eman ta zabal zazu

BILBOKO INDUSTRIA INGENIARITZA TEKNIKOKO UNIBERTSITATE-ESKOLA

KUDEAKETAREN ETA INFORMAZIO SISTEMEN INFORMATIKAREN INGENIARITZAKO GRADUA

GRADU AMAIERAKO LANA

2015 / 2016

TIRADERA AUTOMATIZATU BATEN IREKITZE ETA IXTEAREN ERREGISTROA ALDERANTZIZKO INGENIARITZA ERABILIZ

MEMORIA

IKASLEAREN DATUAK:	**ZUZENDARIAREN DATUAK:**
IZENA: PABLO	NOMBRE: OSKAR
ABIZENAK: CARMONA LUQUE	ABIZENAK: CASQUERO OYARZABAL
	SAILA: SISTEMEN INGENIARITZA ETA AUTOMATIKA
SINATUTA:	SINATUTA:
DATA: 2016/02/11	DATA: 2016/02/11

Laburpena:

Tresnen erabileraren kontrol, erregistro eta automatizazioa gero eta nabariagoa den garai batean gaude. Gaur egun, industrian batez ere, tresnen eta langileen kontrol maximoa eduki nahi da. Honela, zeoze gertatuz gero, kontrol eta erregistro horiei erreparatuko diegu.

GAL honetan, industrian eta konkretuki farmazeutika eta bitxigintza arloetan erabiltzen den tiradera automatizatu baten irekitze eta ixteen denbora errealeko erregistroa gauzatu da. Oso lagungarria izan daiteke zuzenean tiradera automatizatuan mugimendu bat ematen denean berealako erregistro bat sortzea aparteko programak erabili gabe. Honela, erabiltzaileak tiradera erabiltzean ez du erregistroa bere kabuz gorde behar, denbora eta lana aurreztuz batez ere tiradera automatizatu asko mugimendu askorekin dagoen inguruneetan.

Azkenik, GAL hau burutzeko alderantzizko ingeniaritzako ezagutzak erabili eta aplikatu beharko dira, baita aurreanalisi bat egin, tiradera automatizatuaren ezaugarriak ezezagunak baitira. Horregatik, aurrerantzean ikusiko den moduan, tiraderari “kutxa-beltz” izena jartzea erabaki da.

Aurkibidea

Irudien aurkibidea

Taulen aurkibidea

1. Sarrera

1.1 Asmoa

Tiradera automatizatu baten irekitze eta ixtearen denbora errealeko erregistroa egiten duen web eta mahaigaineko aplikazioa sortzea izango da.

1.2 Negozio-esparrua

Gaur egun edozein arlotako aplikazio zein web-aplikazio aurkitu ditzazkegu merkatuan eta sarean: bai lanerako, bai aisirako, bai ikasteko...

Aplikazio honen arloa enpresa izango da. Tiradera automatizatuen kontrola eta erregistroa bermatu nahi duten enpresentzat erabilgarria izango da, batez ere farmazeutika eta bitxigintza sektoreetan. Proiektu honetan egikarituko den aplikazioak sortzen dituen emaitzak aurkezteko, web-aplikazio bat garatuko da. Web-aplikazio hau tiraderan gertatzen ari dena urrunetik bistaratzeko erabilgarri izango da.

1.3 Proiektua aukeratzeko arrazoia

Proiektu hau implementatzeko arrazoi nagusia, proiektuak graduan ematen ez diren konpetentzien ikasketa dakarrelako da. Proiektua gauzatu ostean, produktu ezezagun baten espezifikazioak zein funtzionamenduak lortzeko eman behar diren pausuak jakingo dira. Gaur egun aurrekari gutxi ditu mota honetako aplikazio batek, beraz erronka handi bat da proiektu honetan lan egitea.

Beste arrazoi bat proiektuaren konplexutasuna da. Alde batetik, proiektuak informatika eta elektronika bereganatzen ditu, honek dakarren zailtasunekin. Bestalde, tiraderaren ezaugarri teknikoak ez dira ezagutzen, hau da, ez dago hardware datasheet-ik edo software API-rik(tiradera “kutxa-beltz” bat da), beraz ezinezkoa da bit jarioa zuzenean dekodifikatu. Guzti honek nahitaezkoa egiten du alderantzizko ingeniaritza aplikatzea proiektu honetan.

1.4 Siglak eta akronimoak

- **UART:** Ingesetik datorren *Universal Asynchronous Receiver-Transmitter.* Serie portuak eta serie gailuen kontrola egiten duen gailua. Gailuaren plaka nagusian kokatzen da.

- **I2C:** Ingelesetik datorren *Inter-Integrated Circuit.* Industrian asko erabilitako serie komunikazioen bus bat da, sistema integratuetan periferiko-mikrokontrolagailu komunikazioa ahalbidetzeko.

- **SPI:** Ingelesetik datorren *Serial Peripheral Interface.* Gailu elektronikoen zirkuitu integratuen arteko komunikazioa ahalbidetzen duen komunikazio estandarra da.

- **CAN:** Ingelesetik datorren *Controller Area Network.* Bosch alemaniar enpresak garatutako komunikazio protokoloa.

- **Python:** Multiparadigmikoa den programazio lengoaia. Lengoai interpretatua da, multiplataforma eta kode irekikoa.

- **Csv:** Ingelesetik datorren *comma-separated values.* Formatu zabalduko dokumentu mota bat da, zein datuak taula baten era erraz baten aurkezteko balio duen.

2. Planteamendua

2.1 Helburuak

GAL honek **tiradera automatizatu baten irekitze eta ixtearen denbora errealeko erregistroa** egitea du helburu. Sortutako erregistroa web-aplikazio baten bitartez aurkeztuko da.

Proiektuaren helburu pertsonalak:

- Planifikazio bat duen benetako proiektu batekin lan egitea.
- Graduan ematen ez diren elektronikako ezaguerak bereganatzea.
- Programazio lengoai batzuk sakontzea
- Alderantzizko ingeniaritza aplikatzeko konpetentziak bereganatzea.

2.2 Deskribapena

Proiektua tiradera automatizatu baten irekitze eta ixteen kontrola gauzatera bideratua izango da. Esan bezala, emaitzak web-aplikazio baten aurkeztuko dira era sinple batean bistaratuak izateko.

Proiektua hiru multzo garrantzitsuetan banatuko da:

- **Bit jarioaren aurre-analisia egiten duen python programa:**

Bi programataz osatutako atala:
    - tiraderaParser.py
    - tiraderaAnalizer.py

Programa hauek csv a errekorritzen dute serie mezuak hartuz eta garatutako hiztegi baten bitartez hitzak osatuz. Horretarako saioetan ikusitako hitz berezi baten oinarritzen da programa, hitz hau agertzen denean lerro aldaketa eginez. Jakina denez, tiraderan akzio bat ematen denean sortuko den hitzak luzera handiagoa izango du, beraz tirderaAnalizer delakoak x baino luzera handiagoa duten gertaerak sailkatuko ditu.

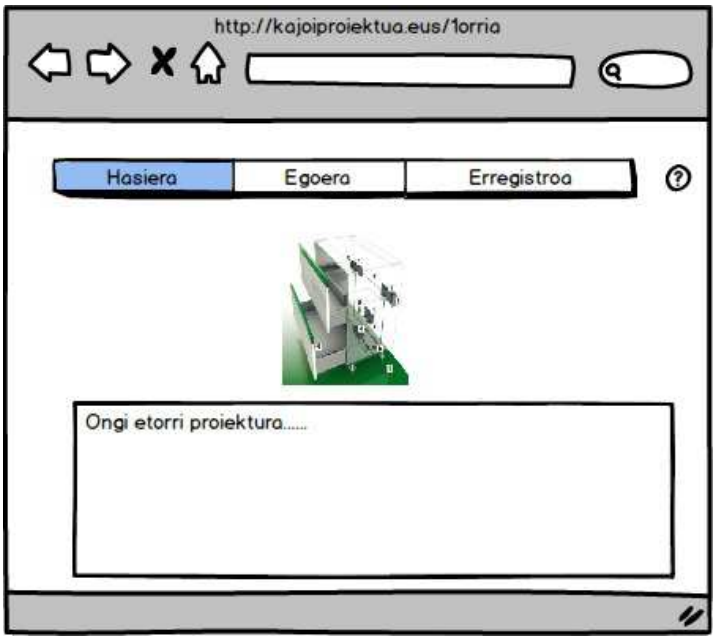
- **Denbora errealean irekitze eta ixteen kontrola gauzatzen duen java programa:**

Bit jarioa etengabe denbora errealean jasoko duen atala da. Etengabe exekutatzen egongo den programa honen funtzioa x katea bit jarioan agertzen den maiztasunaren neurketa egitea izango da. Maiztasunaren edo kateen luzeraren arabera desberdindu ahal izango dugu irekitze eta ixte bat egon dela. Tiraderan mugimendu bat ematen denean programa honek datu basearen eguneraketa gauzatuko du baita ere.

- **Web-aplikazioa:**

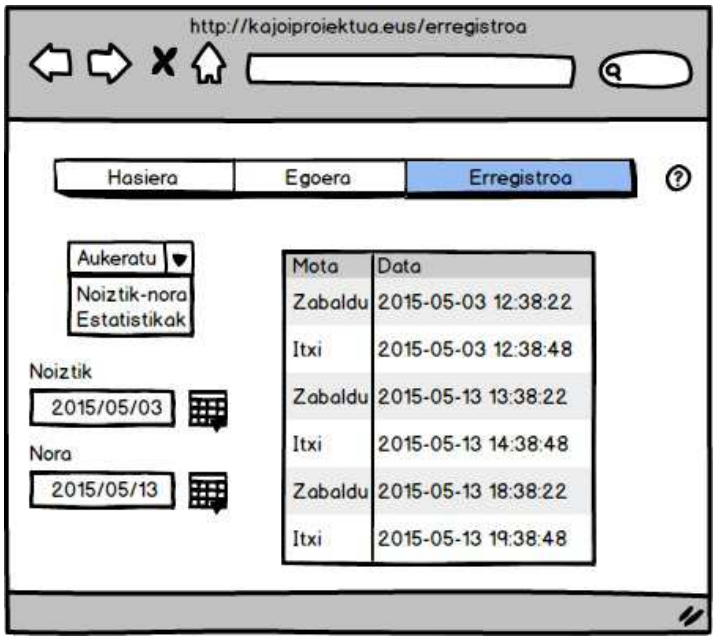
Webgunea emaitzak bistaratzera edo tiraderaren mugimenduen kontsultara bideratua izango da. Bertan hiru atal nagusi bereiztuko dira:

1. **Hasiera orria**



Irudia 1: Hasiera orri zirriborroa

2. **Erregistroen pantaila:** Bertan bi aukera izango ditugu. Erregistro orokorra aukeratuz gero, tiraderan emandako mugimendu guztiak agertuko dira datarekin batera. Noiztik-nora atala aukeratzen badugu, guk aukeratutako daten artean emandako mugimendu guztiak agertuko dira datarekin batera.



Irudia 2: Erregistro pantaila zirriborroa

1. **Laguntza pantaila:** Tiradera automatizatuaren irekitze eta ixteen erregistroa lortzeko eman behar diren pausuak aipatuko ditu.

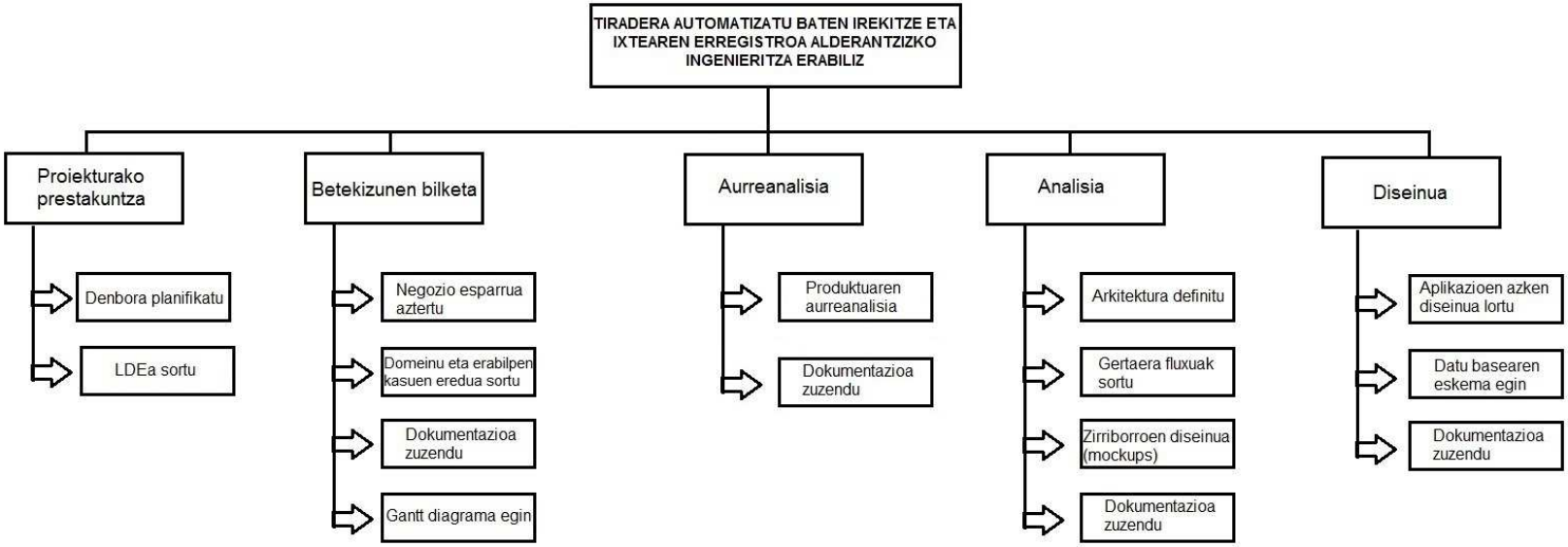
2.3 Plangintza-irismena

2.3.1 Atalak

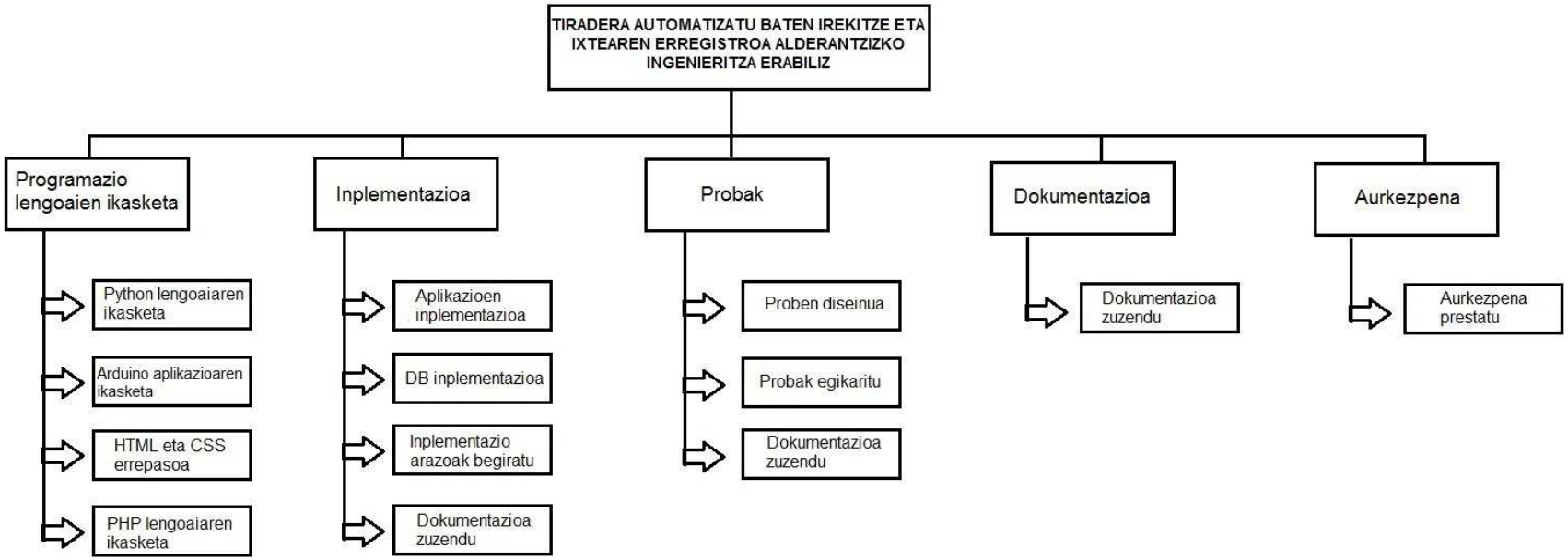
GAL hau garatzeko hainbat zati ezberdin egongo dira, non eginkizun ezeberdinen banaketa egongo den:

1. **Proiekturako prestakuntza:** Proiektua hasteko behar diren tresna guztiak prestatuko dituen fasea izango da.
2. **Betekizunen bilketa:** Proiektuaren betekizunak(funtzionalak eta ez funtzionalak) bilduko dituen fasea. Fase honetan aplikazioren atalak definituko ditugu baita ere diagrama ezberdinekin(DE, klase diagrama...)
3. **Aurreanalisia:** GAL aren analisi fasea hasi baino lehen, lan karga garrantzisu bat izan duen aurreanalisi edo ikerketa prozesu bat eman da. Esan bezala, tiraderaren ezaugarri teknikoak ez dira ezagutzen, hau da, ez dago hardware datasheet-ik edo software API-rik(tiradera “kutxa-beltz” bat da), beraz ezin da bit jario hori zuzenean dekodifikatu. Fase honetan, busetik doan bit jarioa dekodifikatzeko eman den aurreanalisia aurkezten da.
4. **Analisia:** Fase honetan aplikazioak beharrezkoak dituen lengoaiak eta arkitektura definituko dira.
5. **Diseinua:** Fase honetan programatzen hasi baino lehenagoko estruktura orokor bat lortuko dugu.
6. **Programazio lengoaien ikasketa:** Proiektuak bereganatzen dituen programazio lengoaietan sakonduko dugu fase honetan.
7. **Inplementazioa:** Proiektua inplementatuko dugun fasea izango da.
8. **Probak:** Gure aplikazioa ondo dabilen egiaztatzeko fasea izango da, probak diseinatuz eta egikarituz.
9. **Dokumentazioa:** Fase hau proiektu osoan zehar aurkitzen da, non eginkizun guztiak dokumentatu eta aplikazioaren manuala sortuko dugun.
10. **Aurkezpena:** GAL aren defentsa egunerako baliagarria den aurkezpena prestatuko den fasea.

2.3.2 LDE diagrama



Irudia 3: LDE diagrama



Irudia 4: LDE diagrama

2.3.3 Eginkizunen deskribapena

- Eginkizunen deskribapena osatzeko erabiliko den nomenklatura:

Ikurra	Deskribapena
LP x	xx zenbakikuntza duen lan paketea
T x y	xx pakete barnean egin beharreko zeregina yy zenbakikuntzarekin
UE x z	xx pakete barneko entrega unitatea zz zenbakikuntzarekin
EDE	Egiteko denbora estimatua
LK	Lan karga
EU	Entrega unitatea
H	Beharrezko erramintak

Taula 1: Eginkizunen deskribapena

- Zereginen eta lan paketeen deskribapena:

LP1. Proiekturako prestakuntza

Lan pakete honetan proiektua hasi baino lehenagoko ordu planifikazioa eta LDEa sortuko dugu.

EU1	EDE	LK
	4 egun	8 ordu

T11. Denbora planifikatu

Proiektuaren denbora eta epeen planifikazioa gauzatuko da atal honetan.

EDE	LK	H	EU
2 egun	4 ordu	Proiektu kudeaketa irakasgaiko apunteak eta kalkulu orri programa bat(Libreoffice calc,excel...)	EU11

T12. LDEa sortu

Proiektuaren lan dekonposaketa egitura gauzatuko da lan pakete honetan.

EDE	LK	H	EU
2 egun	4 ordu	Diagramak egiteko herraminta bat	EU12

LP2. Betekizunen bilketa

Proiektuaren betekizunak(funtzionalak eta ez funtzionalak) bilduko dituen fasea. Fase honetan aplikazioren atalak definituko ditugu baita ere diagrama ezberdinekin(DE, klase diagrama...)

EU2	EDE	LK
	9 egun	18 ordu

T21. Negozio esparrua aztertu

Garatu nahi dugun antzerako aplikazioen azterketa gauzatuko da atal honetan.

EDE	LK	H	EU
2 egun	4 ordu	Internet	EU21

T22. Domeinuaren eta erabilpen kasuen ereduak sortu

Domeinuaren eredua eta erabipen kasuak sortuko ditugu atal honetan.

EDE	LK	H	EU
3 egun	6 ordu	Eredu hauek irudikatzeko programa bat(visual paradigm)	EU22

T23. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
2 egun	4 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU23

T24. Gantt-a sortu

Atal honetan lan paketeen eta beren atalen denbora planifikatzeko gantt diagrama sortuko dugu.

EDE	LK	H	EU
2 egun	4 ordu	Gantt diagramak sortzeko herraminta bat(Ganttproject)	EU24

LP3. Aurreanalisia

GAL aren inplementazio fasea hasi baino lehen, lan karga garrantzisu bat izango duen analisi edo ikerketa prozesu bat emango da. Esan bezala, tiraderaren ezaugarri teknikoak ez dira ezagutzen, hau da, ez dago hardware datasheet-ik edo software API-rik(tiradera “kutxa-beltz” bat da), beraz ezin da bit jario hori zuzenean dekodifikatu. Beraz lan pakete honetan tiraderaren ezaugarrien aurreanalisia egingo da.

EU3	EDE	LK
	15 egun	30 ordu

T31. Produktuaren aurreanalisia

Fase honetan alderantzizko ingeniaritza erabiliz tiraderaren ezaugarri teknikoetaz zein protokoloetaz jabetzen saiatuko da.

EDE	LK	H	EU
11 egun	22 ordu	Osziloskopio analogiko zein digitala, arduino mega plaka, saleae analizatzailea+softwarea eta logic sniffer softwarea	EU31

T32. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
4 egun	8 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU32

LP4. Analisia

Lan pakete honetan aplikazioak beharrezkoak dituen lengoaiak eta arkitektura definituko dira.

EU4	EDE	LK
	6 egun	12 ordu

T41. Arkitektura definitu

Aplikazioak beharrezkoa duen arkitektura definituko da.

EDE	LK	H	EU
1 egun	2 ordu	Arkitektura irudikatzeko programa bat eta dokumentatzeko programa bat(Libreoffice,word...)	EU41

T42. Gertaera fluxuak egin

Aplikazioa definituko duten gertaera fluxuak sortuko ditugu atal honetan.

EDE	LK	H	EU
2 egun	4 ordu	Gertaere fluxuak irudikatzeko programa bat(visual paradigm)	EU42

T43. Zirriborroen diseinua egin

Emaitzak aurkezteko erabiliko den web-aplikazioaren pantailen zirriborroa diseinatuko da atal honetan.

EDE	LK	H	EU
1 egun	2 ordu	Sarean zirriborroak egiteko aplikazio bat(mockups)	EU43

T44. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
2 egun	4 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU44

LP5. Diseinua

Lan pakete honetan mahaigaineko aplikazio eta web-aplikazioaren programatzen hasi baino lehenagoko estruktura orokor bat lortuko dugu.

EU5	EDE	LK
	10 egun	20 ordu

T51. Aplikazioen azken diseinua

Mahaigaineko zein web-aplikazioen azken diseinua lortuko da fase hontan.

EDE	LK	H	EU
5 egun	10 ordu	Diagramak irudikatzeko programa bat(visual paradigm)	EU51

T52. Datu basearen eskema sortu

Datu basea inplementatzeko beharrezkoa den diseinua gauzatuko da fase hontan.

EDE	LK	H	EU
3 egun	5 ordu	Datu baseak kudeatzeko programa bat(mysql workbench)	EU52

T53. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
3 egun	5 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU53

LP6. Programazio lengoaien ikasketa

Proiektuak bereganatzen dituen programazio lengoaietan sakonduko dugu lan pakete honetan.

EU6	EDE	LK
	18 egun	36 ordu

T61. Python lengoaiaren ikasketa

Python lengoaian trebetasuna hartuko da fase hontan.

EDE	LK	H	EU
5 egun	10 ordu	Internet eta apunteak	EU61

T62. Arduino ikasketa

Arduinorekin trebetasuna hartuko da fase hontan.

EDE	LK	H	EU
4 egun	8 ordu	Internet eta arduino programa probak egiteko	EU62

T63. HTML eta CSS errepasoa

HTML eta CSS lengoaietan trebetasuna hartuko da atal hontan.

EDE	LK	H	EU
4 egun	8 ordu	Internet eta klaseko apunteak	EU63

T64. PHP ikasketa

PHP lengoaiarekin trebetasuna hartuko da fase hontan.

EDE	LK	H	EU
5 egun	10 ordu	Internet eta klaseko apunteak	EU64

LP7. Proiektuaren inplementazioa

Proiektua inplementatuko dugun lan paketea izango da.

EU7	EDE	LK
	71 egun	142 ordu

T71. Aplikazioen inplementazioa

Web zein mahaigai aplikazioen kodea inplementatuko dugu atal honetan.

EDE	LK	H	EU
60 egun	120 ordu	Programazio editore bat(eclipse), web nabigatzaile bat(google chrome,firefox...) eta zerbitzari bat(apache tomcat)	EU71

T72. Datu basearen inplementazioa

Datu basearen inplementazioa gauzatuko da atal honetan.

EDE	LK	H	EU
4 egun	8 ordu	Datu baseak kudeatzeko programa bat(mysql workbench)	EU72

T73. Inplementazio arazoen konponketa

Inplementazioan zehar emandako arazoak konponduko dira atan honetan.

EDE	LK	H	EU
3 egun	6 ordu	Internet arazoen bilaketa egiteko	EU73

T74. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
4 egun	8 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU74

LP8. Probak

Gure aplikazioa ondo dabilen egiaztatzeko lan paketea izango da, probak diseinatuz eta egikarituz.

EU8	EDE	LK
	6 egun	11 ordu

T81. Proben diseinua

Aplikazioa ondo dabilen egiaztatzeko proben diseinua gauzatuko da atal honetan.

EDE	LK	H	EU
2 egun	4 ordu	Amaitutako aplikazioa	EU81

T82. Probak egikaritu

Aplikazioa ondo dabilen egiaztatzeko probak egikarituko ditugu atal honetan.

EDE	LK	H	EU
2 egun	4 ordu	Amaitutako aplikazioa, apachebench eta firebug tresnak	EU82

T83. Dokumentazioa zuzendu

Aurreko ataletan egindako lan guztia dokumentatuko dugu.

EDE	LK	H	EU
2 egun	3 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU83

LP9. Dokumentazioa

Lan pakete honetan dokumentazioa amaitu eta akatsen zuzenketa gauzatuko da.

EU9	EDE	LK
	6 egun	12 ordu

T91. Dokumentazioa zuzendu eta akatsen zuzenketa

Dokumentazioa amaitu eta azken ikutua emango diogu atal honetan.

EDE	LK	H	EU
6 egun	12 ordu	Dokumentatzeko programa bat(Libreoffice,word...)	EU91

LP10. Aurkezpena

Lan pakete honetan GAL a aurkezteko lagungarria den aurkezpena prestatuko da.

EU10	EDE	LK
	3 egun	5 ordu

T101. Aurkezpena prestatu

GAL hau aurkezteko aurkezpena prestatuko da atal honetan.

EDE	LK	H	EU
3 egun	5 ordu	Aurkezpenak prestatzeko programa bat(Libreoffice Impress,power point...)	EU101

- Sortuko diren entregableen zerrenda:

Proiektua aurrerantz doan heinean, hurrengo entregableak sortuko dira:

Kodea	Izena	Deskribapena
EU1	Proiekturako prestakuntza	Lan pakete honetan proiektua hasi baino lehenagoko ordu planifikazioa eta LDEa sortuko dugu.
T11	Denbora planifikatu	Proiektuaren denbora eta epeen planifikazioa gauzatuko da atal honetan.
T12	LDEa sortu	Proiektuaren lan dekonposaketa egitura gauzatuko da lan pakete honetan

Taula 2: Lehenengo entrega unitatea

Kodea	Izena	Deskribapena
EU2	Betekizunen bilketa	Proiektuaren betekizunak(funtzionalak eta ez funtzionalak) bilduko dituen lan paketea. Fase honetan aplikazioren atalak definituko ditugu baita ere diagrama ezberdinekin(DE, klase diagrama...)
T21	Negozio esparrua aztertu	Garatu nahi dugun antzerako aplikazioen azterketa gauzatuko da atal honetan.
T22	Domeinuaren eta erabilpen kasuen ereduak sortu	Domeinuaren eredua eta erabipen kasuak sortuko ditugu atal honetan.
T23	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

T24	Gantt-a sortu	Atal honetan lan paketeen eta beren atalen denbora planifikatzeko gantt diagrama sortuko dugu.

Taula 3: Bigarren entrega unitatea

Kodea	Izena	Deskribapena
EU3	Aurreanalisia	GAL aren inplementazio fasea hasi baino lehen, lan karga garrantzisu bat izango duen analisi edo ikerketa prozesu bat emango da.
T31	Produktuaren aurreanalisia	Fase honetan alderantzizko ingeniaritza erabiliz tiraderaren ezaugarri teknikoetaz zein protokoloetaz jabetzen saiatuko da.
T32	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

Taula 4: Hirugarren entrega unitatea

Kodea	Izena	Deskribapena
EU4	Analisia	Lan pakete honetan aplikazioak beharrezkoak dituen lengoaiak eta arkitektura definituko dira.
T41	Arkitektura definitu	Aplikazioak beharrezkoa duen arkitektura definituko da.
T42	Gertaera fluxuak egin	Aplikazioa definituko duten gertaera fluxuak sortuko ditugu atal honetan.
T43	Zirriborroen diseinua egin	Emaitzak aurkezteko erabiliko den web-aplikazioaren pantailen zirriborroa diseinatuko da atal honetan.
T44	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

Taula 5: Laugarren entrega unitatea

Kodea	Izena	Deskribapena
EU5	Diseinua	Lan pakete honetan mahaigaineko aplikazio eta web-aplikazioaren programatzen hasi baino lehenagoko estruktura orokor bat lortuko dugu.
T51	Aplikazioen azken diseinua	Mahaigaineko zein web-aplikazioen azken diseinua lortuko da fase hontan.
T52	Datu basearen eskema sortu	Datu basea inplementatzeko beharrezkoa den diseinua gauzatuko da fase hontan.
T53	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

Taula 6: Bostgarren entrega unitatea

Kodea	Izena	Deskribapena
EU6	Programazio lengoaien ikasketa	Proiektuak bereganatzen dituen programazio lengoaietan sakonduko dugu lan pakete honetan.
T61	Python lengoaiaren ikasketa	Python lengoaian trebetasuna hartuko da fase hontan.
T62	Arduino ikasketa	Arduinorekin trebetasuna hartuko da fase hontan.
T63	HTML eta CSS errepasoa	HTML eta CSS lengoaietan trebetasuna hartuko da atal hontan.
T64	PHP ikasketa	PHP lengoaiarekin trebetasuna hartuko da fase hontan.

Taula 7: Seigarren entrega unitatea

Kodea	Izena	Deskribapena
EU7	Proiektuaren inplementazioa	Proiektua inplementatuko dugun lan paketea izango da.
T71	Aplikazioen inplementazioa	Web zein mahaigai aplikazioen kodea inplementatuko dugu atal honetan.
T72	Datu basearen inplementazioa	Datu basearen inplementazioa gauzatuko da atal honetan.
T73	Inplementazio arazoen konponketa	Inplementazioan zehar emandako arazoak konponduko dira atan honetan.
T74	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

Taula 8: Zazpigarren entrega unitatea

Kodea	Izena	Deskribapena
EU8	Probak	Gure aplikazioa ondo dabilen egiaztatzeko lan paketea izango da, probak diseinatuz eta egikarituz.
T81	Proben diseinua	Aplikazioa ondo dabilen egiaztatzeko proben diseinua gauzatuko da atal honetan.
T82	Probak egikaritu	Aplikazioa ondo dabilen egiaztatzeko probak egikarituko ditugu atal honetan.
T83	Dokumentazioa zuzendu	Aurreko ataletan egindako lan guztia dokumentatuko dugu.

Taula 9: Zortzigarren entrega unitatea

Kodea	Izena	Deskribapena

EU9	Dokumentazioa	Lan pakete honetan dokumentazioa amaitu eta akatsen zuzenketa gauzatuko da.
T91	Dokumentazioa zuzendu eta akatsen zuzenketa	Dokumentazioa amaitu eta azken ikutua emango diogu atal honetan.

Taula 10: Bederatzigarren entrega unitatea

Kodea	Izena	Deskribapena
EU10	Aurkezpena	Lan pakete honetan GAL a aurkezteko lagungarria den aurkezpena prestatuko da.
T101	Aurkezpena prestatu	GAL hau aurkezteko aurkezpena prestatuko da atal honetan.

Taula 11: Hamargarren entrega unitatea

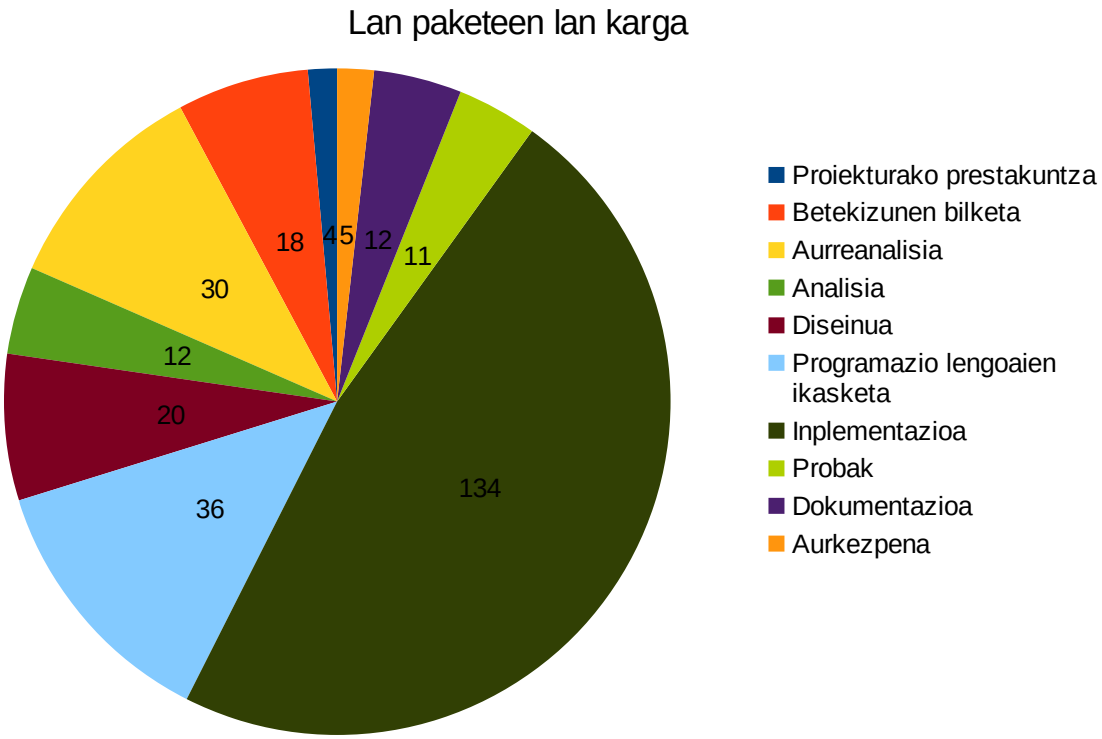
2.4 Denbora planifikazioa

2.4.1 Planifikatutako egutegia

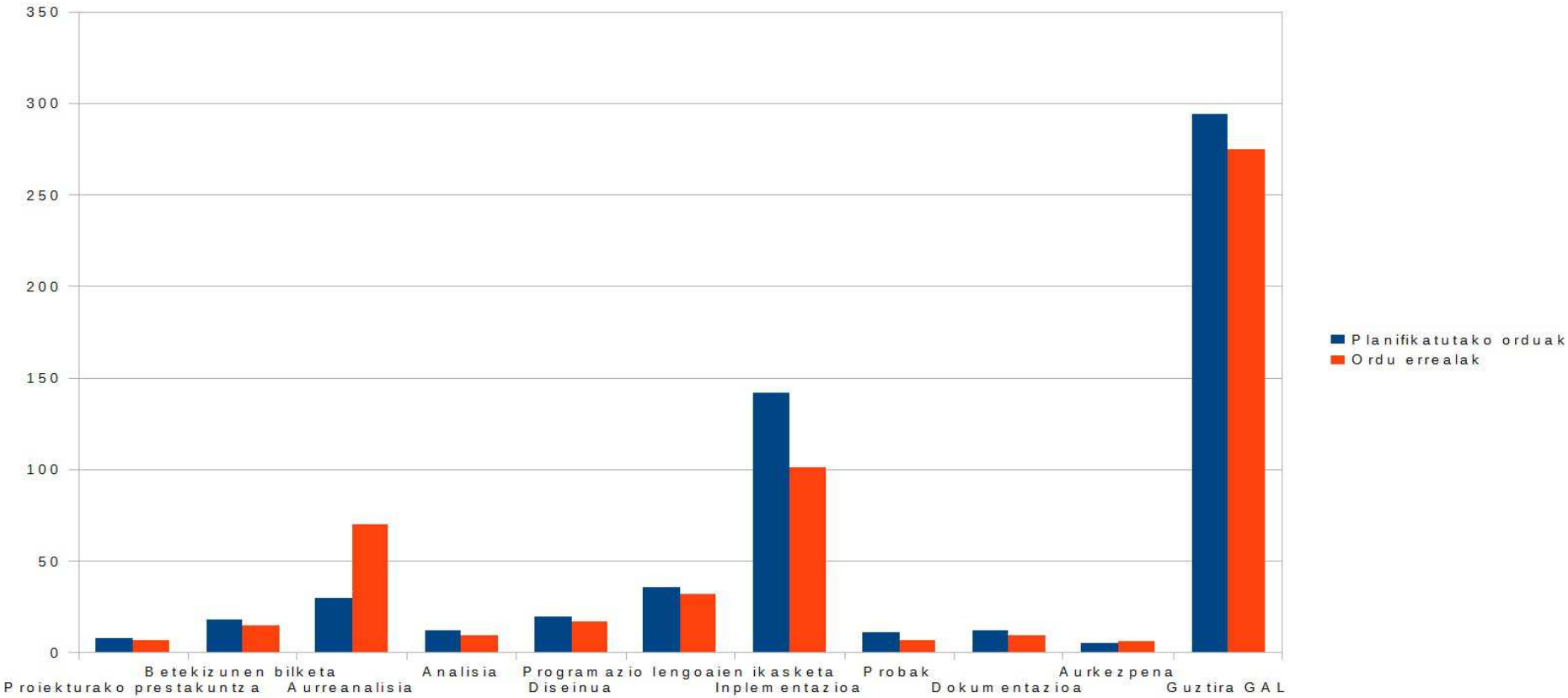
Proiekturako prestakuntza	**2014/11/11-2014/11/13**
Betekizunen bilketa	**2014/11/14-2014/11/26**
Aurreanalisia	**2014/11/27-2014/12/17**
Analisia	**2014/12/18-2015/1/13**
Diseinua	**2015/1/14-2015/1/28**
Programazio lengoaien ikasketa	**2015/1/29-2015/2/23**
Inplementazioa	**2015/2/24-2015/6/2**
Probak	**2015/6/3-2015/6/9**
Dokumentazioa	**2015/6/10-2015/6/17**
Aurkezpena	**2015/6/18-2015/6/22**

Taula 12: Planifikatutako egutegia

2.4.2 Orduen planifikazioa

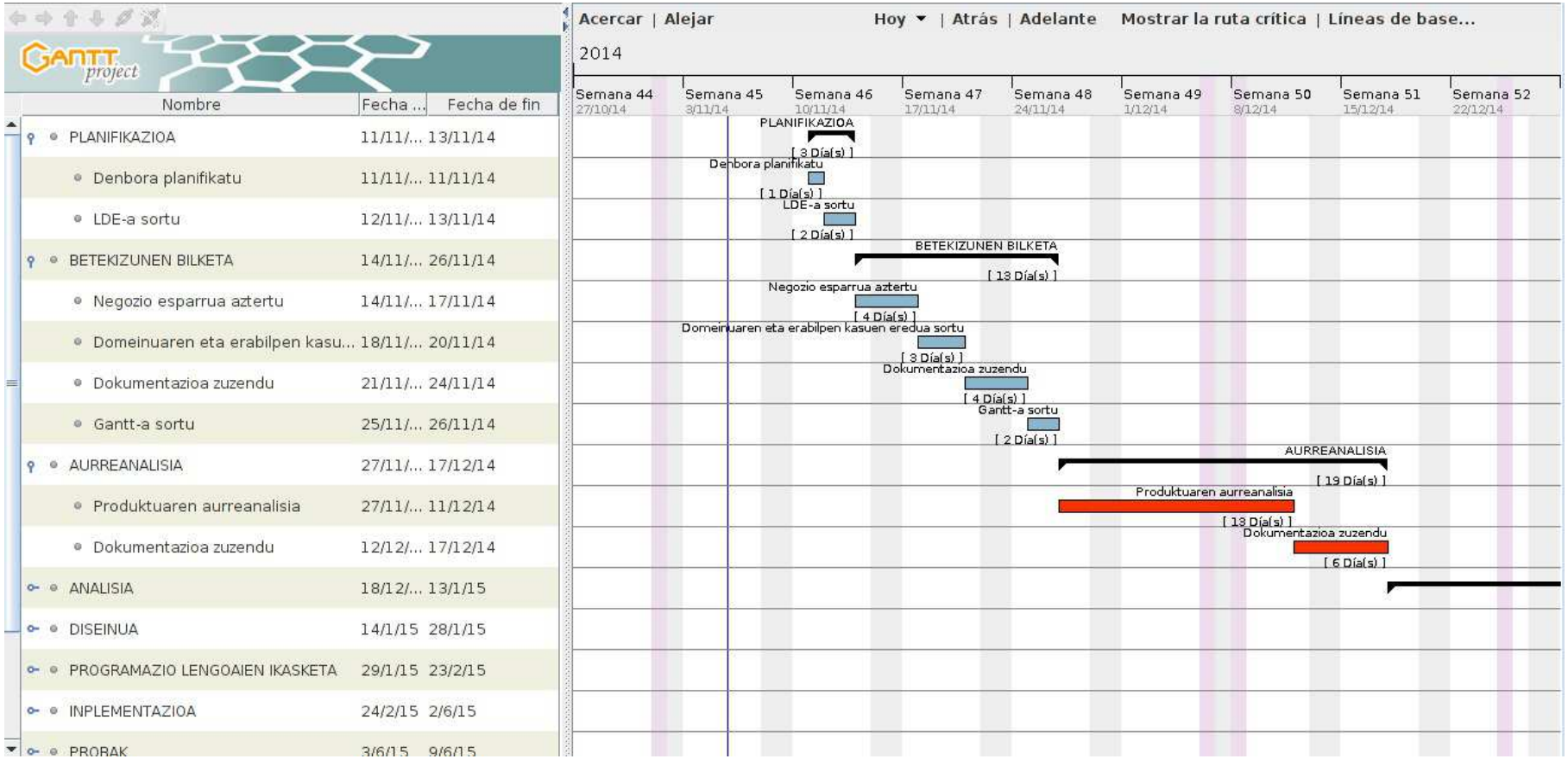


Irudia 5: Ordu planifikazioa

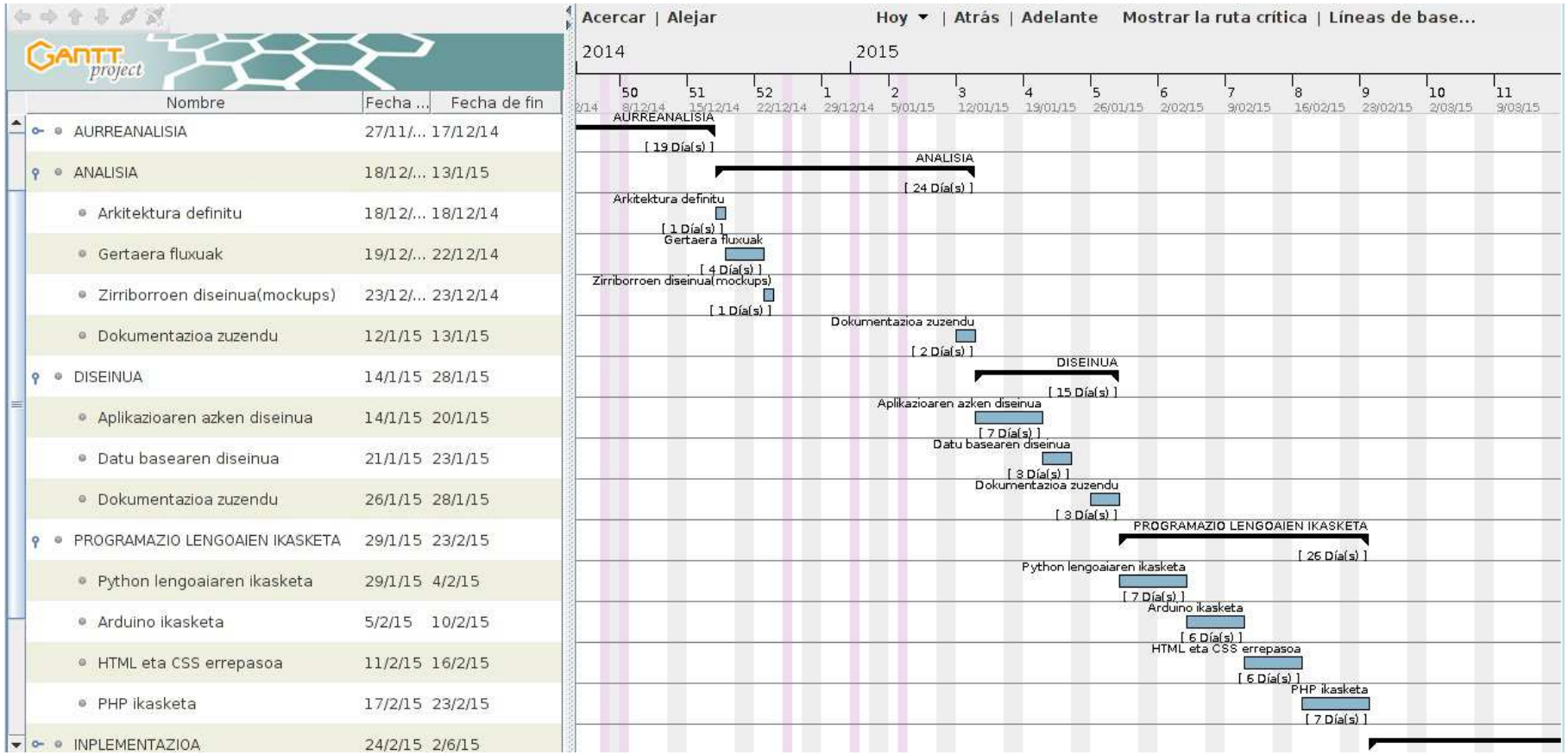


Irudia 6: Estimatutako orduak ordu errealekin alderatuta

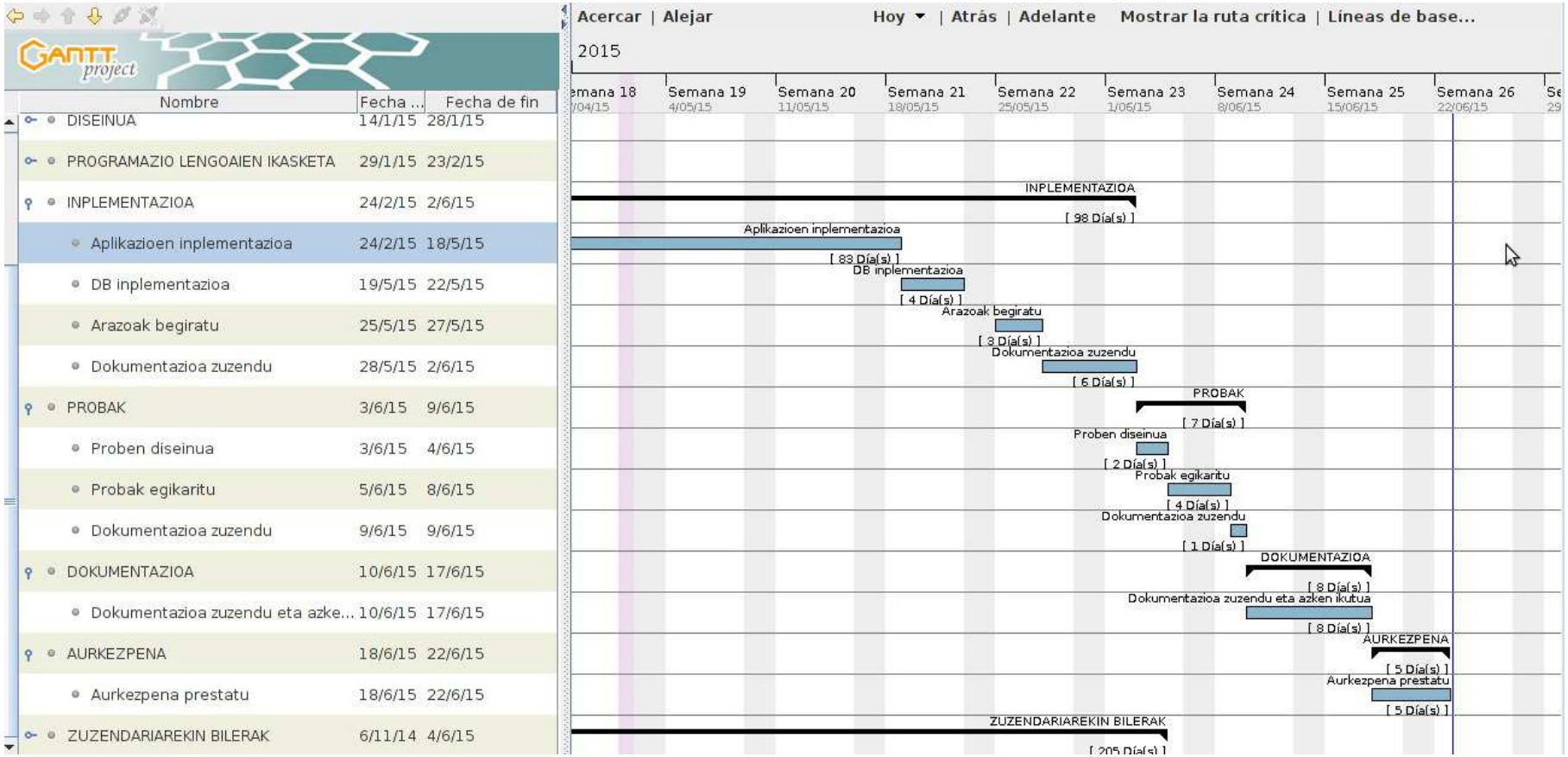
2.4.3 Gantt diagrama



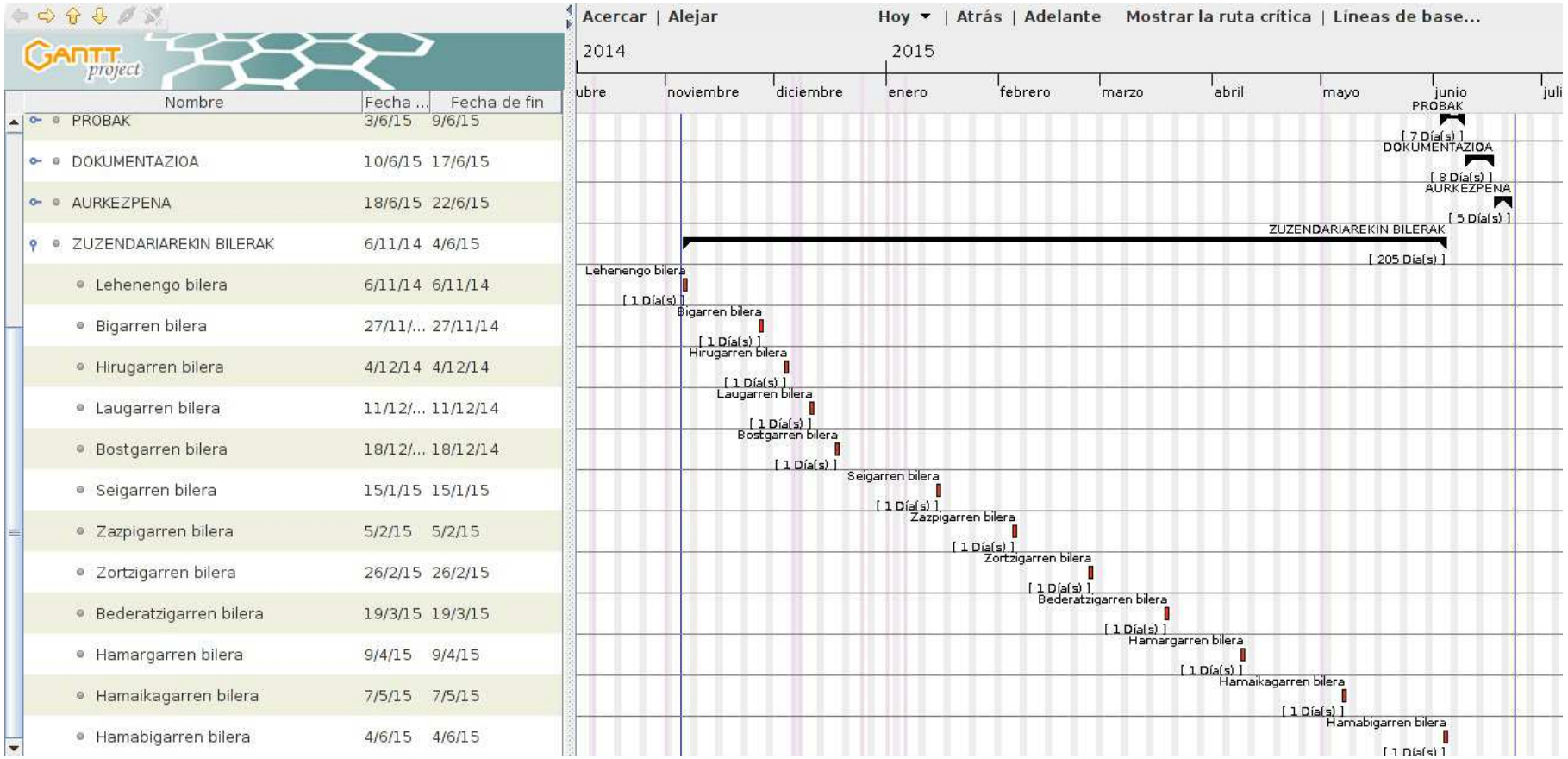
Irudia 7: Gantt diagrama lehenengo zatia



Irudia 8: Gantt diagrama bigarren zatia



Irudia 9: Gantt diagrama hirugarren zatia



Irudia 10: Gantt diagrama laugarren zatia

2.5 Arriskuen analisia

Proiektua bat gauzatzen ari garenean hainbat arrisku suera daietezke. Arrisku hauek ezagututa, arrisku hauek saiheste aldera plan bat izatea oso garrantzitsua da, honela kalteak txiagotu ditzazkegu.

Hona hemen arriskuen zerrenda bat hartu beharreko neurriekin:

Arriskua	Proiektuko informazioa galtzea
Ondorioa	Informazio guztia edo zati bat galtzea
Hartu beharreko neurriak	Galdutako atala berriz egin
Zelan saihestu	Egunero segurtasun kopia bat egin
Suertatzeko probabilitatea	Txikia

Arriskua	Tiraderak ongi ez funtzionatzea
Ondorioa	Programak ez du era egokian funtzionatuko
Hartu beharreko neurriak	Tiraderan ongi funtzionatzen ez duen atala aldatu
Zelan saihestu	Materiala era egokian zaindu
Suertatzeko probabilitatea	Ertaina

Arriskua	Proiektuko partaideen osasun arazoak
Ondorioa	Atzerapenak suerta daitezke
Hartu beharreko neurriak	Sartu beharreko ordu kopurua handitu
Zelan saihestu	Ezin da saihestu
Suertatzeko probabilitatea	Ertaina

Arriskua	Tiraderarekin konexio kableak edo plaka matxuratzea
Ondorioa	Programak ez du era egokian funtzionatuko
Hartu beharreko neurriak	Kablea edo plaka berriztu
Zelan saihestu	Materiala era egokian zaindu
Suertatzeko probabilitatea	Ertaina

Arriskua	Proiektua egiteko beharrezkoa den software tresnaren batek gaizki funtzionatzea
Ondorioa	Atzerapenak suerta daitezke
Hartu beharreko neurriak	Programa berkonfiguratu edo bertsioa aldatu
Zelan saihestu	Programak berriztuta mantendu bertsio egokiekin
Suertatzeko probabilitatea	Altua

Taula 13: Arriskuen analisia

2.6 Aurrekontua

Jarraian “tiradera automatizatu baten irekitze eta ixteen erregistroa alderantzizko ingeniaritza erabiliz” proiektua aurrera ateratzeko inbertitu behar den diruaren analisia egingo da. Kostuak zehazteko orduan eskulan eta material kostuen banaketa egin da.

2.6.1 Eskulanaren prezio unitarioa

Eskulanaren prezioaren kalkulua aurrera eramateko, 2015 urtean espainian informatika ingeniari baten batez besteko mozkin garbiak begiratuko dira. Mozkin hauek **16800€ urtekoak** izan dira.

Langileak enpresarentzat duen benetako kostua kalkulatzeko, gizarte segurantzarako kotizazioa[1] gehitu behar zaio kopuru honi. Portzentai hau soldataren **%28,3** a da(23,6 enpresak eta 4,7 langileak), eurotan **4754,4€** dena. Beraz langileak urte batean enpresarentzat duen kostua **21554,4 €** koa da.

Kontutan edukita langile batek urtean[2] 1800 ordu lan egiten dituela, 21554,4€/1800 ordu= **12€/ordu** ko eskulan kostua izango du proiektu honek.

Izena	Enpresa	Erantzukizuna	Kostua
Oskar Casquero Oyarzabal	EUITI Bilbo	Proiektu Zuzendaria	50€/ordu
Pablo Carmona Luque	EUITI Bilbo	Ingeniaria	12 €/ordu

Taula 14: Proiektuko partaideak

[1] Iturria: http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm
[2] Iturria: https://es.wikipedia.org/wiki/Jornada_de_trabajo_en_Espa%C3%B1ª

2.6.2 Baliabide materialen prezio unitarioa

- Material amortizagarria:

Jarraian proiektua garatzeko beharrezkoak diren material amortizagarrien zerrenda agertuko da.

Deskribapena	Kostua	Amortizazioa	Kostu unitarioa
PC eramangarria	800 €	5 urte	13,33 €/hilabete

GUZTIRA	8 hilabete x 13,33€= **106,64€**

Deskribapena	Kostua	Amortizazioa	Kostu unitarioa
Saleae logic 4	200 €	2 urte	8,33 €/hilabete

GUZTIRA	1 hilabete x 8,33€= **8,33€**

Deskribapena	Kostua	Amortizazioa	Kostu unitarioa
Osziloskopio analogikoa	500 €	5 urte	8,33 €/hilabete

GUZTIRA	1 hilabete x 8,33€= **8,33€**

Deskribapena	Kostua	Amortizazioa	Kostu unitarioa
Osziloskopio digitala	800 €	5 urte	13,33 €/hilabete

GUZTIRA	1 hilabete x 13,33€= **13,33€**

- Material suntsikorra:

Jarraian proiektua garatzeko beharrezkoak diren material suntsigarrien zerrenda agertuko da.

Deskribapena	Kostua
Dokumentazioa eta fotokopiak	20 €
Rs485 serie plaka bihurgailua	20€
Konexio kableak	2€
Arduino mega	50€

Taula 15: Material suntsikorren kostuak

2.6.3 Beste gastu batzuk

Jarraian aparteko gastuen zerrenda agertuko da.

Deskribapena	Kostua
Garraioa	200 €
Internet	20€
Argia	20€

Taula 16: Aparteko gastuak

2.6.3 Giza baliabideen aurrekontua

Jarraian giza baliabideen aurrekontua burutuko da lan pakete bakoitzaren iraupena kontutan izanda eta orduko soldata aurreko atalean egindako kalkuluen arabera estimatuta.

Lan paketea	Orduak	Kostua
LP1	8 ordu	96 €
LP2	18 ordu	216€
LP3	30 ordu	840€
LP4	12 ordu	144€
LP5	20 ordu	240 €
LP6	36 ordu	432 €
LP7	142 ordu	1704€
LP8	11 ordu	132 €
LP9	12 ordu	144 €
LP10	5 ordu	60 €
Zuzendariarekin bilerak	24 ordu	1200€
GUZTIRA	**318 ordu**	**4728 €**

Taula 17: Giza baliabideen aurrekontua

2.6.4 Aurrekontuaren laburpen finala

Azkenik proiektuaren aurrekontu finala aurkeztuko da, non materialen kostuak, aparteko kostuak eta giza baliabide kostuak kontutan hartuko diren.

Deskribapena	Kostua
Material amortizagarria	136,63 €
Material suntsikorra	92€
Beste gastu batzuk	240€
Giza baliabideak	4728 €
AURREKONTU TOTALA	**5196,63 €**

Taula 18: Aurrekontu totala

Planifikatutakoaren arabera proiektu hau garatzeko beharrezkoa den inbertsioa lau mila bostehun eta laurogehita hamasei euro eta hirurogeita hiru zentimokoa da.

3. Aurrekarien analisia

Ondoren aplikazioaren aurrekarien edo antzeko aplikazioen analisia burutuko da. Atal honen helburua gure aplikazioarentzako ideiak ateratzea, gure aplikazioak izan ditzazken akatsak detektatzea eta gure kasuan, jarraitu beharreko prozedimentua erakutsiko digu aurreanalisi on batek. Aurrekarien analisi egoki bat eginda, antzeko aplikazioek ez dituzten ezaugarrietaz ohartuko gara eta honela hauena baino aplikazio sendoago bat lortuko dugu.

Esan beharra dago alderantzizko ingeniaritza erabiliz ebazten diren problementzako ez dagoela metodologia zehatz edo zientifiko bat. GAL honen aurreanalisia alderantzizko ingeniaritza erabiliz burutu da, aurrekarien analisia burutu ondoren behatutako metodoekin. Ikusitakoaren arabera, esan beharra dago mota honetako proiektu guztiek pausu eman beharreko pausu edo urrats batzuk amankomunean dituztela.

Hauek litzateke eman beharreko urratsak:

1. **Produktuaren analisia:**

Lehenengo pausua esku artean daukagun produktuaren behaketa da. Behaketa honetan produktuaren osagaien analisia egingo da. Osagaien artean erreparatuko ditugun lehenengo gauzak busak eta honen pinak izango dira, pinen artean proiektua garatze aldera interesagarriak direnak hartuko ditugu, hau da, datuak maneiatzen dituztenak.

Behin busa eta honen pinak behatuta izanda, mikrokontrolagailura edo unitate zentralera joko dugu. Honen helburua mikrokontrolagailu mota zein den jakitea da, mota jakitea lagungarria izan daiteke honek erabiltzen dituen protokoloetaz ohartzeko.

Irudia 11:
Produktuaren analisia

2. **Bus eta pinen analisia:**

Hurrengo pausua busaren datu pinetatik doan informazioaren analisia litzateke. Horretarako lehenik busaren s/i etatik ikusten diren datuak begiratuko dira, edo beste era batera busean jarraitutasuna dagoen ala ez konprobatuko da. Jarraitutasuna egoteak bus berdinak sistema osoa zeharkatzen duela esan nahi du, hau da, pin ezberdinetan datu berdina ikusiko dela.

Irudia 12: Bus eta pinen analisia

3. **Neurrien neurketa**

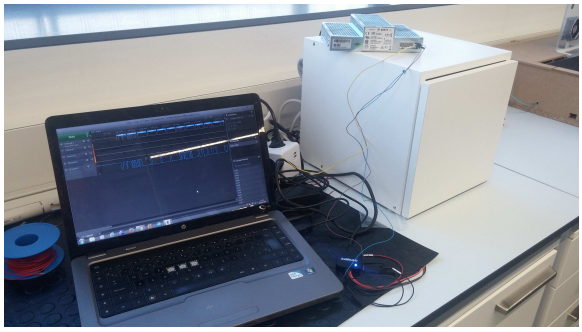
Honekin batera testerraren laguntzaz pin bakoitzaren boltaien neurketa egingo da. Neurketa hau egitearen arrazoia, aurrerantzean plakekin izan ditzazkegun bateraezingarritasunak ekiditea da.

Irudia 13: Testerra

4. **Datuen analisia:**

Hurrengo pausua datua eta honen uhina grafikoki ikustea da. Horretarako **analizatzaile logiko** baten beharra aurreikusten da. Gure proiektuaren kasuan, hasiera baten ez dago analizadore logikorik, beraz osziloskopioen laguntzaz baliatuko da. Osziloskopio digitala analizadore logiko moduan erabiltzen da datuak bistaratzeko eta honen denbora ez errealeko sesio kaptura batzuk egiteko. Esan beharra dago emaitza ez dela espero dena, osziloskopio digitalaren memoria urriak sesio oso txikien kaptura bakarrik ahalbidetzen baitu. Aurrekarien analisia egiterakoan ikusten da datuak ordenagailuan edukitzeko beharra dagoela. Honen bilaketan, **arduino mega plaka eta logic sniffer-a** erabiltzen dira datua ordenagailutik ikusteko asmoarekin. Bigarren kasu honetan ere memoria arazoa sortzen da, arduinok laginak bere baitan gordetzen baititu.

Aurreko arazoak ikusita eta aurrekarietan oinarrituta[3], **saleae analizatzaile logikoa** erostea erabakitzen da. Honen abantaila nahi bezainbeste datu hartu ditzazkegula da, laginak ordenagailuaren memorian gordetzen baititu. Kasu hauetarako saleae tresna ezinhobea da, sesio luzeak gorde ditzazkegulako, sesio hauetan barrena datuen esanahia edo patroi baten bilaketa gauzatzeko.



Irudia 14: Saleaerekin bit jario saio baten kaptura

[3]Aurrekarien iturria:http://www.deepdarc.com/2010/11/27/hacking-christmas-lights/ eta http://soloelectronicos.com/2015/05/15/control-domestico-mediante-un-smartphone-usando-ingenieria-inversa/

4. Betekizunen bilketa

Betekizunen bilketa analisiaren aurreko fasea da. Betekizunek produktuak eduki beharreko forma edo funtzionalitateak definituko dituzte, hau da, inplementatzen hasi baino lehenago oinarri bat edukitzeko, erabiltzaileak egin ahal izango dituen ekintzak edo akzioak definituko ditugu. Bi betekizun mota daude:

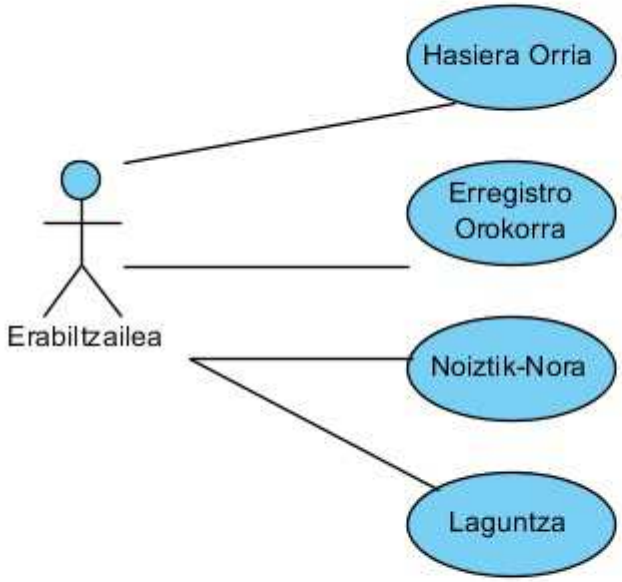
- **Betekizun funtzionalak:** produktuak egin beharreko funtzioak definitzen dituzte.
- **Betekizun ez-funtzionalak:** produktuaren betekizun ez funtzionalak definitzen dituzte, hala nola: erabiltzeko erraztasuna, abiadura, ulerterraza...

Aplikazioaren helburua tiraderaren irekitze eta ixtearen denbora erregistroa gauzatzea da, beraz proiektu honen erabiltzaile nagusiak tiraderaren erabiltzaileak izango dira. Erabiltzaileak erabilpen kasuen eredu bat izango du, non aplikazioan egin ahal izango dituen ekintza guztiak definituko diren.

Azkenik aplikazioak definituko dituzten domeinu diagramak errepresentatuko dira.

4.1 Erabilpen kasuen eredua

Jarraian web-aplikazioaren erabilpen kasuen eredua definituko dugu, hau da, erabiltzaileak egin ahal izango dituen ekintzen edo akzioen definizioa emango da. Erabilpen kasuen eredua sarrera gisa erabiliko da inplementazioa hastean.



Irudia 15: Erabilpen kasuen eredua

Aktorea:

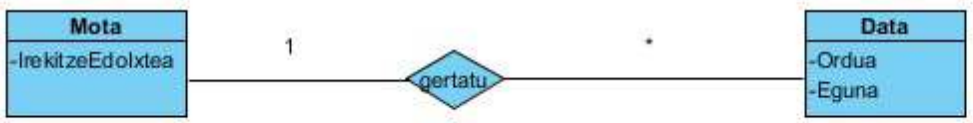
- **Erabiltzailea:** Web-aplikazioa erabiltzea duen erabiltzailea izango da. Erabiltzailea hau tiraderaren erabiltzailea edo tiraderaren erregistroa eta kontrola kontsultatu nahi duen pertsona izango da.

Erabilpen kasuak:

- **Hasiera orria:** Atal hau web-aplikazioaren ongi etorri orria izango da.
- **Erregistro orokorra:** Atal honetan erabiltzaileak tiraderaren irekitze eta ixteen erregistro orokorra edo denbora guztietako erregistroa kontsultatu ahal izango du. Erregistro honetan mugimendu zenbakia, data eta ordua agertuko dira.
- **Noiztik-Nora:** Atal honetan erabiltzaileak tiraderaren irekitze eta ixteen erregistro partzial bat kontsultatu ahal izango du hasiera data eta bukatze data bat sartuta. Erregistro honetan mugimendu zenbakia, data eta ordua agertuko dira.
- **Laguntza:** Atal honetan erabiltzaileari tiraderaren irekitze eta ixtearen denbora errealeko erregistroa egiteko eman behar diren pausuak agertuko dira. Pausu hauek jar programaren exekuzio instrukzioak eta tiradera-ordenagailu konexioa izango dira.

4.2 Domeinuaren eredua

Jarraian web-aplikazioaren domeinuaren eredua aurkeztuko da. Domeinu honek aplikazioa definituko du eta laguntza handikoa izango da inplementazioa hasi baino lehen.



Irudia 16: Web-aplikazioaren domeinuaren eredua

Entitateak:

- **Mota:** Tiraderak egin duen mugimendua desberdinduko da, zabaldu edo itxi izango dira balio posibleak.
- **Data:** Erregistratutako mugimendu bakoitzak berarekin lotuta ekarriko ditu mugimendua emandako ordua eta eguna.

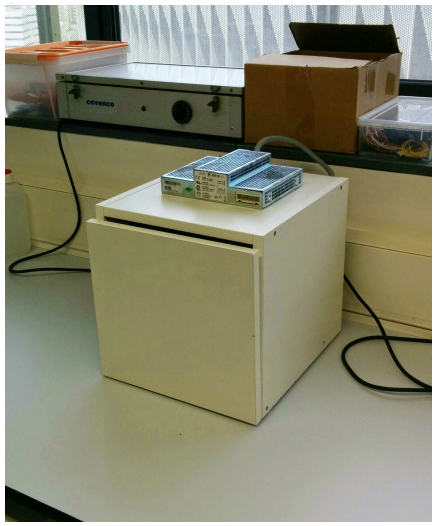
5. Prozesua

GAL aren inplementazio fasea hasi baino lehen, lan karga garrantzisu bat izan duen analisi edo ikerketa prozesu bat eman da. Esan bezala, tiraderaren ezaugarri teknikoak ez dira ezagutzen, hau da, ez dago hardware datasheet-ik edo software API-rik(tiradera “kutxa-beltz” bat da), beraz ezin da bit jario hori zuzenean dekodifikatu. Ondoren busetik doan bit jarioa dekodifikatzeko eman den aurreanalisia aurkezten da.

5.1 Lehenengo pausuak

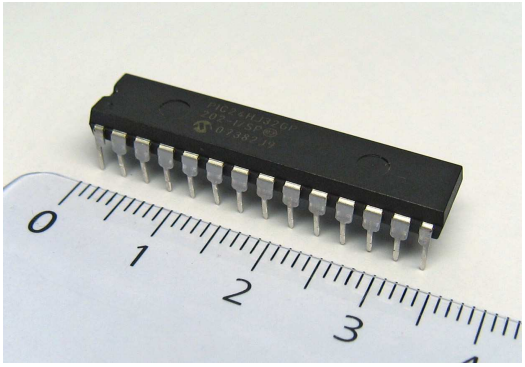
Emandako lehenengo pausua produktuaren gainbegiraketa edo analisia da. Produktua bi zati zagusietan bereizten da:

- **Tiradera:** Unitate zentraletik aginduak jaso eta motoreen bitartez zabaldu edo itxiko den**.**
- **Unitate zentrala edo burmuina:** Hemendik aurrera begiratuko dugun atala. Hemendik ateratzen dira tiraderara doazen aginduak eta atal honetan gertatzen da dekodifikatu beharreko bit jarioa.



Irudia 17: Produktuaren analisia

Esan bezala, unitate zentralean oinarrituko gara. Unitate zentrala zabaldu eta gero PIC motatako mikrokontrolagailua duela ikusten da, zehazki **pic 18f66j15 -i/pt** mikrokontrolagailua.



Irudia 18: Pic motako mikrokontrolagailua

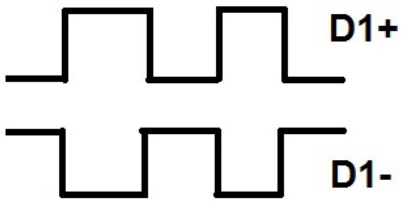
Unitate zentralaren portutik bidaltzen diren seinaleen azterketa beharrezkoa dela ikusten da. Unitate zentralak hainbat portu ditu: V+,+10,GND,+8V/1,GND1,D1-,D1+,GND1.

Irudia 19: Tiraderaren seinale ezberdinak

Osziloskopio analogiakoaren bidez D1+ eta D1- seinaleen azterketa egin ondoren, **busetik bit jarioa etengabe dagoela** ikusten da. Analogikoki, zarataren eraginez ez da askorik ikusten, beraz seinalea digitalean ikusteko beharra dagoela ikusten da.

Osziloskopio digitalean D1+ eta D1- seinaleen 100.000 lagin aztertu ondoren hainbat ondorio ateratzen dira:

- Aurretik esan bezala, nahiz eta tiradera geldirik egon bit jarioa ez da inoiz gelditzen, busetik galdetzen ari da denbora osoan zehar.
- D1+ eta D1- seinaleak diferentzialak dira.



Irudia 20: Bi seinale diferentzial

Osziloskopioarekin kaptura sesio hauek egin ostean, seinalea ordenagailuan edukitzeko beharra ikusten da. Honek analisi erosoago bat ekarriko luke, bit jarioa denbora errealean jasotzeko aukerarekin batera.

5.2 Denbora errealeko bit jario kaptura

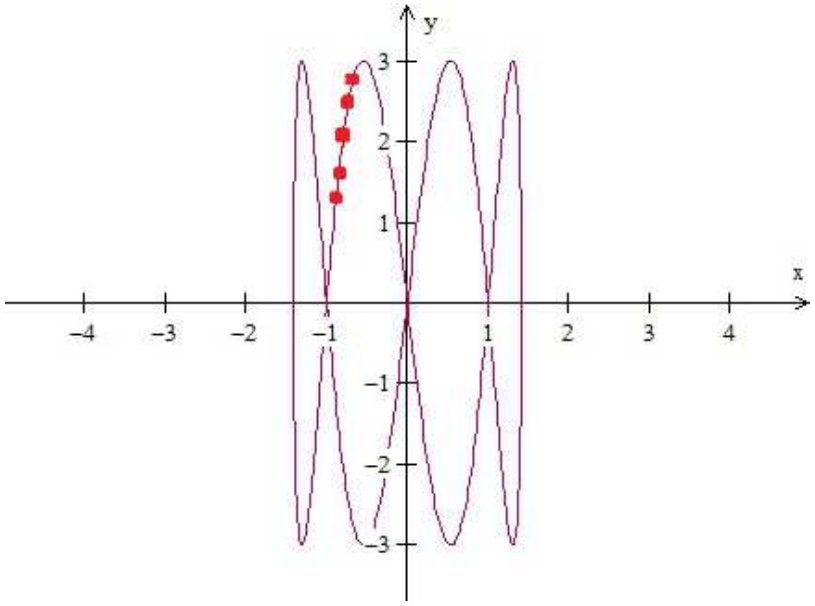
Ordenagailuan seinalea ikusi aurretik seinaleen azterketa egiten da eta boltai neurketa testerraren bidez, arduino plakarekin izan ahal dituen arazoak ekiditeko. Ondorengo ondorioak ateratzen dira:

- Konektore guztiak bus berdinekoak dira.
- Bi hariz(D1+,D1-) osotutako bus bakarra dago, zeinera kontroladorea eta motorrak konektatuta dauden.
- D1+ eta D1- seinaleak diferentzialak eta positiboak(D1+ 3.5V eta D1- 1.35V) dira, hau da, 5V baino txikiagoak.

Eman beharreko hurrengo pausua bit jarioa edo seinalea ordenagailutik denbora errealean kapturatzea da. Horretarako, hiru osagai behar ditugu:

1. **Arduino mega** plaka analizatzaile logiko moduan erabiltzeko.
2. **Logic Sniffer** programa ordenadorean.
3. Bit jarioa jasotzen duen **arduino uno** plaka.

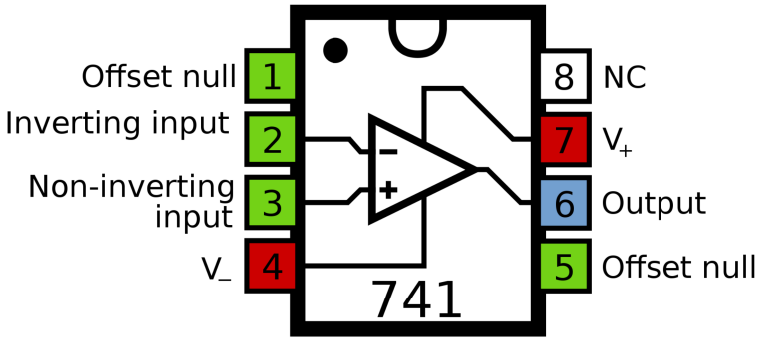
Saiakera honetan ez da datu argirik jasotzen, laginketa-maiztasun ezegoki baten ondorioz. Laginketa maiztasun altuan egiten bada eta uhin zabalera altua bada, datu galera egongo da. Kontrako kasuan, laginketa maiztasuna txikia bada, datu galera egongo da baita ere. Arazo hau ekiditeko, laginketa maiztasuna **Nyquist-Shannon**[4] teoremaren arabera ezarriko da.



Irudia 21: Gaizki egindako laginketa adibidea

Behin laginketa maiztasun egokia definituta D1+ eta D1- portuetatik serie bidez laginak era egokian jasotzen dira 4MHz ra, seinalea 1MHzkoa izanda. Aurretik esan dugun moduan seinale diferentzialak dira, beraz seinale bakarra lortzeko asmoarekin bi seinaleen kenketa planteatzen da **741 integratua** erabiliz. Lortutako seinalea arduinorekin bateragarria izateko asmoarekin seinalea 0-5V artean egokitzea pentsatzen da, **fotoigorle eta fotorrezeptore integratuaren** bidez.

[4]Nyquist-Shannon iturria: https://es.wikipedia.org/wiki/Teorema_de_muestreo_de_Nyquist-Shannon



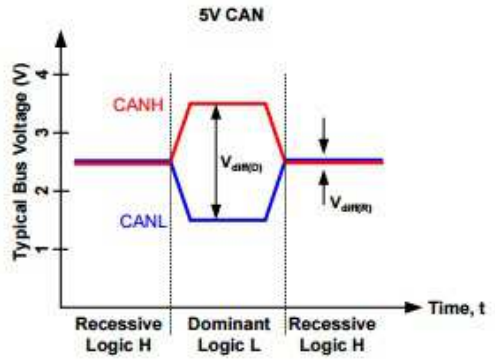
Irudia 22: 741 integratua

5.3 CAN protokoloa eta azken pausuak

Hurrengo pausuak tiraderaren unitate zentralak erabiltzen duen komunikazio protokoloa aurkitzea litzateke.

Hurrengo dokumentuari erreparatuta eta hainbat ebindetzietan oinarrituta, tiraderak **CAN comunikazio protokoloa**[5] 5V tara erabiltzen duela aurreikusten da. Hona hemen ebidentziak:

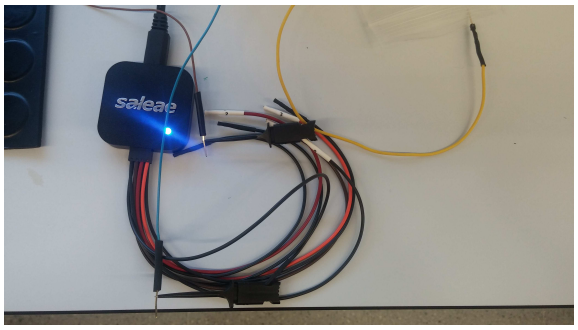
1. Kontrolagailuaren barnean PIC bat aurkitzen da.
2. Bi seinale ditugu.
3. Seinaleak diferentzialak dira.



Irudia 23: CAN protokoloa 5V-ra

[5] CAN protokoloa iturria: http://www.ti.com/lit/an/slla337/slla337.pdf

Eman beharreko hurrengo pausua CAN mezuen bilaketa litzake. Horretarako saleae analizatzailea erabiliko da tresna moduan. Arduinorekin alderatuz lan egiteko erraztasuna ematen digu, nahi bezain beste lagin hartzeko aukera ematen, laginak ordenadorearen memorian gordetzen bait ditu(arduinok bere memoria propioan).



Irudia 24: Saleae analizatzailea

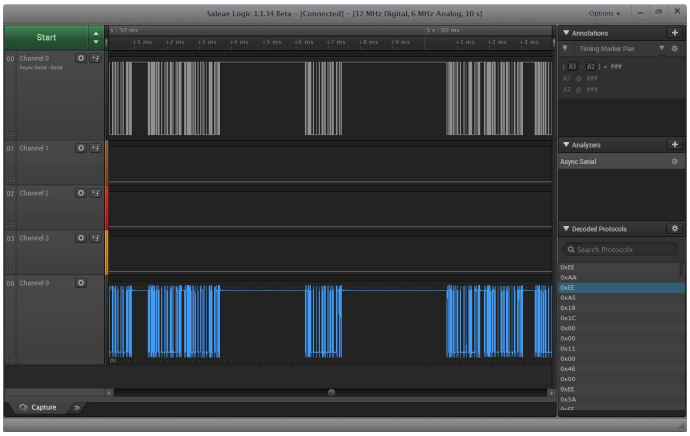
Saleae-ari bit jarioa edo seinaleetako informazioa pasata, ez da CAN protokoloko mezurik aurkitzen. Protokoloaren bilaketarekin aurrera jarraitzeko asmoarekin mikrokontrolagailura jotzen da eta bere datasheet-an[6] bateragarriak diren protokoloen kontsulta egiten da. Hona hemen protokolo horiek:

- UART
- I2C
- SPI

Horretaz gain, D1+ eta D1- pin ak LT 485 integratu batetik irtetzen direla ondorioztatzen da, zeinek RS-485 protokoloa kudeatzen du. Protokolo honek seinale diferentzialak erabiltzen ditu.

[6] Datasheet iturria: http://www.microchip.com/wwwproducts/Devices.aspx?product=PIC18F66J15

Ondorioz, hurrengo bit jario kaptura saioan serie protokoloko mezuen bilaketara jotzen da. Saio honetan serie mezu ereduak aurkitzen dira, beraz saleaearen laguntzaz tiraderaren hainbat fase ezberdinen 10 segundutako saioak gordetzen dira csv etan. Fase hauek tiradera abiatze, pausagune eta zabaldu-ixtekoak dira.



Irudia 25: Serie mezuen bilaketa

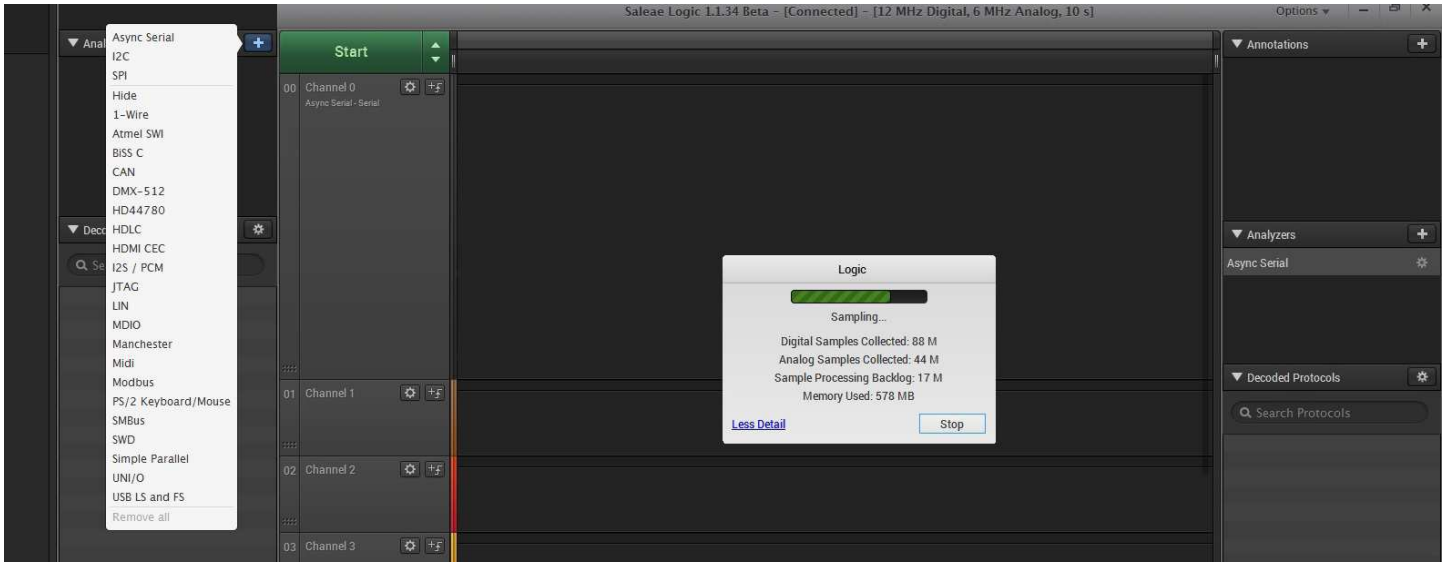
5.4 Parser eta analizer

Saleae analizadore logikoa erabiliz, kajoiaren busetik datuak irakurri ziren 1 minutuko 6 saio eginez. Saio horietan zehar, kajoiko busaren pausaguneko egoeran zein irekitze eta izte eragiketak burutzean busean azaldutako bit jarioak grabatu ziren. Bit jario horretan ez zen inolako eredu edo patroirik antzeman zuzeneko behaketa bat eginez, pausagune zein irekitze eta izteen aurretik edo ostean aldizkako hitz jakin bat izan ezik. Hitz hori ardatz bazela hartuta, beraren inguruan gertatzen zena aztertu zen. Azterketa hau egiteko programa bat ad-hoc garatu zen. Programa hau erabilita, hurrengoa ikusi zen: pausagunean, hitz jakin horren bi agerpenen tartean azaltzen den bit jarioak luzeera jakin bat du; aldiz, irekitze eta izteak burutzen ziranean, hitz jakin horren bi agerpenen tartean azaltzen den bit jarioaren luzeera pausagunean azaltzen dena baino luzeagoa da. Ondoren, prozesu honen xehetasunak azaltzen dira.

- Datuen grabazioa:

Datuen grabazioa Saleae[7] analizadore logikoa erabiliz egin zen. Barne memoria propioa duten beste analizadore logiko batzuren aurrean, Saleaek saioak grabatzeko bera konektatzen deneko ordenagailuaren memoria erabiltzen du. Proiektu honetan ezaugarri hau erabakiorra da, kajoiaren busetik denbora osoan zehar bit jarioa bait dago, eta horrek datu asko grabatu behar izatea suposatzen du. Adibidez, 1 minutuko saioa grabatzeko, Saleaek ~5GB-eko espazioa darabil.

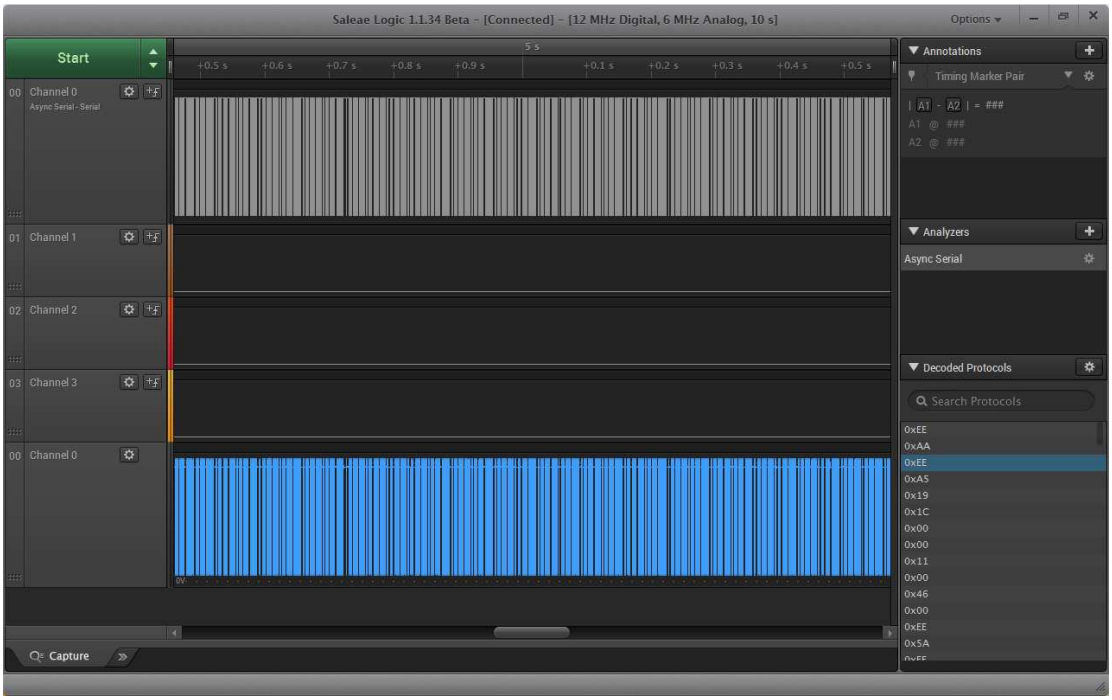
Bestetik , Saleaek analizatu beharreko bit jarioan maila fisikoko protokolo jakin batzuren eredua edo patroia bilatzeko aukera ematen du. Kasu honetan, ez da maila fisikoko protokoloa ezagutzen; hala ere, serie komunikazio batean oinarriturik dagoela ikusten da. Beraz, Saleae serie protokoloa ulertzeko konfiguratu zen (“async serial” irudian). Hori eginda saioak grabatu ziren:



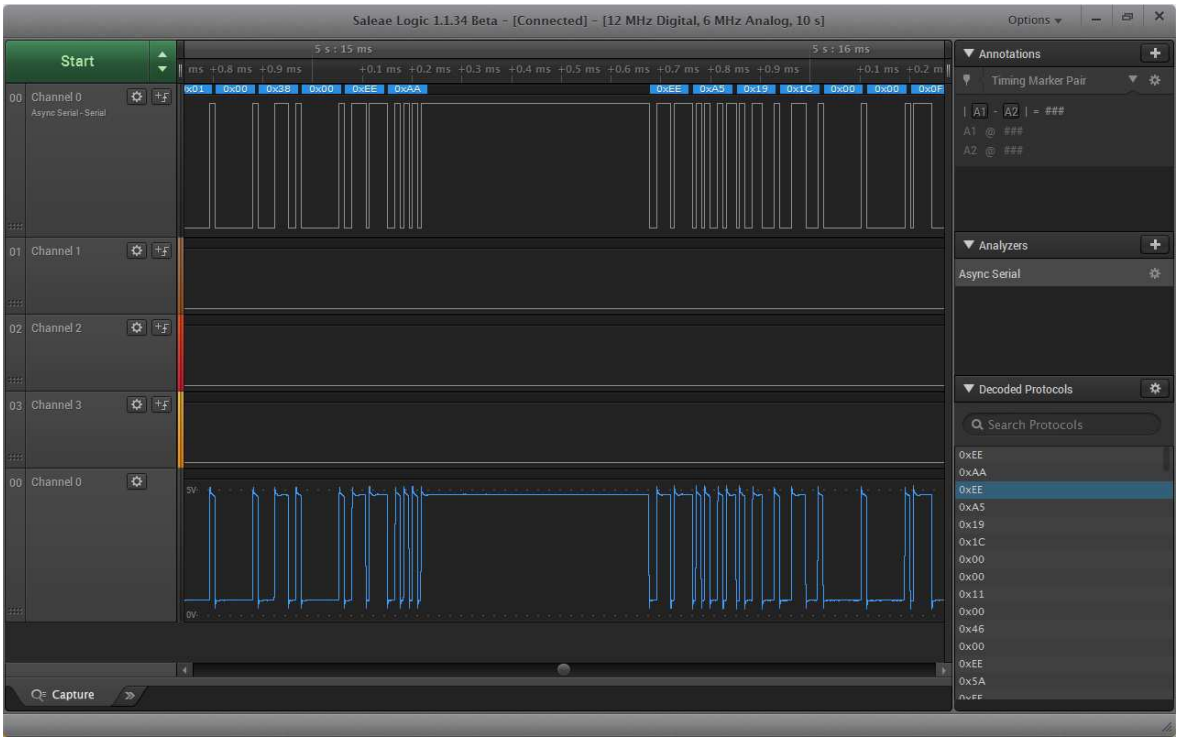
Irudia 26: Saio grabaketa saleaen

Hurrengo irudian ikus daitekeenez, Saleaek bit jarioa kapturatu eta bit jario horretan dagoen informazioa serie protokoloaren egituraren arabera dekodifikatzen du. Serie protokoloa abiadura ezberdinetan konfiguratu daiteke. Saleaen serie protokoloaren abiadura ondo konfiguratzen ez bada, Saleaek dekodifikazioan trama erroreak daudela adierazten du. Proba batzuk egin ostean, errorerik gabeko dekodifikazioa 115200bps-tan egiten zela aurkitu zen. Beraz, abiadura hau hartu zen oinarri bezela.

[7] Saleae analizadore logikoa iturria: https://www.saleae.com/



Irudia 27: Serie mezuen dekodifikazioa saleaen



Irudia 28: Serie mezuen dekodifikazioa saleaen

Saioen grabazioetatik lortutako datuen azterketa analitikoa egin baino lehen, datuen zuzeneko behaketa egin zen. Behaketa honetan, pausagunean zein irekitze eta ixteetan busean aktibitate handia zegoela ikusi zen: hain zen handia pausaguneko aktibitatea, ezen ezin ziren irekitzeak eta izteak bereiztu. Hala ere, alde batetik, hitz guztiak [0xEE, 0xA5] edo [0xEE, 0x55] byte bikoteekin hasi eta [0xEE, 0x5A] edo [0xEE, 0xAA] byte bikoteekin amaitzen direla ikusteko balio izan zuen; bestetik, zuzeneko behaketak bit jario horretan aldizka hitz jakin bat agertzen zela antzemateko balio izan zuen: [0xEE, 0xA5, 0x00, 0x00, 0x00, 0x00, 0x0B, 0x01, 0x0C, 0x18, 0x00, 0xEE, 0x5A]

Time[s]	Analyzer Name	Decoded Protocol Result
0.003186000000000,Async Serial,0xEE		
0.003273500000000,Async Serial,0xA5		
0.003361000000000,Async Serial,0x0C		
0.003448500000000,Async Serial,0x1C		
0.003536000000000,Async Serial,0x00		
0.003623500000000,Async Serial,0x00		
0.003711000000000,Async Serial,0x04		
0.003798500000000,Async Serial,0x00		
0.003886000000000,Async Serial,0x2C		
0.003973500000000,Async Serial,0x00		
0.004061000000000,Async Serial,0xEE		
0.004148500000000,Async Serial,0x5A		
0.004289750000000,Async Serial,0xEE		
0.004376416666667,Async Serial,0x55		
0.004463000000000,Async Serial,0x0C		
0.004549666666667,Async Serial,0x1C		
0.004636250000000,Async Serial,0x00		
0.004722916666667,Async Serial,0x00		
0.004809500000000,Async Serial,0x04		
0.004896083333333,Async Serial,0x02		
0.004982750000000,Async Serial,0x13		
0.005069333333333,Async Serial,0x00		
0.005156000000000,Async Serial,0x41		
0.005242583333333,Async Serial,0x00		
0.005329250000000,Async Serial,0xEE		
0.005415833333333,Async Serial,0xAA		
0.005916000000000,Async Serial,0xEE		
0.006003500000000,Async Serial,0xA5		

- Datuen azterketa analitikoa

Lehen adierazi den bezela, analizadore logikoak eskeinitako datuen behaketa zuzenetik abiatuta, ezin izan zen busetik doan protokoloaren informazio handirik atera. Hala ere, hasierako azterketa honek bit jarioa sinplifikatzeko eta ordenatzeko bidea adierazten du, puntu horretatik abiatuta protokolo ezezagun honetatik informazio gehiago lortzeko asmoz.

Horretarako, datu grabazio saio bakoitzari dagozkion CSV fitxategiak irakurri eta laginak (byte-ak), array batean sartu ziren.

```
def readCSV(file):
    data_path = os.path.expanduser("~") + "/Dropbox/cajon/"
    file_path = data_path + file
    data = csv.reader(open(file_path))

    bytes_array = []
    for row in data:
        if row[0].find('Time') != -1:
            continue
        time, analyzer, result = row
        bytes_array.append([time, result])

    return bytes_array

##############################

bytes_array = readCSV("untitled1.csv")
bytes_array += readCSV("untitled2.csv")
bytes_array.extend(readCSV("untitled3.csv"))

bytes_array = readCSV("untitled4.csv")
bytes_array += readCSV("untitled5.csv")


bytes_array = readCSV("untitled6.csv")

print "Sample size: "
print len(bytes_array)
```

Azpi-prozesu honen irteera CSV ko datuak array batean sarturik izango da.

[['0.000067750000000', '0x00'], ['0.000154375000000', '0x0F'],
['0.000241125000000', '0x01'], ['0.000327625000000', '0x00'],
['0.000414375000000', '0x45'], ['0.000501000000000', '0x00'],
['0.000587625000000', '0xEE'], ['0.000674375000000', '0xAA'],
['0.001746875000000', '0xEE'], ['0.001834375000000', '0xA5']]

Ondoren, bit jarioa sinplifikatzeko, hitz guztiak [0xEE, 0xA5] edo [0xEE, 0x55] byte bikoteekin hasi eta [0xEE, 0x5A] edo [0xEE, 0xAA] byte bikoteekin amaitzen direneko informazioa erabili zen. Horretarako, beste hitzen array bat sortu zen, non hitz bakoitzeren hasierako byte-aren erregistro denbora hitzaren erregistro denbora bezela hartzen zen:

```
word_start_byte1 = '0xEE'
word_start_bytes2 = ['0xA5', '0x55']
word_finish_byte1 = '0xEE'
word_finish_bytes2 = ['0x5A', '0xAA']

words_array = []
word = []
previous_byte = ''
f_word = 0
for each in bytes_array:
    if f_word == 0:
        if (each[1] in word_start_bytes2) and (previous_byte == word_start_byte1):
            word = []
            word.append(previous_byte)
            word.append(each[1])
            f_word = 1
            continue
    elif f_word == 1:
        if (each[1] in word_finish_bytes2) and (previous_byte == word_finish_byte1):
            word.append(each[1])
            words_array.append([each[0], word])
            word = []
            f_word = 0
        else:
            word.append(each[1])
    previous_byte = each[1]

print 'Words Array Length: '
print len(words_array)
```

Azpi-rozesu honen bitartez CSV ko datu berdinak izango ditugu, hitzen arteko banaketa agerian utzita.

```
 ['0.008118000000000', ['0xEE', '0xA5', '0x00', '0x00', '0x00', '0x00', '0x0B',
'0x01', '0x0C', '0x18', '0x00', '0xEE', '0x5A']],

['0.012598750000000', ['0xEE', '0xA5', '0x0C', '0x1C', '0x00', '0x00', '0x04',
'0x00', '0x2C', '0x00', '0xEE', '0x5A']],

['0.014003500000000', ['0xEE', '0x55', '0x0C', '0x1C', '0x00', '0x00', '0x04',
'0x02', '0x13', '0x00', '0x41', '0x00', '0xEE', '0xAA']],

['0.015338000000000', ['0xEE', '0xA5', '0x0C', '0x1C', '0x00', '0x00', '0x11',
'0x00', '0x39', '0x00', '0xEE', '0x5A']],

['0.019530000000000', ['0xEE', '0xA5', '0x0C', '0x1C', '0x00', '0x00', '0x02',
'0x00', '0x2A', '0x00', '0xEE', '0x5A']],

['0.020891500000000', ['0xEE', '0x55', '0x0C', '0x1C', '0x00', '0x00', '0x02',
'0x01', '0x05', '0x30', '0x00', '0xEE', '0xAA']],

['0.022444250000000', ['0xEE', '0xA5', '0x19', '0x1C', '0x00', '0x00', '0x04',
'0x00', '0x39', '0x00', '0xEE', '0x5A']],

['0.023845125000000', ['0xEE', '0x55', '0x19', '0x1C', '0x00', '0x00', '0x04',
'0x02', '0x13', '0x00', '0x4E', '0x00', '0xEE', '0xAA']],

 ['0.025183500000000', ['0xEE', '0xA5', '0x19', '0x1C', '0x00', '0x00', '0x11',
'0x00', '0x46', '0x00', '0xEE', '0x5A']]]
```

Byte-etatik hitzetara pasatzeak bit jarioa asko argitzen du. Hala ere, hitzak beraien berezko byte forman mantentzen badira, hitz banaketak ez du buseko bit jarioaren interpretazioan laguntzen, hau da: byte edo lagin array-a eta hitz array-a gauza bera dira. Horregaitik hitzen hiztegia sortzea eta hitzak laburragoak diren beste ikur batzuren bitartez ordezkatzea erabaki zen.

Zuzeneko behaketan [0xEE, 0xA5, 0x00, 0x00, 0x00, 0x00, 0x0B, 0x01, 0x0C, 0x18, 0x00, 0xEE, 0x5A] hitza aldizka agertzen zela ikusi zenez, hori izan zen hiztegiaren hasierako hitz bezela aukeratu zena.

```
##############################

num = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '@', '#', '$', '%', '&']
alph = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',
        'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z',
        'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N',
        'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
aux_dict = []
for each in num:
    for each2 in alph:
        aux_dict.append(each+each2)

print 'Aux Dict Length: '
print len(aux_dict)

##############################

sample = 0
words_dict = {}
reduced_words_array = []
word_init = ['0xEE', '0xA5', '0x00', '0x00', '0x00', '0x00', '0x0B', '0x01', '0x0C', '0x18',
'0x00', '0xEE', '0x5A']
for each in words_array:
    #print words_array.index(each)
    if sample == 0 and each[1] == word_init:
        reduced_words_array.append([each[0], '0a'])
        words_dict['0a'] = each[1]
        sample = 1
    elif sample == 1:
        aux = []
        for each2 in reduced_words_array:
            if not each2[1] in aux:
                aux.append(each2[1])
        if not each[1] in words_dict.values():
            reduced_words_array.append([each[0], aux_dict[len(aux)]])
            words_dict[aux_dict[len(aux)]] = each[1]
        else:
            reduced_words_array.append([each[0], words_dict.keys()
[words_dict.values().index(each[1])]])
    #print len(words_dict.keys())

print 'Words Dict Length: '
print len(words_dict)
print 'Reduced Words Array Length: '
print len(reduced_words_array)

##############################
```

Azpi-prozesu honen irteera hitz array-aren datuen sinplifikazioa da, hau da, byte kateak beste ikur batzuengatik ordezkatuta agertuko dira.

```
[['0.008118000000000', '0a'],
['0.012598750000000', '0b'],
['0.014003500000000', '0c'],
['0.015338000000000', '0d'],
['0.019530000000000', '0e'],
['0.020891500000000', '0f'],
['0.022444250000000', '0g'],
['0.023845125000000', '0h'],
['0.025183500000000', '0i'],
['0.029375500000000', '0j']]
```

Azkenik , 0A hitza oinarri bezela hartuta, bit jarioa horren arabera zatitu egin zen, zuzeneko beste bigarren behaketa bat egiteko asmoz.

```
aux = ''
for each in reduced_words_array:
    if each[1] == '0a':
        aux += '\n' + each[0] + ' ' + each[1]
    else:
        aux += each[1]

print aux
```

Azpi-prozesu honen irteera sinplifikatutako eta zatitutako bit jarioa da, aldizkako 0A hitza oinarri moduan harturik.

```
0.031658333333333
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.063210666666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.094675500000000
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.126166583333333
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0p

0.161638916666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.193270000000000
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0q

0.229083583333333
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.260382166666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0r

0.295845666666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.327205500000000
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0s

0.362949083333333
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.394623916666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0t
0.430358750000000
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.462234833333333
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0u

0.497812166666667
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o

0.529609416666667
```

```
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0v

0.565571750000000
0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o
```

- Datuen analisia

UntitledParsed fitxategian argi ikusten da zelan **0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0p** luzeeradun katea busaren pausaguneko erreferentzia dela. Datu horren arabera parseatutako datuak oraindik ere gehiago sinplifikatu ahal izateko, bigarren sinplifikazio bat egiten dugu, datuen analisia egiteko balioko diguna:

```
data_path = os.path.expanduser("~") + "/Downloads/cajon/"
file_path = data_path + "untitledParsed2.txt"

data = csv.reader(open(file_path), delimiter=' ')

bytes_array = []
for row in data:
    if row[0].find('Time') != -1:
        continue
    time, result = row
    bytes_array.append([time ,result])

for each in bytes_array:
    if len(each[1]) > len('0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0p'):
        print each
```

Esan bezala, ondoren luzera horretako kateak bakarrik hartuko ditugu, honelako fitxategia sortuz:

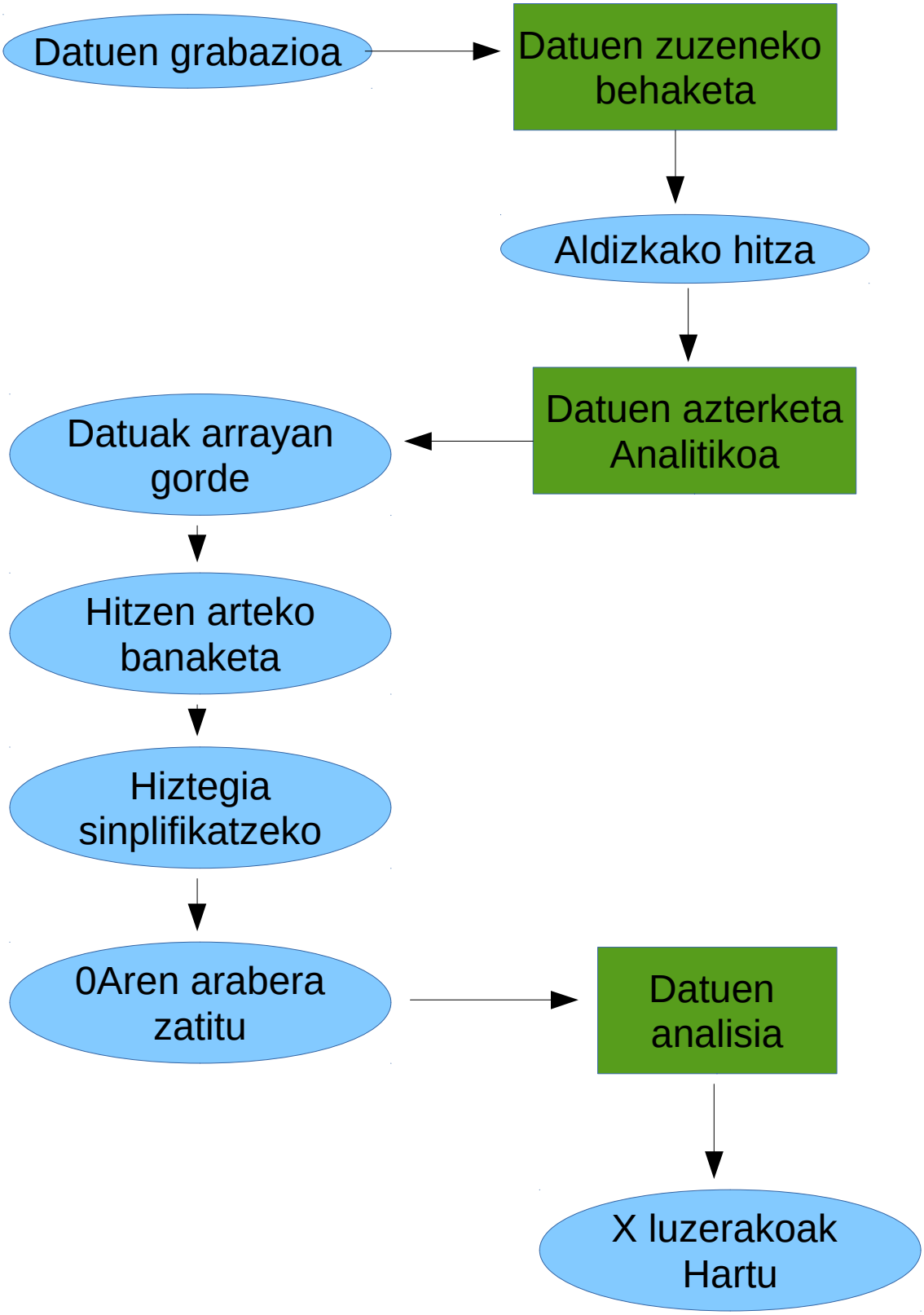
```
['2.941512583333334',
'0a0b0c0d0e0f0g1D0i0j1E0l0m0n0o1F1G1H1I']
--> INICIO APERTURA CAJON EXTERIOR

['5.076044916666667',
'0a0b0c0d0e0f0g2U0i0j0k0l0m0n0o2V1G0Z'] -->
FIN APERTURA CAJON EXTERIOR

['10.800947333333333',
'0a0b0c0d0e0f0g3b0i0j3c0l0m0n0o0b0c0d0e0f0g3
d0i0j3c0b0c0d0e0f0g3e0i0j3c3f1I'] --> INICIO
CIERRE CAJON EXTERIOR

['13.207430416666666',
'0a0b0c0d0e0f0g4s0i0j0k0l0m0n0o2V1G'] -->
FIN CIERRE CAJON EXTERIOR

['13.573502500000000',
```

```
'0a0b0c0d0e0f0g4t0i0j0k0l0m0n4u4v4w'] -->
REBOTE CIERRE CAJON EXTERIOR

['20.202437416666665',
'0a0b0c0d0e0f0g4D0i0j1E0l0m0n0o1F1G1H1I0w']
--> INICIO APERTURA CAJON EXTERIOR

['22.412892916666667',
'0a0b0c0d0e0f0g2W0i0j0k0l0m0n0o2V1G'] -->
FIN APERTURA CAJON EXTERIOR

['27.836351416666666',
'0a0b0c0d0e0f0g5M0i0j3c0l0m0n0o0b0c0d0e0f0g5
N0i0j3c0b0c0d0e0f0g3d0i0j3c3f1I'] --> INICIO
CIERRE CAJON EXTERIOR

['29.626190500000000',
'0a0b0c0d0e0f0g6H0i0j3h0l0m0n0o0I1z']

['30.208593666666665',
'0a0b0c0d0e0f0g4y0i0j0k0l0m0n0o2V1G0R'] -->
FIN CIERRE CAJON EXTERIOR

['30.611180166666667',
'0a0b0c0d0e0f0g4z0i0j0k0l0m0n4u4v4w'] -->
REBOTE CIERRE CAJON EXTERIOR

['36.707197916666665',
'0a0b0c0d0e0f0g1D0i0j1E0l0m0n0o1F1G1H1I']
--> INICIO APERTURA CAJON EXTERIOR


['38.877262916666666',
'0a0b0c0d0e0f0g7G0i0j0k0l0m0n0o2V1G'] -->
FIN APERTURA CAJON EXTERIOR

['44.359191333333335',
'0a0b7H0d0e7I0g2W0i0j0k0l0m0n0o7J7K7L7M1j']
--> INICIO APERTURA CAJON INTERIOR

['46.493126666666669',
'0a0b8Y0d0e0f0g2W0i0j0k0l0m0n0o8Z7K0E'] -->
FIN APERTURA CAJON INTERIOR
```

Une honetan, denbora errealeko bit jarioaren kaptura **sparkfun rs485**[8] aren bidez egingo dela erabakitzen da. Erabaki honen arrazoi nagusia honen tamainaren erosotasuna da, baina baita ere bi datu pin jasotzen dituelako. Sparkfun rs485 en beste alde positiboak irakurri zein idazteko aukera ematen duela da.

[8] Sparkfun rs485-a iturria: https://www.sparkfun.com/products/9822

Ondoren aurreanalisi prozesua egiteko ad hoc tresnaren funtzionamendua laburbiltzen duen fluxu-diagrama aurkezten da:



Irudia 29: Aurreanalisi tresnaren fluxu diagrama

6. Analisia eta diseinua

Behin aurreanalisia burutua edukita gure aplikazioen beharrak eta egitura identifikatzeko unean gaude. Analisia garrantzia handikoa da aplikazioa guk nahi dugun moduan funtziona dezan.

Fase honetan aplikazioak beharrezkoak dituen erramintak, lengoaiak eta arkitektura definituko ditugu, programatzen hasi baino lehenagoko estruktura orokor bat lortzeko asmoarekin.

6.1 Proiektuak beharrezkoak dituen erramintak

6.1.1 Software erramintak

Proiekturako aurreikusten diren software herramintak ondorengoak dira:

- Windows 7
- Libreoffice
- Adobe Reader
- Visual Paradigm
- Paint
- Pycharm
- Eclipse Indigo *Eclipse Java EE IDE for Web Developers* bertsioa
- GanttProject
- Logic Sniffer
- Saleae softwarea
- Mysql workbench
- ApacheBunch
- Firebug osagarria

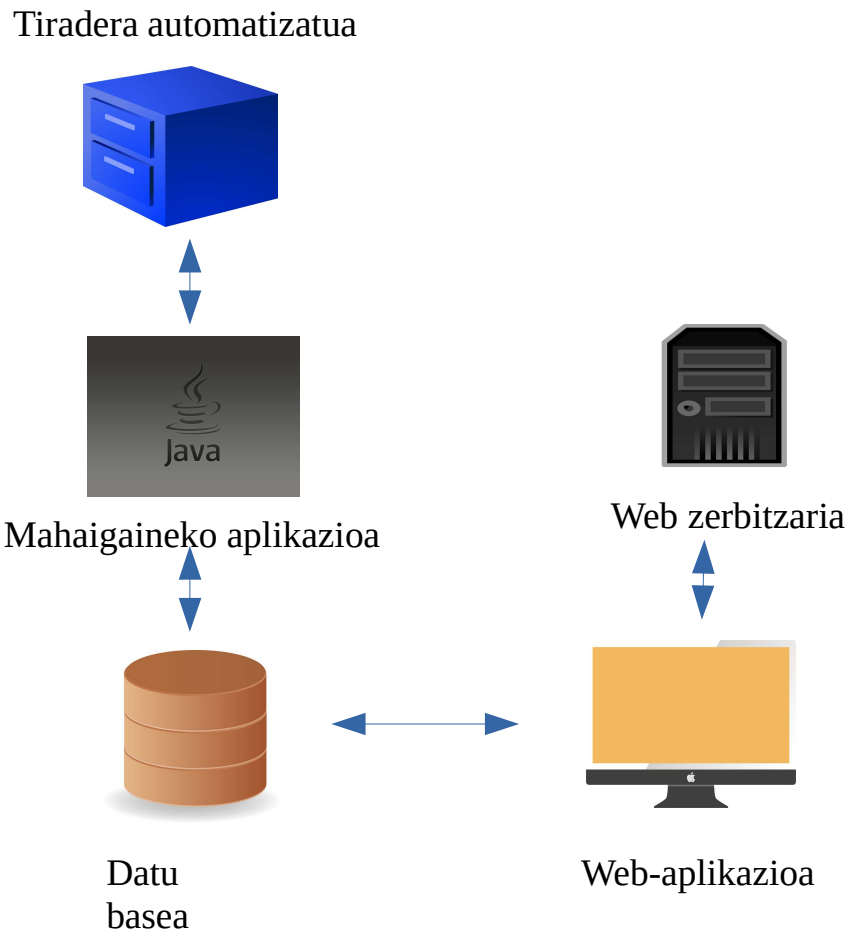
6.1.2 Hardware erramintak

Proiekturako aurreikusten diren hardware herramintak ondorengoak dira:

- HP-G62 eramangarria
- Osziloskopio analogikoa
- Osziloskopio digitala
- Arduino mega
- Saleae analizatzailea
- Sparkfun plaka

6.2 Arkitektura

Hona hemen sistemaren arkitektura:



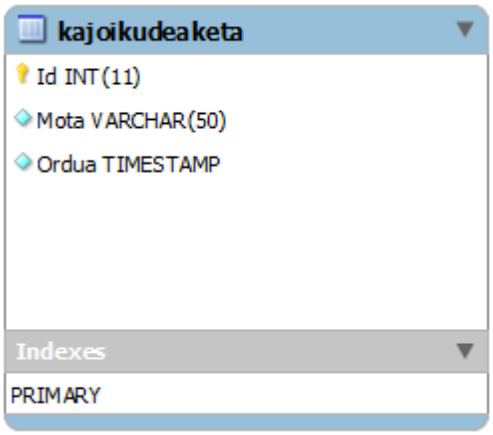
Irudia 30: Sistemaren arkitektura

6.3 Gertaera fluxuak

6.3.1 Erabiltzaileak noiztik-norako erregistroa kontsultatu

- Erabiltzaileak hasiera orritik abiatuta, erregistroa botoian jarriko du arratoia.
- Behin arratoia botoi horren gainean edukita, bi aukera zabalduko zaizkio: erregistro orokorra eta noiztik-nora.
- Noiztik-nora aukera klikatu ondoren, hasiera eta bukaera data bat sartuko ditu, nahi izanez gero egutegitik aukeratzeko aukerarekin.
    - Datak sartzean, erabiltzaileak bidali botoia klikatuko du eta guk sartutako data-tarte horretan emandako mugimenduak pantailaratuko zaizkigu taula baten.

6.4 Datu base eskema erlazionala



Irudia 31: Datu base eskema erlazionala

7. Programazio lengoaiak eta inplementazioa

7.1 Erabilitako programazio lengoaiak

Atal honetan, erabili diren programazio lengoaiak azalduko dira, bakoitzaren azalpen labur batekin.

7.1.1 Python lengoaia

Python interpretatutako programazio lengoai bat da, zeinen helburua irakurgarria den kodea lortuko duen sintaxi egokia lortzea den. Programazio lengoai multiparadigmiko bezala definitzen da, objektuei zuzendutako, programazio funtzionala, etab jasaten dituelako. Kode irekiko lizentzia dauka, Python Software Foundationek zabaldua.

Esandako moduan, python programazio lengoaia multiparadigmikoa da, honek esan nahi du hainbat programazio mota jasaten dituela, programatzailearen eskutan utzita azken honen aukeraketa. Pythonen ezaugarri nagusi bat metodoen lotura dinamikoa da, hau da, metodoen eta aldagaien arteko lotuta dinamikoa exekuzioan zehar. Pythonen beste helburu bat moduluen hedapena da, ezaugarri honen esker c edo c++ en modulu berriak idatzi ditzazkegu.

Pythonek bere baitan komando interpretatzaile bat dauka, zeinetan argibideak idatziko ditugun. Gaitasun honek kodea zatika edo apurka probatzeko aukera emango digu modu interaktibo baten. GAL honen kasuan, gaitasun hau oso erabilgarria izan da pixkanaka edo pausuka aurreanalisia egiten joateko.

Hona hemen osagai nagusiak:

- Ez da aldagaien deklarazioa egin behar
- Operadore logikoak and,or eta not
- Indentazioa tabulazioen bidez

- Funtzioak “def” hitzarekin hasi
- “class” hitza klasea definitzeko

Irudia 32: Python logotipoa

7.1.2 Java lengoaia

Java objektuei zuzendutako programazio lengoai konkurrentea da, zeinen diseinua inplementario dependentziak ekiditera bideratuta dago. Programazio lengoai honen helburua, garatutako programa edozein ingurunetan exekutatua izatea da, birkonpilatu gabe. Lengoai oso ezaguna da, batez ere bezero-zerbitzari motatako aplikazioetan.

Javaren ezaugarri nagusia objektuetara bideratua dela da, honek programazio metodologia eta lengoaiaren diseinuari buruzko informazioa ematen digu. Lehenengo ideia datu ezberdinak euren operazioei lotuta egotea da, hauen konbinazioaz abiatuta objektu izeneko entitateak sortuta. Honen helburua, proiektu handien gestioa erraza izatea da, honela kalitatea bermatuz eta akatsak mugatuz. Beste aldeko atal bat objektuen berrerabilpena da objektu generikoak sortuz, aurrerantzean egiten diren proiektuetan denbora asko aurreztuta.

Esandako moduan, javaren helburua kodea edozein plataformatan exekutatzea da, hau da, kodea java plataformarekiko independentea da. Honek esan nahi du java lengoaian idatzitako edozein kode, edozein hardwarean exekutagarria izango dela. Beraz, esan dezakegu behin idatzitako kodea edozein gailutan exekutagarria izan ahal dela.

Hona hemen osagai nagusiak:

- Klaseetan banaturiko egitura
- Klase nagusiak main() metodoa
- “void” hitzak ez dela ezer bueltatzen esan nahi du
- “static” klaseari loturiko metodoeta
- “public”,”protected” eta “private” metodoetan hierarkia bat osatzeko

Irudia 33: Java logotipoa

7.1.3 HTML lengoaia

HTML(ingelesez HypeText Markup Language) hipertestuentzako marka-lengoaia bat da. Lengoai hau web orrien garapenean oso erabilgarria da.

Standard Generalizated Markup Language etiketetatik sortutako formatu irekia da, zeinen helburua lista batean barruan aurkitzen diren dokumentuak etiketatzea eta ordenatzea den.

HTML lengoaia dokumentuari ordena emango dioten etiketen izenak zehazteko erabiltzen da, zeinetan ordena guztiz librea den. Dokumentuaren estruktura eta bertako elementuen deskripzioak garatu ahal izango ditugu. Azken finean, dokumentuen formatuar zehazteko baligarria eta ulerterraza den lengoaia da. Esan bezala, etiketak sortzen dira, horretarako < eta > zeinuak erabilita. Elementuek lengoaiaren estrukturari gorputza emango diote, bi ezaugarri ezberdinekin: edukia eta ezaugarriak.

HTML lengoaiaren gaitasunen barruan, beste lengoai batzuek eskaintzen dituzten aginduak, *script* izenekoak, exekutatzeko aukera dago. Agindu hauek nabigatzaileentzat zuzenduta daude, eta honek prozesatu ditzan.

Irudia 34: HTML5 logotipoa

7.1.4 CSS lengoaia

CSS lengoaia HTML dokumentuen itxura definitzeko erabiltzen den lengoaia da. Ingelesetik datorren *Cascading Style Sheet* ek dioen moduan, kaskada eran definitutako estiloen orri bat da.

CSS lengoaia HTML orrian bertan definitu daiteke etiketa berezi batzuen bidez, baina baita beste dokumentu bat erabilita definitu daiteke estiloa. Aukerarik onena beste dokumentu bat erabiltzea da, bestela, HTML dokumentuan kode karga handiak ekidituz.

CSS lengoaiaren sintaxia oso erraza da, ingelesezko hitz batzuk estiloa emateko ezaugarri moduan erabiliak izango dira, hauei balio batzuk esleituz.

Barnean hautatzaile bat edo gehiago dituen arau lista batek konposatzen du estilo orria. HTML an hautatzaile hauen erreferentziak aurkitzen dira eta erreferentzia bakoitzean, honeri dagozkion estiloak aplikatuko zaizkio. Azken finean diseinuaren kontrola edukitzeko erraminta bat bezala definitu dezakegu.

Irudia 35: CSS3 logotipoa

7.2 Inplementazioa

Proiektu honen garapena pycharm eta eclipse programazio inguruneen bidez egin da, lagungarriak izan diren bertako plugin batzuetaz baliatuz. Aurretik esan bezala proiektua kronologikoki eman diren hiru zatitan banatzen da:

- Aurreanalisi tresna garapena
- Denbora errealeko java programaren garapena
- Web-aplikazioaren garapena

Aurretik azaldu bezala, hainbat programazio lengoaia erabili dira proiektuan zehar, baita guztiz lagungarriak izan diren **gyovinet**[9] eta **mysql-connector** liburutegiak.

7.2.1 Gyovinet liburutegia

Esan bezala proiektuaren garapena aurrera eramateko, guztiz beharrezkoa da datuak edo bit jarioa seriean kapturatzea ahalbidetzen duen liburutegi baten erabilera. Hainbat liburutegi probatu ostean, **gyovinet** liburutegiaren alde egin da. Liburutegi hau aukeratzearen arrazoia, bere baitan dituen metodoen erabilgarritasuna eta erabiltze erraztazuna da.

Hona hemen liburutegiaren egitura:

- Dokumentazioa
- NativeLibraries
  - libSerialPort.dll
  - libSOSerialPort.so
- GyovinetDriver

[9] Gyovinet liburutegia iturria: http://www.giovynet.com/giovynetDriver_es.html

Aurretik esan bezala, liburutegiak hainbat metodo ditu bere baitan, erabiltzaileari datuen kaptura errazte aldera. Esan beharra dago gyovinet barruan **com eta parameters** klaseen erabilera egin dela.Hona hemen metodo garrantzitsuenak:

- Com
    - getPort():Erabiltzen ari garen komunikazio portuaren izena esango digu.
    - close(): Serie portua ixteko metodoa.
    - receiveToString(): Jasotako ASCII karaktereak stringetara bihurtzen dituen metodoa.
- Parameters
    - setPort(): portua zehazteko metodoa.
    - setBaudRate(): muestreo frekuentzia ezartzeko metodoa.
    - setMinDelayWrite(): karaktere berri bat idatzi baino lehen pasatu beharreko denbora minimoa ezartzeko metodoa.

7.2.2 Java aplikazioa

Java aplikazioaren inplementazioa, aurreanalisi tresnaren inplementazioaren ostean dator. Prozesu atalean azalduta dagoen moduan, aurreanalisi tresnak informazioa era egoki batean antolatuko digu eta bilatu beharreko patroia zein den esango digu, beraz prozesu honen sarrera aurreanalisi prozesuaren irteera izango da. Java aplikazioaren garapenean erabilitako konfigurazioak bai saleae bai aurreanalisi tresnetan oinarrituta daude.

Jarraian kodearen azalpena denbora ordenean emango da:

- Konfigurazioen ezarpena

Egin beharreko lehenengo gauza gure programak izango duen konfigurazioa ezartzea da, esan bezala aurreanalisi eta saleaerekin egindako saioetan oinarritutako konfigurazioa ezarriko dugu.

Ezarri beharreko parametroak honakoak dira:

- Portua
- Baudrate
- Mindelaywrite

Horretan laguntzeko, hurrengo metodoak garatu dira:

```
public void portuaLortu() throws Exception {
		SerialPort serialPort = new SerialPort();
		List<String> portsFree = serialPort.getFreeSerialPort();
		for (String free : portsFree) {
			System.out.println(free);
		}
	}

public Parameters parametroakEzarri(String portua, Baud baudRate,
			int interbaloa) throws Exception {
		Parameters param = new Parameters();
		param.setPort(portua);
		param.setBaudRate(baudRate);
		param.setMinDelayWrite(interbaloa);
		return param;
	}


Parameters param = this.parametroakEzarri("COM12", Baud._115200, 50);
```

- Bit jarioa jaso

Behin konfigurazioa ezarrita bit jarioa edo datuak jasotzen hasiko gara. Horretarako portuari deitu eta jasotako integerra aldagai batean gordeko dugu. Ulergarria izan dadin eta datua aurreko programetan jaso aldera, integer hori hexadezimalera bihurtuko dugu.

```
Integer data = com12.receiveSingleCharAsInteger();
                    String data2 = data.toHexString(data);
                    kont++;
```

- Hitzekin kateak sortu

Behin datua hartu dugula, instantzia bakoitzaren mota begiratuko da kateak sortzeko asmoarekin. Bakarrik **ffee, ffa5, 0, b, 1, c, 18, 5a** motatako instantziei erreparatuko diegu. Honela, aurreko motatako instantzia bat agertzen den bakoitzean, aurretik sortu dugun hitz edo kateari itsatsiko diogu.

```
if (data2.equals("ffee") || data2.equals("ffa5")
                            || data2.equals("0") || data2.equals("b")
                            || data2.equals("1") || data2.equals("c")
                            || data2.equals("18") || data2.equals("5a")) {

                        hitza = hitza + data2;
```

- Sortutako hitzaren analisia

Behin hitz edo kate bat sortu dugula, aurretik azaldu den kate berezia den ala ez begiratuko dugu.

Hitz berezi hori den kasuan, hitz berezi hori agertu den azkenengo alditik irakurri ditugun instantzia kopuruari erreparatuko diogu. Esan bezala hitz berezi hori maiztazun gutxiagorekin ematen da tiraderan mugimendua dagoenean, beste era batera azalduta, guk sortzen ditugun kateak luzeagoak izango dira mugimendua dagoen kasuetan.

Sortutako katea ez denean bilatzen ari garen hitz berezia, hitza aldagaia hutsik utziko dugu hitz berri bat sortzen hasteko asmoarekin.

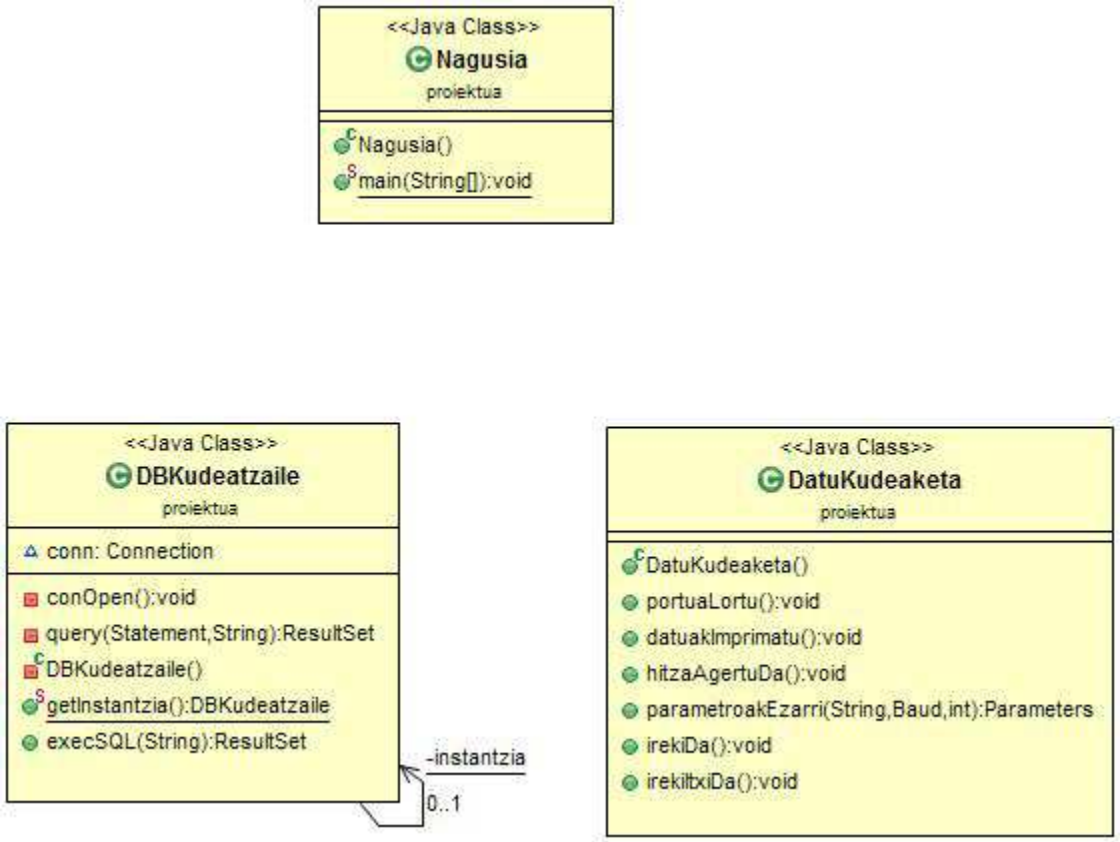
```
if (hitza.length() >= 24) {
                                    if (hitza.contains("b1c")) {
```

Sortutako hitzan “b1c” karaktereak batera agertzea, hitz berezi osoa agertzea esan nahi du, saleareRekin egindako saioetan aurretik behatu baita karaktere hauek ez direla inoiz era independentean agertzen.

Aurretik esan bezala hitz berezi hori agertu den azkenengo alditik irakurri ditugun instantzia kopuruari erreparatuko diogu mugimendua egon den ala ez erabakitzeko. Horregatik algagai baten bidez bi hitz bereziren artean agertu diren instantzia kopuruen neurketa egingo da, 450 instantzia baino gehiagoko kasuak bakarrik hartuz mugimendu bezala. Esan bezala bit jarioa etengabe gertatzen ari da, horrek esan nahi du hitz berezia etengabe agertuko dela eta honekin batera hitzak sortuko direla. Aurreanalisian ikusi den moduan kate luzeenak ixtean sortzen dira, hau da, instantzia kopuru gehiago egongo dira gure bi hitz berezien artean. Horrekin aurreikusten da geldiunean baino informazio gehiago mugitzen dela busetik mugimendu bat egotean.

```
if (kont > 450) {

       i++;

        System.out.println(i + " (e)tan"+ " ZABALDU-ITXI DA!");

        System.out.println(kont);

        Statement st = (Statement) dbk.conn.createStatement();
        st.executeUpdate("INSERT INTO kajoiKudeaketa(mota) " + "VALUES    ('zabaldu')");

                    }
```

- Tiraderaren irekitzeak atzemateko kodea eta klase diagrama



Irudia 36: Denbora errealeko java programaren klase diagrama

Hona hemen denbora errealean tiradera automatizatuaren irekitze eta ixteen erregistroa egiteko garatutako metodo nagusiaren kodea:

```
// Aurrebaldintza: -
// Postbaldintza: Tiraderaren mugimenduak datu basean erregistratuko ditu

public void irekiDa() throws Exception {
	DBKudeatzaile dbk = DBKudeatzaile.getInstantzia();
	// portua="COM12"
	// baudRate=Baud._115200
	// interbaloa=50
	Parameters param = this.parametroakEzarri("COM12", Baud._115200, 50);
	Com com12 = new Com(param);
	String hitza = "";
	int kont = 0;
	int i = 0;
	for (;;) {

		Integer data = com12.receiveSingleCharAsInteger();
		String data2 = data.toHexString(data);
		kont++;
		// System.out.println(data2);

		if (data2.equals("ffee") || data2.equals("ffa5")
				|| data2.equals("0") || data2.equals("b")
				|| data2.equals("1") || data2.equals("c")
				|| data2.equals("18") || data2.equals("5a")) {

			hitza = hitza + data2;

			// System.out.println("hitza: " + hitza);
			// System.out.println("Tamaina: " + hitza.length());

			if (hitza.length() >= 24) {
				if (hitza.contains("b1c")) {

					if (kont > 450) {// 235kin ondo,450kin bakarrik ixteak
						// hartzen ditu(seguro,ezdu
						// fallatzen)
						i++;

						System.out.println(i + " (e)tan"
								+ " ZABALDU-ITXI DA!");

						System.out.println(kont);

						Statement st = (Statement) dbk.conn
								.createStatement();
						st.executeUpdate("INSERT INTO kajoiKudeaketa(mota) "
								+ "VALUES ('zabaldu')");

					}
					hitza = "";
					// System.out.println("Katea agertu da");
					// System.out.println("Distantzia " + kont + " da");
					kont = 0;
					// arrayan gordetzen hasi katea berriz agertu arte
				} else {
					hitza = "";
				}
			}

		}
	}}
```

7.2.3 Web-aplikazioa

Hurrengo lerroetan tiraderaren mugimenduen erregistro orokorra aurkezteko programatutako jsp-a aurkezten da:

```
<!-- Import -->
<%@page import="java.util.SimpleTimeZone"%>
<%@page import="java.util.Calendar"%>
<%@ page language="java" %>
<%@ page import = "java.util.ArrayList"%>
<%@ page import = "proiektua.DBKudeaketa"%>
<%@ page import = "proiektua.Datuak"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<html>
<head>
<title>Erregistro Orokorra</title>
<link rel="stylesheet" type="text/css" href="CSS/estiloak.css" />
</head>
<body BGCOLOR="#A4A4A4">
<h2><%= new java.util.Date()%> -ko Erregistro Orokorra</h2>
      <div id="header">
            <nav>
                  <!-- nav -->
                  <ul class="nav">
                        <li><a href="hasieraOrria.html">Hasiera</a></li>
                        <li><a href="">Erregistroa</a>
                              <ul>
                                    <li><a href="erregistroOrokorra.jsp">Orokorra</a></li>
                                    <li><a href="noiztikNora.jsp">Noiztik-Nora</a></li>
                              </ul></li>
                        <li><a href="">Laguntza</a></li>
                  </ul>
            </nav>
            <!-- nav -->
      </div>
<table border="3">
<tr>
<td><b>Id</b></td>
<td><b>Mota</b></td>
<td><b>Ordua</b></td>
</tr>
<%
ArrayList<Datuak> lista = DBKudeaketa.getDatuak();
for (int i=0;i<lista.size();i++)
{
   out.println("<tr>");
   out.println("<td>"+lista.get(i).getId()+"</td>");
   out.println("<td>"+lista.get(i).getMota()+"</td>");
   out.println("<td>"+lista.get(i).getOrdua()+"</td>");
   out.println("</tr>");
}
%>
</table>
</body>
</html>
```

7.3 Emandako arazoak eta hauei aurre egiteko soluzioak

- Liburutegi egoki bat bilatzea

Hainbat liburutegi probatu ostean, egokiena java hizkuntzan dagoela ikusita, denbora errealeko analisia burutzeko programa javan egitea erabakitzen da.

- Material ezegokia

Bit jarioaren analisia egiteko hasiera baten ez da egon beharrezko materiala. Hasiera batean osliloskopio analogiko zein digitalak erabiltzen dira, baina horrekin ez da nahikoa. Saleae analizagailua erostea erabakitzen da aurreanalisi tresna egoki bat garatzeko aukera izateko.

8. Probak

Edozein aplikazioren sendotasuna probatzerako orduan, probak egikaritzeko momentua dator. Esan beharra dago proba hauek web-aplikazioaren gainean egin direla, beste aplikazioak probatzeko zailtasuna dela eta.

Honetarako apareche-ren **apachebench** eta firefox-en **firebug** tresnen laguntzaz baliatu gara. Apachebench tresna edukitzeko apache instalatuta edukitzearekin nahikoa da eta firebug, firefox nabigatzailearen osagarri bat da.

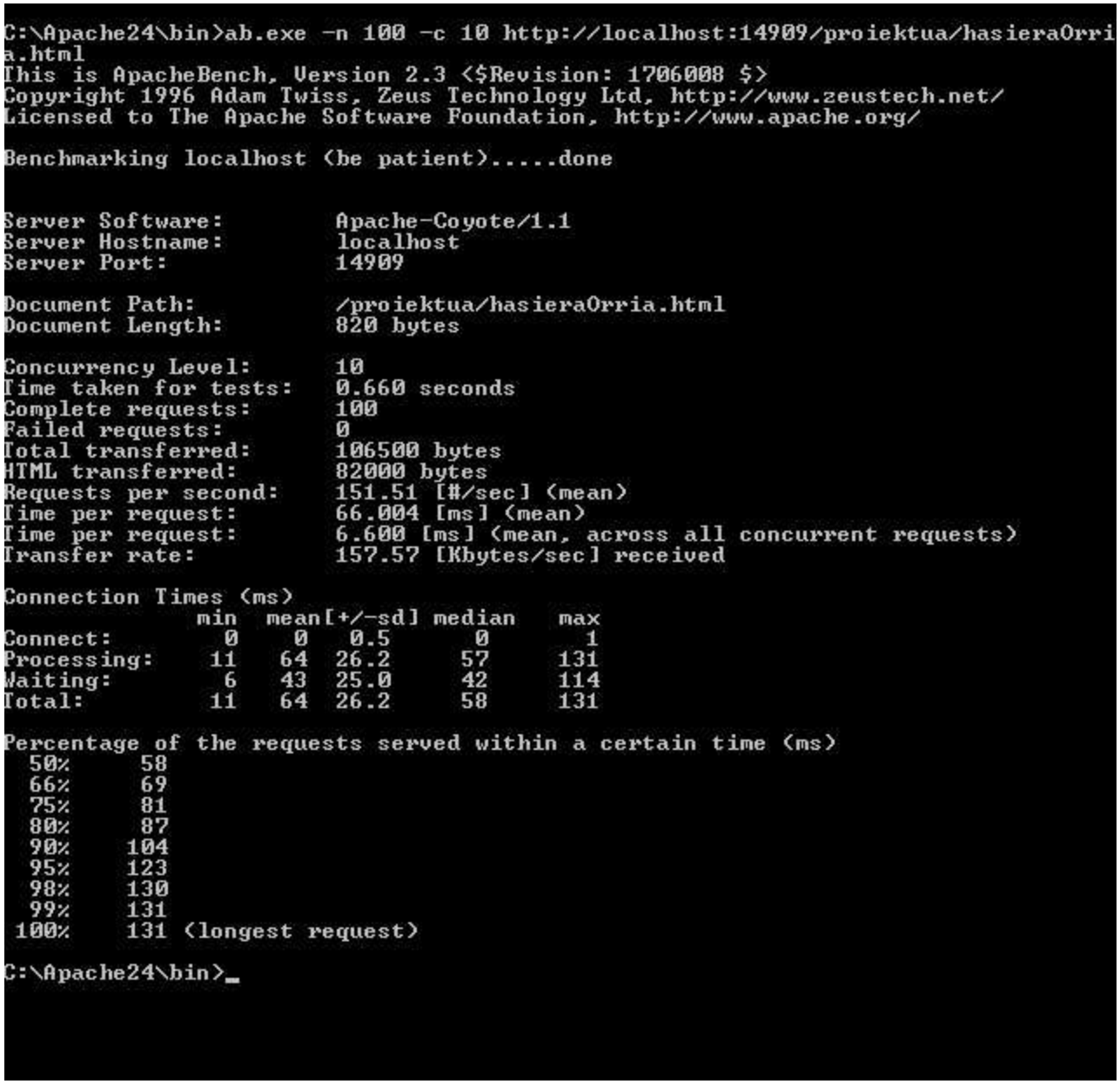
Apachebench aplikazioak zerbitzua frogatuko du, honen bezero kopuru eta konkurrentzia erakutsiz. Tresna honek web-aplikazioa probatzen ari diren bezero kopuru bat simulatuko du, web-aren errendimendua frogatua izan dadin. Aplikazio honek eskaera kopuru handi bat era konkurrentean egitea ahalbidetzen du.

Firebug deitutako firefox nabigatzailearen osagarriak, erabiltzaileak jasotako erantzuna ausnartzea ahalbidetzen digu. Sare, errendimendu eta erantzunen informazioa eskeinita.

8.2 Apachebench

Esan bezala, aplikazio honek web-aplikazioa frogatuko du bezero kopuru bat simulatuz. Aplikazioa exekutatzeko nahikoa da terminalean ab.exe -n 100 -c 10 http://localhost:14909/proiektua/hasieraOrria.html idaztea, non n eskaera kopurua den eta c zatitutako hari kopurua edo konkurrentzia maila den. Aplikazioak irteera moduan hainbat datu eskeintzen ditu, hona hemen garrantzitsuenak:

- Zerbitzariaren softwarea
- Zerbitzariaren portua
- Dokumentuaren luzera
- Konkurrentzia maila
- Eskaera okerrak
- Denbora informazioak



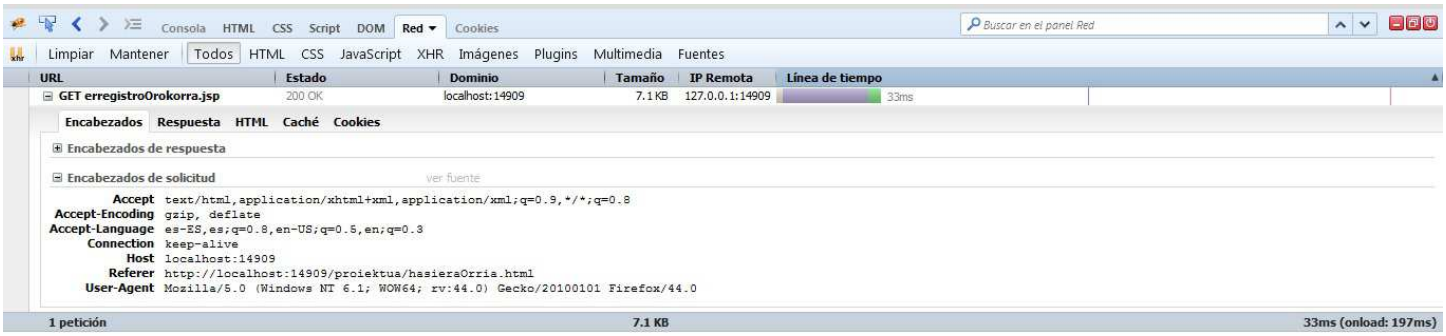
Irudia 37: Web-aplikazioan apachebecnh proba

8.2 Firebug

Esan bezala aplikazio honek erabiltzaileak jasotako erantzuna ikustea ahalbidetzen digu. Honetaz aparte sare, errendimendu eta erantzunen informazioak eskeiniko dizkigu. Aplikazioa erabiltzeko firefox nabigatzailea instalatuta eduki behar da firebug osagarriarekin. Nabigatzailean gure web-aplikazioa zabalduko dugu eta firegub-ek eskaera bakoitzaren informazioa emango digu.

Hona hemen informazio garrantzitsuena:

- Domeinua
- Egoera
- Tamaina
- Denbora



Irudia 38: Web-aplikazioan egindako firebug proba

9. Ondorioak eta etorkizunerako lana

9.1 Betetako helburuak

GAL honen zailtasun nagusia gure produktua “kutxa-beltz” bat dela zen. Esan bezala software APIak zein hardware datasheet-ak ez dira ezagunak, beraz alderantzizko ingeniaritzaren erabilera eskatzen du tiradera automatizatuaren ezaugarrietaz ohartzeko. Bestalde, informatikako graduan ikasten ez diren konpetentzien eta kontzeptu askoren ikasketa eskatzen zuen.

Esan bezala, GAL-a era egokian garatzeko hainbat helburu lortu behar ziren, hona hemen hauetako batzuk:

- **Esanguratsua den ad hoc aurreanalisi tresna baten garapena:** GAL honen hasierako korapiloa desegiteko balio izan duen ad hoc aurreanalisi tresna emaitza egokiekin garatu da.
- **Bit jarioaren denbora errealeko analisia burutzeko programa:** Instantzien atzematea egiteko denbora errealeko aplikazioa era egoki batean garatzea lortu da, ahalik eta instantzia kopuru gutxien utzita analizatu gabe.
- **Web-aplikazio ulerterraz baten sorrera:** Esan beharra dago atal honen helburua emaitzen aurkezpena zela. Horrek esan nahi du webgune sinple bat eskatzen zela, edonork erabili ahal izan duen webgunea non emaitzak era argi baten ikusten diren.
- **Era errazean erabiltzeko aplikazioa:** Beste lorpen bat edornork martxan jarri ahal duen aplikazioa dela. Tiradera automatizatuaren hiru kableak plakarekin konektatzearekin eta jar-a exekutatzearekin nahikoa da. Horretan laguntzeko, laguntza gune bat garatu da webgunean.

9.2 Kudeaketaren aldaketak

Esan beharra dago GAL honetan hiru atal nagusi eskatzen zirela eta horiek entregatu direla funtzionalitateetan aldaketak eman gabe.

9.2.1 Aldaketak planifikazioan

Proiektuaren ordu errealak planifikatutako orduen pareko izan dira, aurreanalisiko lan paketea kenduta, non planifikazio ezegoki bat eman zen. Hona hemen atal bakoitzean planifikatutako orduen taula ordu errealekin alderatuta:

Lan paketea	Planifikatuta	Erreala
LP1	8 ordu	7 ordu
LP2	18 ordu	15 ordu
LP3	30 ordu	70 ordu
LP4	12 ordu	10 ordu
LP5	20 ordu	17 ordu
LP6	36 ordu	32 ordu
LP7	142 ordu	105 ordu
LP8	11 ordu	7 ordu
LP9	12 ordu	10 ordu
LP10	5 ordu	6 ordu
Zuzendariarekin bilerak	24 ordu	34 ordu
GUZTIRA GAL	**318 ordu**	**313 ordu**

Taula 19: Planifikazioan emandako adaketak

Aurreko taulan ikusten den moduan proiektua planifikatutakoaren arabera irten da, aurreanalisia ezik. Atal honetan, aurretik esan dugun moduan ez dago metodo zientifikorik eta zaia da denbora estimazio bat egitea, zailtasunak apurka agertzen doazelako. Gure kasuan optimistegia izan zen aurreanalisirako planifikatutako denbora, eman diren arazoak direla eta(ez zegoen API ez datasheetik) planifikatutakoaren denbora bikoitza behar izan da.

9.2.2 Aldaketak kostuan

Lan paketea	Erreala	Kostua
LP1	7 ordu	84€
LP2	15 ordu	180€
LP3	70 ordu	840€
LP4	10 ordu	120€
LP5	17 ordu	204€
LP6	32 ordu	384€
LP7	105 ordu	1260€
LP8	7 ordu	84€
LP9	10 ordu	120€
LP10	6 ordu	72€
Zuzendariarekin bilerak	34 ordu	1700€
GUZTIRA giza baliabideak	**313 ordu**	**5048€**

Taula 20: Kostuan emandako aldaketak

Deskribapena	Kostua
Material amortizagarria	136,63 €
Material suntsikorra	92€
Beste gastu batzuk	240€
Giza baliabideak	5048 €
AURREKONTU TOTALA	**5516,63 €**

Taula 21: Kostu erreala

Kostuei dagokienez ez da gauza askorik aldatu, kostu erreala planifikatutakoa baino 300€ gehiago izanda. Hau zuzendariarekin egin beharreko bilera gehigarrien eraginez izan da.

Atala	Planifikatutakoa	Erreala
GAL orduak guztira	318 ordu	313 ordu
AURREKONTU TOTALA	**5196,63 €**	**5516,63 €**

Taula 22: Planifikatutako kostua kostu errealarekin alderatuta

9.3 Ondorioak

GAL honetan sortutako aplikazioa oso erabilgarria izan daiteke enpresa arloan, batez ere tiradera automatizatu mota hau erabiltzen dituzten industria farmazeutiko edo bitxigintza sektoreetan. Honela, tiradera automatizatu mota hau erabiltzen dituzten erabiltzaileek era erraz batean hauen kontrol eta erregistro bat izango dute. Merkatuan mota honetako aplikazioetan dagoen hutsuneak aplikazio interesgarri eta erakargarri batetan bihurtzen du aplikazio hau.

Aurretik esan bezala, GAL honen arazo nagusia tiradera automatizatua “kutxa-beltz” bat dela da. Software API eta hardware datasheet-en gabeziak aurreanalisi fase bat eskatu du, zein alderantzizko ingeniaritza erabiliz garatu da. Hona hemen etapa honetako arazo nagusiak:

- Ez dago alderantzizko ingeniaritza aplikatzeko metodo zientifikorik
- Bit jarioan datu kopuru handia denbora gutxitan(30 segundutan 3GB lagin)
- Bit jarioa etengabe, nahiz eta tiradera automatizatua geldirik egon
- Protokoloa aurkitzeko arazoak
- Bit jarioan patroi argi bat bilatzeko arazoa
- Material egokiaren falta hasiera baten
    - Osziloskopioen falta
    - Saleaearen falta

Esan beharra dago aurrean aipaturiko arazo guztiei aurre egin zaiela, GAL honetan graduan ikasi ez diren makina bat konpetentzia eta kontzeptu berri asko ikasita, baita alderantzizko ingeniaritza eta aurreanalisiak burutzeko gai izatera.

9.4 Etorkizunerako lana

Proiektu honek aparteko moduluen gehiketarako edo hedapenerako esparru handia eskeintzen du. Hauetako batzuen konplexutasuna dela eta, beste proiektu bat egiteko erabilgarriak izan daitezke. Hona hemen proiektuaren hedapena gauzatzeko ideia batzuk, honela erakargarriagoa izan dadin:

- **Zabaldu/itxi faseak bereiztea:** Hurrengo pausua tiradera zabaldu edo itxi den detektatzea da. Abantai honekin erabiltzaileak uneoro jakingo du tiraderaren egoera momentu hortan.

- **Ordenagailutik zabaldu/itxi ahal izatea:** Hedapen konplexuena, tiradera automatizatuaren protokoloak ez baitira ezagunak. Hedapen honen helburua busetik irakurri beharream, busean idaztea da.

- **Web-aplikazioan estatistikak eskeintzea:** Hedapen honekin apertura eta ixteeb erregistroekin estatistikak garatuko ziren.

- **Denbora errealeko java programa hari eta prozesuetan banatzea:** Hedapen honekin prozesu ezberdinak sortuko ditugu, lehenengoa instantziak hartzeko, bigarrena analisia egiteko eta hirugarrena datu basean erregistroak gordetzeko. Hain da azkarra bit jarioa(1GB lagin 10 segundutan) non ziurrenik datu basean erregistro bat gordetzean, laginak galtzen diren bidetik. Hori ekiditeko, gomendagarria litzake programa prozesuetan banatzea. Aipatu beharra dago, programa honen kasuan instantzia bat galtzeak ez duela eraginik.

- **Gure aplikazioa sarean egotea[10]:** Gure aplikazioa sarean egonda, erabiltzaileak urrunetik bistaratu ahal izango du tiradera automatizatuan gertatzen ari dena, tiraderara konektatuta egon gabe.

[10] Internet of things iturria: https://es.wikipedia.org/wiki/Internet_de_las_cosas

Bibliografia

1. *Ingenieria inversa. URL:* https://es.wikipedia.org/wiki/Ingenier%C3%ADa_inversa
2. *Bases y tipos de cotización 2016. URL:* http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm
3. *Ganttproject FAQ. 2009. URL:* http://ganttproject.blogspot.com.es/search/label/faq
4. *Robert Sun. Hacking Christmas Lights. 2010. URL:* http://www.deepdarc.com/2010/11/27/hacking-christmas-lights/
5. *Soloelectronicos. Control domestico mediante un smartphone usando ingenieria inversa. 2015. URL:* http://soloelectronicos.com/2015/05/15/control-domestico-mediante-un-smartphone-usando-ingenieria-inversa/
6. *Productos grass. URL*: http://www.grass.at/uebersicht-sensotronic.html?L=5
7. *Jason Blackman and Scott Monroe. Overview of 3.3V CAN (Controller Area Network) Transceivers. 2013. URL:* http://www.ti.com/lit/an/slla337/slla337.pdf
8. *CAN-Bus Data Capture with Wireshark on Raspberry Pi. 2014. URL:* http://skpang.co.uk/blog/archives/1141
9. *Sergegsx. Comunicación OBD con Ford Focus 2007. 2011. URL:* http://secuduino.blogspot.com.es/2011/03/comunicacion-obd-con-ford-focus-2007.html
10. *Parámetros, método GET y POST en Servlets. URL:* http://www.edu4java.com/es/servlet/servlet3.html
11. *Luis Javier Somodevilla. Apache tomcat. 2013. URL:* http://expertoj2ee.blogspot.com.es/2013/06/apache-tomcat.html
12. *Tutorial CSS. URL:* http://es.html.net/tutorials/css/
13. *Consulta de Base de Datos con JSP usando una Clase Java separada. 2013. URL:* http://chuwiki.chuidiang.org/index.php?title=Consulta_de_Base_de_Datos_con_JSP_usando_una_Clase_Java_separada
14. *Pablo Turmero. Captura de datos desde puerto serial con JAVA. URL:* http://www.monografias.com/trabajos104/captura-datos-puerto-serial-java/captura-datos-puerto-serial-java.shtml#ixzz3zseQ03qs

15. *Jhonbo. ApacheBench, una sencilla herramienta para testear servidores web. 2012. URL:* http://www.genbetadev.com/herramientas/apachebench-una-sencilla-herramienta-para-testear-servidores-web
16. *Javier Antoniucci. Manual Básico de Firebug. 2007. URL:* http://www.adictosaltrabajo.com/tutoriales/fire-bug/