko du jarraian. Erabiltzaile-izena eta pasahitza eskatuko dira, hauek biak balioztatzeko botoi baten gainean. Horrez gain, guzti honen azpian, erabiltzaile-konturik ez badago sortuta momentu horretan, Babelium Project-en kontua sortzeko web orrialdera eramaten duen esteka bat erabil dezakegu.

The image shows a login screen for 'Babelium Project'. It has a solid blue background. At the top center, there is a small icon of a building with the word 'babelium' below it, followed by the text 'BabeliumProject' in a large, white, sans-serif font. Below this, there are two white rectangular input fields with rounded corners. The first field contains the placeholder text 'User Name' in a light gray font. The second field contains the placeholder text 'Password' in a light gray font. Below these fields is a blue button with rounded corners and a gradient, containing the text 'Enter' in white. At the bottom center, there is a link that says 'Register Me' in a white, sans-serif font.

*7. irudia: Kautotze-pantaila*

Kautotze-prozesua arrakastaz burutzen badugu, Babelium Project-en barruan sartzea lortuko dugu eta pantaila nagusia bistaratuko da, erabiltzaileek igo dituzten azken bideoak zerrendatuz:

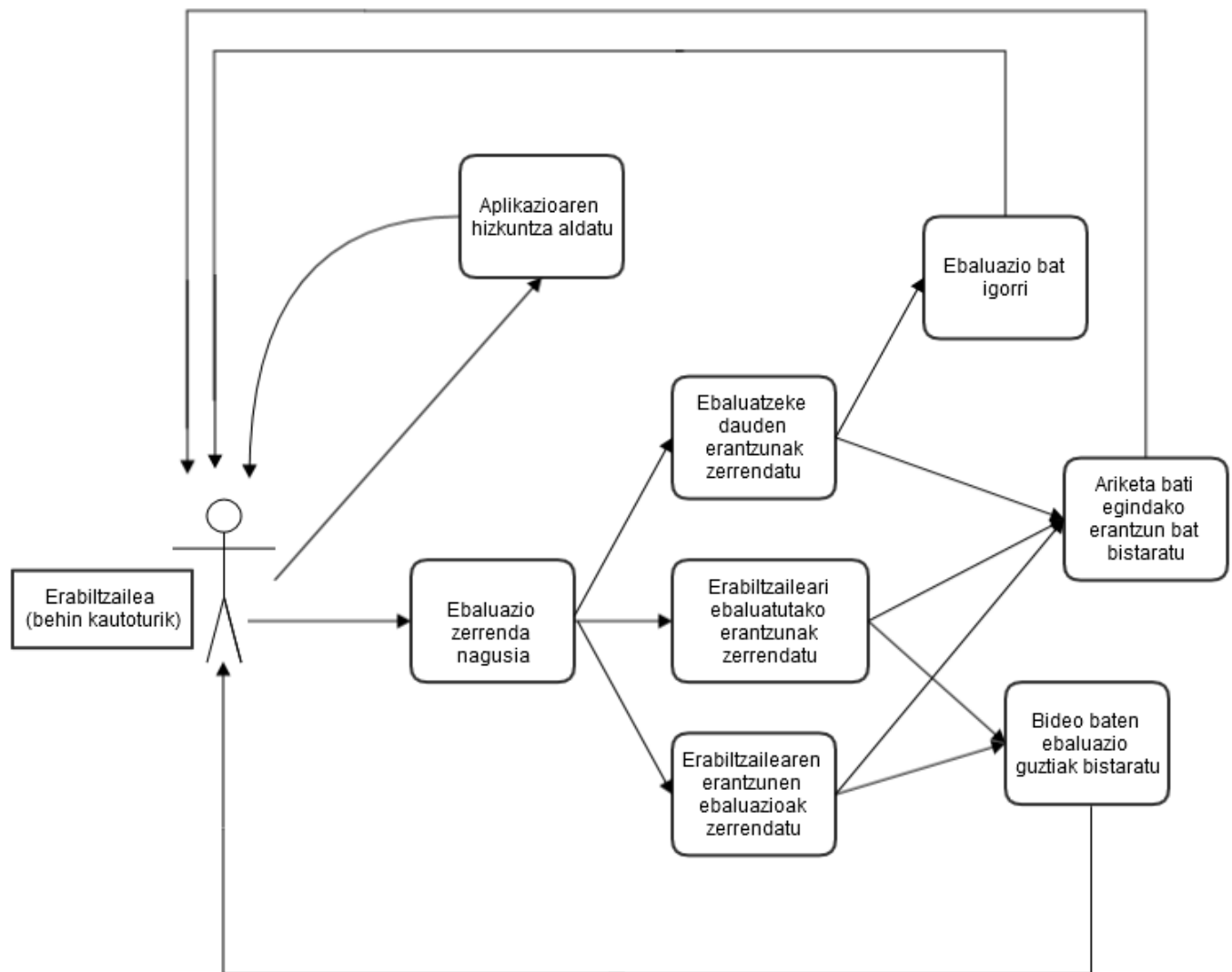


#### 8. irudia: Igotako azken bideoak

Behin kautotze-prozesua era egokian burutu dugularik, GAL honetan Babelium Project aplikazio mugikorrarentzat garatu diren funtzioak azaltzeko prest gaude.



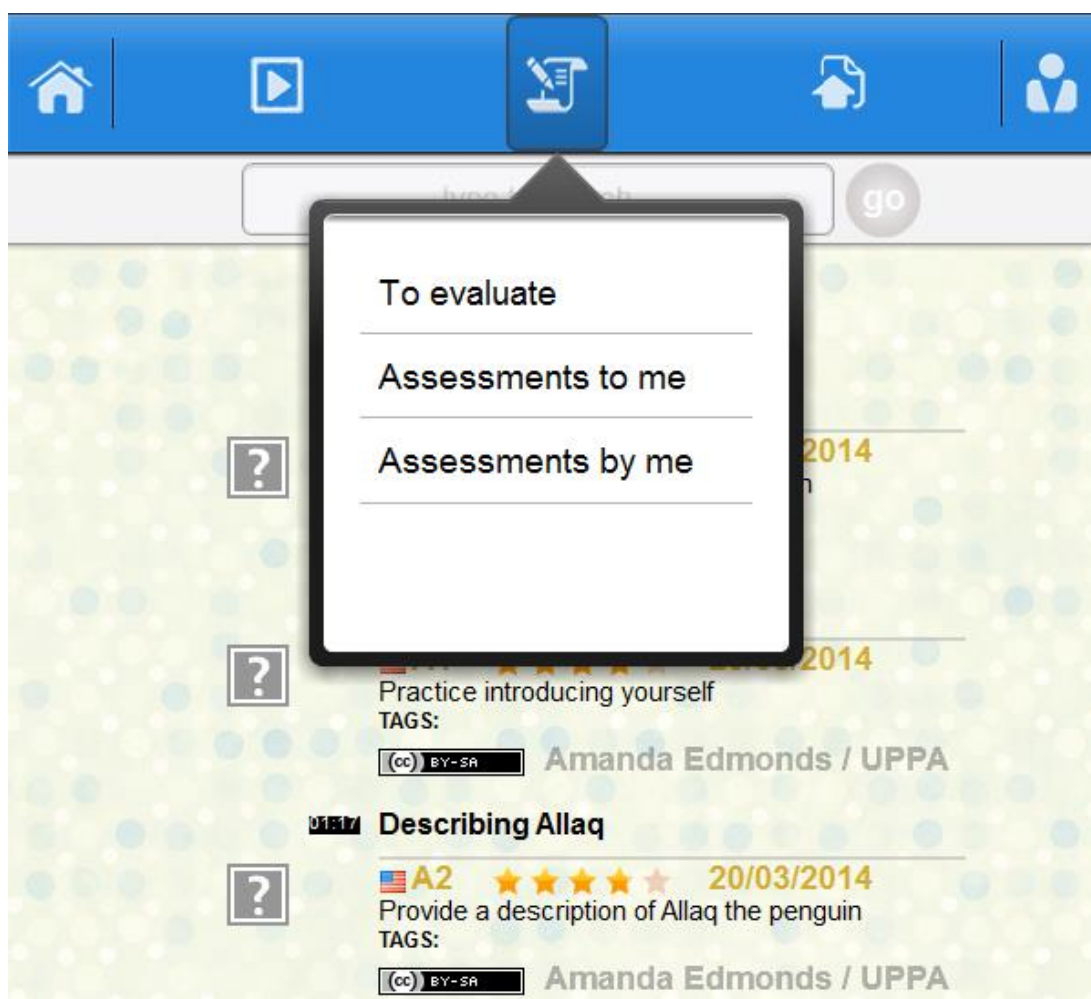
## Erabilpen kasuen eredua



9. irudia: Erabilpen-kasuen eredua

## 1. Ebaluazio zerrenda nagusia

Hemen edozein ariketa aukeratu daiteke erantzun baten grabaketa hasteko, baina ebaluazioarekin erlazionatutako erabilpen kasuetan jarriko dugu arreta. Ebaluatzeko botoia goiko parteko erdialdean kokatzen da, boligrafo eta paper baten ikonoaz errepresentaturik. Botoia sakatu ezkerre, hurrengo menua hedatuko da:



*10. irudia: Ebaluazio zerrenda nagusia*

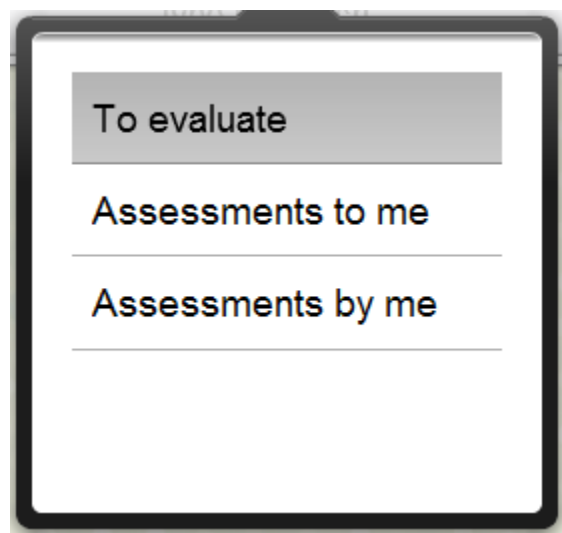
Hiru aukera desberdin ikusiko dira zerrendaren barruan:

- **To evaluate:** botoi hau sakatu ezkerre, oraindik ebaluaziorik jaso ez duten erantzun guztiak agertuko dira zerrenda batean, nahi duguna aukeratu eta ikus dezagun.

- **Assessments to me:** erabiltzaileari egin zaizkion ebaluazioen bideo-zerrenda bistaratzeko sakatu dezakegu botoi hau.

- **Assessments by me:** hirugarren aukera hau hartzen badugu, erabiltzailearen ebaluazioa jaso duten bideo-erantzun guztiak zerrendatuko dira.

## 2. Ebaluatzeke dauden erantzunak zerrendatu



*11. irudia: "To evaluate" aukera*

**To evaluate** aukera hautatzen bada, erabiltzaileak oraindik ebaluatu ez dituen bideo-erantzun guztien zerrenda bat ikusiko du pantailan. Bideo-erantzun bat jada ebaluatu badu, ez da zerrendan agertuko. Hurrengo moduan bistaratzen dira erantzunak:



#### 12. irudia: Ebaluatu ez diren bideoak

Zerrenda honetatik bideo bat aukeratzen denean, ebaluazioak osatu eta bidaltzeko pantaila zabalduko da.

### 3. Ariketa bati egindako erantzun bat bistaratu

Ariketa bati egindako erantzun bat ikusi ahal izateko, aurreko zerrendako edozein bideo aukeratuko dugu. Hauek oraindik ebaluatu ez

ditugun audio edo bideo-erantzunak dira, baina ebaluatutakoak berrikusi ditzakegu **CalloutButton**-aren beste aukeretan klik eginez.

## 4. Ebaluazio bat igorri

Ebaluazio bat igorri ahal izateko, erabiltzaileak **to evaluate** zerrendatik erantzun bat aukeratu behar izango du, bere gainean klikatuz. Hau egitean, bista berri bat zabalduko da:



### 13. irudia: Ebaluatzeko pantaila

Ebaluatzeko pantailan erabiltzaileen bideo-erantzunen (edo soilik audioaren) ebaluazioa egin daiteke. Erabiltzaileak erakutsitako trebetasunaren arabera, balorazio pertsonal bat emango da, hurrengo parametroetan oinarrituz:

- Berezkotasuna
- Hitz-jarioa

- Intonazioa
- Erritmoa
- Puntuazio orokorra.

Bost parametro hauek ez baloratu ezker ezin izango da ebaluazioa bidali, bostak puntuatu behar dira. Horrela egin ezean, errore mezu bat pantailaratuko da.

Horrez gain, testu-iruzkin bat gehitu daiteke, **checkbox** botoi batekin aktibatzen dena. Iruzkin hau gure ebaluazioari esleituta agertuko da etorkizunean erabiltzaileek ebaluazioak bistaratzen dituztenean. Testu-iruzkin hau bidaltzea hautazkoa da, baina **checkbox**-a aktibatu ezker ezin izango da ebaluazioa bidali testua hutsik badago.

Ebaluazioa bideoa ikusten den heinean betetzen joan daiteke, bideo erreproduzitzailerik dugulako bista berdinean. Era berean, ebaluatzailearen hizkuntza-maila ariketaren mailaren azpitik badago, ezin izango du erantzuna ebaluatu.

#### 4. 1. Erabiltzailearen hizkuntza-maila balioztatu

Ebaluazio bat egin eta bidali ahal izateko, logikoa da baloratzen den ariketaren hizkuntza ezagutzearen beharra, bestela egiten den ebaluazioa ez da sinesgarria izango. Babelium-ek dituen erabiltzaile guztiek ezagutzen dituzten hizkuntzen eta hauen ezagutza-maila gordetzen duen zerrenda bana dute. Ariketaren hizkuntza zerrenda horretan agertuz gero, eta bere maila berdindu edo gaitzitzen badu erabiltzaileak, orduan saiakera baloratu ahal izango du.

Ostera, hizkuntza hori bere zerrendan ez badago, edo zerrendan badago baina maila nahikoa ez badu, ezin izango du erantzunari buruz bere iritzia eman: errore mezu bat agertuko da hizkuntza-maila baxuegia

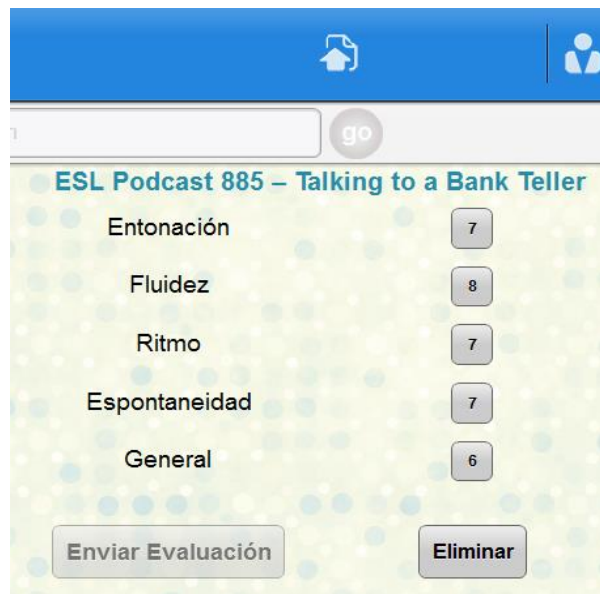
dela esanez, eta ebaluazioa bidaltzeko botoia desgaituko da. 14. irudian ikus dezakegu hau:



*14. irudia: Hizkuntza-maila mezua*

Ikusten denez, “tu nivel de idioma es inferior” mezua agertzen da, eta “enviar evaluación” botoia disabled egoeran aurkitzen da orain. Konprobaketa hau edozein ebaluazio bidaltzen saiatzen garenean burutuko da, baina soilik behin erantzun bakoitzeko.





ESL Podcast 885 – Talking to a Bank Teller	
Entonación	7
Fluidez	8
Ritmo	7
Espontaneidad	7
General	6

Enviar Evaluación      Eliminar

*15. irudia: Ebaluazio botoia desgaituta*

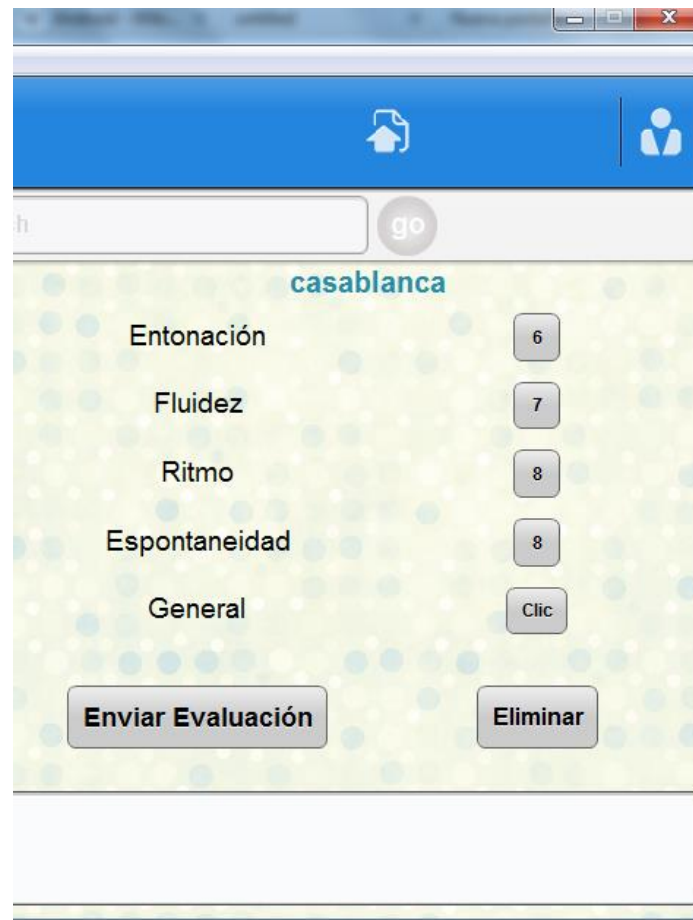
Bideo-zerrendara bueltatu eta berriro sartzen bagara, konprobaketa honen emaitza ez da gogoratuko eta berriro egingo da ebaluatzen saiatu ezkerro.

## 4.2. Ebaluazioa balioztatu

Gure hizkuntza-maila konprobatzeaz gain, interesgarria da ebaluatzaileek ariketa bati buruzko aspektu guztiak kontuan har ditzaten. Horregatik, ebaluatzen diren bost aspektuetatik bat baloratu barik uzten bada, sistemak ebaluazioa baztertu behar du eta erabiltzaileari osorik igorri dezan eskatu.

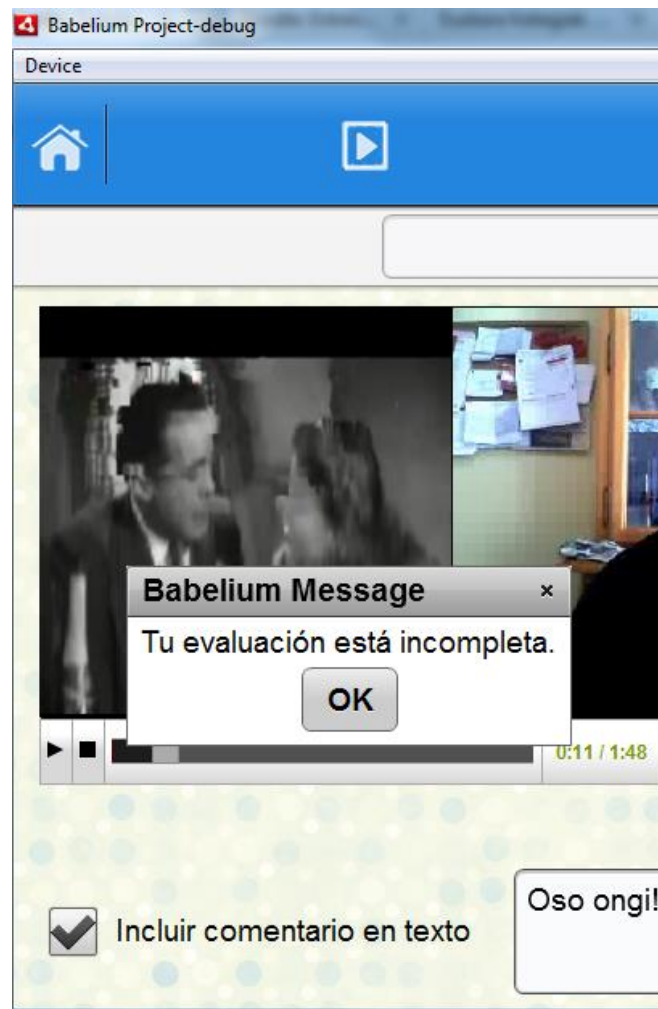
Erabiltzaileak hurrengo ebaluazioa bidaltzen duela demagun:





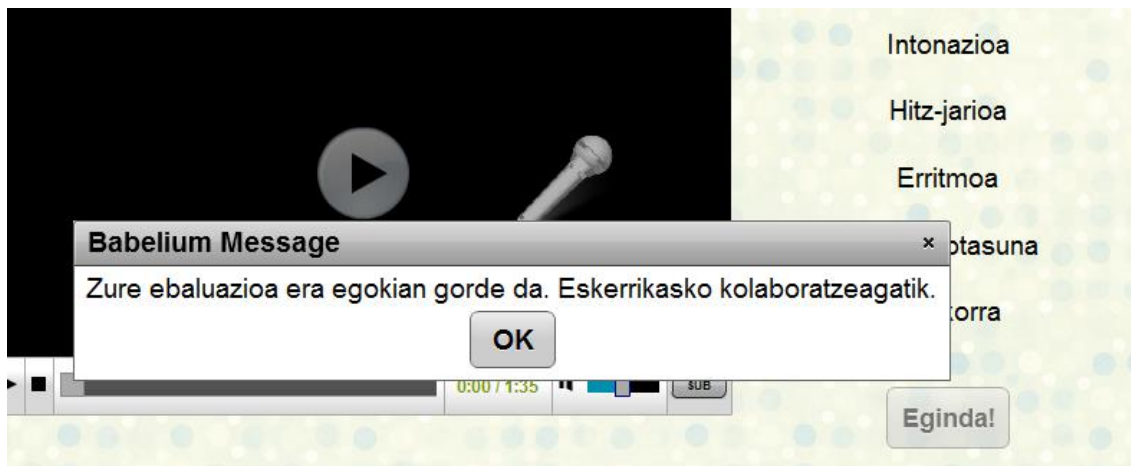
*16. irudia: Amaitzeke dagoen ebaluazioa*

Lehenengo lau atalak baloratu dira, baina azkena, batezbestekoarena, baloratu gabe utzi da. Horrelako ebaluazio bat ez dago osorik, beraz, hurrengo mezua agertuko da pantailan:



*17. irudia: Ebaluazio desegokiaren mezua*

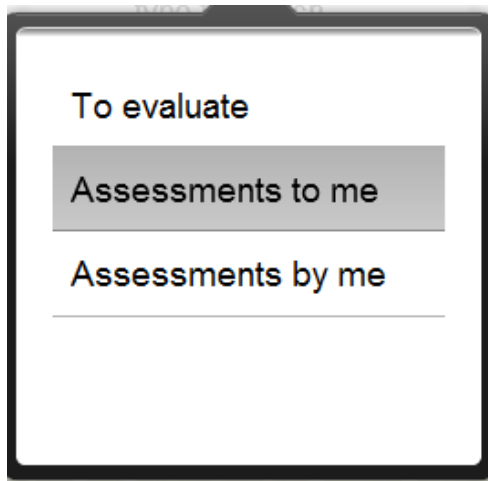
OK botoian sakatu eta ebaluazioa osotzeko aukera izango dugu. Batezbestekoaren puntuazioa ezarriko dugu eta ebaluaketa bidaltzeko botoia berriro sakatu:



*18. irudia: Ebaluazioa ondo gorde da*

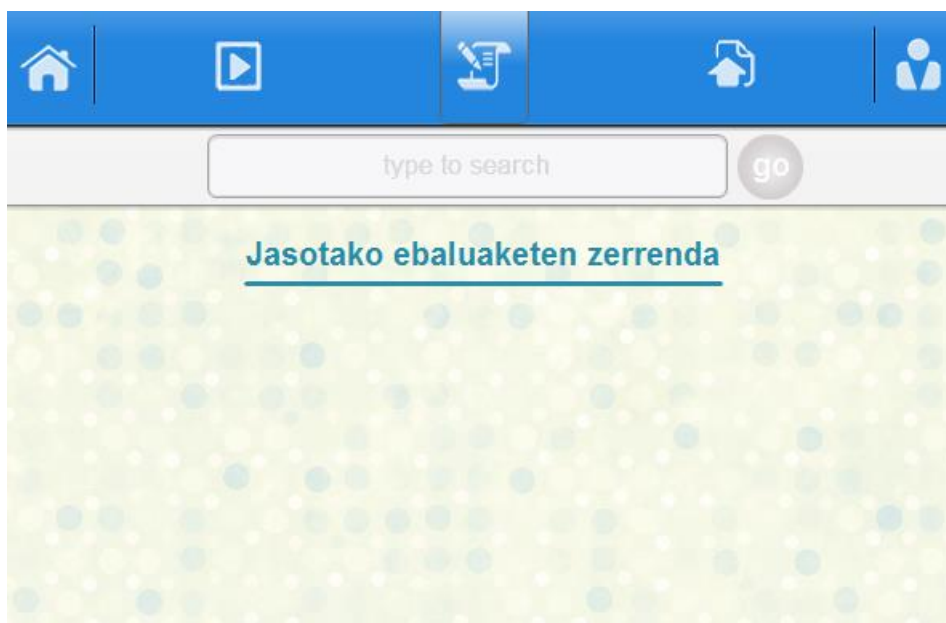
Horrela, ebaluazioa bidalita geratuko da eta erabiltzaile guztientzat publikoa izango da, erantzun baten ebaluazioak zerrendatzean besteekin batera agerraraziz. Ebaluazio bakoitza nork egin duen jakingo dute erabiltzaile guztiek, behin erantzun hori ebaluatu dutenean.

## 5. Erabiltzaileari ebaluatutako erantzunak zerrendatu



*19. irudia: "Assessments to me" aukera*

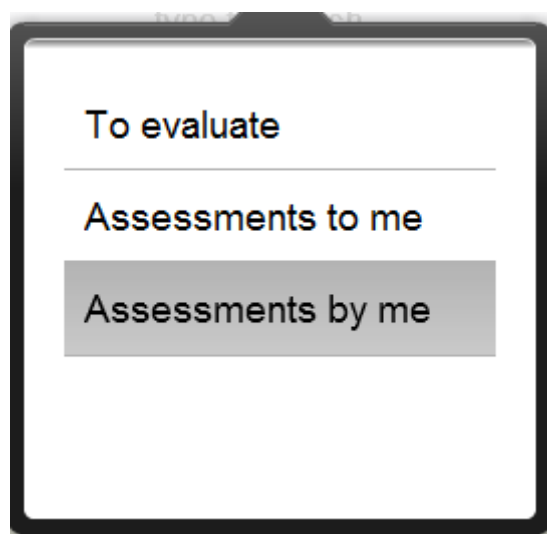
**Assessments to me** aukera hautatzen bada, erabiltzaileak bere saiakerei egindako ebaluazio guztien erantzunen zerrenda bat ikusiko du. Zerrenda honetan soilik bideoak zerrendatuko dira, **To evaluate** aukeran gertatzen den bezala.



*20. irudia: Jasotako ebaluaketak*

Erabiltzaileak ez badu audio edo bideo-erantzunik grabatu eta igo, edo erantzunik ez badu jaso oraindik, 20. irudian bezala agertuko da zerrenda. Ebaluazioak ikusi ahal izateko, bideoa aukeratu eta pantaila berri baten bistaratuko dira, hurrengo kasuan ikusiko den moduan.

## 6. Erabiltzailearen erantzunen ebaluazioak zerrendatu



*21. irudia: Assessments by me aukera*

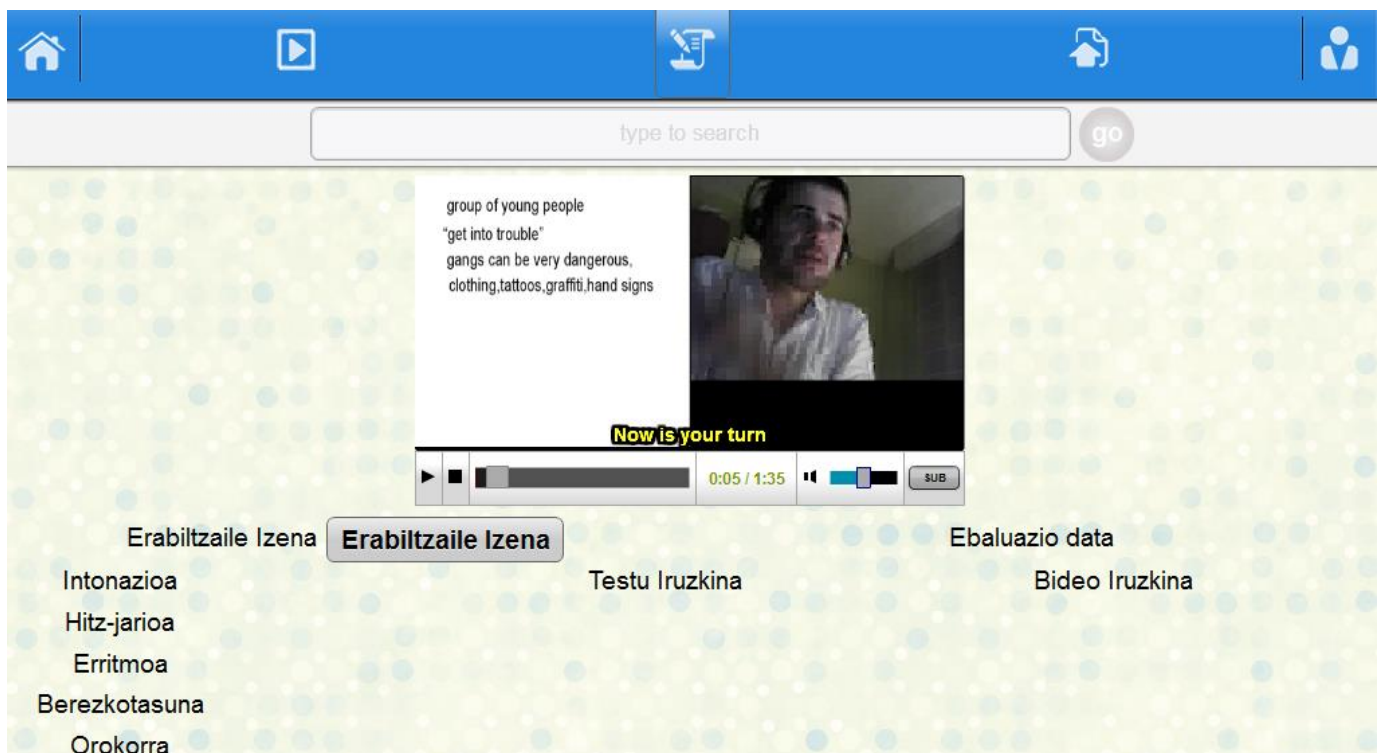
**Assessments by me** aukera hautatzen bada, erabiltzaileak berak ebaluatutako erantzunen zerrenda bat ikusiko du. Zerrenda honetan soilik bideoak zerrendatuko dira, aurreko **To evaluate** eta **Assessments to me** aukeretan gertatzen den bezala:



22. irudia: Egindako ebaluaketak

## 7. Bideo baten ebaluazio guztiak bistaratu

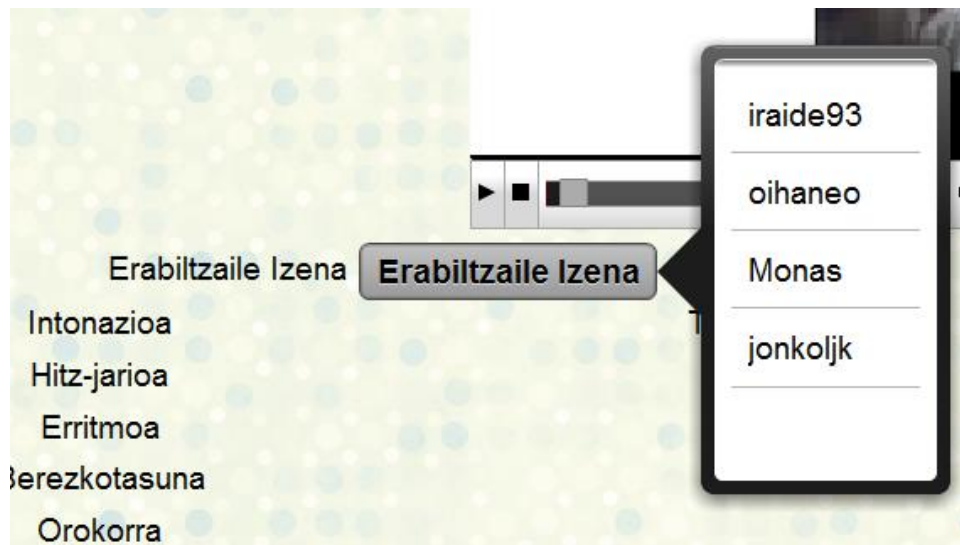
Erabiltzaile baten saiakerari egindako ebaluazioak ikusi ahal izateko, lehenengo ebaluatzaileak berak ebaluatu behar izan du ariketa. Berari egin badizkiote ebaluazioak, orduan ere ikusi ahal izango ditu, prozesua berdin-berdina da. Ebaluazioak ikusteko pantaila hurrengoa da:



### 23. irudia: Ebaluazioak bistaratzeko pantaila

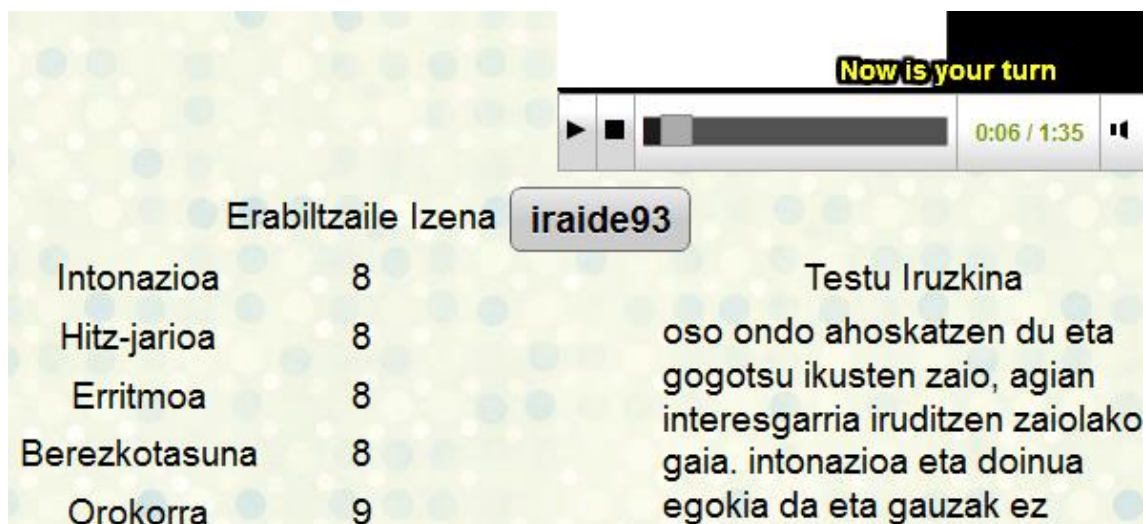
Hasieran zuri daude informazioa bistaratzeko erabiltzen diren eremu guztiak, erabiltzaileak bat aukeratzen duen arte. Ebaluazio bat aukeratzeko, "Erabiltzaile izena" duen botoia sakatu beharko da. Hau egitean, erabiltzaileen izenekin zerrenda bat hedatuko da, ikusi nahi den ebaluazioa aukeratu ahal izateko:





24. irudia: Ebaluatzaileen zerrenda

Adibidez, lehenengo erabiltzailearen ebaluazioa aukeratzen badugu, pantailan agertuko da egin zuen ebaluazioaren informazio guztia:



25. irudia: Ebaluazio baten informazioa

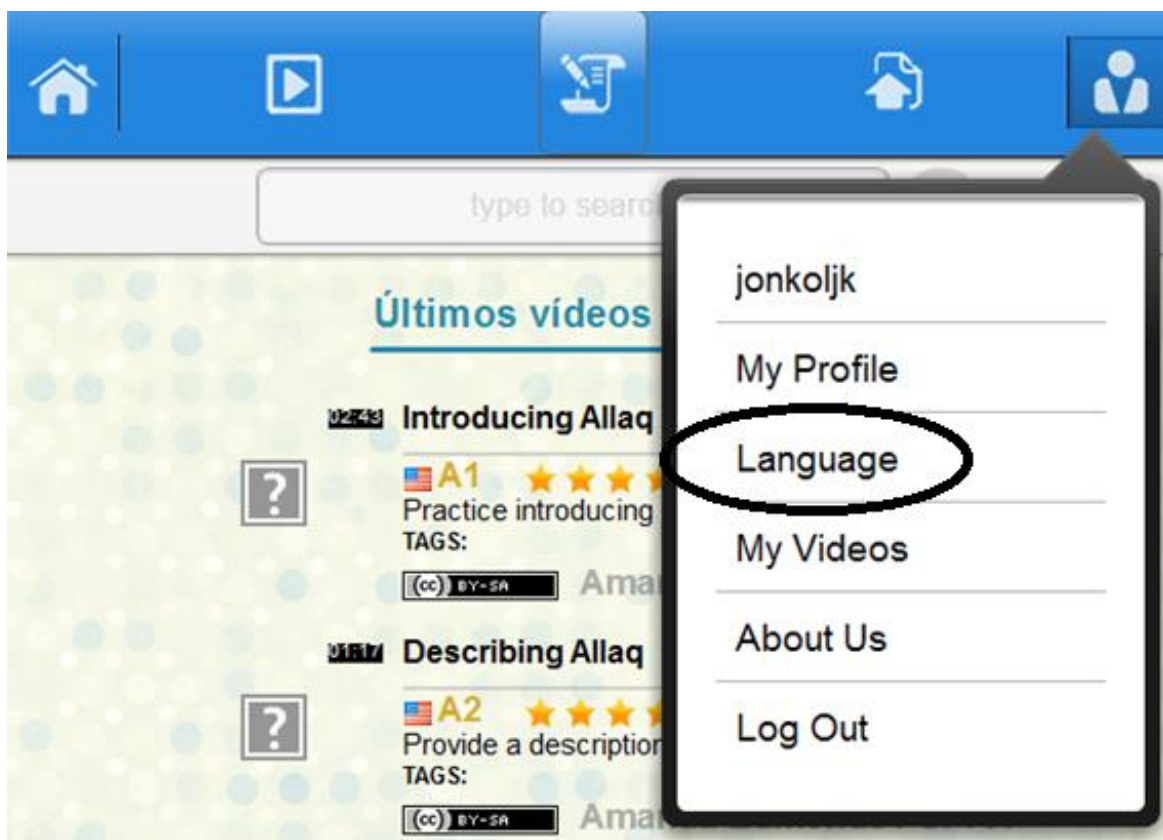
Prozesu berdina eginez beste erabiltzaile baten ebaluazioa aukeratzen bada, momentuan pantailan dagoen informazioa ebaluazio berriaren informazioarekin berridatziko da.



## 8. Aplikazioaren hizkuntza aldatu

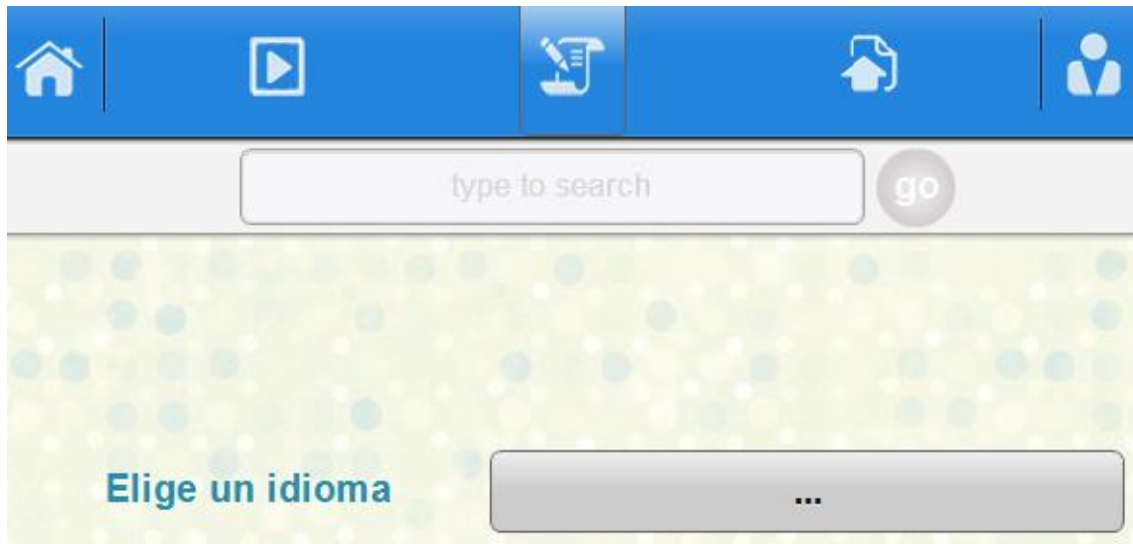
Babelium Project hainbat hizkuntzatan dago aukeran: gaztelera, euskera, ingelesa, frantsesa eta alemanera. Hauen artean hautatzea posible zen web bertsioan, baina ez bertsio mugikorrean. Beraz, hau egiteko aukera ere inplementatu da.

Hizkuntza aldatzeko, erabiltzaileak eskumako goiko aldean dagoen botoia sakatu beharko du (pertsona baten ikonoa duena) eta “Language” aukera sakatu:



26. irudia: Hizkuntza aldatzeko botoia

Hau egin eta gero, hurrengo pantaila ikusiko da:



*27. irudia: Hizkuntza aldatzeko pantaila*

Hemen, erdiko aldean dagoen botoia sakatuz gero, erabiltzaileak lehen aipatutako bost lengoaiak izango ditu eskuragarri.



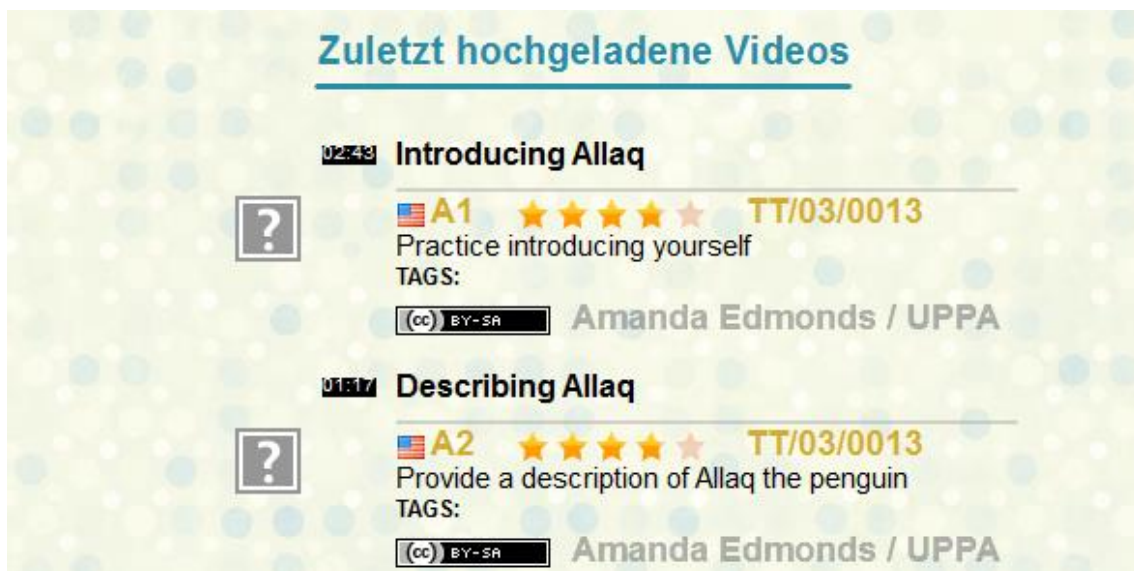
*28. irudia: Hizkuntzen zerrenda*

Hauetako bat aukeratzen denean, leihoa bakarrik itxiko da, eta hizkuntza dinamikoki kargatuko da. Hizkuntza aldatu nahi dugun bakoitzean, pantaila honetara sar gaitezke eta behin aldaketa egin dugula, Back botoia sakatu terminalean eta aurreko pantailara bueltatu. Honen zergatia kodea azaltzen den momentuan argituko da:



*29. irudia: Hizkuntza aukeratu eta gero*

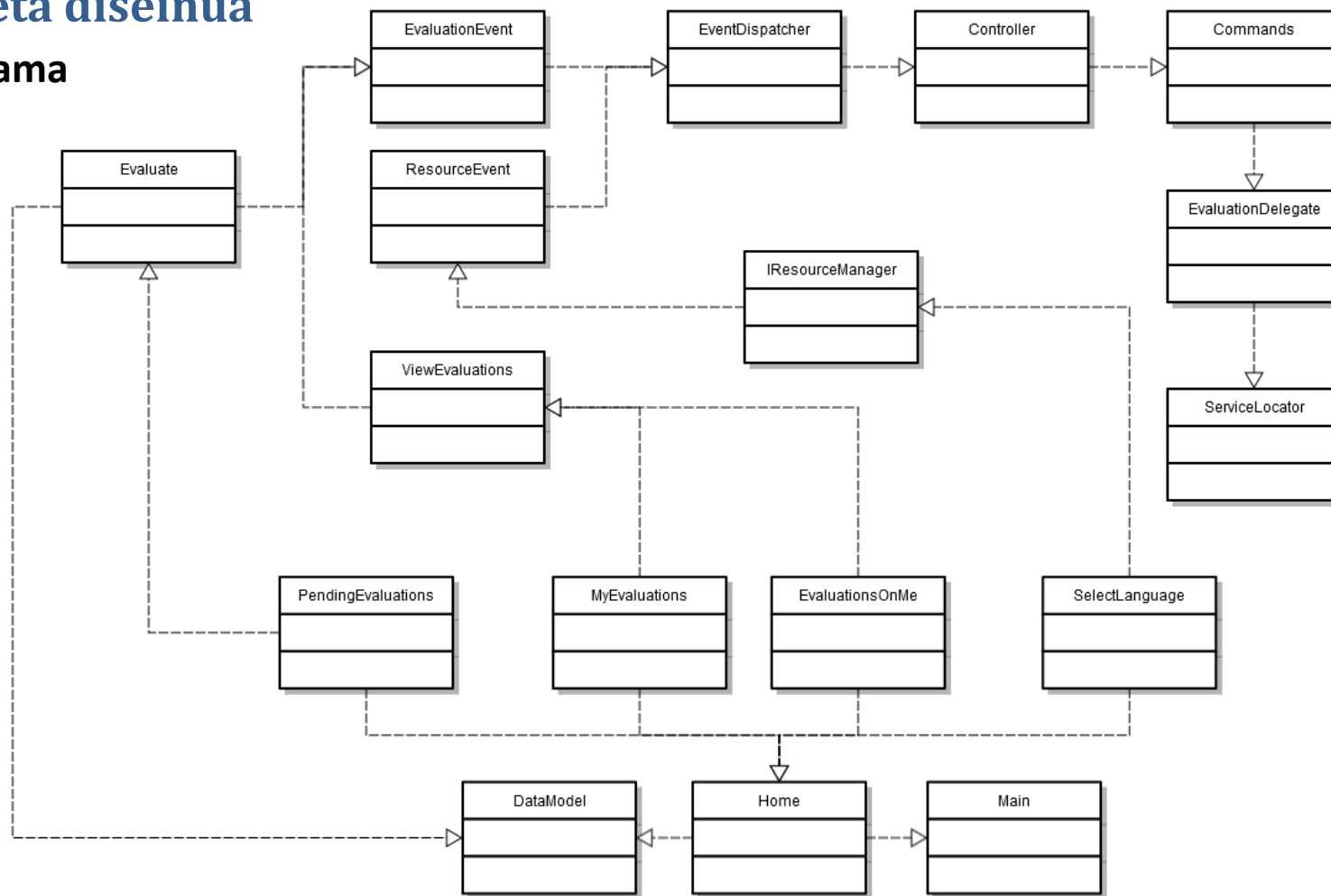
Orain hizkuntza aldatu dugula, aurreko pantailara bueltatu Back botoiarekin eta aldaketa ikusiko dugu:



*30. irudia: Hasierako pantaila hizkuntza aldatuta*

## Analisia eta diseinua

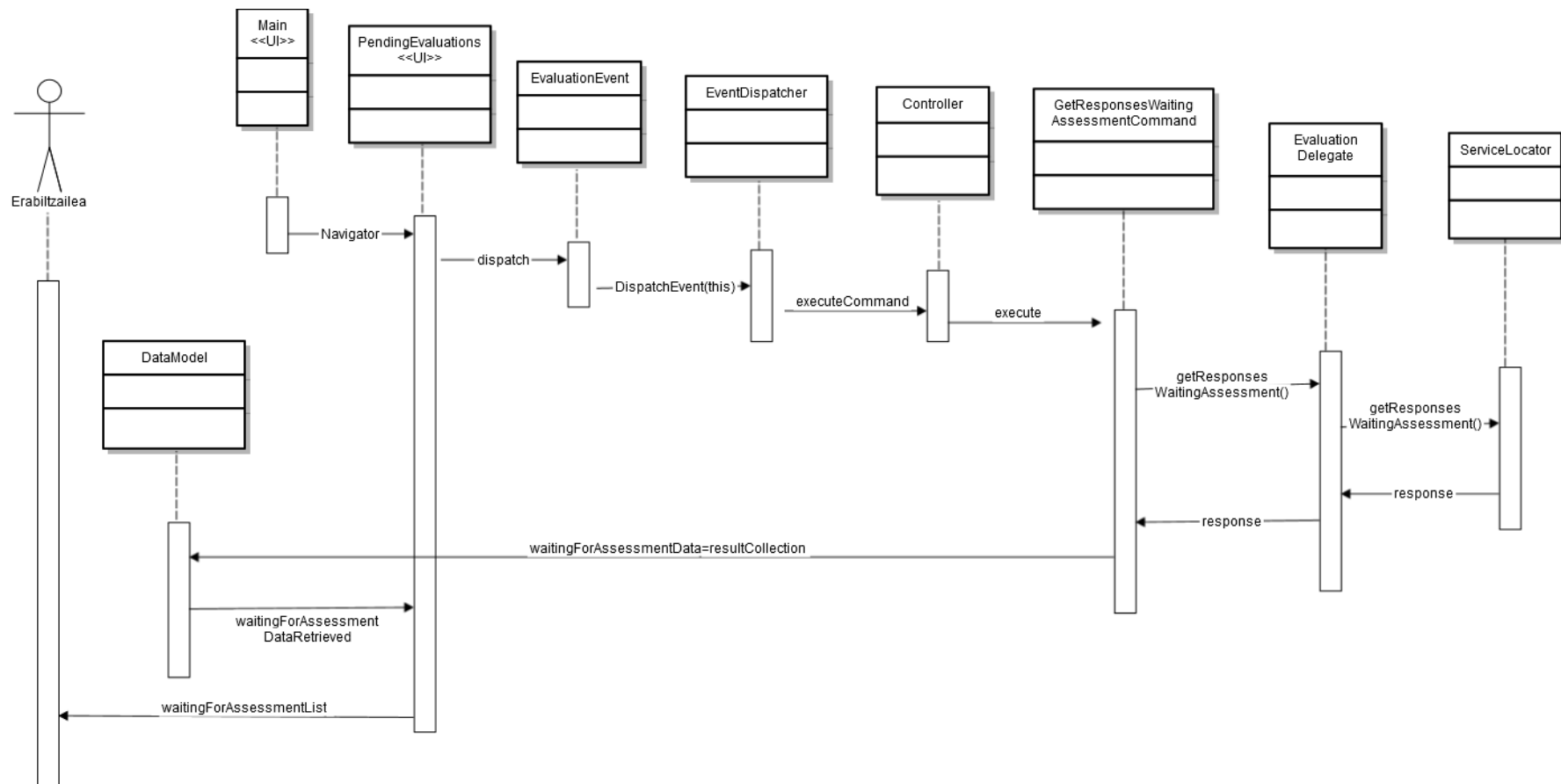
### Klase diagrama



31. irudia: Klase diagrama. "Commands" klasean komando-klase guztiak sartzen dira, exekutatzen denaren arabera

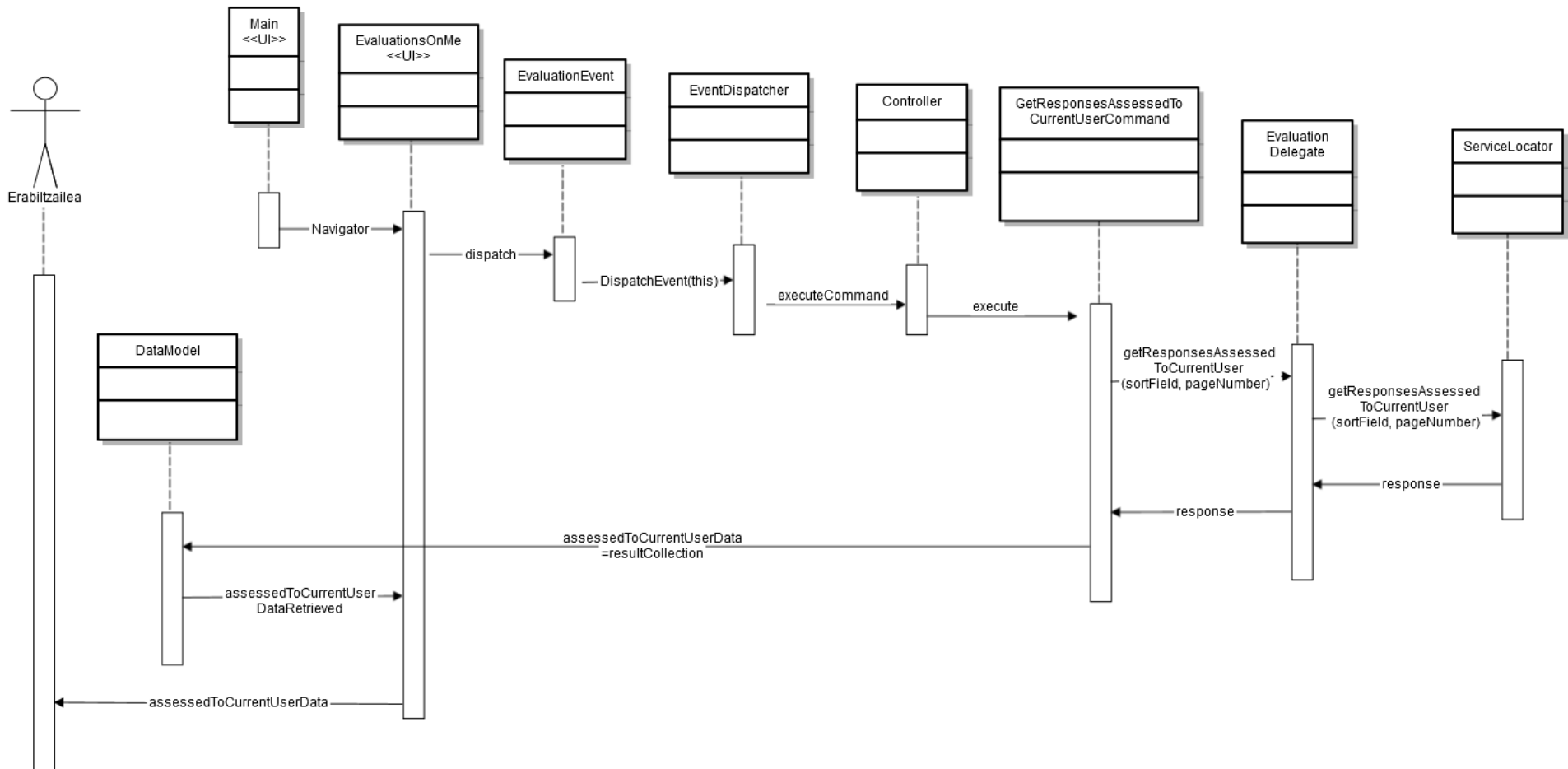
### Sekuentzia diagramak

Ebaluatzeke dauden erantzunak zerrendatu



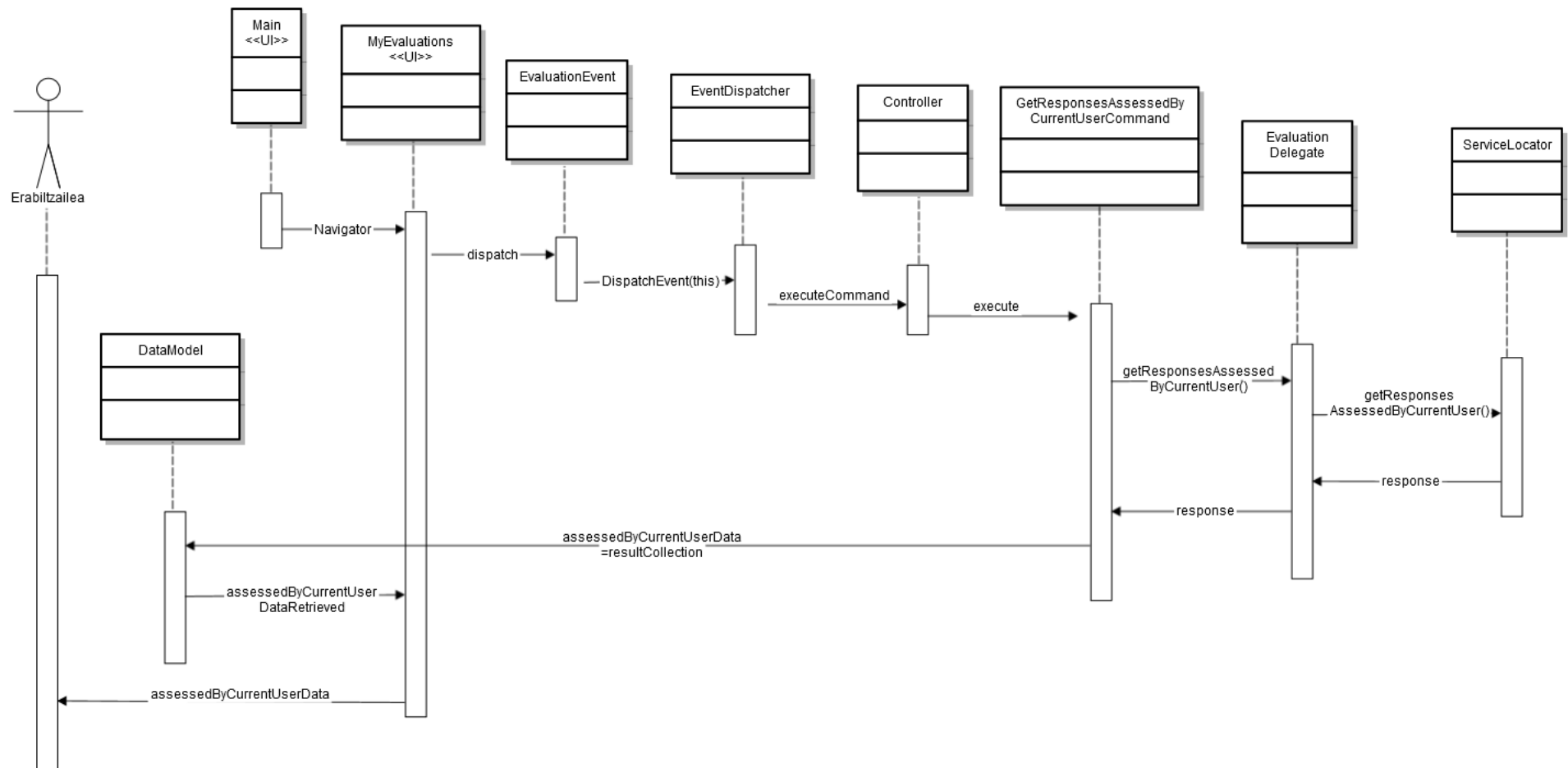
32. irudia: 1. Sekuentzia diagrama

Niri ebaluatutako erantzunak zerrendatu



33. irudia: 2. Sekuentzia diagrama

Nik ebaluatutako erantzunak zerrendatu



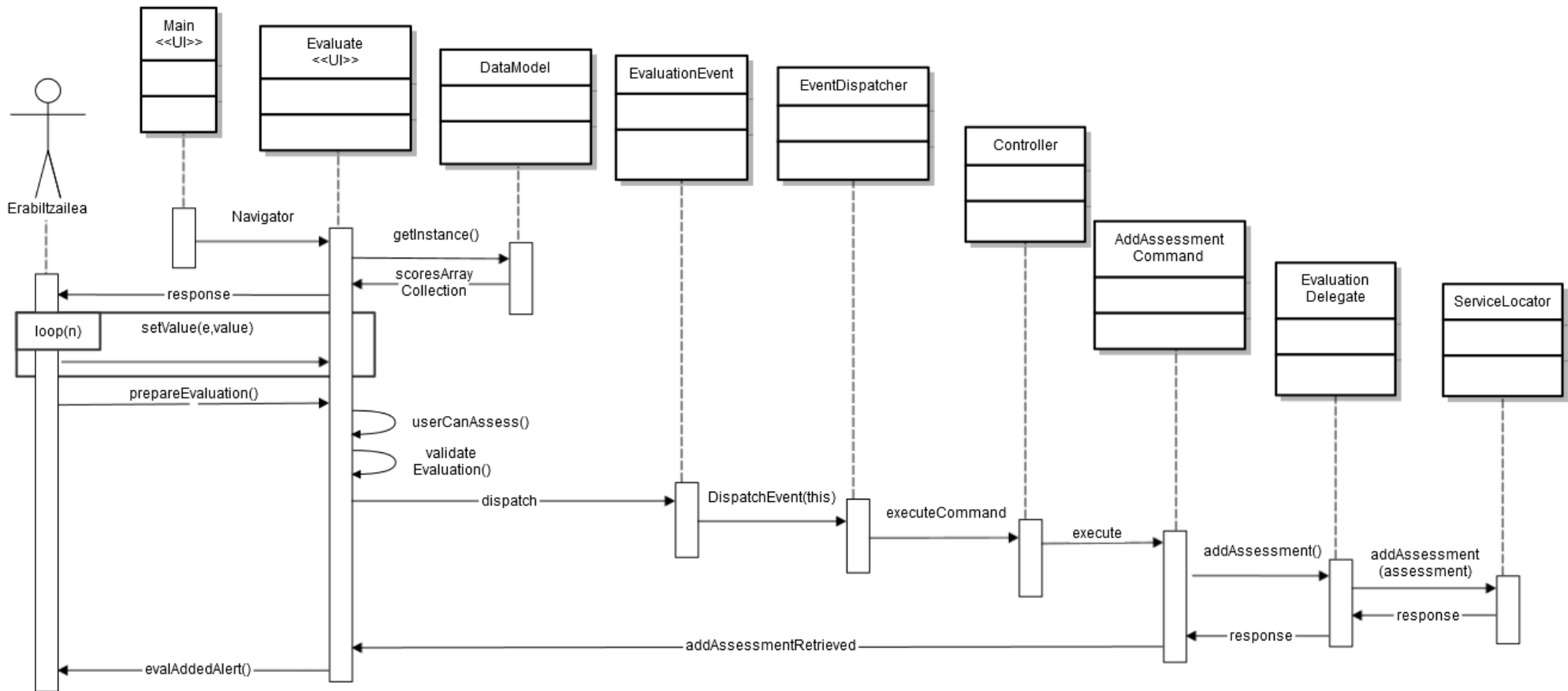
34. irudia: 3. Sekuentzia diagrama

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

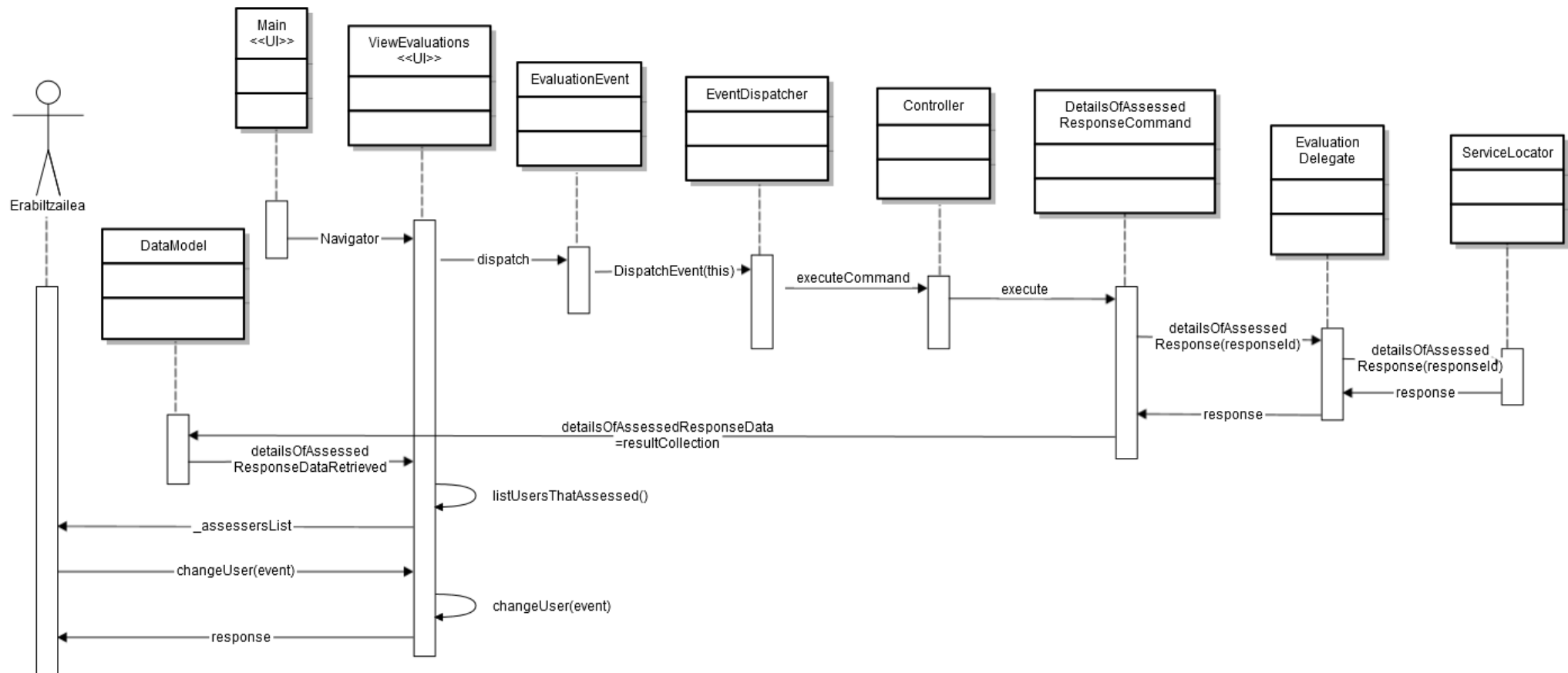
#### Ebaluazio bat bidali



35. irudia: 4. Sekuentzia diagrama

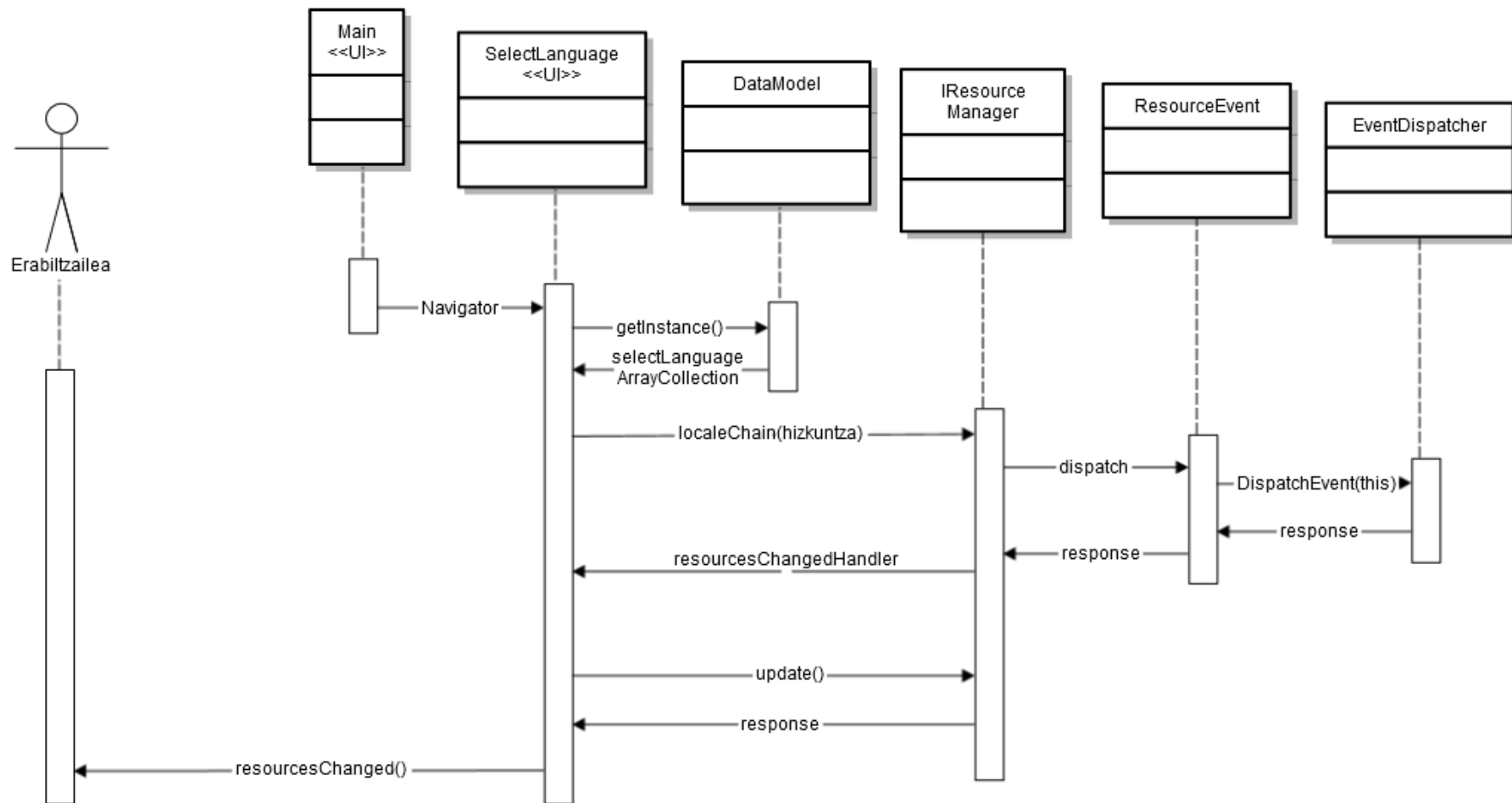


### Ebaluazioak bistaratu



36. irudia: 5. Sekuentzia diagrama

Hizkuntza aldatu



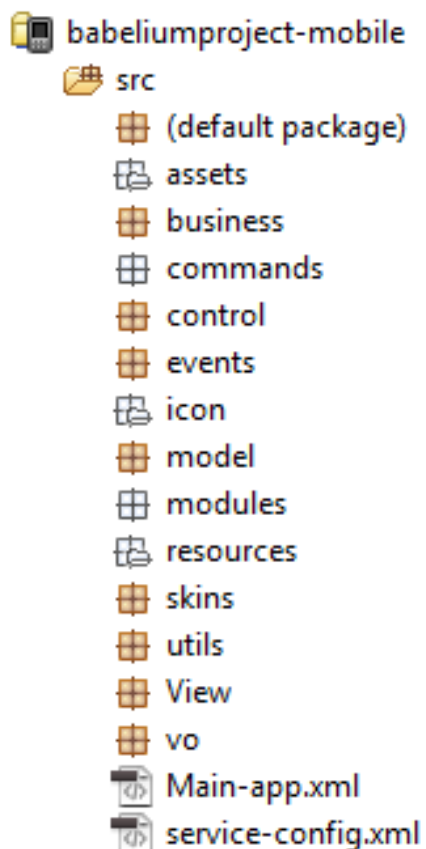
37. irudia: 6. Sekuentzia diagrama

## Inplementazioa

### Babelium Mobile proiektuaren egitura

#### Egitura SDK-aren barne

Babelium Project-en bertsio mugikorraren proiektua lan egiteko erabiliko den SDKren barruan inportatzean, hurrengo egitura hartuko dute pakete desberdinek:



*38. irudia: SDK-aren barneko egitura*

Garrantzitsua da kodea nola egituratuta dagoen jakitea hau aldatu edo eguneratu behar dugunean. Hurrengoak dira proiektuaren azpi-paketeak:

## Proiektuaren azpi-atalak

### Assets

Karpeta honen barruan aurkituko ditugu aplikazioaren zati estatikoek erabiltzen dituzten irudiak, diseinuaren atzekaldeko patroia, eta abar.

### Business

Hemen eskuordetutako komandoak kudeatzen dituzten .as fitxategiak gordetzen dira, Cairngorm gertakarien (events) datuak *FrontController*-etik pasatu eta gero komando egokiarekin prozesatu daitezten, *ServiceLocator*-era pasatu baino lehen.

### Commands

Cairngorm gertakarietan datozen datuekin prozesatzeko erabiltzen diren komandoak gordetzen dira hemen, aurreko atalean aipatutakoak, hain zuzen ere. Proiektu honetan erabilitako garrantzitsuenetariko bat *AddAssessmentCommand.as* izan da, ebaluazio bat jaso eta gero zerbitzarira bidaltzeko ibilbidearen zati bat izanik.

### Control

Cairngorm egituraren barruan, Kontrolatzaile-mailako funtsezko osagaiak barneratzen ditu atal honek. Aipatutako *FrontController* osagaia (proiektuan *Controller.as* izenaz), adibidez. Gertakariak jasotzen ditu eta hauen motaren arabera komando bat edo beste bat hasieratzen du.

## Events

Pakete honek gordetzen ditu lehen aipatutako Cairngorm gertakarien definizioak. Gertakaria bera eta datu-sortaz gain, testu-kate bat gordetzen du kontrolatzaileak komando egokia aukeratu dezan, estatikoki definitu eta dinamikoki bidaltzen dena.

## Icon

Aplikazioaren ikono nagusia gordetzen da, erresoluzio desberdinetan.

## Model

Hasieran kargatzen diren datu guztien definizioa gordetzen da hemen: hizkuntzak, aplikazio osoan konstanteak diren aldagaiak (testu-kateak, zerrendak, beste datu batzuk). *Singleton* patroiarekin eraikitzen da eredu hau eta ondoren atzitzeko *getInstance()* motako funtzio bat erabiltzen da.

## Modules

Aplikazioan arbitrarioki kargatzen diren elementuak dira. Kodean definitzen dira eta gero, kudeatzaile baten laguntzaz (*ResourceManager*, adibidez), datuak esleitzen zaizkie. Adibidez, bideo-zerrenda baten bideoak (modulu bat bideo bakoitzeko) bistaratzeko oso erabilgarriak dira, egitura berdina dutelako baina datuak eta aldagaiak desberdinak izango direlako. Hau dela eta, kudeatzaile dinamikoa beharrezkoa da.

## Resources

Aplikazioaren barruan, bistaren arabera beharko diren errekurtsuak biltzen ditu, irudiak eta ikonoak gehienak. Bestalde, **script** eta **datu-base**

kontsultak ere gordetzen ditu, nahiz eta hauek zerbitzarian exekutatzen diren eta hauek aplikazioarekin paketatzea beharrezkoa ez den.

### Skins

Aplikazioan erabiltzen diren hainbat baliabide dira, soilik grafikoak ez izanik (aurreko **resources** atalekoak ez bezala). MXML kode-zati txikiak dira gehienetan, beste klase eta bista orokorragoetatik inportatu eta berrerabiltzeko.

### Utils

Pakete honetan aplikazioak erabiltzen dituen modulu erabilgarri batzuk gordetzen dira: aurreko kamara (mugikor eta tablet-etan erabiltzeko), bideoak bilatzeko **TextField** grafikoa (**Search.mxml**) eta metodoen implementazioa, bideo-ariketen zerrendetako elementuak pantailaratzeko klaseak, eta abar.

### View

Aplikazioaren bista grafikoak daude hemen bilduta: kautotzeko pantaila, igotako azken bideoen bista, bideoen zerrenda desberdinen pantailak, erabiltzailearen profila, ebaluazioa egiteko interfazea, eta abar. Bista bakoitza MXML fitxategi batean kodetu eta gordetzen da, eta **Home.mxml** bistaren gainean pila gisa jartzen dira pantailak erabiltzaileak aplikazioaren barruan ekintzak burutzen dituen heinean.

### VO

ValueObject-ak gordetzen dira hemen, aplikazioaren informazioa gordetzen duten datu-egiturak. Datu-egitura hauek zerbitzari eta aplikazioaren artean informazioa bidali eta jasotzeko erabiltzen dira. Ebaluazio bat egitean, bideo-erantzun bat grabatzean, zerbitzarian

kautotzean, eta abar, ValueObject bat sortzen da ekintzaren datuak gordetzeko, eta beharrezkoa izanez gero, gertakari baten barruan zerbitzarira bidaltzeko.

#### **Main-app.xml:**

Main-app fitxategian, AIR aplikazioak identifikatu, instalatu eta exekutatzeko hainbat parametro espezifikatu daitezke: aplikazioaren identifikatzaile bakar eta errepikaezina, APK fitxategiaren izena, instalatzailean agertuko den aplikazio-izena, etabar.

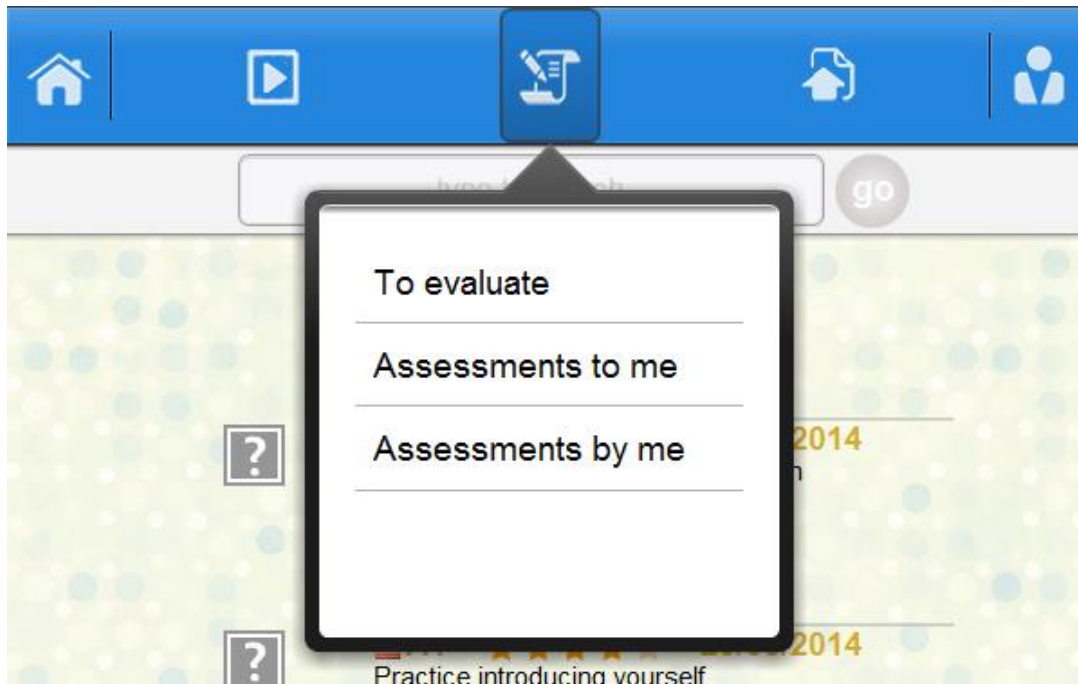
Horrez gain, aplikazioaren ikono nagusia eta Android baimenak ere espezifikatu daitezke, Adobe Flash Builder-ren proiektua sortzean egiten den bezala, baina fitxategi hau editatuz.

#### **Service-config.xml:**

Service-config fitxategian, aplikaziotik kanpoko zerbitzuetara konektatzeko parametroak definitzen dira. Adibidez, proiektu honetan Babelium zerbitzariaren URI (Uniform Resource Identifier) edo Baliabide Identifikatzaile Uniformea espezifikatzen da fitxategi honetan, zerbitzariarekiko atzipenak balioztatzeko beste hainbat atributuz gain.

## Aplikazioaren barneko bistak

### Ebaluaketa aukeren CalloutButton osagaia



39. irudia: Ebaluazio CalloutButton-a

Babelium Project aplikazioari gehitutako ebaluazio-sistemaren oinarria da botoi hau. Aurretik aplikazioak erabiltzailearen aukerak (goiko eskumako aldean) kudeatzeko implementatu zen botoiaren oso antzekoa da. Flex aldeari dagokionez, kodeak hurrengoaren antza du:

```
<s:CalloutButton id="evaluate_group" height="75"
    <s:calloutLayout>
        <s:VerticalLayout horizontalAlign="center" gap="5">
            </s:VerticalLayout>
        </s:calloutLayout>
        <s:calloutContent >
            <s:VGroup horizontalAlign="center">
                <s:List id="evalList"
dataProvider="{model.DataModel.getInstance().evalListArrayCollection}" height="200" width="215"
change="list_evaluate(event)"/>
            </s:VGroup>
        </s:calloutContent>
    </s:CalloutButton>
```



**CalloutButton** da osagai nagusia, baina bere barruan beste azpi-osagai batzuk ikus ditzakegu:

- **calloutLayout** zatian elementuen disposizioa definituko da.
- **calloutContent** zatiak barneratuko ditu botoian klik egitean hedatuko den zerrendaren elementuak, **List** aldagai baten laguntzaz.
- **List** hau aplikazioaren **DataModel** klase estatikotik eskuratuko da.

**DataModel** klasean botoiak hartzen duen zerrenda aldatzen badugu, kendu edo gehitzen diren aukerak desagertu edo agertuko dira. Baina testua konparatzen duen funtzioa ere eguneratu beharko da:

```
protected function list_evaluate(event:IndexChangeEvent):void{
    if(evalList.selectedItem=="To evaluate"){
        navigator.pushView(View.PendingEvaluations,null,null,slideTrans);
    } else {
        if (evalList.selectedItem=="Assessments to me"){
            navigator.pushView(View.EvaluationsOnMe,null,null,slideTrans);
        } else {
            if (evalList.selectedItem=="Assessments by me"){
                navigator.pushView(View.MyEvaluations,null,null,slideTrans);
            }
        }
    }
    evaluate_group.closeDropDown();
    List(event.target).selectedItem=null;
}
```

Ikusten denez, egiten dugun aukeraren arabera, **navigator** klaseak **View** edo bista bat edo beste bat bidaliko du pantailaren aurrealdera, beste hiru aldagairekin. Pantaila honetatik daturik ez kontesturik bidali behar ez denez hurrengo pantailara, bigarren eta hirugarren atributuak **null** dira. Laugarren atributua, aldiz, pantaila aldatzeko animazioa aukeratzeko erabiltzen da. Hurrengo bista irristatze animazio batekin ikusiko dugu sartzen, edo **slideTrans** elementua deritzonarekin.

## PendingEvaluations.mxml, EvaluationsOnMe.mxml eta MyEvaluations.mxml



### 40. irudia: Egindako ebaluaketen zerrenda

Hiru pantaila hauek funtzionamendu berbera erabiltzen dute helburu berdinerako: egindako aukeraren arabera, erabiltzaileen saiakeren zerrenda bat eskaintzen dute. Flex lengoia, hurrengo antza du kodeak:

```
<s:VGroup width="100%" height="100%">
    <s:VGroup horizontalAlign="center" width="100%">
        <s:Label
            text="{resourceManager.getString('myResources','ASSESSMENT_PENDING_RESPONSES_LIST')}"
            styleName="sectionTitleLabel"
            id="userProfileTitleLabel"/>
    </s:VGroup>
    <s:List id="assessmentPendingVideosDataGroup" width="100%" height="100%"
        itemRenderer="modules.evaluation.ListItemWaitingForAssessment"
        dataProvider="{waitingForAssessmentList}">
        <s:layout>
            <s:VerticalLayout/>
        </s:layout>
    </s:List>
</s:VGroup>
```

Hurrengo elementuek osatzen dute zerrenda hau eratzeke kode-zatia:

- **VGroup**: grafikoki elementuak barneratzeko taldekatze klase bat da, **HGroup** bezala. **VGroup** bertikalki multzokatzen ditu elementuak, eta **HGroup**, aldiz, horizontalki.

- **Label**: editatu ezin diren testu-etiketak errepresentatzeko erabiltzen da. Kasu honetan, eta aplikazio ia osoan, **resourceManager** izeneko klase batek kudeatzen ditu, hizkuntzaren arabera testu-kate egokia agertarazteko.

- **List**: lehen esan bezala, zerrendak grafikoki irudikatzeko balio du, eta bere aldagaien bidez kudeatzen da erakusten den informazioa.

Hemen, ebaluazio-modulu batzuez osatuta dago,

**ListItemWaitingForAssessment** izenekoak, audio/bideoak grafikoki zerrendatzeko erabiltzen diren edukiontziak.

Aplikazioak pantaila sortzen amaitzen duenean, gertakari bat bidaltzen du zerbitzarira, `EvaluationEvent` motakoa:

```
protected function view1_creationCompleteHandler(event:FlexEvent):void{
    new EvaluationEvent(EvaluationEvent.GET_RESPONSES_WAITING_ASSESSMENT).dispatch();
}
```

**Binding** izeneko deklarazio baten bidez, azpian dagoen kodeko boolearraren balioa aldatzen da eta bista hau aldaketaz ohartzen da:

```
<fx:Binding source="{DataModel.getInstance().waitingForAssessmentDataRetrieved}"
    destination="waitingAssessmentRetrievedHandler"/>
```

Zerbitzaritik jasotzen den informazio guztia kargatzen denean, hau da, aurreko **binding**-a aktibatzen denean, hurrengo kodea exekutatzen da elementuak erakutsi ahal izateko:

```
private function set waitingAssessmentRetrievedHandler(value:Boolean):void{  
    var waDataProvider:ArrayCollection=DataModel.getInstance().waitingForAssessmentData;  
    waitingForAssessmentList=waDataProvider;  
}
```

Zerrendatik elementu bat aukeratzen dugunean, berriro **pushView** funtzioa ikusten da. Baina orain, hurrengo pantailan zein informazio erakutsi behar den jakiteko, saiakeraren datuak pasatzen dizkiogu **obj** aldagaiaren bidez, ebaluazioaren informazioa gordetzen duen **EvaluationVO (ValueObject)** bat.

```
protected function videosDataGroup_changeHandler(event:IndexChangedEvent):void{  
    var obj:EvaluationVO=List(event.target).selectedItem as EvaluationVO;  
    navigator.pushView(View.Evaluate,obj,null,slideTrans);  
    List(event.target).selectedItem = null;  
}
```

### Evaluate.mxml



#### 41. irudia: Ebaluatzeko pantaila

Ebaluatzeko pantaila honetan aurkitzen dugun elementu nagusia bideoa erreproduzitzeko modulua da, VideoPlayerBabelia klasearekin instantziatzen dena Flex erabiliz:

```
<s:HGroup id="videoScreen" width="55%" height="100%" minWidth="0" minHeight="0"
    horizontalAlign="left" paddingLeft="16" paddingTop="6">

    <videoPlayer1:VideoPlayerBabelia id="VideoPlayer"
        skin="white" seek="true" subtitles="true"
        autoPlay="false" autoScale="true" width="{this.width/2}"
        height="{this.height/1.8}"
        streamSource="{DataModel.getInstance().streamingResourcesPath}"
        controlsEnabled="true"/>

    <!-- <s:HGroup id="recording_options" width="100%" minWidth="0" minHeight="0"
        paddingLeft="16" verticalAlign="top">
    </s:HGroup> -->

</s:HGroup>
```

Ikusteko errazagoak diren hurrengo elementuak ere aurki ditzakegu **Evaluate.mxml** pantailaren kodean:

- **HGroup** eta **VGroup**: lehen aipatu bezala, hauen barnean beste elementu batzuk barneratzeko erabiltzen dira, lortu nahi den orientazioaren arabera bata edo bestea aukeratuz.
- **CalloutButton**: klase hauek ere erabili dira puntuaketak esleitu ahal izateko. **DropDownList**-ekin topatutako *bug*-ak erabakigarriak izan dira zerrenda-botoiak era honetan egitea aukeratzeko.
- **Label** eta **List**: erabili diren beste elementu zeharo orokorrak.
- **TextArea**: erabiltzaileak, testu-iruzkin bat ekarpen osagarri gisa egin nahi izatekotan, eremu honetan egin ahal izango du, Android sistema-eragileak eskaintzen duen teklatu natiboa hedatuz gainean klik egiten denean.
- **CheckBox**: aurreko TextArea gaitu eta desgaitzeko erabiliko den beste botoi bat.

Erabiltzailearen hizkuntza-maila konprobatzeko hurrengo metodoa erabiliko da:

```
private function userCanAssess():void{
    var knowsLanguage:Boolean = false;
    var tmpUser:UserVO = model.DataModel.getInstance().loggedUser;
    for (var i:int=0; i<tmpUser.userLanguages.length; i++){
        if (tmpUser.userLanguages[i].language == _currentEvaluation.exerciseLanguage){
            knowsLanguage = true;
            if (knowsLanguage && (tmpUser.userLanguages[i].level >=
                _currentEvaluation.exerciseAvgDifficulty)){
                _canAssess = true;
            }
            break;
        }
    }
    _assessCheck = true;
}
```

Ebaluazioak balioztatzeko funtzioa hurrengoa da:

```
private function validateEvaluation() : void {
    if (punt1.selectedItem != null && punt2.selectedItem != null &&
        punt3.selectedItem != null && punt4.selectedItem != null &&
        punt5.selectedItem != null) {
        _validEvaluation = true;
    }
}
```

Ebaluazioa bidaltzeko botoia sakatzean, aurreko bi funtzioak exekutatu dituzte aplikazioak. Behin erabiltzaileak ebaluatu dezakeela eta ebaluazioa era zuzenean osatu dela konprobatuta, ebaluazioa zerbitzarira bidaltzeko funtzioa exekutatu da:

```
private function prepareEvaluation() : void {
    if (!_assessCheck) {
        userCanAssess();
    }
    if (_canAssess) {
        validateEvaluation();
        if (_validEvaluation) {
            evalButton.enabled = false;
            evalButton.label = resourceManager.getString('myResources', 'LABEL_SENDING');
            new EvaluationEvent(EvaluationEvent.ADD_ASSESSMENT, createEvaluation()).dispatch();
        }
    } else {
        //Invalid evaluation alert dialog throw
        new invalidEvalAlert().open(this, true);
    }
} else {
    evalButton.enabled = false;
    new langLevelAlert().open(this, true);
}
}
```

**EvaluationEvent** gertakaria bidaltzen denean, `createEvaluation()` funtzioaren ebaluazio `ValueObject`-a bidaltzen da zerbitzarira. Horrela sortzen da ebaluazioa, pantailan sarturiko datuak bilduta:

```
private function createEvaluation():EvaluationVO{
    var newEval:EvaluationVO = new EvaluationVO;
    newEval.responseId = _currentEvaluation.responseId;
    newEval.intonationScore = (uint)(punt1.selectedItem);
    newEval.fluencyScore = (uint)(punt2.selectedItem);
    newEval.rhythmScore = (uint)(punt3.selectedItem);
    newEval.spontaneityScore = (uint)(punt4.selectedItem);
    newEval.overallScore = (uint)(punt5.selectedItem);
    if (checkTestua.selected = true){
        newEval.comment = testulruzkina.text;
    } else {
        newEval.comment = "";
    }
    return newEval;
}
```

Bideoen erreprodukzio egokiari dagokionez, hurrengo pantailan azalduko dira kodearen xehetasunak, era berdinean inplementatzen baitute ezaugarri hau.



## ViewEvaluations.mxml



### 42. irudia: Ebaluazioak ikusteko pantaila

Ebaluazioak bistartzeko pantailan, aurreko Evaluate.mxml pantailan bezala, zatitutako VideoPlayer bat dago, eta azpian ebaluazioen informazioa bistartzeko eremua. Saiakera hau ebaluatu duten bideo guztiak lortzeko, aplikazioak kode hau exekutatzen du pantaila sortzean:

```
protected function viewEvaluations_creationCompleteHandler(event:FlexEvent):void{
    new EvaluationEvent(EvaluationEvent.DETAILS_OF_ASSESSED_RESPONSE,data as
    EvaluationVO,data.responseId as uint).dispatch();
}
private function set detailsOfAssessedResponseDataRetrievedHandler(value:Boolean):void{
    var doaDataprovider:ArrayCollection=DataModel.getInstance().detailsOfAssessedResponseData;
    _assessmentDetailsList=doaDataprovider;
    if (_assessmentDetailsList != null) {
        listUsersThatAssessed();
    }
}
```

Aurreko pantailatik jasotako **data** aldagaia bidaltzen du zerbitzarira `EvaluationEvent` batekin bideo-erantzun honen ebaluazio guztiak jasotzeko, eta **\_assessmentDetailsList** aldagai batean gordetzen ditu geroago tratatu ahal izateko. Izan ere, **listUsersThatAssessed()** metodoak erabiltzailea aukeratzeko botoiaren zerrendan sartuko ditu saiakera hau ebaluatu dituzten erabiltzaile guztiak:

```
private function listUsersThatAssessed():void{
    _assessersList.removeAll();
    for (var i:int=0; i<_assessmentDetailsList.length; i++){
        _assessersList.addItemAt(_assessmentDetailsList[i].userName,i);
    }
}
```

Horrela, erabiltzaileak ebaluazioen artean aukeratzean, hurrengo kodea exekutatuko da ebaluatzaile horren puntuaketa eta (existitzen bada) iruzkina pantailaratzeko:

```
private function changeUser(e:Event):void{
    var index:uint = e.currentTarget.selectedIndex;
    punt1.text = _assessmentDetailsList[index].intonationScore;
    punt2.text = _assessmentDetailsList[index].fluencyScore;
    punt3.text = _assessmentDetailsList[index].rhythmScore;
    punt4.text = _assessmentDetailsList[index].spontaneityScore;
    punt5.text = _assessmentDetailsList[index].overallScore;
    textCommentArea.text = _assessmentDetailsList[index].comment;
    assessmentDate.text = _assessmentDetailsList[index].addingDate;
    userButton.label = _assessmentDetailsList[index].userName;
    userButton.closeDropDown();
}
```

Hasieran ez da daturik agertuko testu-eremuetan, eta erabiltzaile batetik bestera aldatzean datuak berridatzi egingo dira, aurreko erabiltzailearenak desagerraraziz.

Bideoaren erreprodukzioari dagokionez, aurreko pantailatik viewPush egitean jasotako informazioa tratatuz lortzen dira estekak, eta bestelako informazio guztia **DataModel** instantziatik lortzen da:

```
private const EXERCISE_FOLDER:String=DataModel.getInstance().exerciseStreamsFolder;
private const RESPONSE_FOLDER:String=DataModel.getInstance().responseStreamsFolder;

_currentEvaluation=data as EvaluationVO;
EXERCISE_NAME=_currentEvaluation.exerciseName as String;
RESPONSE_NAME=_currentEvaluation.responseFileIdentifier as String;

_exerciseName=_currentEvaluation.exerciseName;
_exerciseTitle=_currentEvaluation.exerciseTitle;
_exerciseId=_currentEvaluation.exerciseId;
_selectedRole=_currentEvaluation.responseCharacterName;
```

Atributu hauek erabiliz, bideoari dagokion informazio guztia pantailaratu daiteke. Gero, hurrengo metodoak exekutatu dira bata bestearen atzean:

- **prepareExercise()**
- **resetCueManager()**
- **setupVideoPlayer()**
- **setupRecordingCommands()**

Lehenengo metodoan, **VideoPlayer**-aren hasieraketa egingo da, bi bideo batera erreproduzitu ditzan, eta bi bideo hauek non aurkitu esanez. Horrez gain, ariketan erabiltzen diren azpтитuluak atzitu eta deskargatuko ditu zerbitzaritik. Bigarren eta hirugarren metodoak, azpтитuluak eta *listener* guztiak prestatzeko erabiliko dira:

```
private function prepareExercise():void{
    VideoPlayer.state=VideoPlayerBabelia.PLAY_BOTH_STATE;
    VideoPlayer.stopVideo();
    VideoPlayer.videoSource=EXERCISE_FOLDER + '/' + EXERCISE_NAME;
    VideoPlayer.secondSource=RESPONSE_FOLDER + '/' + RESPONSE_NAME;

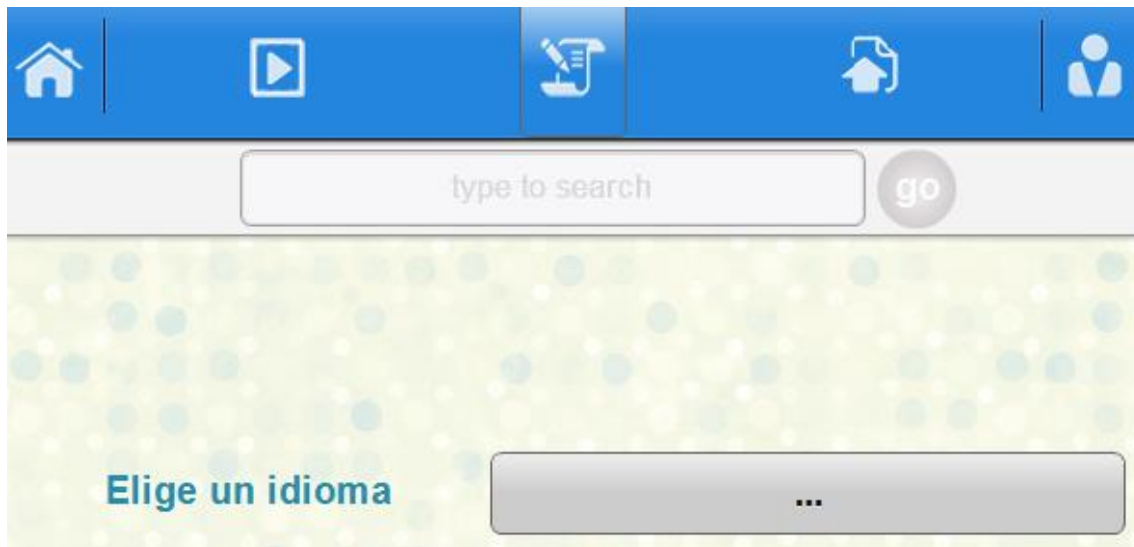
    var auxExRol:ExerciseRoleVO=new ExerciseRoleVO();
    auxExRol.exerciseId=_exerciseId;
    new ExerciseRoleEvent(ExerciseRoleEvent.GET_EXERCISE_ROLES, auxExRol).dispatch();

    var auxEx:ExerciseVO=new ExerciseVO();
    auxEx.id=_exerciseId;
    new ExerciseEvent(ExerciseEvent.GET_EXERCISE_LOCALES, auxEx).dispatch();
}
```

Azkenik, **setupRecordingCommands** metodoak erabiltzaileak zein pertsonaiaren rol aukeratu duen determinatuko du aurreko **\_selectedRole** atributuarekin konparaketa bat eginez, eta honen arabera bideoaren audioa mututu edo birgaituko du, erabiltzailearen saiakera entzun dadin jatorrizko bideoarenaren orde.

```
private function setupRecordingCommands():void {
    var auxList:ArrayCollection=_cueManager.getCueList();
    if (auxList.length <= 0)
        return;
    for each (var cueobj:CueObject in auxList){
        if (cueobj.role != _selectedRole){
            cueobj.setStartCommand(new RecordingOtherRoleCommand(cueobj, VideoPlayer));
            cueobj.setEndCommand(new ShowHideSubtitleCommand(null, VideoPlayer));
        } else {
            cueobj.setStartCommand(new StartRecordingSelectedRoleCommand(cueobj, VideoPlayer));
            cueobj.setEndCommand(new StopRecordingSelectedRoleCommand(VideoPlayer));
        }
    }
    _cueManagerReady=true;
}
```

## SelectLanguage.mxml



43. irudia: Hizkuntza aldatzeko pantaila

Hizkuntza aukeratzeko pantaila oso sinplea da: hizkuntza aukeratzeko esaten duen etiketa edo Label bat eta beste **CalloutButton** batek osatzen dute. Zerrenda-botoi honek hizkuntzen zerrenda bat kargatzen du **DataModel**-etik, eta aukeratzeko den testu-katearen arabera, hizkuntza bat edo beste bat aukeratzeko da:

```
<s:CalloutButton id="cb" width="55%" label="..."
    <s:calloutLayout>
        <s:VerticalLayout paddingTop="10" paddingBottom="10" paddingLeft="10" paddingRight="10"
horizontalAlign="center" gap="5">
            </s:VerticalLayout>
        </s:calloutLayout>
        <s:calloutContent >
            <s:VGroup paddingTop="10" paddingBottom="10" paddingLeft="10" paddingRight="10"
horizontalAlign="center">
                <s:List id="list" dataProvider="{model.DataModel.getInstance().selectLanguageArrayCollection}"
change="list_changeHandler(event)"/>
            </s:VGroup>
        </s:calloutContent>
    </s:CalloutButton>
```

Hizkuntza aldatzen denean botoiarekin, **list\_changeHandler** metodoa altxatzen da, gertakari bat bidaliz berarekin:

```
protected function list_changeHandler(event:Event):void{
    resourceManager.addEventListener(ResourceEvent.COMPLETE,resourcesChangedHandler);
    resourceManager.addEventListener(ResourceEvent.ERROR,resourcesErrorHandler);
    switch(list.selectedItem){
        case "Spanish (Spain)":
            resourceManager.localeChain = [ "es_ES" ];
            break;
        case "English (United States)":
            resourceManager.localeChain = [ "en_US" ];
            break;
        case "Basque":
            resourceManager.localeChain = [ "eu_ES" ];
            break;
        //Falta diren hizkuntzak...
        navigator.popToFirstView(null);
    }
    cb.label = list.selectedItem;
    cb.closeDropDown();
}
```

Metodo honetan bi *listener* gehiago jarri dira, **resourceManager** klasearen **localeChain** katea aldatzen denean **update()** metodo bat hasieratu behar delako, eta horretarako beharrezkoa da aldaketa modu egokian egin dela jakitea. Gauzak horrela, ikusten da aurreko **CalloutButton**-ean aukeratutako testu-katea konparatuz hizkuntza hori jartzen dela **localeChain** atributuan. Behin hau egin dela, eta errorerik egon ez bada, **resourceManager**-ari jarritako salatzaileak hurrengo metodoa altxatuko du:

```
protected function resourcesChangedHandler(event:FlexEvent):void{
    resourceManager.update();
    resourceManager.removeEventListener(ResourceEvent.COMPLETE,resourcesChangedHandler);
    resourceManager.removeEventListener(ResourceEvent.ERROR,resourcesErrorHandler);
}
```

Ikusten denez, **update()** metodoa exekutatu da eta bi salatzaileak ezabatu dira. Hizkuntza aldatzean errorerik egotekotan, hurrengo metodoa exekutatu da:

```
protected function resourcesErrorHandler(event:FlexEvent):void{
    trace("Error changing language");
    new langSetErrorAlert().open(this,true);
    resourceManager.removeEventListener(ResourceEvent.COMPLETE,resourcesChangedHandler);
    resourceManager.removeEventListener(ResourceEvent.ERROR,resourcesErrorHandler);
}
```

Salatzaileak ere ezabatzen dira hemen, baina erabiltzaileari hizkuntza aldatzeko momentuan errore bat gertatu dela jakinarazten zaio. Errore mezuak sortzeko kodea hurrengoa (**SkinnablePopUpContainer** klasekoa) da, eta <fx:Declarations> eta </fx:Declarations> tarteko kode-eremuan kokatu behar da:

```
<fx:Component className="langSetErrorAlert">
    <s:SkinnablePopUpContainer x="70" y="300">
        <s:TitleWindow title="Babelium Message" close="close()">
            <s:VGroup horizontalAlign="center" verticalAlign="middle" paddingTop="8" paddingBottom="8"
paddingLeft="8" paddingRight="8" gap="5" width="100%">
                <s:Label text="{resourceManager.getString('myResources','ERROR_WHILE_CHANGING_LANGUAGE')}" />
                <s:Button label="OK" click="close()" height="40" width="60" />
            </s:VGroup>
        </s:TitleWindow>
    </s:SkinnablePopUpContainer>
</fx:Component>
```

Hizkuntza katea dinamikoki aldatzen denez **resourceManager** klasearen bidez, **back** botoia sakatzean ez da pantailan sartzean zegoen lengoaia berriro jarriko. Hizkuntza katea aplikazio osorako eguneratzen da, eta berriro **localeChain()** eta **update()** era egokian eta pantaila honen bidez exekutatu arte, ez da hizkuntza aldatuko.

## Proiektuaren kudeaketa

Proiektua kudeatzeko hainbat eginkizun bete dira: kode guztia kudeatzeko erremintak erabili, proiektuaren zuzendariarekin bilerak egin, planifikazio bat eraman, eta abar. Honi buruzko xehetasunak emango dira orain eta konklusioak atera ere bai.

### Proiektuaren aurrerapenak ebaluatzeko bilerak

Hainbat bilera egin dira proiektuaren zuzendariarekin lana aurrera joan den heinean. Batzuetan zalantzak argitzeko (bai zalantza teknikoak, bai zalantza metodologikoak), beste batzuetan proiektua aurrera nondik jo beharko lukeen ebaluatzeko.

### Denbora-plangintzarekiko desbiderapena

Denboren plangintza egitea eta %100eko zehaztasuna izatea ezinezkoa da. Hauek dira proiektu honetan izan diren denbora-plangintza eta errealitatearen arteko desberdintasunak:

Eginkizuna	Iraupen estimatu	Iraupen erreal
Proiektuaren zuzendariarekin bilerak finkatu eta elkartu	10 ordu	12 ordu
Informazioa bilatu eta barneratu	120 ordu	171 ordu
Bideragarritasuna aztertu	5 ordu	3 ordu
Betekizunen bilketa garatu	15 ordu	12 ordu
Funtzionalitateak eta erabilpen kasuak planteatu	15 ordu	16 ordu
Web bertsioaren funtzioen azterketa egin	10 ordu	7 ordu
Klase diagramak garatu	15 ordu	12 ordu
Sekuentzia diagramak garatu	20 ordu	18 ordu
Aplikazioaren interfaze grafikoaren zirriborroak landu	10 ordu	6 ordu



## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

Metodo eta atributu lokalen definizioa	60 ordu	48 ordu
API eta beste klase batzuen berrerabilpena	40 ordu	18 ordu
Interfaze grafikoaren inplementazioa	50 ordu	28 ordu
Proben diseinua	5 ordu	2 ordu
Proben burutzea eta kodearen arazketa	10 ordu	23 ordu
Eratorritako arazoak zuzendu eta hobekuntzak burutu	15 ordu	31 ordu
Memoria eta eranskinen osaketa	80 ordu	95 ordu
Aplikazioaren kodea eta baliabideak paketatuta	5 ordu	2 ordu
Proiektuaren defentsa prestatu	15 ordu	15 ordu
<b>Guztira</b>	<b>500 ORDU</b>	<b>519 ORDU</b>

Ikusten denez, plangintzari dagokionez, ikaskuntzan ematen da desbideratze handiena. Ikaskuntza-fasea espero baino luzeagoa izan da, eta inplementazio-fasea espero baino laburragoa. Beste atazetan, denborak nahiko antzekoak dira.

Azkenean, guztira, 19 orduko desbiderapena egon da soilik, beraz, planifikazioaren desbiderapena orokorrean %3,66koa izan da soilik.

### Kodearen kudeaketa

Tamaina handiko proiektuetan kodea kudeatzeko euskarri-sistema bat erabiltzea oso interesgarria da, eta proiektu honen kasuan oso ondo funtzionatu duen **Git** sistema goraipatzekoa da. **GitHub** web orrialdean sortutako proiektu publiko batean dago Babelium Project-en kode guztia: <https://github.com/babeliumproject/>

Kode horretatik aurrera nik egin ditudan gehipenak hurrengo estekan aurkitu daitezke: <https://github.com/jonkolik/babeliumproject-mobile>

## **Bug edo programazio-akatsak**

Programatzeko orduan, nabarmenena izan den eta arazo gehiago sortu dituen *bug*-ari buruz hitz egingo da. Flex-ekin lan egitean izan dira arazoak akats honekin. Hurrengo estekan aurkitu daiteke honen deskribapena eta ematen dituen arazoei buruz informazio gehiago (ingelesez): <https://issues.apache.org/jira/browse/FLEX-27892>

Zerrendak hedatzen dituen botoiekin gertatzen da arazoa. Flex-ek kasu hauetarako **DropDownList** izeneko osagai bat eskaintzen du, erabiltzaileak zerrenda batetik balio bat aukeratu dezan. Mugikorrentzako Flex bertsioan ezaugarri hau ez dago ondo inplementatuta eta erabiltzaileak ezin ditu balioak aukeratu zerrenda hori hedatzean.

Horregatik, **CalloutButton** klasea erabili da hau behar izan den kasuetan. Inplementazio atalean azaldu da hori sakonago, baina labur esanda, botoiaren barruko balioak gordetzen dituen zerrenda bat pasatzen zaio **CalloutButton**-ari, eta botoi honekin egiten diren interakzioak programatzaileak definitutako metodoekin kontrolatzen dira aplikazioan (**changeIndex()**, **setValues()**, eta abar, adibidez).

## Ondorioak eta konklusioak

### Zer ikasi dut?

Gaur egungo iraultza teknologikoan murgilduta gaudelarik, garrantzitsua da informatikako ikasle batentzat gailu mugikorrenzako aplikazioei buruz ezagutza izatea. Proiektu honetan asko ikasi dut **Android** gailuei buruz: aplikazio txikiak hasieratik garatzen, gailuaren ezaugarrien arabera aplikazioak moldatzen, internetek eskaintzen dituen baliabideak berrerabiltzen, etabar.

Beti dago gehiago ikasteko eta munduan barrenago sartzeko aukera, baina hasiera moduan oso proiektu aprobetxagarria izan da, multimedia arloko aspektu asko **Android** ikuspuntutik aztertzeke aukera eman baitit: kamera, mikrofonoa, audio eta bideoa, eta geolokalizazioa, beste batzuen artean.

Horrez gain, Adobek eskaintzen dituen programazio-lengoaia eta *framework*-ei buruz asko ikasi dut. **ActionScript**, **Adobe Flex** eta **Adobe AIR** teknologiekin egin dut lan proiektu honetan eta hauen alde onak eta ez hain onak ezagutu, eta horri esker beste alternatiba batzuekin konparatu ahal izango ditut etorkizunean.

Lan egiteko metodologiari dagokionez ere dezente ikasi dut. Aurretik ezagutzarik ez bada izaten lan egin beharreko arloan, hobe da honi behar den denbora guztia eskaintzea, ikaskuntza denbora-tarte hori optimoagoa izan dadin, eta proiektuaren garapenarekin ahalik eta gutxien nahastu dadin.

## Zer hobetu dezaket?

Hobetu daitezkeen gauzak ere badaude, noski. Dena den, gauza bat esan behar da lehenengo: proiektu hau ingeniaritza teknikoko ikaskuntza-planean hasi nuen, eta azkenean Bolonia formatuko graduan amaitu dut. Beraz, proiektu honetan lan egin duen bakarra ez naizenez izan, ikaskuntza-plana aldatu nuenean aurrera jarraitu zuen aplikazioaren garapenak, eta berrartu nuenean, baldintzak zeharo aldatu ziren.

Dena den, honek ere lagundu dit proiektuaren plangintzari estu lotuta egotearen garrantzia ulertzen. Izan ere, momentu batzuetan zalantza asko izan ditut egin nuen aukerarekin, ez bainekien hautabide egokia zen edo ez. Horregatik, hasiera batean helburuak garbiago uzten badira, egiten den lan guztiari errendimendu hobe ateratzen zaio.

Bestalde, beste ikasle batzuek aurretik egindako lana hobeto aztertzea falta izan zait, agian, edo lehenago jabetu izan banintz honetaz, nire esfortzuak modu eraginkorragoan bideratu ahal izango nituzke hasieratik, aurreko lanetik nik egindako lanaren oinarriak hartu izan banitu, hain zuzen.

Azkenik, proiektuaren dimentsioa dela eta, programazio-lengoaia guztiz berri batekin lan egin behar izana kontuan hartzeaz gain, kodea ez dago guztiz optimizatuta. Funtzio batzuk berridatzi ahal izango lirateke errendimendu hobe eta kostu txikiagoa lortzeko.

## Zer gertatuko da proiektu honekin etorkizunean?

Proiektuaren etorkizunari dagokionez, eta zuzendariarekin hitz egin eta gero, hurrengo pausua Flex eta AIR erabiltzen uztea izango dela ematen du, HTML5 aukeratu eta oinarri honetan aplikazioa sendotuz. Flex erabiliz *bug* bat edo besteri aurre egin behar izan zaio, eta gero eta sostengu gutxiago du *framework*-ak Adoberen eskutik.

Izan ere, Adobek Flex eman zion Apache Software Foundation irabazi-asmorik gabeko erakundeari 2011. urtean. Dena den, HTML5 ez dago oraindik guztiz prest, fase esperimentalean baitaude oraindik honen barneko arlo batzuk, nahiz eta garatzaile askok erabiltzen duten, eskaintzen dituen abantaila asko direla eta.

Horrez gain, beste ikasle batek proiektu honekin jarraitzea aukeratzen badu, aurreko ikasle eta beste batzuen lanean aurkitu ahal izango du oinarria, agian Adobe AIR bertsio hau amaitzeko, agian HTML5 bertsio batera *port* bat egiteko. Dena den, gailu mugikorrentzako Babelium aplikazioa ez dago amaiturik, eta horrekin lanean jarraitu beharko da.

## Bibliografia

### Liburuak

**Izenburua:** “Professional Cairngorm”

**Egilea:** Jeremy Wischusen

**Argitaletxea:** Wiley Publishing, Inc.

**Urtea:** 2010

**Izenburua:** “Developing Applications with Adobe AIR”

**Egilea:** Véronique Brossier

**Argitaletxea:** O’Reilly Media, Inc.

**Urtea:** 2011

**Izenburua:** “Flex Mobile in Action”

**Egilea:** Jonathan Campos

**Argitaletxea:** Manning Publications Co.

**Urtea:** 2012

### Webguneak

**Babelium Project**-en webgune nagusia: <http://babeliumproject.com/>

**Babelium Project GitHub**-en: <https://github.com/babeliumproject/>

**Wikipedia**-ko hainbat orrialde: <http://www.wikipedia.org/>

**Gliffy** online diagrama erreminta: <https://www.gliffy.com>

## Eranskinak

### 1. eranskina: Lan-ingurunearen hasieraketa eta softwarea

Oso garrantzitsua da lanean jarri baino lehen, lan-ingurune apropos bat zehaztu eta prestatzea, lan-fluxua jarraitua izan dadin eta software-mailan ezustekoak suertatu ez daitezen. Babelium Project Mobile bertsioaren garapenean hurrengo software-tresnak izango dira beharrezkoak:

- **Adobe Flash Builder** SDK
- **Adobe AIR** runtime-a
- **Git** bertsio-kontrol softwarea
- **GitHub** web-orrialdean kontu bat sortzea
- **Red5** zerbitzaria
- **OflaDemo** aplikazioa

Nahiz eta azken biak Babeliumekin erabiliko ez ditugun, oso garrantzitsuak dira RTMP protokoloaren funtzionamendua aztertu eta ulertzeko. Hau aurrerago azalduko da.

### Adobe Flash Builder

Lan-ingurunea prestatzerako orduan, **SDK** edo garapenerako erreminta multzo bat behar da, hau baita tresnarik garrantzitsuena. Erabil daitezkeen **SDK** asko daude: **Eclipse**, **NetBeans**, **Android SDK**... gure kasuan, Adobe Flash Builder erabiliko da, **Eclipse**-ren gainean eraikitako IDE bat (Integrated Development Environment). Software honek **Adobe Flex** eta **ActionScript** lengoaietan programatzeko ingurune apropos bat eskaintzen du, eta **Flex**-en sortutako egiturak grafikoki bistaratzeko aukera ere eman.

Zergatik aukeratu **Adobe Flash Builder**? Hurrengo hiru aspektuak hartu nituen kontuan:

- **Linux** ingurune baten lan egitea (**Ubuntu**, esate baterako).
- Ahalik eta ingurune zehatz eta aproposena aukeratzea programazio-beharren arabera.
- Arazoak izatekotan, modu errazean konpondu ahal izatea (internet-en erreferentziak bilatuz gehien bat).

**Linux**-en lan egiteko ez dago **Adobe Flash Builder** erabiltzeko aukera %100 segururik: **fb4linux** proiektua aurkitu daiteke sarean baina ez da ofiziala, eta horrek aurreko hiru aspektuetatik gehiegi baztertzen gaitu. Izan ere, funtzionalitateekin arazorik izan ezkerro, aurkitu daitekeen feedback-a nahikoa izan ez liteke, lan-erritmoa motelduz.

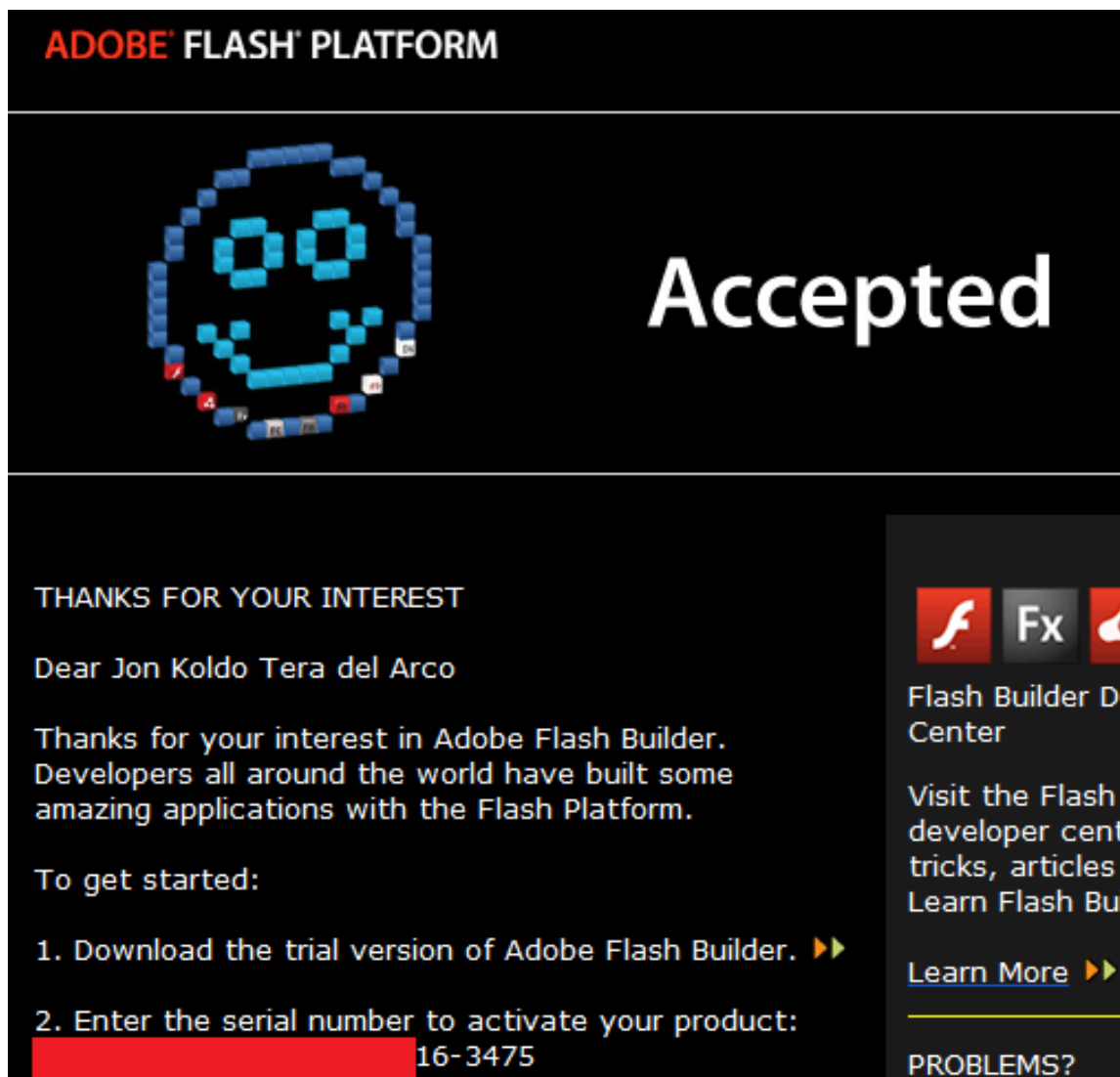
**Adobe Flash Builder** softwareak ez du **Linux**-en lan egiteko aukera eskaintzen, beraz, **Windows 7** sistema eragilearekin jarduteko alternatiba hautatu nuen. Honek, nahiz eta **Linux** batekin programatzeko nahia baztertu, eraginkortasun handiagoa erdiesten du, Adobe teknologiekin integratuta baitago. **ActionScript**, **Flex** eta **AIR** teknologiekin lan egiterako orduan oso garrantzitsua suertatuko da ezaugarri hau, kodea eraikitzeko funtzionalitateek lana asko erraztuko baitigute.

**Adobe Flash Builder** softwarea ordaindu behar da, baina ikasle eta software-garatzailleentzako doako lizentziak ere eskaintzen dituzte unibertsitateetako ikasle, irakasle eta ikertzaileentzat. Adobe Flash Builder-ren froga-bertsioa deskargatu ahal izateko, esteka hau jarraitu behar da: <https://creative.adobe.com/es/products/flash-builder>

Memoria honen osaketaren egunean ez da jada ikerkuntza-lizentzia hau gehiago eskaintzen (<http://www.adobe.com/products/flashbuilder-free-ria-tools.html> web orrian ikus daitekeen moduan), baina bere momentuan eskaera bat bidaltzen zen interneten bidez eta, ikertzaile-



egoera hori baieztatzen duen dokumentazio egokia erantsiz, **Adobe**-tik datorren maila bat jasotzen zen eskaera baieztatuz edo errefusatzuz. Nire kasuan, honen antzeko posta elektroniko bat jaso nuen kode batekin, programa aktibatu ahal izateko:



*44. irudia: Adobe Flash Builder-ren lizentzia onartua*

44. irudian ikusten denez, serial number bat entregatzen da produktua aktibatu ahal izateko eta softwarea libreki erabiltzeko.

## Adobe AIR

Adobe AIR runtime-a beharrezkoa da Adobe Flex exekutatzeko, biak integratuta datozelako Flex 3 bertsiotik aurrera. Adobe AIR deskargatzeko hurrengo orrialdera joko dugu: <http://get.adobe.com/es/air/>

Behin barruan, deskargatzeko esteka sakatu eta fitxategiaren bidalketa hasiko da. Ordenagailuan dugunean, instalazioa abiaraziko dugu fitxategia exekutatzuz.

## Git + GitHub

Kodea eguneratuta mantentzeko, oso erabilgarria da aukeran dauden online errepositorioetako bat erabiltzea. Nire kasuan, GitHub erabiltzeko aukera hartu nuen, Git bertsio-kontrol sistemarekin integratzen dena. Honen bidez, taldean garatzen den software guztia lankide

Horretarako, <https://github.com/> orrialdera jo eta erabiltzaile-kontu bat sortuko dugu. Web orrialde honetatik ere Git deskargatu eta instalatu dezakegu, kodearen errepositorioa gure ordenagailuan instalatuko den aplikazio natibo batetik eguneratuta mantendu ahal izateko.

Komando-lerroa erabiltzen badugu, lau komando hauek izango dira garrantzitsuenak (**git** aurretik idatziz):

- **add**
- **commit**
- **push**
- **rm**

Komando hauei esker, fitxategiak gehitu, konfirmatu, zerbitzarira igo, eta ezabatu ahal izango ditugu, hurrenez hurren.

## Red5 Media Server

Telefono mugikor, tablet edo beste edozein gailu elektronikori bideoak eskaintzeko, zerbitzari bat behar da. Zerbitzari honek, RTMP protokoloaren bidez eskainiko ditu bideoak streaming bidez erreproduzitzeko, FLV (Flash Video) formatua erabiliz.

Red5 multimedia zerbitzaria erabiliko da, open-source aukeren artean erakargarrienetakoa izanik. Adobe-k alternatiba eskaintzen digu, baina berriro ordaindu beharreko softwarearen arazoa suertatzen da.

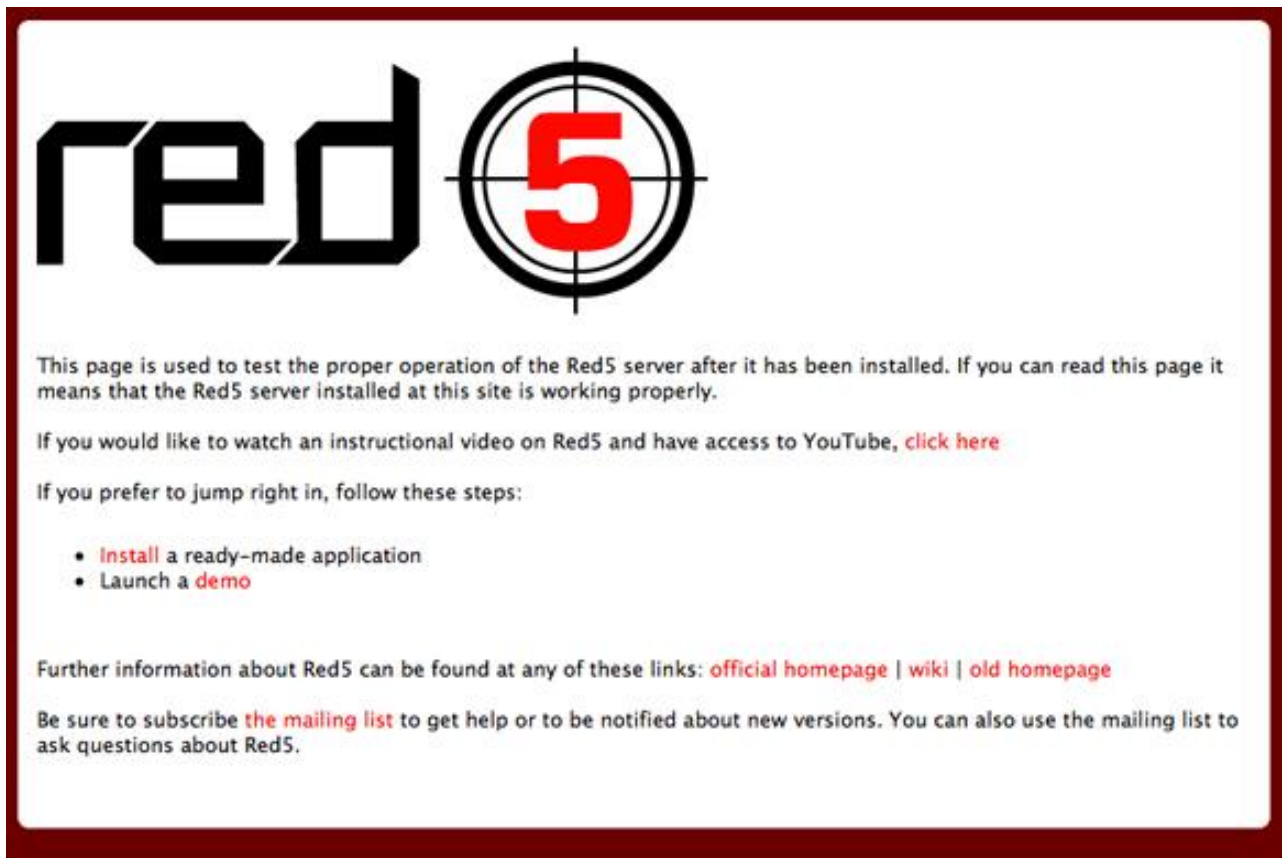
Red5 Media Server erabiliz hurrengo motatako aukerak ematen dira:

- Bideo streaming-a (FLV formatuan)
- Audio streaming-a (MP3 formatuan)
- Partekatutako objektuak
- Streaming-ak zuzenean publikatzea
- Action Message Format (AMF) formatua erabiltzea

Zerbitzaria instalatzeko, Red5 proiektuaren web-orrialdera jo behar da (<http://red5.org>) eta han deskargatzeko esteka bilatu. Proiektu honetan erabilitako bertsio konkretua hau da: [http://red5.org/downloads/red5/1\\_0/setup-Red5-1.0.0.exe](http://red5.org/downloads/red5/1_0/setup-Red5-1.0.0.exe)

Fitxategi hau deskargatu eta exekutatu dugu zerbitzaria instalatzeko. Behin instalazioaren barne gaudelarik, programak

Defektuz, 5080 portua erabiltzen du Red5 zerbitzariak, beraz, instalatu eta gero <http://localhost:5080> esteka jarraituz Red5 zerbitzariaren proba orrialdea ikusi beharko litzateke:

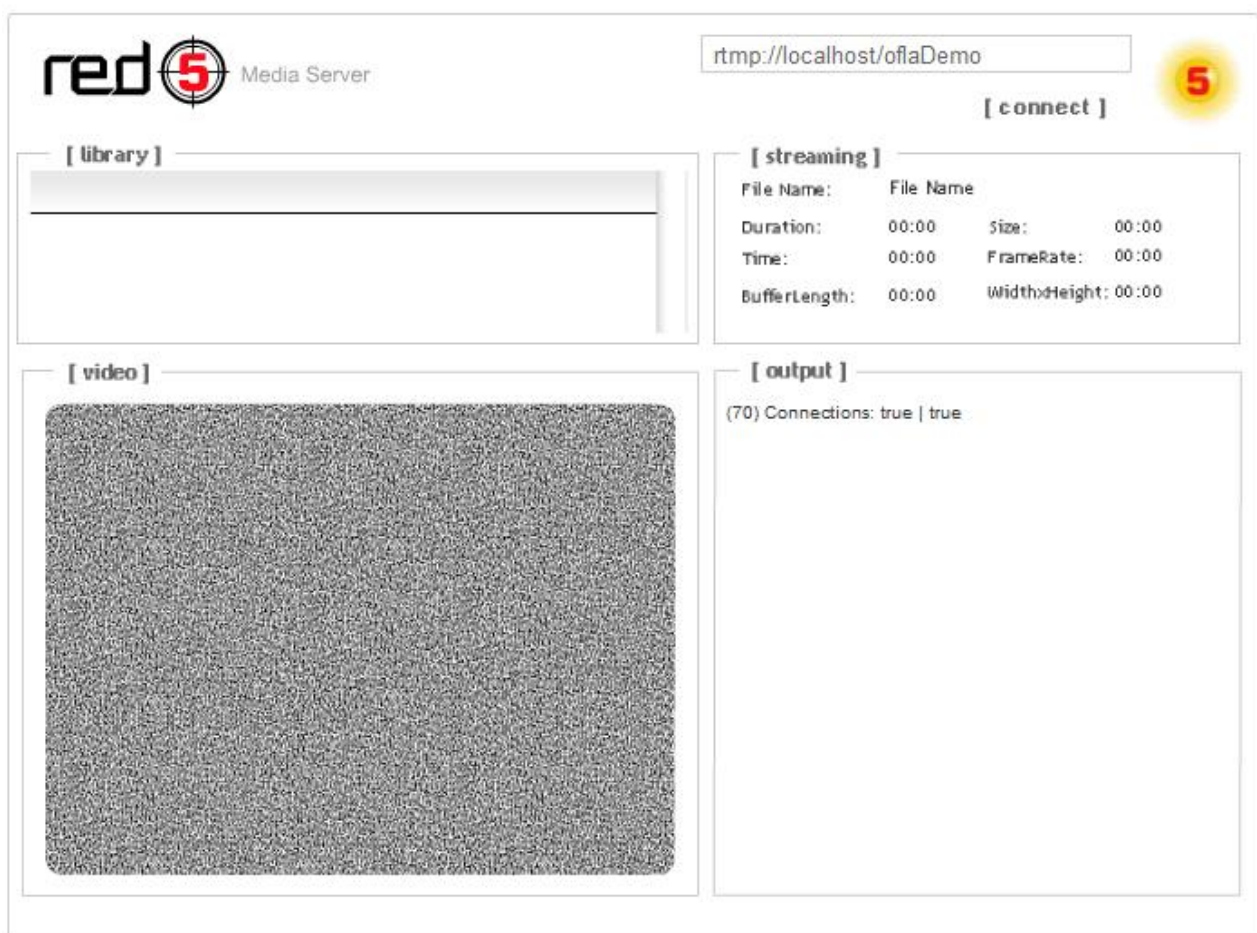


#### *45. irudia Red5 zerbitzari lokalaren pantaila nagusia*

Web-orrialde hau Apache web-zerbitzariak eskaintzen duenaren ("It works!" testu famatua erakusten duena, hain zuzen) funtzio berdina betetzen du: gure-zerbitzaria ikusgai dagoen beste zerbitzarietatik eskaerak jaso eta zerbitzatu ditzake.

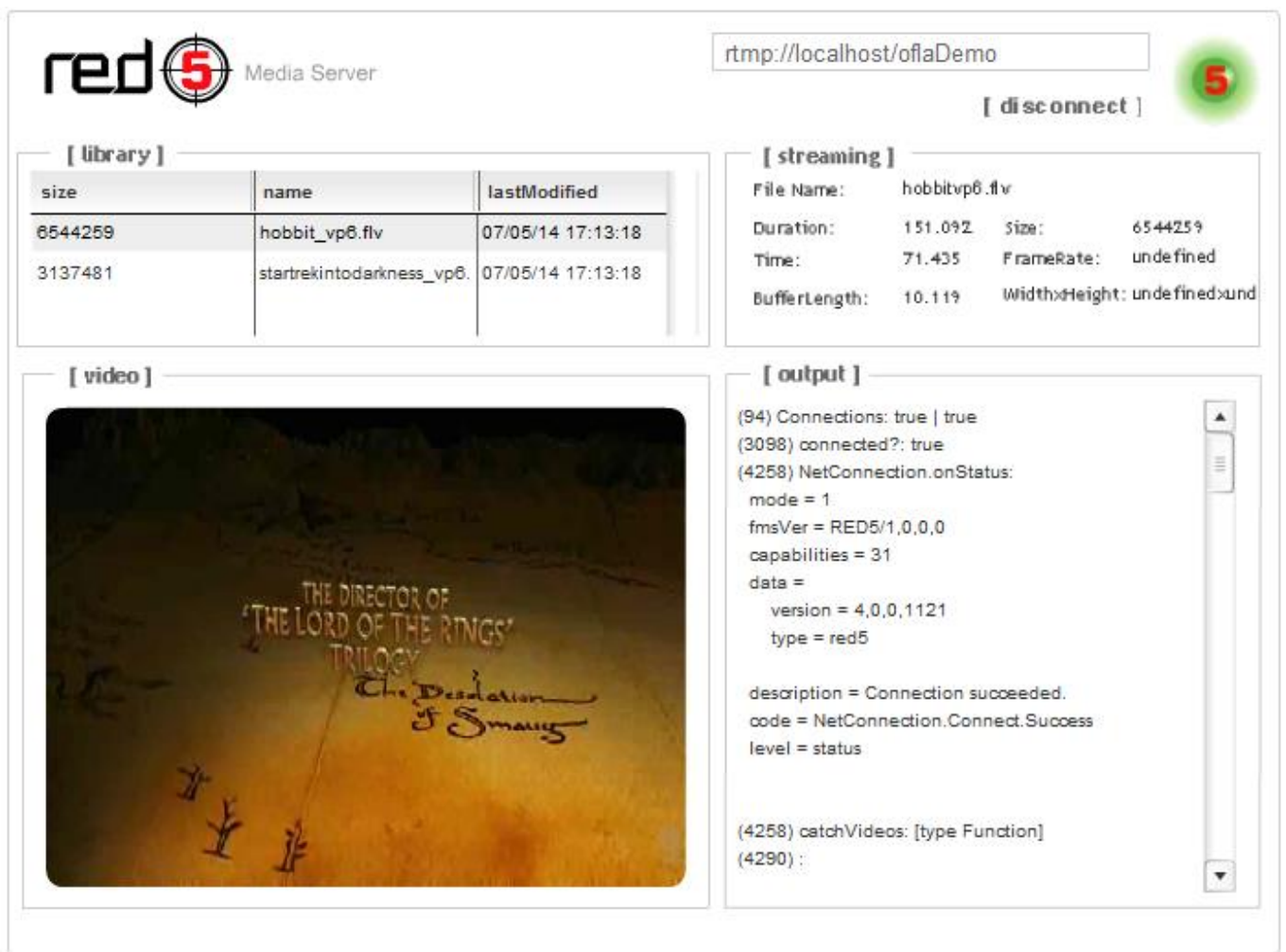
## OflaDemo aplikazioa

OflaDemo aplikazioa Red5 zerbitzariaren barruan sartuta datorren demo bat da, RTMP protokoloaren funtzionamendua probatu ahal izateko. Behin <http://localhost:5080> orrialdearen barruan gaudela, “Install a ready-made application” klik eginez aukeratu eta “App Name” zutabean “OFLA Demo” instalatuko dugu. Horren ondoren, aurreko pantailara bueltatu eta “Launch a demo” esteka sakatuz Red5-ek dakartzan demoen zerrenda bat ikusiko dugu. Lehen instalatutako OflaDemo aukeratuz gero, hurrengo pantaila ikus dezakegu:



46. irudia: OflaDemo aplikazioa

“Connect” botoian sakatuko dugu zerbitzarira konektatzeko. Dena ondo badoa, “output” leihoan NetConnection.Connect.Success egoera altxatuko da, eta “library” leihoan erabilgarri dauden streaming-en zerrenda bat bistaratuko da. FLV fitxategi hauetako batean klik bikoitza egin ezkerre, erreprodukzioa hasiko da eta “streaming” leihoan ikusiko da honen informazioa.



47. irudia: OflaDemo aplikazioa martxan

Demo honek ez ditu aukera gehiago ematen deskonektatzeaz aparte, baina RTMP protokoloaren funtzionamendua nabarmentzen du: zerbitzariarekin NetConnection konexio bat ezartzen da, eta hau ondo joan dela ikusten duenean (Connect.Success), bideoen zerrenda ikusgai jartzen du. Konexioa ezarrita dagoenez, bideoen streaming-a has daiteke.

Memoria honen hurrengo orrietan eskuragarri dagoen bigarren eranskinean, Android gailu batetik konexioa ezartzeko prozesua deskribatzen da, Adobe Flash Builder eta Red5 zerbitzaria erabiliz.

## 2. eranskina: RTMP protokoloa testatzeko aplikazio bat

### Behar diren elementuak

OflaDemo erabili barik erreproduzituko da bideo bat, ActionScript lengoia idatzitako aplikazio txiki bat erabiliz. Hurrengo elementuak programatuko dira:

- IP helbidea eta bideoaren izena sartzeko bi **TextField**.
- **NetConnection** konexioa ezartzeko botoia.
- **Video** klasearen instantziak.
- Bideoa kontrolatzeko botoi batzuk: **resume**, **pause**, **mute**, etabar.
- Aplikaziotik irtetzeko botoi bat.
- Botoi hauentzako **EventListener**-ak eta metodoak.

Aplikazioari jartzen zaizkion botoiak ActionScript lengoiaz programatuko dira, **Sprite** klasea erabiliz. Beraz, benetako botoiak ez direnez, *listener* edo *trigger* bat beharko dute hauen eremuan klik egiten dugunean ekintza bat altxatu dadin. Horregatik idatziko ditugu EventListener-ak.

### Proiektua sortu eta konfiguratu

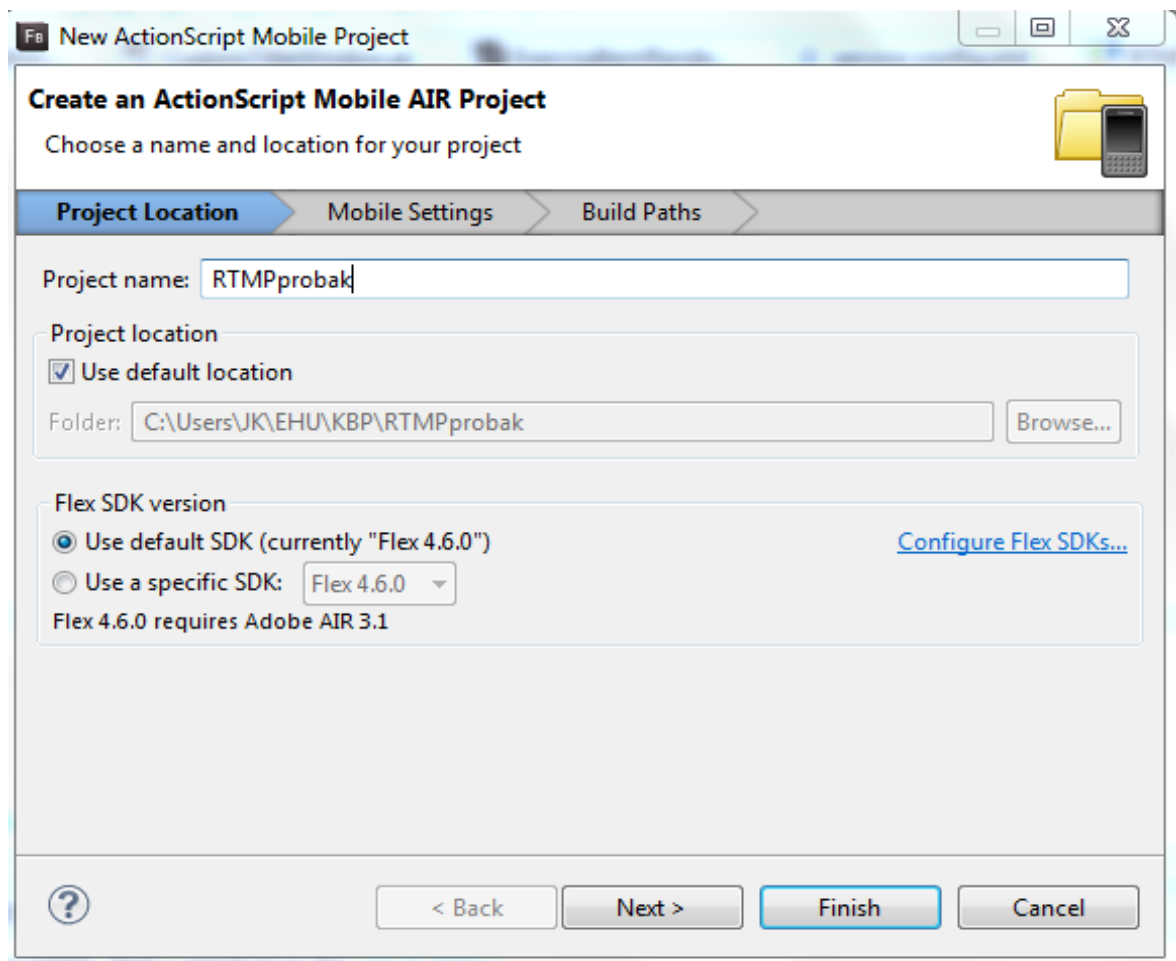
Hasteko, Adobe Flash Builder programa hasieratuko dugu. Jada instalatuta eta martxan dagoela suposatuko dugu, aurreko ataletako pausoak jarraitu izanez gero horrela izango baita. Behin gure SDK zabalik dela, goian “File” → “New” → “ActionScript Mobile Project” aukeratuko dugu. Proiektuaren konfigurazioa ezartzeko hurrengo leihoa altxatuko da:



Leiho honetan konfigurazio hiru atal nagusi ikusi eta konfiguratuko dira:

- Proiektuaren kokapena
- Telefono mugikorra eta bere ezaugarrien konfigurazioa
- Aplikazioa eraikitzeko elementuak

## Proiektuaren kokapena

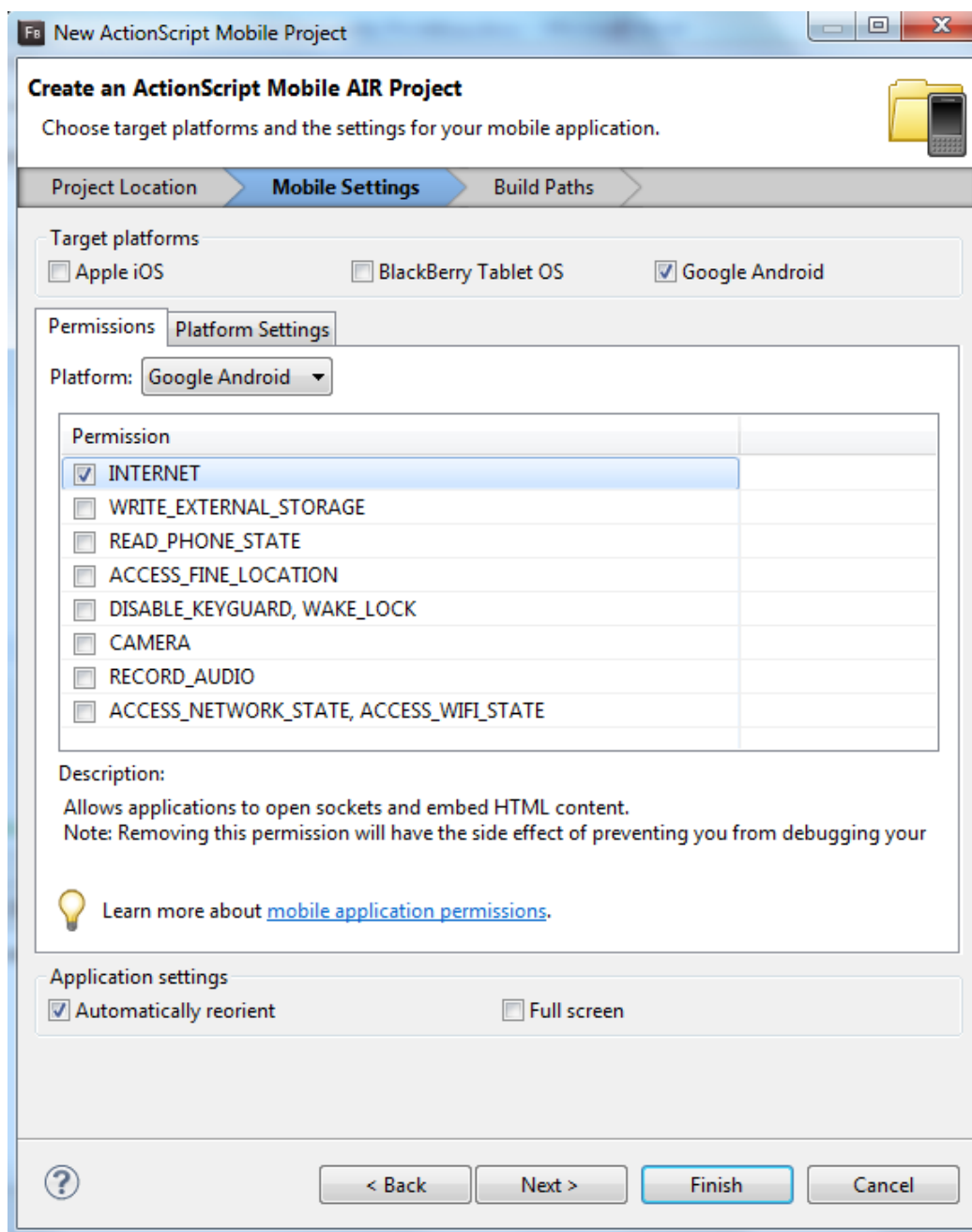


*48. irudia: Proiektua sortzeko lehen leihoa*

Hemen hainbat aukera ikusiko ditugu: proiektuaren izena, hau ordenagailuko zein lekutan gordeko den, Flex SDK-ren zein bertsio

erabiliko den... kasu honetarako, proiektuari edozein izen ezarri eta “Next” botoia sakatuko dugu. Hurrengo leihoa ikusiko da:

## Gailuaren konfigurazioa



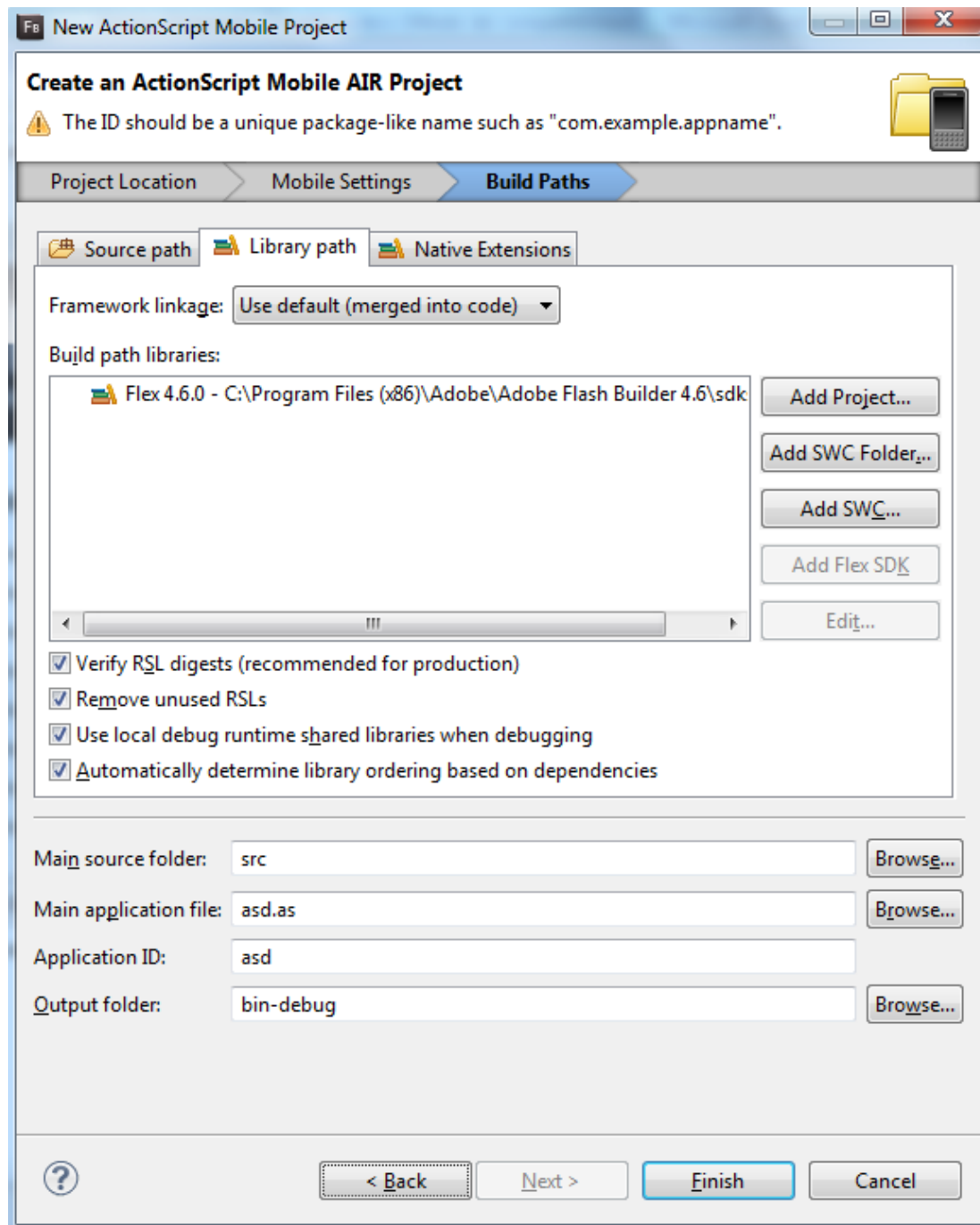
49. irudia: Proiektua sortzeko bigarren leihoa

Aplikazioa zein plataformarako garatuko den aukeratuko dugu orain: Google Android. Beste bi aukerak baztertzeko ditugunean, leihuan ikusten den laukia agertuko da. Hemen aplikazioari emango zaizkion baimenak definituko ditugu:

- **INTERNET:** portuak zabaldu eta gailutik kanpoko edukia atzitzeko baimena ematen zaio aplikazioari.
- **WRITE EXTERNAL STORAGE:** kanpoko gailu batean informazioa idazteko baimena ematen dio aplikazioari.
- **READ PHONE STATE:** aplikazioaren audioa mututzeko baimena ematen dio AIR runtime-ari, telefonoak dei bat jaso ezker, adibidez.
- **ACCESS FINE LOCATION:** aplikazioari GPS kokapena atzitzeko baimena ematen dio.
- **DISABLE KEYGUARD, WAKE LOCK:** pantaila ez erabiltzea dela eta automatikoki itzaltzeko ezaugarria kontrolatzeko baimena ematen dio aplikazioari.
- **CAMERA:** gailuaren kamera atzitzeko baimena emateko balio du.
- **RECORD AUDIO:** gailuaren mikrofonoarekin soinu-grabazioak baimentzen dira.
- **ACCESS NETWORK STATE/ACCESS WIFI STATE:** gailuaren sare-interfazeen informazioa eskuratzeko aukera ematen zaio aplikazioari.

Ezaugarri hauek kontrolatuz baimendu ez diren atzipenak ekiditi daitezke, Android gailuaren segurtasuna eta errekurtsuen kudeaketa egiaztatuz. Behin proiektuaren ezaugarriak aukeratuta, hurrengo pantailara joateko “Next” sakatuko dugu.

## Aplikazioa eraikitzeko elementuak



50. irudia: Proiektua sortzeko hirugarren leihoa

Azken pantaila honetan aplikazioaren eraikuntza zein parametrorekiko egingo den definituko da. Adibidez, Flex liburutegiak non kokatzen diren zehaztuko da, zein izango den *main* fitxategia edo fitxategi nagusia exekutatzeko orduan, zein karpetan gordeko diren *debugging* edo kodearen arazketa egiten denean sortzen den eraikuntza... hurrengo pantailan ikus daiteke guzti hau nola dagoen antolatuta:

Ikusten denez, Flex 4.6.0 liburutegia dago erabiltzeko prest. Liburutegi honetan aurkituko ditugu propietate eta metodo asko aplikazioa programatzen dugun heinean, Java eta beste lengoaia batzuetan erabiltzen diren liburutegiekin egiten den era berean.

## Frogatzeko aplikaziorako kodea (laburtuta)

Behin proiektua sortuta, kodea implementatzen has gaitezke. Proba proiektu honetarako, hurrengo kodea erabil daiteke:

```
package
{
    import flash.desktop.NativeApplication;

    import flash.display.Sprite;
    import flash.display.StageAlign;
    import flash.display.StageScaleMode;

    import flash.events.MouseEvent;
    import flash.events.Event;
    import flash.events.NetStatusEvent;

    import flash.media.SoundTransform;
    import flash.media.Video;

    import flash.net.NetConnection;
    import flash.net.NetStream;

    import flash.text.TextField;
    import flash.text.TextFieldType;
    import flash.text.TextFormat;
```

**Package** atala irekiko dugu eta jarraian berrerabiliko ditugun klase guztiak inportatuko ditugu. Hauetako asko Babelium Project-en erabiliko dira, beraz, azalpen txiki bat merezi dute:

## Desktop

Desktop klasearen barruan **NativeApplication** dago, eta honen *exit()* metodoa erabiliko dugu aplikazioa itxi nahi dugunerako.

## Display

Display-ri esker, froga txiki honetako botoiak marraztuko ditugu **Sprite** erabiliz. Horrez gain, **StageAlign** eta **StageScaleMode** erabiliko dira pantailaren orientazioa kontrolatzeko.

## Events

Botoietan jarriko diren **EventListener**-ak inplementatzeko erabiliko dugu klase hau. Gainera, RTMP konexioa hasieratzeko **NetStatusEvent** bat ere altxatu behar da, konexioa ondo edo gaizki joan dela adierazteko eta horren arabera jarduteko.

## Media

**Media** klaseak audio eta bideoarekin jokatzeko aukera emango digu, **SoundTransform** eta **Video** azpiklaseen bitartez.

## Net

**NetConnection** eta **NetStream** ezinbestekoak dira RTMP protokoloaz konexioa ezarri eta bideoaren *streaming*-a arrakastaz erreproduzitzeko.

## Text

**Text** klasean berrerabiliko ditugun baliabideak testuari formatua eman eta botoien barruan jartzeko dira. **TextField**-ak ere inportatuko ditugu bideoaren izena eta IP helbidea emateko.

```
public class RTMPprobak extends Sprite
{
    var streamClient:Object = new Object();
    var fullscreen:Boolean = false;
    var mute:Boolean = false;

    public function RTMPprobak(){
        stage.align = StageAlign.TOP_LEFT;
        stage.scaleMode = StageScaleMode.NO_SCALE;
        drawBackground();

        var transform:SoundTransform = new SoundTransform();
        var connection:NetConnection = new NetConnection();
        var connectServer:TextField = createTextField(10, 10, stage.stageHeight / 18, stage.stageWidth * 0.65,
14, "rtmp://192.168.1.131");
        var videoPathServer:TextField = createTextField(10, 15 + (stage.stageHeight / 18), stage.stageHeight /
18, stage.stageWidth * 0.65, 14, "hobbit_vp6");
```

Aldagai batzuk definitu ditugu hemen: *streaming*-a egiteko bezero bat, bi aldagai boolear, soinua kontrolatzeko beste aldagai bat, sare konexioa, eta bi TextField. Hainbat gehiago dira (botoi guztiak falta dira hemen), baina ez dira guztiak hemen jarriko. Kode guztia memoriaren amaierako eranskinean aurkitu daiteke.

```
var video:Video = new Video();
video.width = stage.stageWidth;
video.height = stage.stageHeight / 2;
video.y = stage.stageHeight*0.15;

function onConnect():void{
    if (!connection.connected){
        connection.addEventListener(NetStatusEvent.NET_STATUS, netConnectionEvent);
        connection.connect(connectServer.text + "oflaDemo");
    } else {
        streamClient.close();
        connection.close();
        removeChild(video);
        stage.removeEventListener(Event.RESIZE, onResize);
    }
}

function netConnectionEvent(event:NetStatusEvent):void {
    event.target.removeEventListener(NetStatusEvent.NET_STATUS,
netConnectionEvent);
    if (event.info.code == "NetConnection.Connect.Success") {
        var stream:NetStream = new NetStream(connection);
        stream.addEventListener(NetStatusEvent.NET_STATUS, onStreamEvent);
        streamClient = stream;
        video.attachNetStream(stream);
        stream.play(videoPathServer.text + ".flv");
        addChild(video);
        stage.addEventListener(Event.RESIZE, onResize);
    }
}
```

Hemen, bideo aldagaia eta konexioa egiteko bi metodo: lehenengo metodoak konexiorako *listener* bat ezartzen du, eta konexioa hasi zerbitzariarekiko. *Event*-a jarritako *listener*-rak jasotzen du eta bigarren metodoa hasieratzen du, *streaming*-a hasieratu dadin.

```
function drawBackground():void{
    //Atzekaldea marrazteko kodea
}

function createSprite(xPos: int, yPos: int, x:int, y:int, col: int, intext:String):Sprite{
    //Botoiak marrazteko kodea
}

function createTextField(x:int, y:int, height:int, width:int, fontSize: int, text:String):TextField{
    //Testu leihoak marrazteko kodea
}
```

Kode zati honetan atal grafikoa implementatzen da. ActionScript kode hutsa da; Babelium aplikazioan Flex erabiltzen da interfaze grafikoari dagokion zatia kodetzeko. Froga aplikazio txiki honetarako ez gara Flex lengoian sartuko.

```
function onExit(event:MouseEvent):void{
    NativeApplication.nativeApplication.exit();
}

function onResume():void{
    streamClient.resume();
}

function onPause():void{
    streamClient.pause();
}

function onMute():void{
    if (!mute){
        streamClient.soundTransform = new SoundTransform(0);
        mute = true;
    } else {
        streamClient.soundTransform = new SoundTransform(1);
        mute = false;
    }
}

function onResize():void{
    //Pantaila biratzean birdoitzeko kodea
}

function onStreamEvent():void{
}
```



## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

```
function onMetaData():void{  
    }  
}
```

Azken zati honetan, *listener*-ak deklaratzeko esleitu zaizkien metodoak definitzen dira, laburbilduz, botoiak sakatzeko zein izango den burutu behar den ekintza.

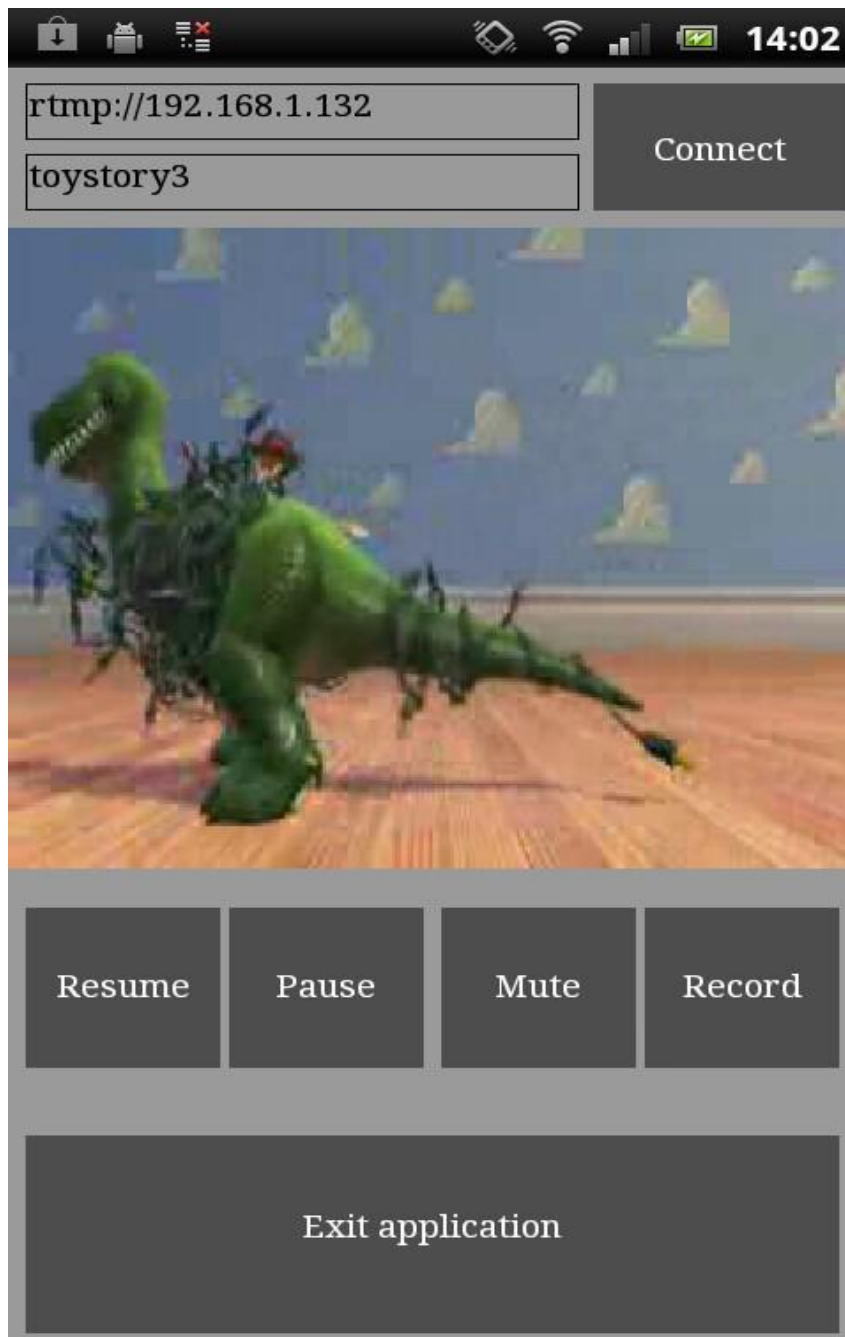
## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

Hona hemen aplikazioak duen gutxi gorabeherako itxura, Sony Ericsson Xperia Neo V telefono mugikor baten testeatzerako orduan:



*51. irudia: Aplikazioa Sony Ericsson Xperia Neo V telefonoan*

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

Hurrengo hau, berriz, Samsung Galaxy S5 mugikor batean egina. Telefonoaren ezaugarri teknikoak aldatzen direnez, objektu batzuen proportzioak ere egokitzen dira:



*52. irudia: Aplikazioa Samsung Galaxy S5 telefonoan*

### 3. eranskina: RTMP froga aplikaziorako kodea

Hurrengo orrialdeetan aurkituko dugu “Lan-ingurunearen hasieraketa” ataleko RTMP frogan erabili den kodea, ActionScript lengoia oinarritzen dena guztiz:

```
package
{
    import flash.desktop.NativeApplication;

    import flash.display.Sprite;
    import flash.display.StageAlign;
    import flash.display.StageScaleMode;

    import flash.events.MouseEvent;
    import flash.events.Event;
    import flash.events.NetStatusEvent;

    import flash.media.SoundTransform;
    import flash.media.Video;

    import flash.net.NetConnection;
    import flash.net.NetStream;

    import flash.text.TextField;
    import flash.text.TextFieldType;
    import flash.text.TextFormat;

    public class RTMPprobak extends Sprite
    {
        var streamClient:Object = new Object();
        var fullscreen:Boolean = false;
        var mute:Boolean = false;

        public function RTMPprobak() {
            stage.align = StageAlign.TOP_LEFT;
            stage.scaleMode = StageScaleMode.NO_SCALE;
            drawBackground();
            var transform:SoundTransform = new SoundTransform();
            var connection:NetConnection = new NetConnection();
            var connectServer:TextField = createTextField(10, 10, stage.stageHeight / 18, stage.stageWidth * 0.65,
14, "rtmp://192.168.1.131");
            var videoPathServer:TextField = createTextField(10, 15 + (stage.stageHeight / 18), stage.stageHeight /
18, stage.stageWidth * 0.65, 14, "hobbit_vp6");

            var exitButton:Sprite = new createSprite(10, ((stage.stageHeight)/6)*5, stage.stageWidth - 20,
(stage.stageHeight/6) - 10, 0x000000, "Exit application");
            exitButton.addEventListener(MouseEvent.CLICK, onExit);

            var resumeButton:Sprite = new createSprite(10, (stage.stageHeight*0.65) + 5, (stage.stageWidth / 4) -
10, stage.stageHeight/8, 0x000000, "Resume");
            resumeButton.addEventListener(MouseEvent.CLICK, onResume);

            var pauseButton:Sprite = new createSprite((stage.stageWidth / 4) + 5, (stage.stageHeight*0.65) + 5,
(stage.stageWidth / 4) - 10, stage.stageHeight/8, 0x000000, "Pause");
            pauseButton.addEventListener(MouseEvent.CLICK, onPause);
        }
    }
}
```

# Babelium Project: Android gailuetarako Adobe AIR prototipoa

## PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

```
var muteButton:Sprite = new createSprite((stage.stageWidth / 2) + 5, (stage.stageHeight*0.65) + 5,
(stage.stageWidth / 4) - 10, stage.stageHeight/8, 0x000000, "Mute");
muteButton.addEventListener(MouseEvent.CLICK, onMute);

var recordButton:Sprite = new createSprite((stage.stageWidth / 4)*3, (stage.stageHeight*0.65) + 5,
(stage.stageWidth / 4) - 10, stage.stageHeight/8, 0x000000, "Record");
//recordButton.addEventListener(MouseEvent.CLICK, onRecord);

var connectButton:Sprite = new createSprite(stage.stageWidth * 0.69, 10, stage.stageWidth * 0.30,
stage.stageHeight / 10, 0x000000, "Connect");
connectButton.addEventListener(MouseEvent.CLICK, onConnect);

var video:Video = new Video();
video.width = stage.stageWidth;
video.height = stage.stageHeight / 2;
video.y = stage.stageHeight*0.15;

function onConnect():void{
    if (!connection.connected){
        connection.addEventListener(NetStatusEvent.NET_STATUS, netConnectionEvent);
        connection.connect(connectServer.text + "/oflaDemo");
    } else {
        streamClient.close();
        connection.close();
        removeChild(video);
        stage.removeEventListener(Event.RESIZE, onResize);
    }
}

function netConnectionEvent(event:NetStatusEvent):void {
    event.target.removeEventListener(NetStatusEvent.NET_STATUS,
    netConnectionEvent);
    if (event.info.code == "NetConnection.Connect.Success") {
        var stream:NetStream = new NetStream(connection);
        stream.addEventListener(NetStatusEvent.NET_STATUS, onStreamEvent);
        streamClient = stream;
        video.attachNetStream(stream);
        stream.play(videoPathServer.text + ".flv");
        addChild(video);
        stage.addEventListener(Event.RESIZE, onResize);
    }
}

function drawBackground():void{
    var background:Sprite = new Sprite();
    background.graphics.beginFill(0x333333, 0.5);
    background.graphics.drawRect(0, 0, stage.stageWidth, stage.stageHeight);
    background.graphics.endFill();
    addChild(background);
}

function createSprite(xPos: int, yPos: int, x:int, y:int, col: int, intext:String):Sprite{
    var temp:Sprite = new Sprite();
    temp.graphics.beginFill(col, 0.5);
    temp.graphics.drawRect(xPos, yPos, x, y);
    temp.graphics.endFill();
    var temptext:TextField = new TextField;
    var mytextformat:TextFormat = new TextFormat;
    mytextformat.size = 20;
    mytextformat.align = "center";
```

```
mytextformat.color = 0xFFFFFFFF;
temptext.text = intext;
temptext.width = x;
temptext.x = xPos;
temptext.y = yPos + (y*0.35);
temptext.setTextFormat(mytextformat);
addChild(temp);
temp.addChild(temptext);
return temp;
}

function createTextField(x:int, y:int, height:int, width:int, fontSize: int, text:String):TextField{
    var temp:TextField = new TextField;
    temp.type = TextFieldType.INPUT;
    temp.border = true;
    var mytextFormat:TextFormat = new TextFormat;
    mytextFormat.size = fontSize;
    temp.defaultTextFormat = mytextFormat;
    temp.text = text;
    temp.x = x;
    temp.y = y;
    temp.height = height;
    temp.width = width;
    temp.multiline = true;
    temp.wordWrap = true;
    addChild(temp);
    return temp;
}

function onExit(event:MouseEvent):void{
    NativeApplication.nativeApplication.exit();
}

function onResume():void{
    streamClient.resume();
}

function onPause():void{
    streamClient.pause();
}

function onMute():void{
    if (!mute){
        streamClient.soundTransform = new SoundTransform(0);
        mute = true;
    } else {
        streamClient.soundTransform = new SoundTransform(1);
        mute = false;
    }
}

function onResize():void{
    if (!fullscreen){
        video.x = 0;
        video.y = 0;
        video.height= stage.stageHeight;
        video.width = stage.stageWidth;
        fullscreen = true;
        removeChild(connectServer);
        removeChild(videoPathServer);
        removeChild(exitButton);
        removeChild(resumeButton);
        removeChild(pauseButton);
        removeChild(muteButton);
    }
}
```

## Babelium Project: Android gailuetarako Adobe AIR prototipoa

### PROIEKTUAREN MEMORIA

Jon Koldo Tera del Arco

---

```
        removeChild(recordButton);
        removeChild(connectButton);
    } else {
        video.width = stage.stageWidth;
        video.height = stage.stageHeight / 2;
        video.y = stage.stageHeight*0.125;
        fullscreen = false;
        addChild(connectServer);
        addChild(videoPathServer);
        addChild(exitButton);
        addChild(resumeButton);
        addChild(pauseButton);
        addChild(muteButton);
        addChild(recordButton);
        addChild(connectButton);
    }
}
function onStreamEvent():void{

}

function onMetaData():void{

}
}
}
```

## 4. eranskina: myResources.properties fitxategietan gehitutako sarrerak

Hizkuntzak ezartzeko erabiltzen diren fitxategietan gehitutako testu-kateak, etorkizunerako egin beharreko aldaketak egin daitezzen erreferentzia gisa:

**Hizkuntzak:** eu\_ES, es\_ES, en\_US

**Gehitutako sarrerak:**

ERROR\_INSUFFICIENT\_LEVEL

LABEL\_DONE

LABEL\_SENDING

YOUR\_ASSESSMENT\_IS\_INCOMPLETE

ERROR\_WHILE\_CHANGING\_LANGUAGE