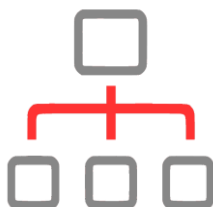


Informatika Fakultatea

Informatika Ingeniaritzako Gradua

▪ Gradu Amaierako Lana ▪

Software Ingeniaritza



ProMeta

Softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema metaereduetan oinarrituta

Egilea: Julen Etxaniz Aragoneses
Tutorea: Juan Manuel Pikatza Atxa
2021eko iraila

Laburpena

Proiektuaren helburua metaereduetan oinarritutako softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema eraikitzea da. Izan ere, softwarea garatzeko egokiak diren hainbat metodologia existitzen dira. Garrantzitsua da metodologia horien informazioa ereduen bidez definitzea, etorkizunean malgutasunez kudeatu eta hobekuntzak egin ahal izateko. Gainera, beharrezkoa da ereduen informazioa erabiliz metodologia ezartzen duen sistema eraikitzea, garapen taldeak proiektuetan erabiltzeko. Proiektuaren garapenerako, OpenUP metodologia erabili da eta proiektuaren dokumentazioa eta memoria idazteko CCII-N2016-02 estandarra.

Hitz gakoak: metaeredua, garapen metodologia, aurkezpen araua, softwarearean bizi-zikloa, MDE, CMS, sistematizazioa, automatizazioa.

Resumen

El objetivo del proyecto es construir un sistema para la definición e implementación de procesos de desarrollo de software basado en metamodelos. De hecho, existen varias metodologías que son adecuadas para el desarrollo de software. Es importante definir la información de estas metodologías a través de modelos para que se puedan gestionar de forma flexible en el futuro y se puedan realizar mejoras. Además, es necesario construir un sistema que establezca una metodología utilizando información del modelo para uso de los equipos de desarrollo en los proyectos. Para el desarrollo del proyecto se utilizó la metodología OpenUP y el estándar CCII-N2016-02 para la redacción de la documentación y memoria del proyecto.

Palabras clave: metamodelo, metodología de desarrollo, norma de presentación, ciclo de vida del software, MDE, CMS, sistematización, automatización.

Abstract

The objective of the project is to build a system for the definition and implementation of software development processes based on metamodels. In fact, there are several methodologies that are suitable for software development. It is important to define the information of these methodologies through models so that they can be managed flexibly in the future and improvements can be made. In addition, it is necessary to build a system that establishes a methodology using information from the model for use by development teams in projects. The OpenUP methodology was used for the development of the project and the CCII-N2016-02 standard for the drafting of the project documentation and memory.

Keywords: metamodel, development methodology, presentation rule, software life cycle, MDE, CMS, systematization, automation.

Aurkibide Orokorra

Laburpena.....	3
Resumen.....	3
Abstract	3
Aurkibide Orokorra.....	5
Irudien Aurkibidea	13
Taulen Aurkibidea.....	17
Laburdurak	19
1 Sarrera	21
2 Helburuak	23
2.1 Arazoaren Planteamendua	23
2.2 Produktuaren Planteamendua.....	24
2.3 Interesatuen Deskribapena.....	24
3 Aurrekariak	25
3.1 Software Kalitatea.....	25
3.2 Bezeroen Eskakizun Gogorrak.....	25
3.3 Garapen Metodologiak	26
3.4 Proiektuen Aurkezpenerako Arauak	28
3.5 Prozesuetan Oinarritutako Garapena	28
3.6 Ezagutzaren Kudeaketa eta Inferentzia Motorrak	29
3.7 Ereduek Bideratutako Ingeniaritza	31
3.8 Edukiak Kudeatzeko Sistemak.....	32
3.9 ProWF Sistema	32
4 Egungo Egoera	35
4.1 Deskribapena	35
4.2 Abantailak eta Desabantailak	35
4.2.1 Abantailak.....	35
4.2.2 Desabantailak	36
4.3 Proposatutako Hobekuntzak	36
4.4 Prestakuntza	37
5 Arauak eta Erreferentziak.....	39
5.1 Aplikatutako Legedia eta Araudia	39
5.2 Bibliografia	40
5.3 Metodoak.....	41

5.3.1	OpenUP.....	41
5.3.2	ABRD.....	41
5.3.3	MDE.....	41
5.4	Tresnak.....	42
5.4.1	Git.....	42
5.4.2	GitHub.....	42
5.4.3	GitHub Pages.....	42
5.4.4	Toggle Track.....	42
5.4.5	Java.....	43
5.4.6	Eclipse IDE.....	43
5.4.7	EPF Composer.....	44
5.4.8	Visual Studio Code.....	44
5.4.9	Xtext.....	45
5.4.10	XSLT.....	45
5.4.11	Microsoft Office.....	45
5.4.12	PlantUML.....	45
5.4.13	PHP.....	45
5.4.14	Drupal.....	45
5.4.15	XAMPP.....	46
5.4.16	Pantheon.....	46
5.4.17	MariaDB.....	47
5.4.18	phpMyAdmin.....	47
5.5	Ereduk.....	48
5.5.1	CCII-N2016-02.....	48
5.5.2	OpenUP.....	48
5.5.3	ABRD.....	48
5.5.4	UMA.....	48
5.5.5	Ecore.....	48
5.6	Metrikak.....	48
5.6.1	Irismena.....	48
5.6.2	Denbora.....	49
5.6.3	Kostua.....	49
5.6.4	Kalitatea.....	49
5.7	Prototipoak.....	49
5.7.1	ProMeta ModelEditor.....	49

5.7.2	ProMeta IO-System	49
6	Definizioak eta Laburdurak	51
6.1	Laburdurak	51
6.2	Definizioak	53
6.2.1	Bizagi	53
6.2.2	BPM	53
6.2.3	CCII	53
6.2.4	CCII-N2016-01	53
6.2.5	CCII-N2016-02	53
6.2.6	CMMI	53
6.2.7	CMS	53
6.2.8	DOT	53
6.2.9	Drupal	53
6.2.10	IBM	53
6.2.11	ISO	53
6.2.12	OpenUP	54
6.2.13	PlantUML	54
6.2.14	ProMeta	54
6.2.15	ProWF	54
6.2.16	RUP	54
6.2.17	SPICE	54
6.2.18	UML	54
6.2.19	UNE	54
6.2.20	URPS	54
6.2.21	Workflow	54
6.2.22	WYSIWYG	54
7	Hasierako Betekizunak	55
7.1	Betekizun Funtzionalak	55
7.2	Betekizun Ez-Funtzionalak	55
7.3	Sistemaren Ezaugarriak	56
7.3.1	Erabilgarritasuna	56
7.3.2	Fidagarritasuna	56
7.3.3	Errendimendua	56
7.3.4	Mantengarritasuna	56
7.4	Sistemaren Interfazeak	56

7.4.1	Itxura eta Sentsazioa	56
7.4.2	Diseinu eta Nabigazio Betekizunak	56
7.4.3	Trinkotasuna	56
7.4.4	Erabiltzailearen Pertsonalizazio Betekizunak	56
8	Irismena	57
8.1	Bizi-zikloa	57
8.2	Artefaktuak	57
9	Hipotesiak eta Murriztapenak	59
9.1	Hipotesiak	59
9.2	Murriztapenak	59
10	Aukeren Azterketa eta Egingarritasuna	61
10.1	Arkitektura	61
10.1.1	Bizagi	61
10.1.2	ProWF	62
10.1.3	ProMeta	62
10.2	CMS	63
10.2.1	Wordpress	63
10.2.2	Drupal	64
10.2.3	Joomla	64
10.3	Dokumentazioa <i>Hostinga</i>	64
10.3.1	GitHub Pages	64
10.3.2	GitLab Pages	64
10.3.3	Netlify	65
10.4	Drupal <i>Hostinga</i>	65
10.4.1	000webhost	65
10.4.2	Heroku	65
10.4.3	Pantheon	65
10.5	Datu-base Kudeaketa Sistema	65
10.5.1	MySQL	65
10.5.2	PostgreSQL	65
10.6	Datu-basearen Sorrera	65
10.6.1	Inferentzia Motorra	66
10.6.2	Teneo	66
10.6.3	Xtext	66
10.7	Lanerako Ingurunea	66

10.7.1	Makina Birtuala	66
10.7.2	Ordenagailu Pertsonala.....	66
10.8	Bertsio Kontrola	66
10.8.1	GitHub	66
10.8.2	GitLab	66
10.9	Metaeredua	67
10.9.1	SPEM	67
10.9.2	UMA	67
10.10	Metodologia	67
10.10.1	OpenUP	67
10.10.2	RUP	67
10.10.3	ABRD.....	67
10.11	Metodologia Editorea.....	67
10.11.1	EPF Composer	67
10.11.2	Rational Method Composer	68
10.11.3	Editore Grafikoa	68
10.11.4	Testu Editorea	68
10.12	Prozesua Bistaratu	68
10.12.1	DOT.....	68
10.12.2	XPDL	68
10.12.3	Webgunea	68
10.13	Denboraren Kontrola.....	68
10.13.1	Toggl Track	68
10.13.2	Clockify	68
10.13.3	WakaTime	69
10.14	Diagramak.....	69
10.14.1	PlantUML.....	69
10.14.2	Draw.io	69
10.14.3	UMLDoclet.....	69
11	Proposatutako Sistemaren Deskribapena	71
11.1	Azpisistemak	71
11.2	ModelEditor	71
11.2.1	Arkitektura	71
11.2.2	Erabilpen Kasuak	72
11.2.3	Diseinua.....	73

11.2.4	Dokumentazioa	74
11.3	IO-System.....	83
11.3.1	Arkitektura	84
11.3.2	Erabilpen Kasuak	84
11.3.3	Diseinua.....	86
11.3.4	Dokumentazioa	87
11.4	Proba.....	96
11.5	Hedapena.....	97
11.6	Etorkizunerako Hobekuntzak.....	98
12	Arriskuen Analisia	99
12.1	Arriskuak	99
12.2	Arriskuen Analisia	100
13	Proiektuaren Antolamendua eta Kudeaketa	101
13.1	Proiektuaren Antolamendua	101
13.1.1	Taldeakideak eta rolak	101
13.1.2	Informazio-sistema.....	101
13.1.3	Komunikazio-kanalak	101
13.2	Proiektuaren Kudeaketa	101
13.2.1	Integrazioaren Kudeaketa	101
13.2.2	Irismenaren Kudeaketa	102
13.2.3	Epeen Kudeaketa.....	102
13.2.4	Kostuen Kudeaketa	102
13.2.5	Kalitate Kudeaketa	103
13.2.6	Giza Baliabideen Kudeaketa	103
13.2.7	Komunikazioen Kudeaketa.....	103
13.2.8	Arriskuen Kudeaketa	104
13.2.9	Erosketen Kudeaketa	104
13.2.10	Interesatuen Kudeaketa	104
14	Denbora Planifikazioa	105
15	Aurrekontuaren Laburpena	107
16	Dokumentuen Lehentasun Ordena	109
	ERANSKINAK	111
	I. Memoriaren Eranskinak.....	111
	A1 - Sarrerako Dokumentazioa.....	111
	A2 - Analisia eta Diseinua	112

A3 - Tamaina eta Esfortzu Estimazioa	112
A4 - Kudeaketa Plana	112
A5 - Segurtasun Plana	112
A6 - Beste Eranskinak	113
II. Sistemaren Espezifikazioa	113
III. Barne Informazioa	114
III.1 Barne Kudeaketa	114
III.2 Artefaktuen Txantiloak	115
III.3 Aurreko Proiektuak	115
III.4 Ingurunea	115
III.5 Tresnak	116
III.6 Trebatzeko Materialak	116
IV.1 Arriskuen Analisia	117
IV.1.1 Arriskuak	117
IV.1.2 Arriskuen Analisia	119
IV.2 Denbora Planifikazioa	121
IV.2.1 Mugarriak	121
IV.2.2 LDE Diagrama	122
IV.2.3 Gantt Diagrama	123
IV.2.4 Iterazioak	124
IV.2.5 Atazen Denborak	126
IV.2.6 Denboraren Neurketa	126
IV.2.7 Desbiderapenak	128

Irudien Aurkibidea

1.1. Irudia. Proiektuaren webgunean memoria bistaratu.	21
1.2. Irudia. Webguneko eranskinen eta barne informazioaren menuak.	22
3.1. Irudia. RUPen prozesu iteratiboa, bizi-zikloaren faseak, iterazioak eta jarduerak.	27
3.2. Irudia. OpenUP-en geruzak: mikro-gehikuntzak, iterazio bizi-zikloa eta proiektu bizi-zikloa.	27
3.3. Irudia. BPM <i>Gartner</i> -en Koadrante Magikoa eta <i>Forrester Wave</i> txostena.	29
3.4. Irudia. EHSISen garapen ingurunea.	30
3.5. Irudia. Programazio tradizionalaren eskema.	30
3.6. Irudia. Erregeletan oinarritutako sistemen eskema: klaseak, objektuak, erregelak eta egitateak.	30
3.7. Irudia. MDERen 3 abstrakzio maila: eredua, metaeredua eta meta-metaeredua.	31
3.8. Irudia. ProWF Workflow Editor azpisistemaren arkitektura.	33
3.9. Irudia. ProWF IO-System azpisistemaren arkitektura.	33
5.1. Irudia. ABRD metodologiaren prozesuko faseak.	41
5.2. Irudia. ProMeta proiektuko webgunearen GitHub biltegia.	42
5.3. Irudia. Toggle Track tresnan hilabete bateko egun bakoitzeko denbora.	43
5.4. Irudia. ModelEditor azpisistemako proiektuak Eclipse IDE editorean.	43
5.5. Irudia. OpenUP bizi-zikloa EPF Composer tresnan.	44
5.6. Irudia. ProMeta dokumentazio webgunearen errepositorioa Visual Studio Code-n.	44
5.7. Irudia. PlantUMLren lengoain idatzitako sekuentzia-diagrama.	45
5.8. Irudia. Proiektuan sortutako Drupal webgunea.	46
5.9. Irudia. XAMPP kontrol panela Apache eta MySQL hasieratzeko.	46
5.10. Irudia. <i>ProMeta</i> webguneko <i>Pantheon</i> kontrol panela.	47
5.11. Irudia. phpMyAdmin tresnarekin datu-baseko taulak kudeatzen.	47
5.12. Irudia. UMA metodologiako metodoaren eta prozesuaren kontzeptuak.	48
8.1. Irudia. OpenUP metodologiako bizi-zikloaren faseak.	57
10.1. Irudia. Bizagi Modeler erabiliz sortutako prozesua.	61
10.2. Irudia. BPMn eta workflowetan oinarritutako arkitekturak.	62
11.1. Irudia. ModelEditor eta IO-System azpisistemen arkitektura.	71
11.2. Irudia. ModelEditor azpisistemaren arkitektura.	72
11.3. Irudia. ModelEditor azpisistemako erabilpen kasuen eredua.	72
11.4. Irudia. UMA metaeredua graphviz formatuan.	73
11.5. Irudia. UMA metaereduaren zati bat graphviz formatuan.	74
11.6. Irudia. OpenUP bizi-zikloa EPF Composer tresnan.	74
11.7. Irudia. OpenUP metodologiaren webgunea.	75
11.8. Irudia. Eraldaketa egiteko XSLT fitxategia.	75
11.9. Irudia. EMF proiektuaren sorrera Eclipsen.	76
11.10. Irudia. Uma metaeredua inportatzen Eclipsen.	76
11.11. Irudia. Uma.ecore fitxategia Eclipsen.	77
11.12. Irudia. Uma metaereduaren klase diagrama.	77
11.13. Irudia. Editore grafikoaren kodearen sorrera.	77
11.14. Irudia. Editore grafikoa martxan jartzeko konfigurazioa.	78
11.15. Irudia. Editore grafikoarekin library.uma fitxategia bistaratzen.	78
11.16. Irudia. Xtext proiektua sortzen Eclipsen.	79
11.17. Irudia. Uma metaeredutik Xtext proiektua sortzen.	79
11.18. Irudia. Xtext proiektuaren eta lengoaiaren izenak aukeratzen.	80
11.19. Irudia. Testu editorearen proiektuak Eclipsen.	80

11.20. Irudia. Xtext proiektuko lengoaiaren gramatika.....	80
11.21. Irudia. Editore grafikoa ezkerrean eta testu editorea eskuinean.	81
11.22. Irudia. Datu-basearen datuak dituen SQL fitxategia.	81
11.23. Irudia. Metodologiaren datu-basea phpMyAdmin-en.	82
11.24. Irudia. ModelEditor-en paketeak eta paketeen arteko dependentziak.	82
11.25. Irudia. Pakete diagrama adibidea.....	83
11.26. Irudia. Klase diagrama adibidea.	83
11.27. Irudia. IO-System azpisistemaren arkitektura.	84
11.28. Irudia. IO-System azpisistemako erabilpen kasuen eredua.....	85
11.29. Irudia. OpenUP metodologiako rolak multzokatuta.....	85
11.30. Irudia. Proiektu kudeatzaileak bete beharreko ataza eta artefaktuak.	85
11.31. Irudia. Datu-base erlazionalaren diseinua.	86
11.32. Irudia. Drupal datu-base baten diseinua.	87
11.33. Irudia. Drupal webgunera datu-basetik edukia inportatzeko moduluak.	88
11.34. Irudia. ProWF sistemaren hasierako orria.	88
11.35. Irudia. Drupal webgunearen hasierako orria.....	89
11.36. Irudia. Drupal webguneko edukia ikusteko orria.	89
11.37. Irudia. Drupal webguneko metodologiak ikusteko orria.	90
11.38. Irudia. Drupal webgunean OpenUP metodologia ikusteko orria.	90
11.39. Irudia. Drupal webgunean maiatzean gehitutako edukia.	91
11.40. Irudia. Drupal webguneko glosario orria.....	91
11.41. Irudia. Drupal webgunean edukia bilatzeko aukera.	92
11.42. Irudia. Drupal webguneko saioa hasteko orria.....	92
11.43. Irudia. Drupal webguneko kontua sortzeko orria.....	93
11.44. Irudia. Drupal wenguneko pasahitza berrezartzeko orria.	93
11.45. Irudia. Drupal webguneko erabiltzailearen kontuaren orria.	94
11.46. Irudia. Drupal webguneko kontua editatzeko orria.	94
11.47. Irudia. Drupal webguneko OpenUP metodologiaren orria.	95
11.48. Irudia. Drupal webguneko metodologia sortzeko orria.	95
11.49. Irudia. Drupal webgunean rolak kudeatzeko orria.....	96
11.50. Irudia. Drupal webgunean baimenak kudeatzeko orria.	96
12.1. Irudia. Arriskuen inpaktu eta probabilitate analisia.	100
0.1. Irudia. Eranskinak webguneko menuan.	111
0.2. Irudia. Memoriaren Eranskinak webguneko menuan.	111
0.3. Irudia. Sarrerako Dokumentazioa webguneko menuan.	112
0.4. Irudia. Analisia eta Diseinua webguneko menuan.	112
0.5. Irudia. Beste Eranskinak webguneko menuan.	113
0.6. Irudia. Sistemaren Espezifikazioa webguneko menuan.	113
0.7. Irudia. Barne Informazioa webgunearen menuan.....	114
0.8. Irudia. Barne Kudeaketa webguneko menuan.	114
0.9. Irudia. Artefaktuen Txantiloia webguneko menuan.	115
0.10. Irudia. Aurreko Proiektuak webguneko menuan.	115
0.11. Irudia. Ingurunea webguneko menuan.	115
0.12. Irudia. Tresnak webguneko menuan.	116
0.13. Irudia. Trebatzeko Materialak webguneko menuan.	116
0.1. Irudia. Arriskuen inpaktu eta probabilitate analisia.	119
0.1. Irudia. Plangintzako mugarri garrantzitsuen diagrama.	121

0.2. Irudia. Birplangintzako mugarri garrantzitsuen diagrama.....	122
0.3. Irudia. LDE diagrama.....	122
0.4. Irudia. Plangintzako Gantt diagrama.	123
0.5. Irudia. Birplangintzako Gantt diagrama.....	123
0.6. Irudia. Proiektuaren hilabete bakoitzeko denbora.....	127
0.7. Irudia. Apirileko egun bakoitzeko denbora.	127
0.8. Irudia. Proiektuaren denbora atazaka.	128

Taulen Aurkibidea

2.1. Taula. Arazoaren Planteamendua.	23
2.2. Taula. Produktuaren Planteamendua.....	24
2.3. Taula. Interesatuen deskribapena.	24
3.1. Taula. CMMI-ren 5 heldutasun-mailak.	26
7.1. Taula. Hasierako betekizun funtzionalak.....	55
7.2. Taula. Hasierako betekizun ez-funtzionalak.	55
8.1. Taula. OpenUP metodologiako bete diren artefaktuak domeinutan sailkatuta.	58
10.1.Taula. CMS ezberdinen ezaugarrien konparaketa.....	63
12.1. Taula. Identifikatutako arriskuen zehaztasunak.	100
13.1. Taula. Kalitate kudeaketako rolak eta ardurak.....	103
15.1. Taula. Aurrekontua giza baliabideen eta erreminten kostuekin.	107
15.2. Taula. Software taldeko kideen kostu baremoa.	108
0.1. Taula. Identifikatutako arriskuen zehaztasunak.	119
0.1. Taula. Plangintza eta birplangintzako mugarri garrantzitsuak.	121
0.2. Taula. Proiektuko iterazioen fasea, helburuak eta datak.	125
0.3. Taula. Lan-atazen zehaztasunak.	126

Laburdurak

ABRD	Agile Business Rules Development
BETRADOK	Betekizunen Trazabilitatea eta Dokumentazio sorkuntza automatikoa
Bizagi	BPM hornitzailea
BPM	Business Process Management
BPMN	Business Process Model and Notation
CCII	Consejo de Colegios de Ingeniería Informática
CCII-N2016-01	Norma de Visado de Proyectos y Actuaciones Profesionales en Ingeniería Informática
CCII-N2016-02	Norma Técnica para la realización de la Documentación de Proyectos en Ingeniería Informática
CLIPS	C Language Integrated Production System
CMMI	Capability Maturity Model Integration
CMS	Content Management System
COOL	Classroom Object Oriented Language
CSS	Cascading Style Sheet
DOT	Grafoak definitzeko lengoaia
Drupal	CMS hornitzailea
DSL	Domain Specific Language
Ecore	Meta-metaeredua
EHSIS	Erabakiak Hartzen Laguntzeko Sistemen Sortzailea
EHU	Euskal Herriko Unibertsitatea
EMF	Eclipse Modeling Framework
EPF	Eclipse Process Framework
Freemium	Dohako oinarritzko zerbitzuak gehi ordaindu beharrekoak
GrAL	Gradu Amaierako Lana
HTML	HyperText Markup Language
IBM	International Business Machines
IDE	Integrated Development Environment
ISO	International Organization for Standardization

LDE	Lan Deskonposaketa Egitura
MDE	Model-Driven Engineering
NASA	National Aeronautics and Space Administration
OpenUP	Open Unified Process
PDF	Portable Document Format
PHP	Hypertext Preprocessor
PlantUML	Plant Unified Modeling Language
PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
ProMeta	Professional Metamodel
ProWF	Professional Workflow
RUP	Rational Unified Process
SOA	Service-oriented Architecture
SPEM	Software Process Engineering Metamodel
SPICE	Software Process Improvement and Capability dEtermination
SQL	Structured Query Language
SWEBOK	Software Engineering Body of Knowledge
UMA	Unified Method Architecture
UML	Unified Modeling Language
UNE	Una Norma Española
URPS	Usability, Reliability, Performance and Supportability
Workflow	Lan-fluxua
WYSIWYG	What You See Is What You Get
XAMPP	Cross-platform Apache, MariaDB, PHP and Perl
XMI	XML Metadata Interchange
XML	eXtensible Markup Language
XPDL	XML Process Definition Language
XSLT	eXtensible Stylesheet Language Transformations

1 Sarrera

Dokumentu hau Julen Etxaniz Aragonesesen Informatika Ingeniaritzako Gradu Amaierako Lanaren memoria da. Egilearen inguruko informazio gehiago jakiteko <https://julenetxaniz.eus/> webgunea kontsulta daiteke. Lan hau Euskal Herriko Unibertsitateko (UPV-EHU) Donostiako Informatika Fakultatean landu da, Juan Manuel Pikatza izanik tutorea.

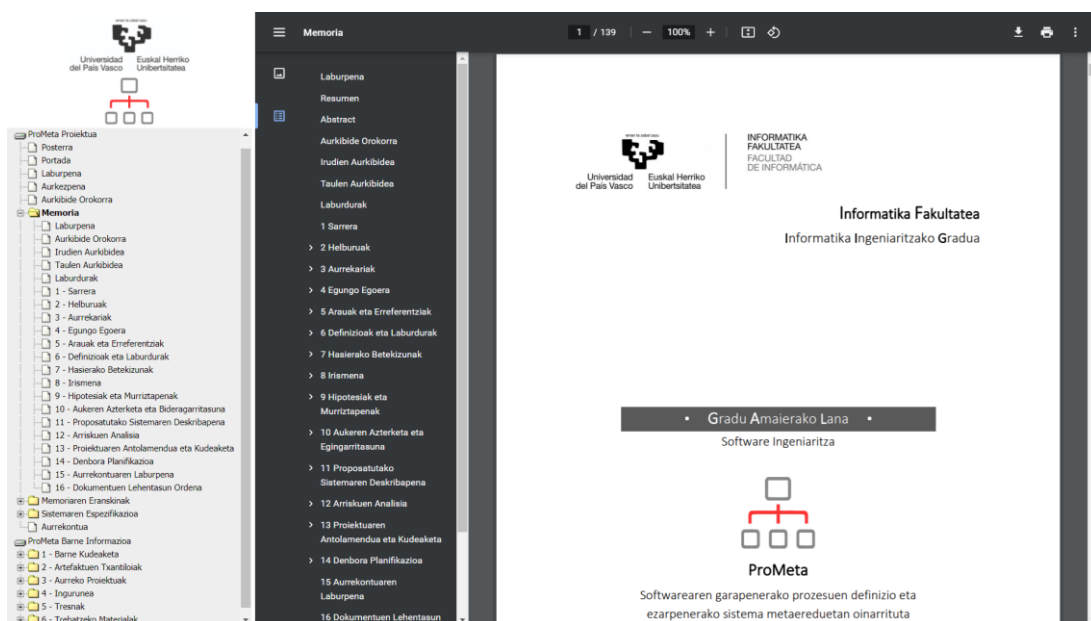
Dokumentu honetan ProMeta proiektuaren inguruko aspektu guztiak azalduko dira. ProMeta izena prozesu edo profesional eta metaeredu hitzetatik dator. Portadan ikusten den proiektuaren logoak metaeredu bat irudikatzen du. Proiektuaren izenburua **Softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema metaereduetan oinarrituta** da. Beraz, proiektuak bi helburu nagusi ditu: garapenerako prozesuen definizioa eta prozesuaren ezarpena.

Proiektua OpenUP metodologia erabiltzaile gauzatuenez, bertako pausoak jarraitu dira. Gainera, dokumentazio formala eta profesionala lortzeko asmoz, ezaguna den CCII-N2016-02 arau estandarria eta Informatika Fakultateko GrAL eredua erabili dira memoriaren eta proiektuaren webgunearen atalak definitzeko.

Proiektuaren webgunea honakoa da: <https://juletx.github.io/ProMeta/>. Webgune honen helburua ProMeta proiektuaren dokumentazio guztia biltzea eta proiektuko bezero zein interesdunek eskura izatea da. Memorian atal batzuetan webgunean dauden eranskinak aipatuko dira.

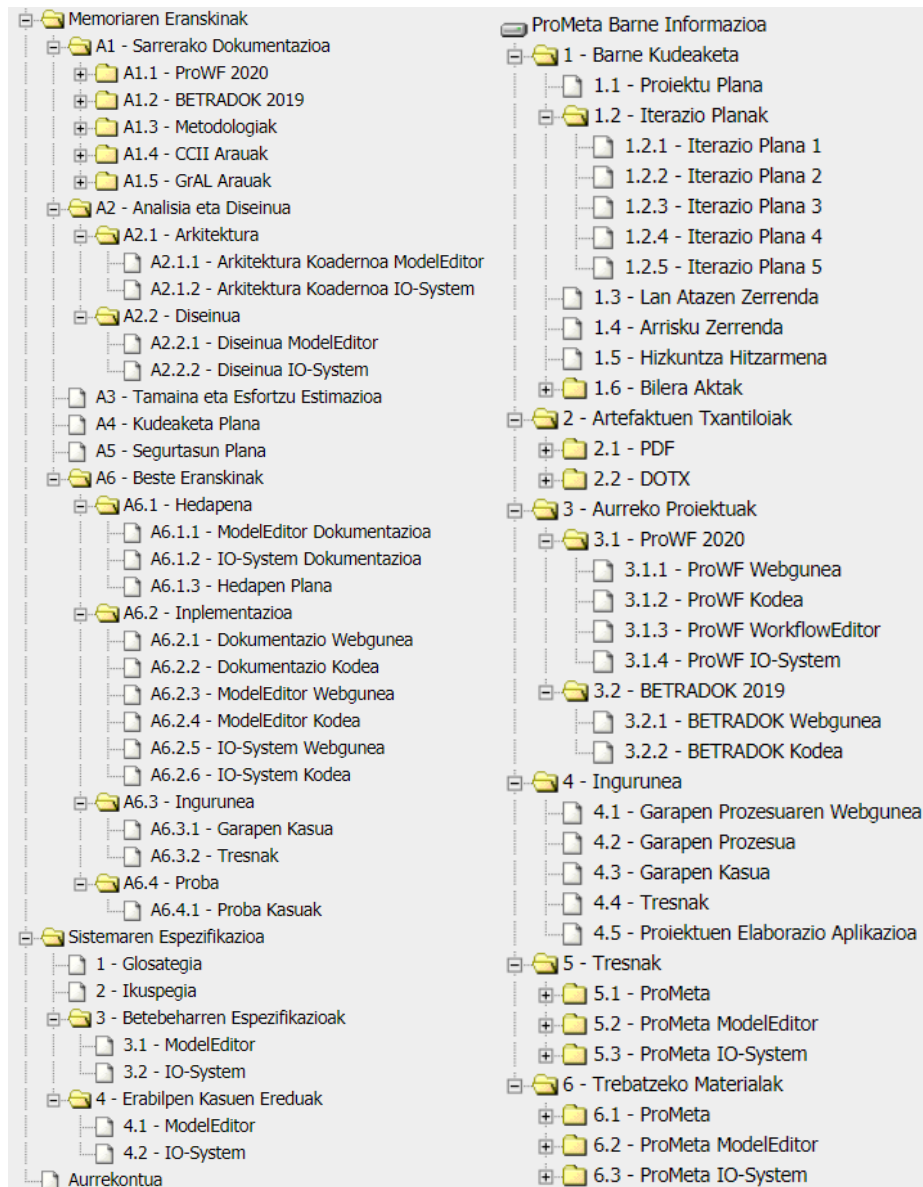
Webgunea bi zatitan dago banatuta, ezkerrean nabigazio menua dago eta eskuinean nabigazio menuan aukeratutakoa agertuko da. Ezkerreko menua erabiliz, proiektuko edozein dokumentu ikustea lortu daiteke: memoria, eranskinak, posterra, barne kudeaketarako dokumentuak etab. Gainera, webguneak CCII-N2016-02 estandarria betetzen duenez, bezero aldeko teknikari edo ebaluatzaileak oso azkar identifikatuko du behar duen dokumentua. Behin menuko aukera batean klikatzen denean, zati nagusian PDF formatu gisa irekiko da dokumentua.

Adibidez, 1.1. Irudian memoria bistaritzen da eskuineko aldean. Memoriako ataletara nabigatu daiteke menuko aukerak erabiliz edo PDF dokumentuko menua erabiliz. PDF dokumentuko menuan maila guztietako izenburuak agertzen dira, aurkibide orokorrean bezala.



1.1. Irudia. Proiektuaren webgunean memoria bistaratu.

Esan bezala, memoriak gain dokumentu eta esteka asko daude webgunean. 1.2. Irudian menuko dokumentu gehienak ikus daitezke. Ezkerrean memoriaren eranskinak, sistemaren espezifikazioa eta aurrekontua daude, CCII arauak eta OpenUP metodologiak zehazten dituenak. Eskuinean proiektuaren barne informazioa agertzen da, bezeroari ez zaiona bidaliko. Bertan proiektuaren barne kudeaketa, txantiloak, aurreko proiektuak, ingurunea, tresnak eta trebatzeko materialak daude.



1.2. Irudia. Webguneke eranskinen eta barne informazioaren menuak.

2 Helburuak

Izenburuak dioten moduan, ProMeta **Softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema metaereduetan oinarrituta** da. Beraz, proiektuak bi helburu nagusi ditu: garapeneko prozesuen definizioa eta prozesuaren ezarpena.

Lehenengo helburuan, softwarearen garapeneko prozesuak definitzeko metaeredu bat definitu beharko dugu eta gutxienez OpenUP metodologiaren eredua. Eredua aldatu ahal izateko, editore grafiko bat eta testu editore bat sortuko ditugu. Bi editoreen arteko bateragarritasuna bermatu beharko dugu, edozein momentutan bien artean aldatu ahal izateko.

Bigarrengean, metodologiaren ereduaren informazioa erabiliz prozesua ezartzen duen webgune bat sortu beharko dugu. Horrek garapen taldeari prozesua jarraitzen lagunduko dio, bakoitzak uneoro egin behar duena argi utziz. Webguneak prozesuen informazioa eta proiektuena bistaratu eta aldatzeko aukera emango du, garapen prozesurako behar den informazio guztia bateratuz.

Enpresa edo garatzaileen ikuspegitik bi dira helburuak. Alde batetik, **ekoizpen-prozesu sistematiko bat izatea**, ezinbestekoa etengabeko hobekuntza gauzatu eta kalitatezko produktuak sortzeko. Bestetik, **ekoizpen-prozesu hori sistematizatzeko baliabideak metodologia, arau eta estandarretatik ateratzea**. Beraz, ekoizpen-prozesuko artefaktuak berrerabili ahal izateko azpiegitura teknologiko bat sortu beharko dugu.

Informatika Ingeniaritza Gradu ikasle bezala, Software Ingeniaritzan espezializatuta, lan honen egilearen helburua proiektuaren motibazioarekin bat dator: software garapenerako garrantzitsuak diren aspektuak bereganatzea. Adibidez, gaur egungo bezeroen eskakizun ez-funtzionalak betetzea, softwarearen kalitateari dagozkionak.

Proiektuaren helburuak hobeto ulertzeko, komenigarria da hiru aspektu zehaztea: arazoa, produktua eta interesatuak. Ondorengo azpiataletan deskribatuko dira horiek taulen bidez.

2.1 Arazoaren Planteamendua

Proiektuaren testuingurua ulertu ahal izateko, lehenik konpondu nahi dugun arazoa ulertu behar da. 2.2. Taulan agertzen den informazioa jakin behar dugu: arazoa zein den, nori eta nola eragiten dion eta gure soluzioa zein den.

Arazoaren Deskribapena	Proiektu informatikoen elaboraziorako ezagutza erabilgarria pilatzen da zenbait jardura modu automatikoan egiteko, adostutako gida bati jarraituz. Gida hauek softwarea garatzeko prozesuetan, metodologietan eta estandarretan oinarritzen dira. Beharrezkoa da gida hauek definitu eta kudeatu ahal izatea, mantendu eta hobetu ahal izateko. Horrek giden edukia eta horiek exekutatzen dituen sistema definitzea eskatzen du. Honetan oinarrituta bezeroen CMMI eskaerak bete daitezke.
Interesatu Kaltetuak	Proiektu informatikoak garatzen dituzten erakundeak.
Arazoaren Eragina	Estandarizazio maila baxua software proiektuen garapen prozesuan. Desadostasunak, atzerapenak eta akatsak eragiten ditu honek.
Soluzioaren Abantailak	Softwarea garatzeko prozesua definitu, kudeatu, mantendu eta hobetzeko erraztasuna. Garapen denbora asko gutxituko du eta bizi-ziklorako metodologia bat erabiliz estandarizazio-maila handituko du.

2.1. Taula. Arazoaren Planteamendua.

2.2 Produktuaren Planteamendua

Arazoa ulertu ondoren, produktuaren inguruko planteamendua azalduko dugu 2.2. Taulan. Bezeroa eta beharra, produktua, alternatibak eta horiekiko abantaila eta hobekuntzak zein diren jakin behar dugu.

Bezeroa	Software proiektuak garatzen dituen enpresa, domeinuko bezeroen epaien menpe dagoena.
Beharra	Softwarea garatzeko prozesuaren bizi-zikloa definitu eta exekutatzeko duen sistemaren beharra, prozesua mantendu eta hobetu ahal izateko.
Produktua	ProMeta: Softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema metaereduetan oinarrituta.
Abantaila	Softwarea garatzeko prozesua definitu, kudeatu, mantendu eta hobetzeko erraztasuna.
Alternatiba	ProWF: Software proiektuen elaboraziorako workflowetan oinarritutako sistemaren sorkuntza eta bizi-zikloa definitzeko metodologia baten ezarpena. Ez ditu OpenUp bizi-zikloaren fase guztiak, hasiera eta elaborazio fasearen zati bat bakarrik.
Hobekuntzak	ProMeta sistemak metaereduak erabiltzen dituzenez erraztasuna ematen du aldaketak egiteko. Etorkizunean sistemaren atal bat aldatzea erabakitzen bada, metaeredua edo ereduak aldatzea nahikoa da. OpenUp metodologiaren bizi-zikloa osatzea du helburu. IO-System sistemaren editorea hobetzea ere aurreikusten da. Gainera, sistema zerbitzari batean jarriko da, edozein erabiltzailearentzat eskuragarri egon dadin.

2.2. Taula. Produktuaren Planteamendua.

2.3 Interesatuen Deskribapena

Arazoa eta produktuaren planteamendua jakinik, proiektuaren interesatuak eta horien erantzukizunak deskribatzen dira 2.3. Taulan.

Izena	Deskribapena	Erantzukizunak
Software Enpresa	Software proiektuak sortu eta sortutako proiektuaren partaideak aukeratzeko dituen.	Proiektua sortu Partaideak aukeratu
Administratzailea	Sistema kudeatzeaz arduratzen da.	Erabiltzaileen kontuak kudeatu Sisteman aldaketak egin
Proiektuko Kudeatzailea	Software proiektuaren liderra da, interesdunen arteko harremanak koordinatzen ditu, eta proiektuko taldea helburuetan enfokaturik egotea du ardura.	Proiektuaren ardurak eta rolak esleitu Prozesuaren gida automatikoak jarraitu
Proiektuko Analista	Software proiektuaren eskakizunak eta betebeharrak harrapatzen ditu.	Proiektuaren eskakizunak eta beharrak eskuratu Prozesuaren gida automatikoak jarraitu
Proiektuko Prozesu Ingeniaria	Software proiektuaren eskakizunak eta betebeharrak aztertuz, prozesuak sortzen ditu.	Prozesua sortu/editatu, metaeredu eta ereduak aldatzeko eginez. Prozesuaren gida automatikoak jarraitu
Proiektuaren beste taldekideak	Software proiektuaren arkitektoak, garatzaileak, testerrak...	Prozesuaren gida automatikoak jarraitu

2.3. Taula. Interesatuen deskribapena.

3 Aurrekariak

Kapitulu honetan proiektua burutu ahal izateko garrantzia izan duten iraganeko elementu esanguratsuak jasotzen dira. ProMeta proiektua ProWF proiektuaren jarraipena denez, aurrekariak antzekoak dira eta aurreko proiektuan oinarrituta daude.

Hasteko, software kalitatea eta gaur egungo bezeroen eskakizun gogorrak aztertuko dira. Ondoren, bezeroen eskakizunengatik agertu diren beharrak azalduko dira: softwarea garatzeko metodologiak eta proiektuak aurkezteko araua. Hori azaldu eta gero, *BPM (Business Process Management)* sistemek eskaintzen dituzten abantailak eta sortzen dituzten menpekotasunak aztertuko dira. Jarraian, inferentzia motorrak eta haien ahalmena komentatuko da, *CLIPS* eta *EHSIS* inguruneak aztertuz. Segidan, *Ereduek Bideratutako Ingeniaritza (MDE)* gauzatzeko softwarea garatzeko metodologia azalduko da eta *Edukiak Kudeatzeko Sistemak (CMS)* softwareak deskribatuko dira. Amaitzeko, ProWF sistemaren ezaugarriak aztertuko dira, aurretik azaldutako kontzeptuak kontuan hartuz.

3.1 Software Kalitatea

Softwarearen kalitatea bezeroen beharrak asetzen dituen softwarearen ezaugarriak kontrolatzean eta ziurtatzean datza, bezero eta erabiltzaileen onurarako. Softwarea produktua ez da fabrikatzen eta ez da fisikoki degradatzen, baina garapen-prozesu bat du. Hala ere, softwarea ez da ia inoiz perfektua izaten, asebate behar den bezeroaren arazoetan ezaugarri berriak agertzen direlako. Proiektu ororen helburua ahalik eta kalitate oneneko softwarea ekoiztea da, bezero eta erabiltzaileen itxaropen eta betekizunak gaindi ditzan. Softwareak akatsak eta gorabeherak izan ditzake, baina ez dira edozein ekipo fisikoren antzekoak, horietan ausazko hutsegiteak eman baitaitezke. Softwarearen kasuan, denak sistematikoak dira eta, ondorioz, zuzendu beharrekoak.

Softwarearen ziurtagiria bere kalitatea ziurtatzeko prozesuaren ondorioa da, baina ez da inoiz azken helburua. Softwarearen kalitatea ez da ziurtatzen, bermatu behar direnak kalitatezko softwarea eraikitzekeko prozedurak dira. Prozedurek, bezeroek eskatutako kontsentsu maila altuko nazioarteko estandarretan oinarritutakoak izan behar dute eta, herrialde bakoitzean, normalizazio agentzia ofizialaren berariazko lan-taldeak egokitutakoak. Prozeduren kalitate-eredu ezberdinak daude, garrantzitsuenak *CMMI* eta *SPICE* dira, helburu berdina dutenak. Kalitate-ereduek, softwarea garatzeko praktika onenak definitzen dituzte, softwarea garatzen duten erakundeek prozesuak hobetzen laguntzeko. Ziurtagiri-emaeleek, praktika on horiek hartu eta beraien emaitza neurgarriak egiaztatzen dituzte garatzaileekin lankidetzan. Ebaluazio-prozesu batean parte hartu ahal izateko inplikatu guztiak ziurtagiri egokiaren jabe izan behar dira.

3.2 Bezeroen Eskakizun Gogorrak

2007ko urriaren 30ean onartutako *Ley de Contratos del Sector Público* legeak, *BOE-261-2007-18874*, bere 69. artikuluan, hornitzaileek kalitate bermeak erakusteko arau europarrei erreferentzia egin eta, dagozkien ziurtagiriak arauarekin bat datozen erakundeek emanak izan behar dutela ezarri zuen.

Ondoren, aurreko legearekin bateratuta, 2011ko azaroaren 14an Estatuko Aldizkari Ofizialean onartu zen legearen ostean, *BOE-A-2011-17887*, bezeroen eskakizunak sendo gogortu ziren, softwarearen kalitateari dagokionez. Lege horren 80. artikulua kalitatea bermatzeko arauak betetzen direla egiaztatzea zuen helburu, horretarako erakunde independenteek emandako ziurtagiriak beharrezkoak ziren merkatuak exijitzen bazituen. Erakunde horiek kalitatea bermatzeko Europako arau jakin batzuei erreferentzia egin

edo baliokideak izan behar ziren. Hortaz, lege horren ostean software garapenean kalitatea bermatzeko *CMMI* edo *SPICE* erakundeei lotutako ziurtagiriak lortzea beharrezkoa bilakatu zen.

CMMI erakunde baten softwarea garatzeko prozesuaren heldutasuna ebaluatzeko eta neurtzeko metodoak erabiltzen dituen prozesua da. Erakundeen heldutasun-maila neurtzeko bost etapa bereizten ditu, 3.1. Taulan ikusten direnak.

Maila	Etap	Prozesuaren egoera
I	Hasierakoa	Kaotikoa da, kontrol zaila du, erreaktiboa da
II	Kudeatuta	Proiektuen ezaugarriak ditu, askotan erreaktiboa da
III	Definituta	Erakundeen ezaugarriak ditu, proaktiboa da
IV	Kuantitatiboki kudeatuta	Erdi-mailako prozesua da, kontrolatuta dago
V	Optimizazioa	Etengabeko hobekuntzan dago

3.1. Taula. CMMI-ren 5 heldutasun-mailak.

*CMMI*ren 3. heldutasun-maila lortzea ezinbestekoa zen merkatuan mantentzeko, hori dela eta, hainbat enpresa kexatu ziren merkatutik kanpo geratzeagatik, baina alferrikakoa izan zen. Horren adibidea, 2016an *OESIA NETWORKS, S.L.* enpresak jarritako erreklamazioa da, *Recurso 0006-2016*, honek *CMMI*ren 3. heldutasun-maila lortzear zegoela erreklamatzan zuen merkatuan mantentzeko, tramitazioaren faltan zegoen. Hala ere, administrazioaren erantzuna errekurtsua baiestearen aurka egotea izan zen, lehen aipatutako 80. artikulua ez betetzeagatik. Arrazoia enpresak oraindik ez zuela eskatutako kalitate maila bermatzen zuen ziurtagirik izan zen eta gainera, maila hori zuten software enpresa asko zeudela.

Ebidentzia horiek direla eta, gaur egun proiektu informatiko baten bezeroek dute pisu handiena edo agintea, haien exijentzia gogorrak betetzea funtsezkoa da merkatuan mantentzeko. Bezeroen behar eta ametsetan fokatzea, 2018ko *CMMI 2.0* bertsioan eta estandar internazionaletan sendotuta geratu da.

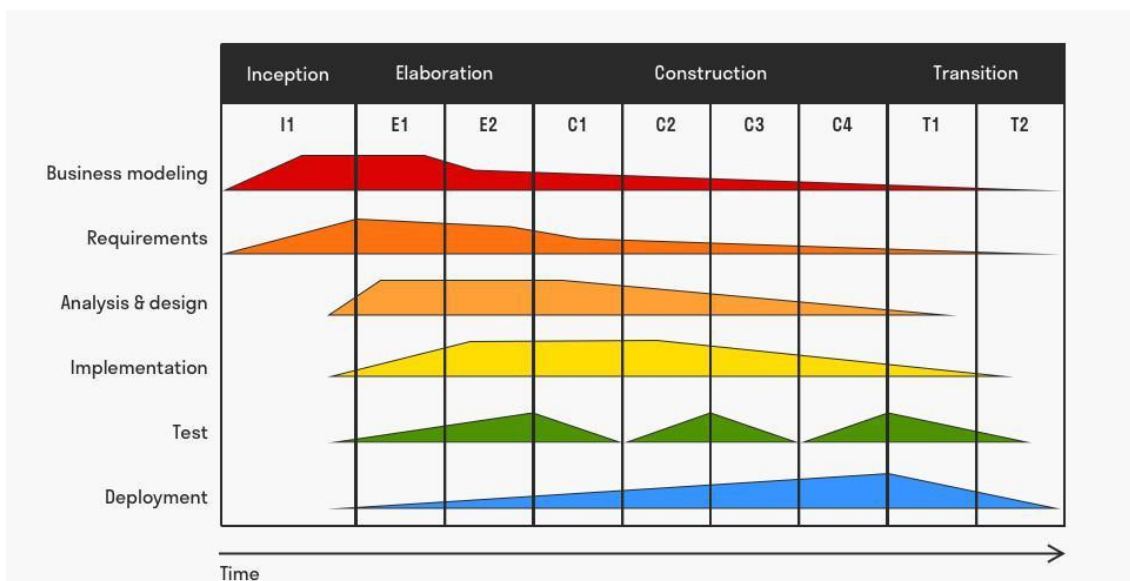
3.3 Garapen Metodologiak

Gaur egun pentsaezina bilakatu da software proiektu bat aurrera ateratzea metodologiarik jarraitu gabe. Software munduan artisautza lanak ez du etorkizunik, metodologia baten ezarpenak hori ekiditeko balio du. Software ingeniariak edozein momentutan jakin behar du zer egin, noiz eta nola, bestela arazoak eta galerak agertzeko probabilitatea handituz joango da proiektuaren garapena luzatzen doan heinean.

Gainera, proiektuak zerotik egitea garestiegia da software garapen enpresentzat. Berrerabilpenean oinarritutako metodologia bat aukeratuz, enpresaren kostu ekonomikoak gutxituko dira epe laburrean.

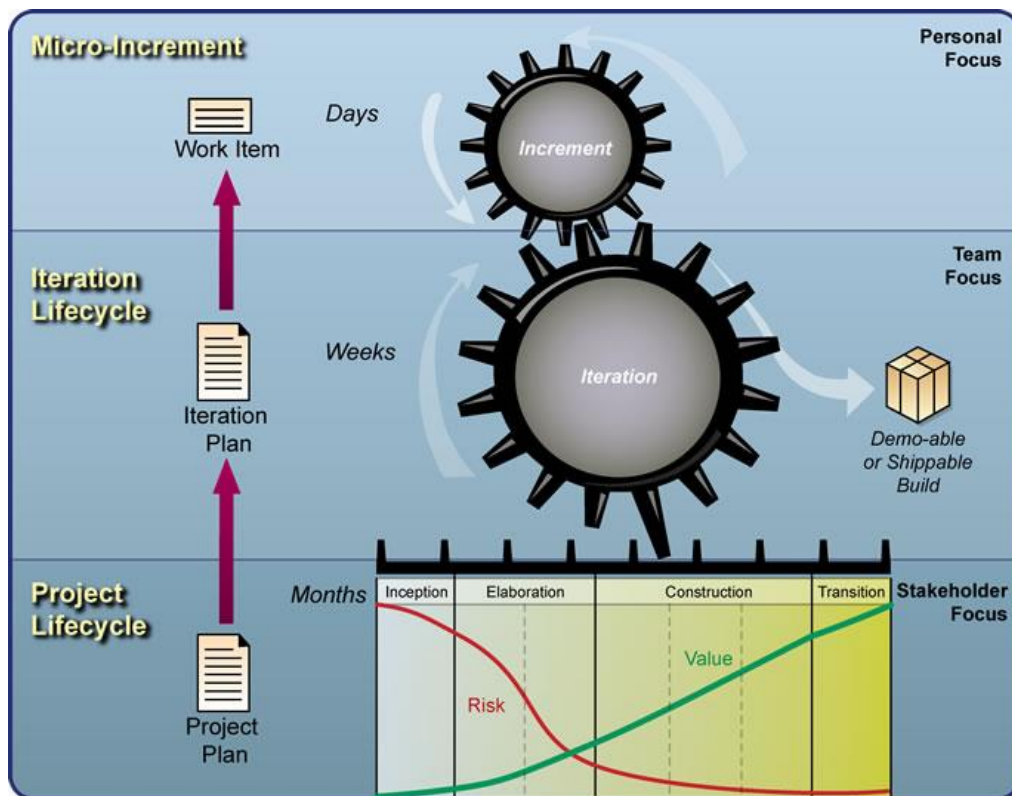
Gehien erabiltzen direnak metodologia arinak edo bizkorrak dira. Arina kontzeptua softwarea sortzeko urratsak arindu behar direlako sortzen da, urrats bakoitzeko prototipo bat sortuz. Giza interakzioetan zentratzen da, aldean arteko elkarrikketa-fluxuari eutsi ahal izateko, garapen dinamikoagoa eta parte-hartzaileagoa ahalbidetzeko. Metodologia bizkorrek garapen-sistema egokitzaile bat erabiltzen dute, eta ez prediktiboa. Horrek esan nahi du lantaldeak buruan duela nahi duen emaitza, baina ez daki zehatz-mehatz zer produktu mota sor dezakeen.

Software garapenaren metodologia arinen artean RUP aurkitzen da. Softwarea garatzeko prozesu iteratibo bat da, Rational Software Corporation erakundeak sortua, IBM enpresaren dibisio bat. 3.1. Irudian ikus daiteke RUP prozesuaren egitura. Hala ere, RUP ez da zehatz-mehatz jarraitu behar den prozesua, baizik eta prozesu moldagarria da, garapen-erakundeen eta software-proiektuen taldeen egokitzeko asmoarekin, bakoitzaren premietarako egokiak diren elementuak hautatuz.



3.1. Irudia. RUPen prozesu iteratiboa, bizi-zikloaren faseak, iterazioak eta jarduerak.

RUPen oinarritutako metodologia simple eta erabiliena OpenUP da. Metodologia horrek RUPen funtsezko ezaugarriak gordetzen ditu, garapen iteratiboa, erabilpen-kasuak, arriskuen kudeaketa eta arkitekturan oinarritutako ikuspegia bultzatzen duten agertokiak barne. RUPen erabiltzen ez diren aukerako zati gehienak baztertu eta elementu asko bateratzen ditu. Eraitza prozesu askoz sinpleagoa da, eta RUP printzipioekiko leiala izaten jarraitzen du. 3.2. Irudian ikus daiteke OpenUP prozesuaren laburpen bat. Proiektuaren tamainaren arabera OpenUPtik RUPera pasatzea komenigarri ikusiko dugu, informazio zehatzagoa gorde ahal izateko.



3.2. Irudia. OpenUP-en geruzak: mikro-gehikuntzak, iterazio bizi-zikloa eta proiektu bizi-zikloa.

3.4 Proiektuen Aurkezpenerako Arauak

Proiektuen garapenean zehar sortzen diren dokumentazio multzoaren aurkezpenerako ezinbestekoa da ezarrita dagoen araudi ofiziala betetzen dela bermatzea, nazioarteko estandarren oinarritutakoa. Helburu nagusia proiektuan parte hartzen duten alderdi guztien aldeko dokumentazioa osoa eta gardena izatea izanik, bezeroaren gogobetetasuna handituz.

Naziortean ISO erakundeak ezarritako estandarrak jarraituz, Espainian araurik finkatuenak *UNE* eta *CCII* erakundeek aurkeztutakoak dira: **CCII-N2016-02. Norma Técnica para la realización de la Documentación de Proyectos en Ingeniería Informática**. Zehatz-mehatz deskribatzen du nola egin behar den Informatika Ingeniaritzako proiektu baten dokumentazioa. Proiektuko dokumentazioa egiteko ereduak eta dokumentazioaren antolamendua deskribatzen du ere. Erreferentziazko nazioarteko esparru eta estandarrak kontuan hartzen ditu. Hala nola, *UNE 157801:2007* – Informazio-sistemen proiektuak egiteko irizpide orokorrak, *UNE-ISO 21500:2013* – Proiektua zuzentzeko eta kudeatzeko jarraibideak eta PMIren *PMBOK* – Proiektuen Zuzendaritzarako Oinarrien Gida. Proiektu baten bisa edo ikuskaritzak egiteko **CCII-N2016-01** araua ere eskuragarri dago.

Proiektuak aurkezteko araudien gabeziak proiektu informatiko batean gatazkak ekarri ditzake. Hau da, proiektuan esku hartzen duten aldeentzat nahi ez diren ondorioak eragin, bezero, hornitzaile zein interesdunen arteko gatazkak sortuz. Epaitegietan bukatzea oso kaltegarria da edozein software enpresentzako.

3.5 Prozesuetan Oinarritutako Garapena

Business Process Management (BPM) negozio-prozesuak definitzera eta aplikatzera bideratutako metodologia edo ikuspegi estrategiko gisa ikus daiteke. Negozio-prozesuak konplexuak eta dinamikoak direnez malguak izan behar dute, negozioa aldatzen joango delako eta etengabe eguneratu behar delako.

Hona hemen BPMrekin lan egitean lortzen diren abantailak:

- Ataza errepikakorrak ezabatzea eta automatizatzea.
- Eraginkortasuna handitzea: prozesuetan akatsak minimizatuz, itxarote-denbora murriztuz, giza esku-hartzeak murriztuz eta lana berregitea saihestuz
- Negozio-arauak betetzen direla ziurtatzea.
- Zerbitzu-maila bermatzea, salbuespenak maneiatuz, egoeren jarraipena eginez, gertakariak mailakatuz, prozesuen sendotasuna eta trazabilitatea bermatuz, etab.
- Lan egiteko modua aldatzeko aukera eskaintzea, inpaktua murriztuz eta etengabe hobetuz.

Gartner eta *Forrester* aholkularitza-enpresek BPM merkatua ikertzeko, urtero, txosten bat argitaratzen dute. Txosten horiek patentatutako datu kualitatiboak aztertzeko metodoetan oinarritzen dira, merkatuaren joerak frogatzeko, hala nola zuzendaritza, heldutasuna eta parte-hartzaileak.

Gartner-en Koadrante Magikoa industria teknologikoaren azpisektore nagusiei buruz egiten duen merkatu-ikerketari bati erantzuten dio. 3.3. Irudian lau koadrantetan banatutako grafiko gisa aurkezten da. X ardatzak exekutatzeko gaitasuna adierazten du, eta Y ardatzak, berriz, balio-proposamen osoa. Ezkerretik eskuinera eta goitik behera, sektore bakoitzeko enpresak honako koadrante hauetan kokatzen dira: *challengers*, *leaders*, *niche players* eta *visionaries*. Antzeko informazioa aurkitu dezakegu 3.3. Irudian agertzen den *Forrester Wave* txostenean. Bi txosten ezagun hauek, ondo kokatutako hornitzaileek komertzialki erabiltzen dituzte.



3.3. Irudia. BPM Gartner-en Koadrante Magikoa eta Forrester Wave txostena.

Bizagi bi txostenetan agertzen da, ondo kokatuta, gainerako liderrak tamaina handiko enpresa eta erakundeen prozesu kudeaketan fokuratzeko dute. *Bizagi* bezalako teknologia erabiliz arkitektura konplexuko web-aplikazioa inplementatu daiteke, prozesu, erregela eta *workflow*-ak exekutatzeko motorrak integratuak dituen. Apustu hori garestia izan daiteke eta proiektu edo enpresaren menpekotasun teknologikoa handituko sortu. Beste apustu merkeago bat, metodologia eta estandarretara hurbiltzen joateko, teknologia propioa sortzea da, *workflow*-lengoaia propioa sortuz eta lengoaia horretan adierazitako ezagutza exekutatzeko inferentzia motorrak erabiliz.

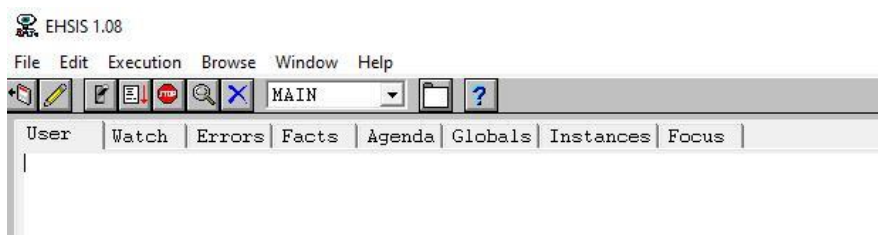
3.6 Ezagutzaren Kudeaketa eta Inferentzia Motorrak

ProWF proiektuan teknologia propioa sortzea erabaki zen, lan-fluxu edo *workflow*-en bidezko lengoaia propioa sortuz eta berekin adierazitako ezagutza inferentzia motorren bidez exekutatu.

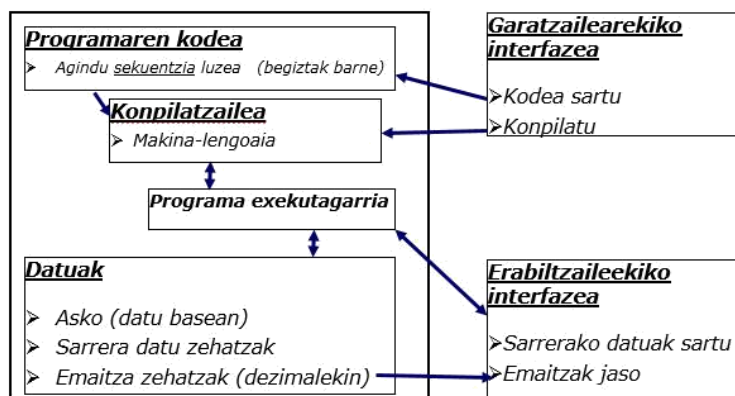
CLIPS softwareak inferentzia motor baten erabilera errazten digu, sistema adituak ekoizteko eta exekutatzeko garapen-ingurune bezala erabili daiteke. *NASAK* sortu eta erabilia jabari publikoan utzita dago. Bere lengoaiak erregelak, objektuetara bideratutako programazioa eta programazio prozedurala erabiltzen du ezagutza adierazteko. Lengoaia sinplea baino oso ahaltsua da, hurrengo ezaugarrietan nabarmenduz:

- **Garraigarritasuna.** C lengoaiak ematen diona.
- **Integrazioa edo zabalkortasuna.** Programazio prozeduralari esker funtzio berri ahaltsuak sor daitezke. *Service-oriented Architecture* (SOA) estandarra erabilia BPM edo beste edozein sistemekin integragarria da.
- **Interakzio edo diziplinarteko garapena.** Formakuntza ezberdinetako pertsonen ideiak azkar inplementatzeko aukera, objektu eta erregeletan oinarritutako lengoaia erabiliz.
- **Egiaztapen edo balidazio errazagoa.** Horretarako, funtzio bereziak erabiliz.

EHSIS, ordea, Euskal Herriko Unibertsitateko (EHU) *ERABAKI* taldeak hedatutako ingurunea da, *CLIPS* 6.04, *FuzzyCLIPS* 6.04, objektuetara eta gertaeretara bideratutako programazioa, interfazeen garapena, komunikazioa eta leihoetan oinarritutako ingurunea integratzen duena. *EHSIS* inguruneak (ikusi 3.4. Irudia), softwarea garatzeko baliabide tradizionaletan oinarritutako aplikazioak sortzeaz gain (ikusi 3.5. Irudia), *COOL* lengoaia erabilia, ezagutzan oinarritutakoak ere sor ditzake sistemaren arkitektura egoki batekin baliatuz.

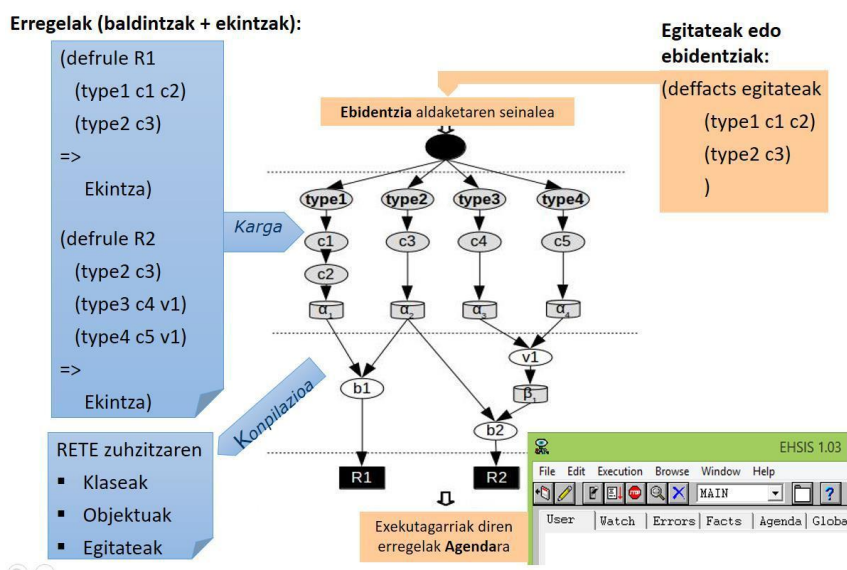


3.4. Irudia. EHSISen garapen ingurunea.



3.5. Irudia. Programazio tradizionalaren eskema.

Erregelak idazteko lengoai sinplea denez, klase eta objektuak erabilia ere, domeinuko ezagutza adierazteko ondo diseinatutako lengoai bat lortuz gero, ezagutza hori exekutatzeko kodea automatikoki sortzea ere posible da. Gainera, inferentzia motorrarekin abiadura handiko exekuzio eraginkorra lortu daiteke kode guztia *RETE* sare bezala gordeta (ikusi 3.6. Irudia), hau konpilazio mota bat bezala erabili daiteke. *RETE* sarearen egitura bereziak *RETE* algoritmoaren abiadura, egitate eta erregela askorekin ere, aprobetxatzeko aukera ematen du. Erreminta hau, bere eraginkortasuna eta jabari publikoa dela eta, konplexutasun handiko proiektuetan erabili daiteke produktu mantengarriak lortzeko.



3.6. Irudia. Erregeletan oinarritutako sistemen eskema: klaseak, objektuak, erregelak eta egitateak.

EHSISen garapen inguruneak badu bere produkzio bertsioa, *EHSIS_RT* deitutakoa. Web-zerbitzuetaarako bertsioa ere badu, *Mairi* deitutakoa.

3.7 Ereduek Bideratutako Ingeniaritza

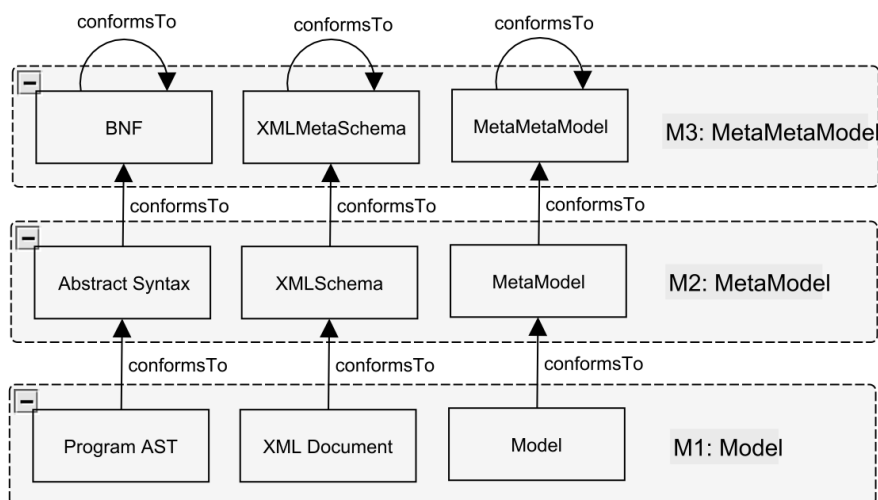
Model Driven Engineering (MDE) edo eredu bidezko ingeniari-tza softwarea garatzeko metodologia da. Domeinu ereduak erabiltzen ditu, hau da, arazo zehatz bati lotutako gai guztien eredu kontzeptualak. Hori dela eta, aplikazioen domeinu jakin baten ezagutzaren eta jardueren irudikapen abstraktuak nabarmentzea du helburu, kontzeptu informatikoetan sartu gabe.

MDEren helburua produktibitatea handitzea da. Horretarako, sistemen arteko bateragarritasuna maximizatzen du eredu estandarizatuak berrerabiliz, diseinu prozesua sinplifikatu aplikazioaren domeinuko diseinu patroia errepikakorren eredu bidez eta sisteman lan egiten duten pertsonen eta taldeen arteko komunikazioa sustatu praktika onen estandarizazioaren bidez.

MDEren modelatze paradigma bat eraginkorra dela deritzo bere ereduak domeinua ezagutzen duen erabiltzaile baten ikuspuntutik zentzua badute eta sistemak ezartzeko oinarri gisa balio badute. Ereduak produktuen kudeatzaileen, diseinatzaileen, garatzaileen eta aplikazioaren domeinuko erabiltzaileen arteko kolaborazioarekin garatzen dira. Ereduak amaitzen doazen heinean, softwarea eta sistemak garatzea ahalbidetzen dute.

MDEn 4 modelatze maila daudela esan dezakegu. Maila handiagoa den heinean abstrakzio-maila igotzen doa. 3.7. Irudian ikus daitezke 3 abstrakzio-maila altuenak: eredu, metaeredua eta meta-metaeredua.

- **M0. Instantziak.** Maila hau exekuzio sistemari dagokio. Maila honetan negozioko elementuak daude, edo mundu errealeko elementuen errepresentazioak (software errepresentazioak).
- **M1. Eredua.** Eredu honek software sistemen itemak adierazten dituzten kontzeptuak dauzka. M1 mailan dauden kontzeptuak M0 mailan dauden instantziak kategorizatu edo sailkatzen dituzte.
- **M2. Metaeredua.** M1 mailan dauden kontzeptuen inguruan arrazoitze beharrezkoak diren kontzeptuak dauzka. M2 mailako elementu batek M1-eko elementuak espezifikatzen ditu. Ereduen eredu honi metaeredu esan ohi zaio.
- **M3. Meta-metaeredua.** M2 mailan dauden kontzeptuen inguruan arrazoitze beharrezkoak diren kontzeptuak dauzka. M3 mailako elementuek M2-ko elementuak kategorizatzen ditu. Meta-metaeredu esan ohi zaio.



3.7. Irudia. MDEren 3 abstrakzio maila: eredu, metaeredua eta meta-metaeredua.

3.8 Edukiak Kudeatzeko Sistemak

Content Management System (CMS) izeneko softwareak dokumentuak eta bestelako edukiak antolatu eta kudeatzeko erremintak dira, webguneak eta web edukiak kudeatzeko balio dutenak. Gaur egun sistema asko daude arlo honetan, bai kode irekikoak eta baita jabedunak ere.

Edukiak kudeatzeko sistema zerbitzarian kokaturik egon ohi da. CMS batera atzipena erabiltzaile motaren arabera maila ezberdinetan egin daiteke. Adibidez, edukiaren sortzaileek sistema osatuko duten dokumentuak sortuko dituzte. Argitalpen-teknikariek dokumentu hauek aipatu, onartu edo baztertuko dituzte. Editoreen burua gure web orrian eduki hori argitaratzeaz arduratuko da. Dena interfaze grafiko baten bidez kontrolatuko da, honen erabilera erraztu ahal izateko.

ProWF eta ProMeta proiektuen IO-System sistema garatzeko Drupal CMSa erabili da.

3.9 ProWF Sistema

Esan bezala, ezinbestekoa da ProWF sistema [1] ulertzea proiektu hau horren jarraipena delako. Horretarako, bertan zehazten diren rola, sistemaren azpisistemak eta osagaiak deskribatuko dira.

ProWF sistemaren erabiltzaileek hurrengo rol hauekin lan egin behar dute:

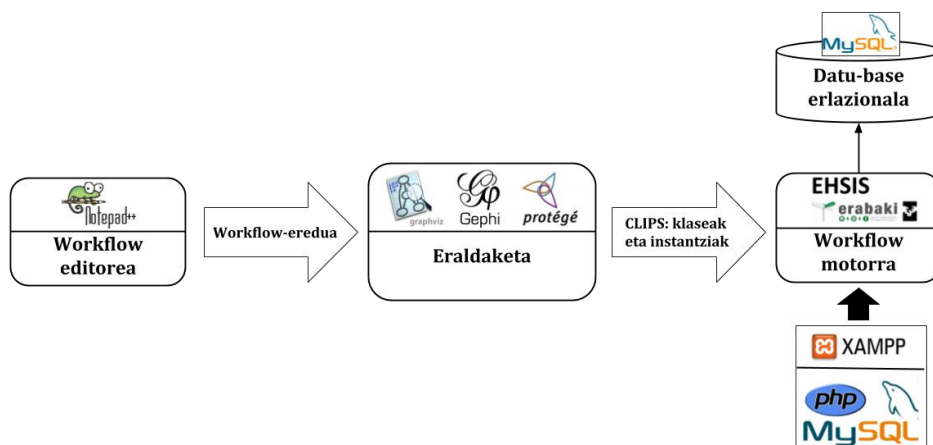
- **Prozesu Sortzailea:** sortutako workflow-lengoaia baliatuz, softwareen garapenerako bizi-zikloa ezartzen duen metodologia definituko duena.
- **Analista:** sortutako workflow-ereduak adierazten duen bizi-zikloa exekutatzeaz eta aplikatzeaz arduratuko da.
- **Proiektuko Zuzendaria:** proiektuak sortu eta proiektuko partaideen rola esleitzeko ditu.
- **Kalitate Arduraduna:** workflowaren exekuzioan zehar sortuko diren artefaktuen kalitatea bermatzea du helburu, balorazioak emanez.

ProWF sistema bi azpisistema ezberdinetan bananduta dago:

- **Workflow Editor:** sortutako workflow-lengoaia erabiliz workflow-ereduak sortzeko balio du, ondoren, eredu horiek workflow motorrak exekutatzeke eraldatuko ditu. Testu-editore bat izango da. Rolak: Prozesu sortzailea.
- **IO-System:** workflow kudeatzaileraren sarrera/irteerak maneiatzea du helburu. CMS baten bitartez kudeatutako web-aplikazioa izango da. Rolak: Proiektuko zuzendaria, analista eta kalitate arduraduna.

3.8. Irudian ikusten den moduan, hauek dira **Workflow Editor** azpisistemaren arkitekturaren osagaiak:

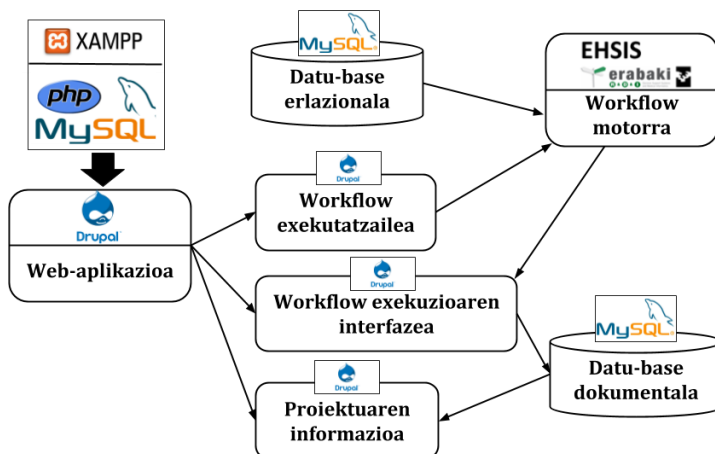
- **Workflow editorea:** definitutako lengoaia grafiko bat erabilita, testu editore batekin, DOT lengoiaren bitartez workflow-ereduak sortzeko balio du.
- **Eraldaketa-prozesua:** DOT lengoian idatzitako workflow-eredua CLIPS lengoiako klase eta instantzietan eraldatzea du helburu.
- **Datu-base erlazionala:** sortutako workflow-ereduen informazio gordetzeaz arduratzen da.
- **Workflow motorra:** CLIPS lengoiara eraldatutako workflow-ereduaren klaseak eta instantziak baliatuz, workflowa exekutatzeke erregela batzuen bidez, informazio guztia datu-base erlazionalan gordeko du.



3.8. Irudia. ProWF Workflow Editor azpisistemaren arkitektura.

3.9. Irudian ikusten den bezala, hauek dira **IO-System** azpisistemaren arkitekturaren osagaiak:

- **Web-aplikazioa:** CMS baten bitartez, lokalean erabiltzeko (zerbitzari batean ere kokatu daiteke) pentsatuta dagoen sistema kudeatzailea. Hurrengo osagai nagusiak integratuko ditu:
 - **Workflow exekutatzailea:** workflow motorrari aginduak emango dizkio eta ondoren, bere irteerak jasoko ditu.
 - **Workflow exekuzioaren interfazea:** erabiltzaileak exekuzioan zehar ikusiko duena da, workflow motorraren irteeran adierazitako ekintzak betetzeko interfazea da.
 - **Proiektuaren informazioa:** CCII-2016N-02 araua jarraituz, erabiltzailearen dokumentuak antolatuta dituen webgune antzeko bat da.
- **Workflow motorra:** Datu-base erlazionalari kontsultak eginez, workflow-eredua prozesatuko du eta jardueraz-jarduera joango da egin beharreko ekintzak irteera lez bueltatuz.
- **Datu-base erlazionala:** sortutako workflow-ereduen informazioa gordetzeaz aparte, erabiltzaileen proiektu eta artefaktuen informazioa biltegitratuko du.
- **Datu-base dokumentala:** erabiltzaileek idatzitako dokumentuen kokapena eta sekzio bakoitzaren edukia gordeko du.



3.9. Irudia. ProWF IO-System azpisistemaren arkitektura.

4 Egungo Egoera

Kapitulu honetan, proiektuaren egungo egoera deskribatuko da, ProWF aurreko proiektuaren egoera eta egilearen prestakuntza azalduz.

4.1 Deskribapena

Egungo egoera ulertzeko, ezinbestekoa da proiektu honen aurrekaria den ProWF proiektuaren egoera ulertzea. Proiektuaren izenburuak dioten moduan ProWF sistemak *software proiektuen elaboraziorako workflowetan oinarritutako sistemaren sorkuntza eta bizi-zikloa definitzeko metodologia baten ezarpena* ahalbidetzen du. Sistemak bi osagai nagusi ditu, workflow editorea eta sarrera irteera sistema.

Horretarako, proiektuan zehazten diren abantailak eta desabantailak aztertu eta osatuko ditugu. Gainera, proposatzen diren hobekuntzak ere zehaztuko ditugu, eta gure ideiekin osatu. Izan ere, hobekuntza horietako batzuk aurrera eramango dira ProMeta proiektuan.

4.2 Abantailak eta Desabantailak

ProWF proiektuan sistemaren abantaila eta desabantaila batzuk identifikatu ziren. Osatu egin dira proiektua aztertzerakoan identifikatu diren beste batzuekin. Garrantzitsua da hauek kontuan hartzea ProMeta proiektuaren planteamendua egiteko.

4.2.1 Abantailak

Sistemaren abantailen artean bi mota aurki ditzakegu, workflow-lengoiari lotutakoak eta workflowetan oinarritutako sistemari lotutakoak.

Workflow-lengoiari lotutakoak:

- Sortutako workflow-ereduen irudien nabigagarritasunak garbitasuna eta ulergarritasuna ematen dio prozesuari. Gainera, OpenUP metodologiaren webgunean agertzen diren formak eta koloreak erabiltzen ditu.
- Workflow-eredua aldagarria da, baldin eta sortutako lengoia grafikoa errespetatzen bada. Bestela, workflow-lengoia exekutatzeko erregelak egokitu beharko dira.
- Lengoiak softwarearen bizi-zikloaren ezaugarri esanguratsuenak harrapatzen ditu.

Workflowetan oinarritutako sistemari lotutakoak:

- Interfaze sinple eta intuitiboa du, itxura profesionalarekin.
- Drupal CMSari esker, erabiltzaileen erregistro eta kudeaketa erraza du.
- Funtzionalitate berriak gehitzeko erraztasuna Drupal CMSaren bidez.
- Edukia kudeatzeko erraztasuna Drupal CMSa erabiliz.
- Workflow motorra, workflow-lengoia erabiliz sortutako edozein prozesu exekutatu dezake, prozesuaren objektuak automatikoki sortzen dira eta prozesuak exekutatzeko erregelak berrerabili daitezke.
- Workflow motorraren prozesaketa-denbora asko murrizten da, erabiltzen dituen instantzia eta erregelak RETE sarean "konpilatu" izanari esker.

4.2.2 Desabantailak

Sistemaren desabantailak ere multzo berdinetan sailka daitezke.

Workflow-lengoiari lotutakoak:

- Workflow-eredua nabigagarria denez, hainbat fitxategi eraldatu behar dira CLIPS lengoiako klase eta instantziak sortzeko. Prozesu errepikakor eta neketsua da.
- OpenUP bizi-zikloko workflow eredua eskuz definitu beharra. Horrek prozesua definitzeko denbora asko behar izatea eragiten du. Hobe izango litzateke webguneko informazioa erabiliz automatikoki sortzea.

Workflowetan oinarritutako sistemari lotutakoak:

- Web-aplikazioak lokaleko instalazioa behar du. Zerbitzari batera eraman daiteke eta horrela instalazio prozesua asko murriztuko litzateke, bakarrik Workflow Editor azpisistemaren osagaiak instalatuz.
- Drupalen bidez sortutako web-aplikazioak ez ditu erantzun azkarrak ematen. Gunearen orrialdez aldatzean kargatu behar diren modulu eta beste aspektuek errendimendua murrizten diote.
- Elaboratzen den proiektuaren webguneko editorea testu soilean dagoenez, ez du aukerarik ematen formatua emateko. Adibidez, ezin da letra lodia erabili eta ez dago taulak betetzeko aukerarik.
- Webgunean OpenUP artefaktuen informazioa betez ez dira beraien txantiloien formatuko dokumentuak lortzen.
- Drupaletik sortzen diren datuak kanpoko datu-base baten daude. Horrek Drupalek eskaintzen dituen aukerak murrizten ditu, datuak bistartzeko eta editatzeko aukerak, adibidez.

4.3 Proposatutako Hobekuntzak

ProWF proiektuan hobekuntza interesgarri asko proposatzen ziren. Garrantzitsua da horiek kontuan hartzea ProMeta proiektua haren jarraipena baita. Gainera, hobekuntza posible gehiago ere gehitu ditugu aurrekoak osatzeko.

ProWF proiektuan hurrengo hobekuntzak proposatzen dira etorkizunerako:

- OpenUP bizi-zikloko workflow eredua amaitu eta ahal bada, hobetu. Prozesuan gelditzen diren faseak gehitu eta bigarren fasea (*elaboration*) guztiz definitu. Horretarako, “*Workflow-lengoiaren Eskuliburua*” eta “*Workflow Editor – Eskuliburua*” dokumentuak jarraituz.
- *Workflow*-ereduen eraldaketa-prozesu errepikakorra ekiditeko metaeredu bat definitzea, DOT lengoia deskriptiboaren eta COOL lengoiaren arteko eredu bat sortuz, urrats bakar baten bidez eraldaketa eginez eta kanpoko softwareak (Gephi, Protégé) erabiltzea ekidituz. Produktibitatea, azkartasuna eta mantengarritasuna bilatuz.
- *Workflow*ak kudeatzeko sistema zerbitzari batean jartzea. Zerbitzari batean egonda, erabiltzaileak ez du instalaziorik beharko.
- *Workflow*ak kudeatzeko sisteman, *IO-System* azpisisteman, artefaktuen sekzioak idazterako orduan *HTML* edo *WYSIWYG* motako testu-editore bat inplementatzea. Softwarearen bizi-zikloa definitzen duten metodologia askotan taulak eta Excel orriak bete behar dira, prototipo honetan, ordea, ez dago taulak txertatzeko aukerarik.

- Bezero ezberdinen eskakizunak asetzeko gaitasuna izateko asmotan, metodologia ezberdinak integratzen dituen garapen-prozesuak definitzea.
- Gure enpresak ondo egiten duena garapen-prozesuan sartzea. Hori CMMi 2.0 kalitate-ereduak eskatzen du. Prozesu berriekin integratzeko lanak konplexuak izan daitezke.
- Garapen-prozesua grafikoki adieraztea xehetasun maila handiagorekin eta funtzionalitate gehiagorekin. Lehen fase batean, lan-fluxuen eredua erabiliz eta, bigarren fase batean, BPMN estandarrak definitzen duen lengoaia grafikoa erabiliz, partekatze eta adoste lanak erraztu eta azkartzeko.
- Garapen-prozesua beste metodologia batzuen baliabideekin edo adostasun-maila handiko artefaktuen txantiloiekin aberastea, adibidez, *RUP* metodologia arina.
- CMMI 2.0 kalitate-ereduaren 2. maila lortzeko garapen-prozesua osatzea.
- CMMI 2.0 kalitate-ereduaren 3. maila lortzeko garapen-prozesua osatzea.

Hobekuntza posible gehiago ere identifikatu dira proiektua aztertzerakoan:

- Drupaletik zuzenean inferentzia motorrari deitu ahal izatea, tarteko fitxategirik erabili gabe. Orain sistemak erabiltzaileari esaten dio fitxategi bat exekutatu behar duela.
- OpenUP bizi-zikloko workflow eredua eskuz definitu beharrik ez edukitzea. Horrek prozesua definitzeko denbora asko behar izatea eragiten du. Hobe izango litzateke webguneko informazioa erabiliz automatikoki sortzea.
- Prozesuaren informazioa gordetzeko metaeredua eta ereduak erabiltzea. Metaereduak erabiltze malgutasuna ematen du etorkizunean eraldaketak egiteko komeni den formatura.
- Drupaletik sortzen diren datuak kanpoko datu-base baten gorde ordez Drupalen datu-basean gordetzea. Horrek Drupalek eskaintzen dituen aukerak aprobetxatzen ditu, datuak bistartzeko eta editatzeko aukerak, adibidez.
- Drupal webgunearen itxura hobetu, defektuzko itxura aldatuz. Itxura egoki bat aurkitu webgunerakomenu hedagarriak onartzen dituen.

4.4 Prestakuntza

Proiektu honen egileak bazituen proiektu honetarako erabilgarriak diren hainbat ezagutza, Informatika Ingeniaritzako Gradu hainbat irakasgaitan ikasitakoak. Esaterako, software proiektuen, softwarearen bizi-zikloaren, metodologia zein estandarren oinarritzko ezagutzak.

Softwarearen Kalitatea irakasgaien proiektu honetan interesa duten hurrengo ekintzak jorratu ziren:

- *BPMn* oinarritutako software bat probatu, *Bizagi*. Software horren bidez, prozesuetan oinarritutako web-aplikazioa bat sortu zen. Lehenengo, *Bizagi Modeler* softwarearen bitartez prozesua modelatu, eta ondoren, prozesu horretan oinarritutako web-aplikazioa eraiki zen *Bizagi Studio* softwarearekin.
- *OpenUP* metodologia jarraitzen zuen proiektu bat osatu, softwarearen bizi-zikloa definituz. Ez ziren metodologiako artefaktu guztiak bete, baina bai hasierako fasekoak, betekizunen ingeniaritzari buruzkoak.
- Proiektu bat aurkezteko webgunea sortu eta antolatu *CCII-2016N-02* araua jarraituz.

Softwarearen Garapen Industrialak irakasgaiak beste gai hauek landu ziren:

- Model Driven Engineering (MDE) edo ereduak bideratutako ingeniariaren oinarriko kontzeptuak: metaeredua eta ereduak.
- Domain Specific Language (DSL) edo domeinu zehatzeko lengoaien sorrera.
- ATLAS Transformation Language (ATL) erabilera ereduaren arteko eraldaketak egiteko.
- Eclipse Modelling Framework (EMF) tresnen erabilera MDE-rako.

Web Sistemak irakasgaiak ikasitakoak:

- XAMPP-en erabilera webguneak ordenagailu lokalean garatzeko.
- PHP programazio lengoaiaren erabilera web garapenerako.
- MYSQL-ren erabilera web garapenerako.

Bestetik, egileak prestakuntza zuen erabilgarriak izan diren beste gai batzuetan:

- Git eta GitHub-en erabilera bertsio kontrolerako.
- GitHub Pages-en erabilera webgune estatikoak sortzeko.
- Java eta Eclipse tresnen erabilera.

5 Arauak eta Erreferentziak

Kapitulu honetan, proiektuan zehar erabilitako araudia, bibliografia, metodoak, tresnak, ereduak, metrikak eta prototipoak deskribatuko dira.

5.1 Aplikatutako Legedia eta Araudia

Hainbat esparrutako legedia eta araudia aplikatzen da. Alde batetik, GrALari eta dokumentazioari buruzkoak eta bestetik erlazionatuta dauden administrazio publikoaren legeak.

Gradu Amaierako Lanen inguruko bete beharreko arautegia:

- [UPV/EHuko gradu amaierako lanen araudia](#). Euskal Herriko Unibertsitatean gradu amaierako lana egin eta defendatzeari buruzko arautegia.
- [Informatika Fakultateko gradu amaierako lanen araudia](#). Informatika Fakultateko Gradu Amaierako Lanari buruzko arautegia.
- [BOE-A-2009-12977](#). Informatika Ingeniaritzako Graduko edo Ingeniaritza Teknikoko titulazioak bete beharreko kompetentzia profesionalak eta Gradu Amaierako Lanen izaera profesionala ezartzen duen Errege Dekretua.

Administrazioa Publikoak ezarritako Sektore Publikoko Kontratuen legeak eta aurkeztutako kexak:

- [BOE-261-2007-18874](#). 30/2007 Legea, urriaren 30ekoa, Sektore Publikoko Kontratuei buruzkoa.
- [BOE-A-2011-17887](#). 3/2011 Legegintzako Errege Dekretua, azaroaren 14koa, Sektore Publikoko Kontratuei buruzko Legearen testu bategina onartzen duena.
- [BOE-A-2017-12902](#). 9/2017 Legea, azaroaren 8koa, Sektore Publikoko Kontratuena, Europako Parlamentuaren eta Kontseiluaren 2014ko otsailaren 26ko 2014/23/EB eta 2014/24/EB zuzentarauen transposizioa egiten duena Espainiako ordenamendu juridikora. 93. Artikulua: Kalitatea bermatzeko arauak betetzen direla egiaztatzea.
- [Recurso nº 6/2016 Resolución nº 100/2016](#). Kontratu baliabideen administrazio zentralaren erresoluzioa OESIA NETWORKS, S.L enpresaren errekurtsoari.

Kalitate-eredu eta giden inguruko informazio eta baliabideak:

- [CMMI-DEV, V1.3](#). Improving processes for developing better products and services.
- [CMMI 2.0](#). Capability and performance model. Berarekin batera bakiabide multzo bat dakar:
 - Eredua erraztua (erredundantziak murriztuz) eta azken joera metodologikoak gehituta
 - Ebaluazio metodoa erakundeen maila jakiteko.
 - Prestakuntza ikastaroak dagozkio ziurtagiriak lortzeko.
 - Sistema eta erreminta berriak. Kalitate-eredua online atzitzeko eta ebaluaziok egiteko.
 - CMMI V2.0 hartzeko edo CMMI 1.3-tik migratzeko gidak.
- [PMBOK](#). Project Management Body of Knowledge.
- [SWEBOK](#). Software Engineering Body of Knowledge.

Proiektu honen dokumentazioen antolaketarako eta proiektuaren aurkezpenerako aplikatu den araua CCII-N2016 estandarra da.

- [CCII-N2016-01](#). Ingeniaritza informatikoko proiektuen ikuskaritza edo bisa egiteko araua CCII N2016-01 estandarra da. Estandarrak dokumentuen osotasuna berrikusteko zerbitzuen prozesua deskribatzen du.
- [CCII-N2016-02](#). Estandar honek ingeniaritza informatikoko proiektuen dokumentazioaren antolaketa eta bere aurkezpena zehazten du. Memoria eta bere eranskinak estandar honen arabera antolatu dira, baita memoriarekin batera entregatu den webgunearen antolaketa.

5.2 Bibliografia

Jarraian, proiektuan zehar informazioa bilatzeko eta datuak lortzeko erabili diren erreferentzia bibliografikoak zerrendatuko dira.

- [1] J. Rojo, «ProWF proiektua: Software proiektuen elaboraziorako workflowetan oinarritutako sistemaren sorkuntza eta bizi-zikloa definitzeko metodologia baten ezarpena. Gradu Amaierako Lana,» 2020. [Online]. Available: <https://juletx.github.io/ProWF/>.
- [2] Eclipse Foundation, «OpenUP: Open Unified Process,» 2012. [Online]. Available: <https://420-gel-hy.github.io/EPF/openup/index.htm>.
- [3] Eclipse Foundation, «ABRD: Agile Business Rules Development,» 2012. [Online]. Available: <https://420-gel-hy.github.io/EPF/ABRD/index.htm>.
- [4] L. Torvalds, «Git Documentation,» [Online]. Available: <https://git-scm.com/doc>.
- [5] Microsoft, «GitHub Documentation,» [Online]. Available: <https://docs.github.com/>.
- [6] Microsoft, «GitHub Pages Documentation,» [Online]. Available: <https://docs.github.com/en/pages>.
- [7] Eclipse Foundation, «Xtext Documentation,» [Online]. Available: <https://www.eclipse.org/Xtext/documentation/index.html>.
- [8] Drupal, «Drupal Documentation,» 2021. [Online]. Available: <https://www.drupal.org/documentation>.
- [9] Apache Friends, «XAMPP Documentation,» [Online]. Available: <https://www.apachefriends.org/docs/>.
- [10] Pantheon, «Pantheon Documentation,» 2021. [Online]. Available: <https://pantheon.io/docs/>.
- [11] CCII, «Norma CCII-N2016-01: Norma de Visado de Proyectos y Actuaciones Profesionales en Ingeniería Informática,» 2016. [Online]. Available: <https://juletx.github.io/ProMeta/Proiektua/Memoriaren%20Eranskinak/A1%20-%20Sarrerako%20dokumentazioa/CCII-N2016-01.pdf>.
- [12] CCII, «Norma CCII-N2016-02: Norma Técnica para la realización de la Documentación de Proyectos en Ingeniería Informática,» 2016. [Online]. Available:

<https://juletx.github.io/ProMeta/Proiektua/Memoriaren%20Eranskinak/A1%20-%20Sarrerako%20dokumentazioa/CCII-N2016-02.pdf>.

- [13] P. Kruchten, «RUP: Rational Unified Process,» [Online]. Available: https://www.researchgate.net/publication/220018149_The_Rational_Unified_Process--An_Introduction.
- [14] J. Legarda, «BETRADOK proiektua: Betekizunen trazabilitate inpaktu-analisi automatikoa eta dokumentazio formalaren sorkuntza automatikoa modeloetan oinarritutako ekosistemetan. Gradu Amaierako Lana,» 2019. [Online]. Available: <https://juletx.github.io/BETRADOK/>.

5.3 Metodoak

Proiektuan hainbat metodo erabili dira, garapena baldintzatu dutenak.

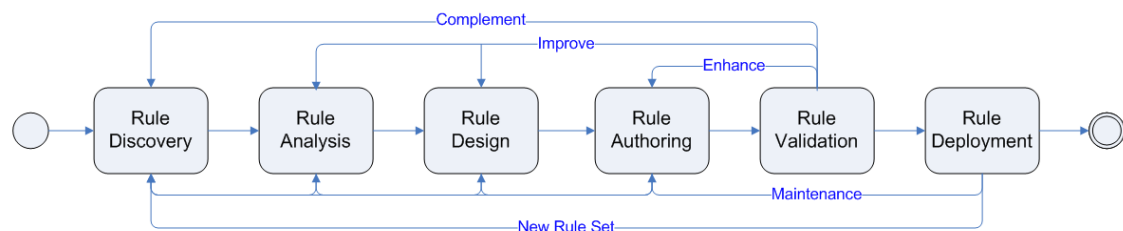
5.3.1 OpenUP

OpenUP softwarea garatzeko metodo eta prozesu bat da, teknologien sektoreko enpresa multzo batek proposatutakoa, zeintzuk 2007an Eclipse Fundazioari dohaintzan eman zioten [2]. Fundazioak lizentzia libre bezala argitaratu du eta eredu gisa mantentzen du Eclipse Process Framework (EPF) proiektuaren barruan.

Metodologia honek garrantzi handia izan du proiektu osoan zehar. Batetik, proiektuaren helburuetako bat metodologia baten definizioa eta ezarpena izan da eta OpenUP izan da aukeratutako metodologia. Bestetik, proiektuaren elaborazio prozesurako OpenUP metodologia jarraitu da, dokumentazioa bilduz eta proiektuaren kontrola eramanez.

5.3.2 ABRD

Agile Business Rules Development metodologiaren eredua ere erabili da. Horrela, bi metodologia erabiliz ziurtatzen da sistemaren egitura egokia dela. ABRD negozio erregela aplikazioa garatzeko metodologia da, erregela motorra eta erregelak kudeatzeko sistema erabiliz [3]. Metodologia honek erregelen prozesua eta sistema hedatzeko beharrezkoa diren arkitektura diziplinak zehazten ditu. 5.1. Irudian metodologiaren prozesua ikus daiteke, fase eta jarduerekin.



5.1. Irudia. ABRD metodologiaren prozesuko faseak.

5.3.3 MDE

Model Driven Engineering (MDE) edo eredu bidezko ingeniarietza softwarea garatzeko metodologia da. Domeinu ereduak erabiltzen ditu, hau da, arazo zehatz bati lotutako gai guztien eredu kontzeptualak. Hori dela eta, aplikazioen domeinu jakin baten ezagutzaren eta jardueren irudikapen abstraktuak nabarmentzea du helburu, kontzeptu informatikoetan sartu gabe.

5.4 Tresnak

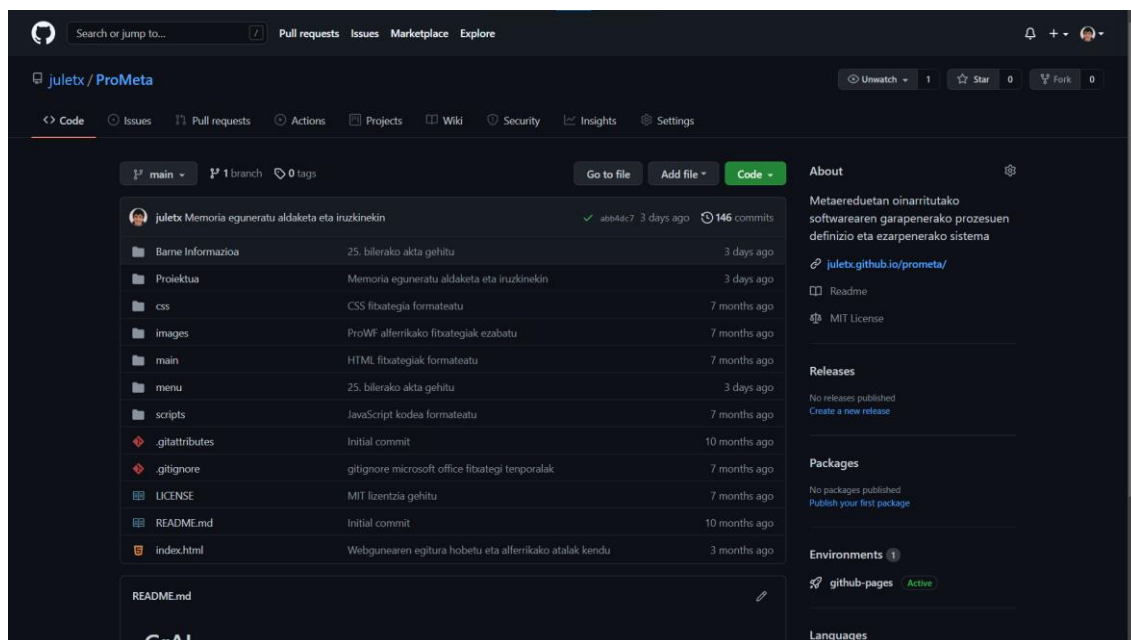
Atal honetan erabili diren tresna nagusien deskribapen labur bat egingo da.

5.4.1 Git

Linus Torvaldsek garatutako bertsio-kontrol software bat da [4]. Hain zuzen, produktu edo konfigurazio batean egin daitezkeen aldaketak kudeatzeko programa da. Proiektuaren fitxategi guztien bertsio-kontrolerako erabili da, hainbat biltegitan banatuta.

5.4.2 GitHub

Bertsio kontrolerako web-ostatu zerbitzua da, Git erabiltzen duena [5]. Gehienbat iturburu koderako erabiltzen da. Git-en bertsio-kontrol banatu eta iturburu-kode kudeatzaile funtzionalitate guztiak eskaintzeaz gain bere ezaugarri propioak gehitzen ditu. Proiektuan Git biltegi guztiak igotzeko erabili da. Adibidez, 5.2. Irudian dokumentazio webgunearen biltegia ikus daiteke.



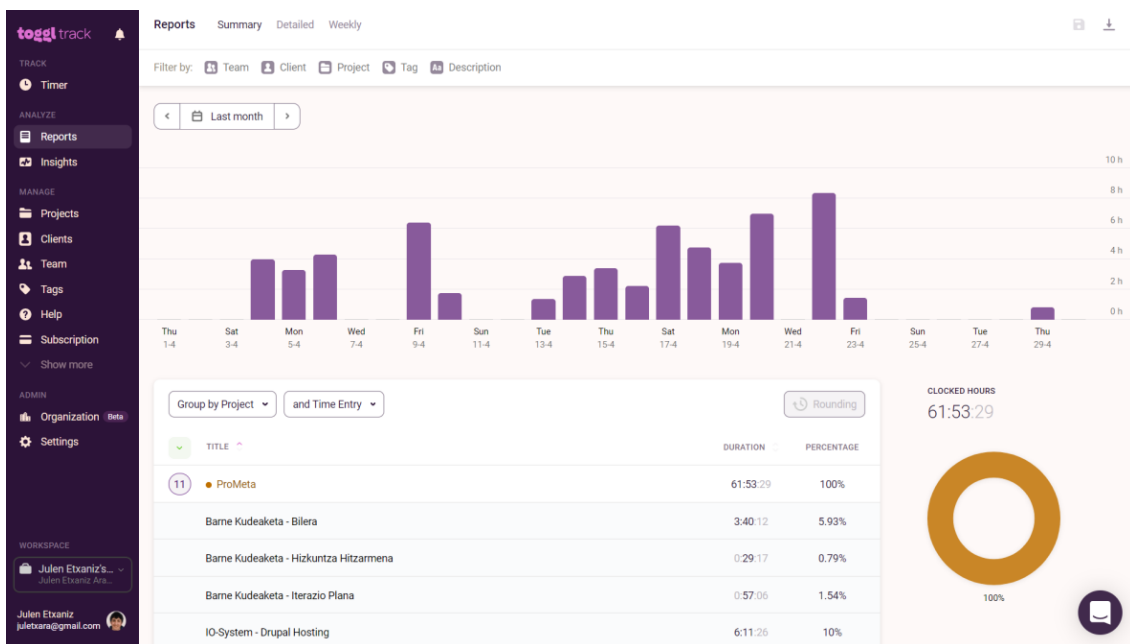
5.2. Irudia. ProMeta proiektuko webgunearen GitHub biltegia.

5.4.3 GitHub Pages

GitHub-ekin integratuta, automatikoki eraikitzen da webgunea kodea GitHub-era igotakoan [6]. Konfigurazio oso erraza, biltegitik bertatik egiten da. Webgune estatikoak bakarrik onartzen ditu, beraz, drupal-erako ez du balio, PHP erabiltzen baitu. Webguneak HTML, CSS, eta JavaScript-en idatzita egon daitezke edo Jekyll webgune estatiko sortzailea erabiliz.

5.4.4 Toggle Track

Proiektuko denbora neurtzeko erabili den tresna. Tenporizadore baten bidez ataza bakoitzean pasatako denbora neurtu daiteke. Eskuz ere aldatu daiteke denbora ahaztu egin bazaigu. Abisatu egiten du inaktibo bagaude eta denbora neurtzen ari bagara. Aspaldian denborarik ez badugu neurtu ere abisatzen du, ez ahazteko. Denbora proiektutan, atazatan eta kategoriatan sailkatu daiteke. Edozein momentuko denboraren estatistikak ikusi eta deskargatu daitezke. Adibidez, 5.3. Irudian hilabete bateko egun bakoitzeko denbora ikus daiteke.



5.3. Irudia. Toggl Track tresnan hilabete bateko egun bakoitzeko denbora.

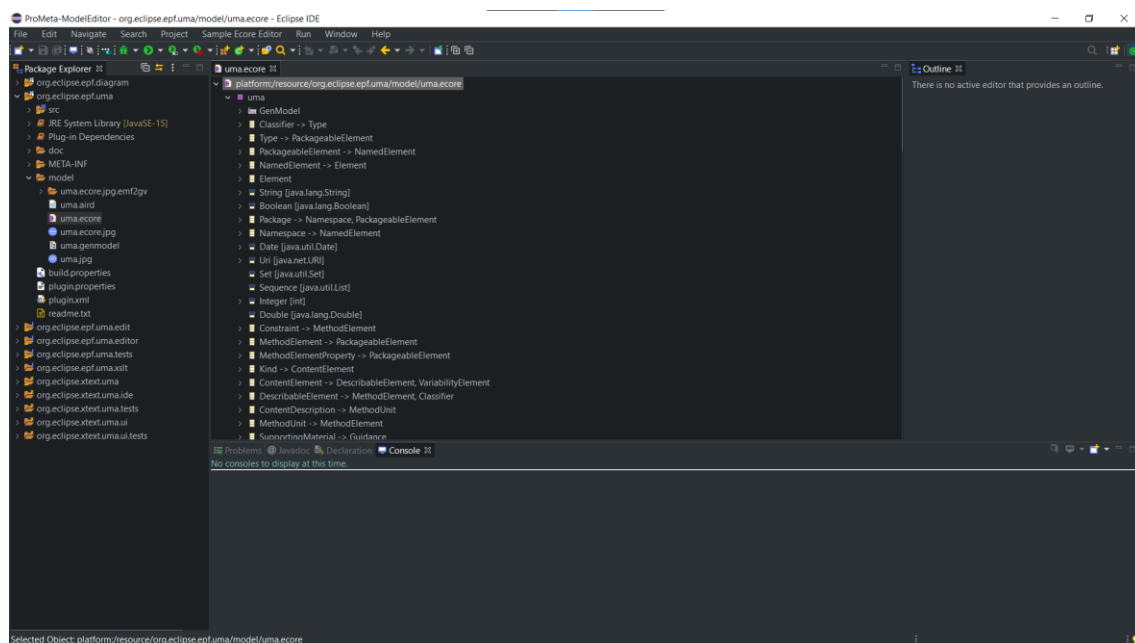
5.4.5 Java

Sun Microsystems-ek garatutako programazio lengoia eta plataforma informatikoa da. Plataforma makina birtual bat da eta Java programazio lengoia eta garapen tresnak erabiliz garatutako aplikazioak exekutatzeko gai da. Proiektuan Java 8 bertsioa erabili da EPF Composer-erako eta Java 11 eta 15 Eclipseko garapenerako. Java programazio lengoia erabili da ModelEditor azpisistemaren garapenerako.

5.4.6 Eclipse IDE

Kode irekiko software plataforma bat da. Ematen zaion erabilera nagusia Javaz programatzeko garapen ingurune integratuarena (IDE) da. Proiektuan zehar ModelEditor azpisistemaren garapenerako erabili da,

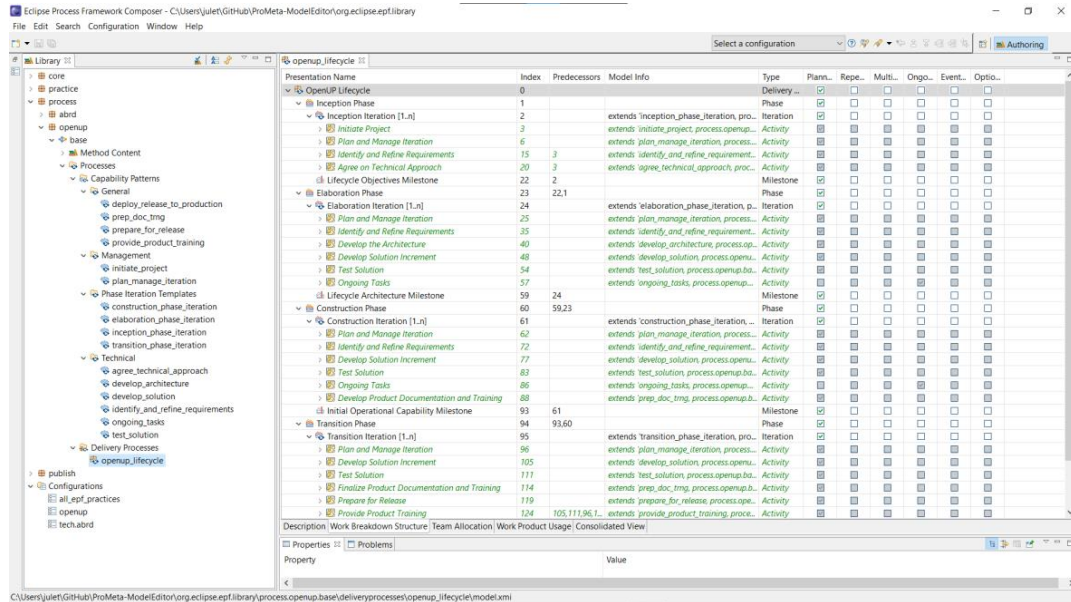
5.4. Irudian ikus daitekeen bezala.



5.4. Irudia. ModelEditor azpisistemako proiektuak Eclipse IDE editorean.

5.4.7 EPF Composer

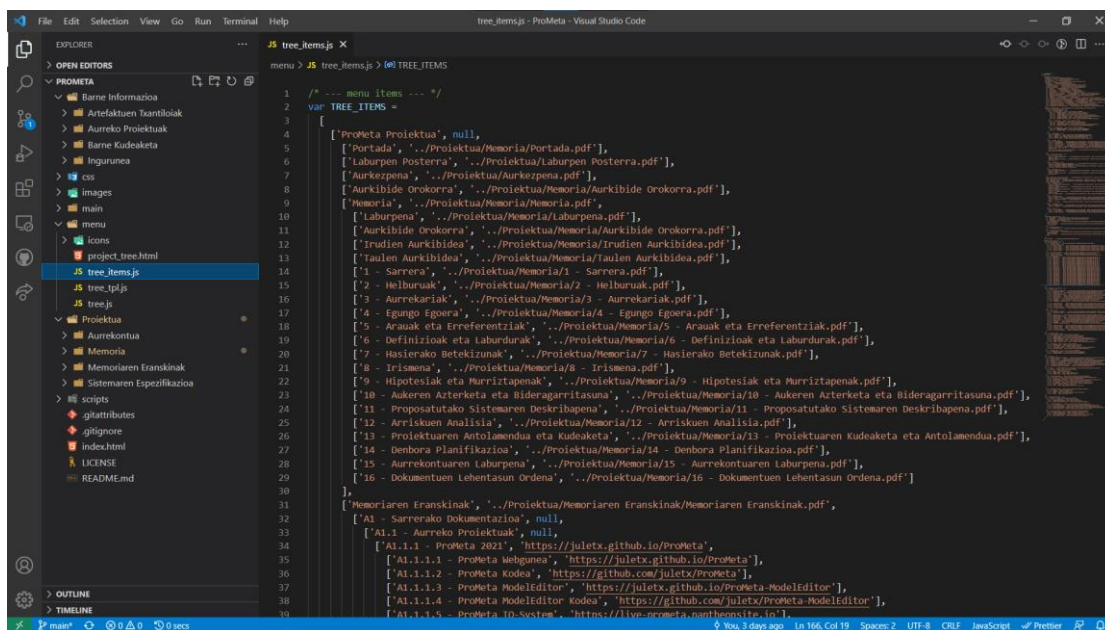
EPF Composer tresna erabili da metodologiaren informazioa eskuratu eta bistartzeko. Gainera, metodologia aldatzeko erabil daiteke eta metodologiaren webgunea sortzeko. 5.5. Irudian ikus daiteke EPF Composer tresnaren erabilera adibide bat.



5.5. Irudia. OpenUP bizi-zikloa EPF Composer tresnan.

5.4.8 Visual Studio Code

Microsoftek garatutako kode editorea. Bertsio kontrola, sintaxi nabarmentzea eta kode osatze automatikoa bezalako aukerak eskaintzen ditu. Software libre eta doakoa da. Proiektuan zehar hainbat fitxategi mota editatzeko erabili da. IO-System sistemaren garapenaren editore nagusia izan da. Gainera, bertsio kontrolerako ere erabili da, *commit* eta *push* guztiak bertatik eginez. 5.6. Irudian ProMeta dokumentazio webgunearen errepositorioa ikus daiteke Visual Studio Code-n.



5.6. Irudia. ProMeta dokumentazio webgunearen errepositorioa Visual Studio Code-n.

5.4.9 Xtext

Programazio lengoaiak eta domeinu espezifikoko lengoaiak (DSL) garatzeko kode irekiko ingurunea da [7]. Xtext-ek analizatzaile bat, sintaxi zuhaitz abstraktuaren klase eredua eta Eclipse-n oinarritutako IDE pertsonalizagarria sortzen ditu. Proiektuan testu editorea sortzeko eta SQL kodea sortzeko erabili da.

5.4.10 XSLT

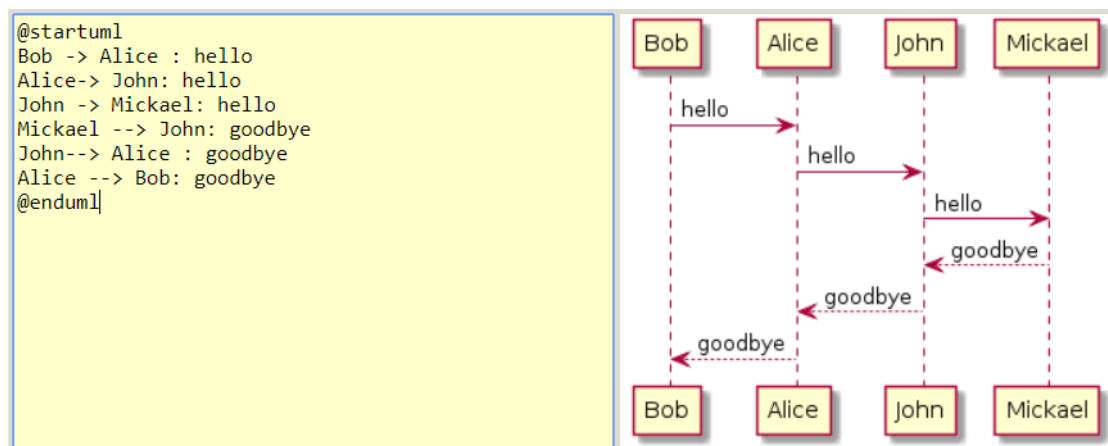
eXtensible Stylesheet Language Transformations (XSLT) XMLn oinarritutako lengoia da, XML fitxategiak transformatzeko balio duena. Proiektuan ModelEditor azpisteman XMI eredua UMARA transformatzeko erabili da.

5.4.11 Microsoft Office

Microsoftek garatutako ofimatika aplikazioa suitea. Ordainpekoa da Office 365 programaren harpidetza bitartez. Proiektuan zehar Word eta Excel erabili dira memoria eta eranskinak idazteko. PowerPoint erabili da posterra eta aurkezpena egiteko.

5.4.12 PlantUML

Kode irekiko tresna. Testu planotik abiatuta eta etiketa bidezko lengoia definitu bat erabiliz, UML diagramak sortzeko balio du. Adibidez, 5.7. Irudia lengoia deskriptiboa erakusten da eta horretatik sortutako diagrama sinple baten eredua. Proiektu honetan, OpenUP metodologiak eskatzen dituen erabilpen kasuen ereduak sortzeko erabili da.



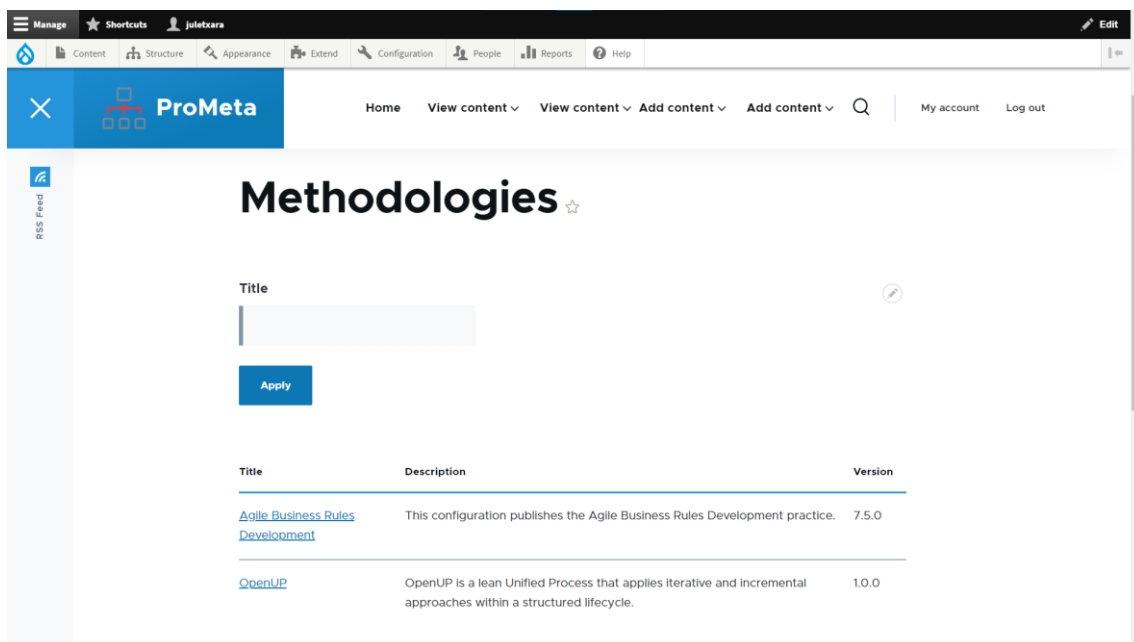
5.7. Irudia. PlantUMLren lengoain idatzitako sekuentzia-diagrama.

5.4.13 PHP

PHP (PHP: Hypertext Preprocessor) interpretatutako programazio lengoia bat da, batez ere webgune dinamikoak sortzeko erabili ohi dena. Datu-base sistema ugarirekin funtzionatzeko aukera izatea eta sistema eragile gehienetarako eskuragarri izatea dira beronen abantaila nagusiak. Proiektuan Drupal-erako programazio lengoia moduan erabili da.

5.4.14 Drupal

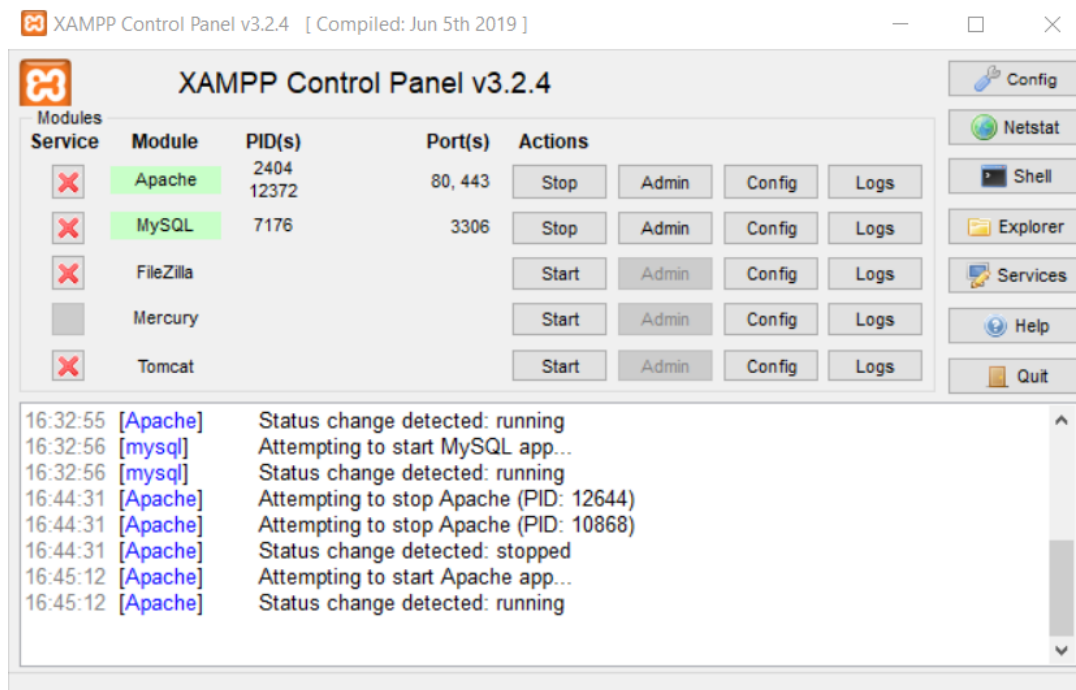
Drupal edukiak kudeatzeko sistema edo CMS librea, modularra eta oso konfiguragarria da [8]. Sistema dinamikoa da, hau da, zerbitzariaren edukiak modu finkoan biltegitatu beharrean, orrien testu-edukia eta beste konfigurazio batzuk datu-base batean biltegitatzen dira eta web-ingurune bat erabiliz editatzen dira. 5.8. Irudian proiektuan sortutako Drupal webgunea ikus daiteke. ProMeta IO-System sistemaren interfaze moduan erabili da, datuen sarrera/irteerak kudeatzeko.



5.8. Irudia. Proiektuan sortutako Drupal webgunea.

5.4.15 XAMPP

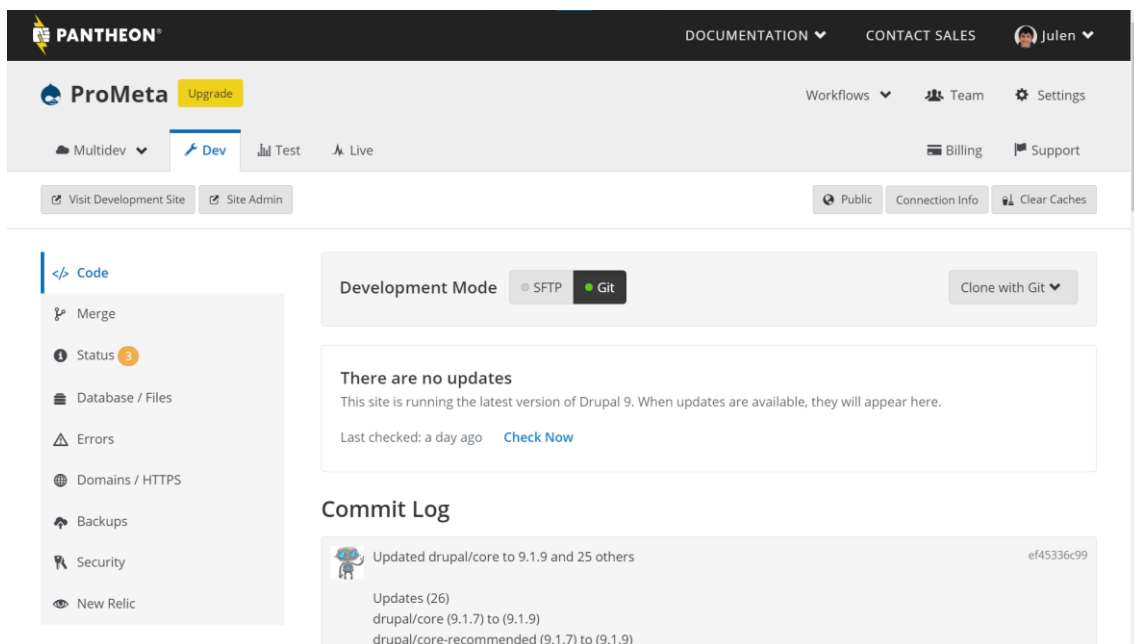
XAMPP software libreko pakete bat da, nagusiki *MySQL* datu-baseen kudeaketa sistema eta *Apache* web zerbitzaria integratzen duena [9]. 5.9. Irudian ikus daiteke XAMPPen erabileraren adibide bat. Proiektuan honetan Drupal webgunea lokalean garatzeko erabili da.



5.9. Irudia. XAMPP kontrol panela Apache eta MySQL hasieratzeko.

5.4.16 Pantheon

Pantheon kode irekiko Drupal eta *WordPress* webguneetarako *hosting* plataforma da [10]. 5.10. Irudian *Pantheon* kontrol panela ikus daiteke.



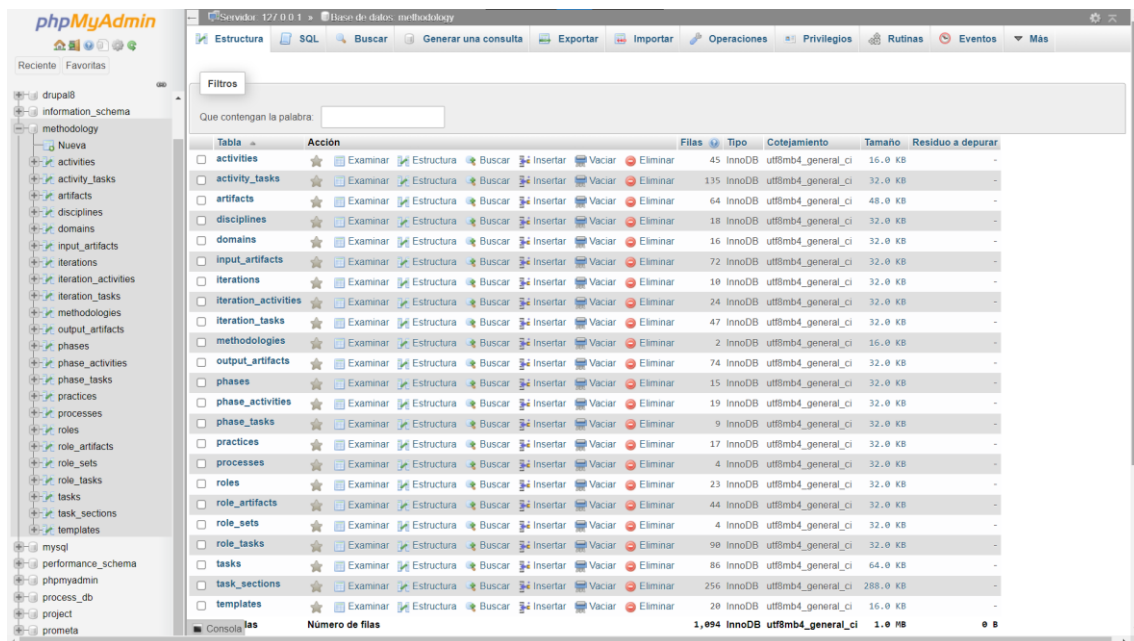
5.10. Irudia. *ProMeta* webguneko *Pantheon* kontrol panela.

5.4.17 MariaDB

MariaDB datu-baseak kudeatzeko sistema erlazional libre bat da. Oracle Corporationek MySQLren jabea zen Sun Microsystems erostean MySQLren jatorrizko garatzaileetako batzuk GNU Lizentzia Publiko Orokorra lizentziapean libre mantentzeko asmoz sortutako *fork* bat da. Datu-base erlazional moduan erabili da proiektuan.

5.4.18 phpMyAdmin

phpMyAdmin PHP-n idatzitako tresna da MySQL administrazioa web orrien bidez kudeatzeko. Proiektuan datu-base erlazionalak kudeatzeko erabili da, 5.11. Irudian ikus daitekeen moduan.



5.11. Irudia. *phpMyAdmin* tresnarekin datu-baseko taulak kudeatzen.

5.5 Ereduk

Proiektuan hainbat eredu erabili dira, batzuek dokumentazioarekin erlazionatuak eta beste batzuk inplementazioarekin.

5.5.1 CCII-N2016-02

Arauen atalean aipatu den moduan, estandar honetan oinarrituta antolatu da memoria eta proiektuaren webgunea. CCII erakundeak sortutako araua da, ingeniariaritz informatikoko proiektuetarako dokumentazioaren egitura eta beharrezkoak diren dokumentu eta sekzioak definitzen dituen.

5.5.2 OpenUP

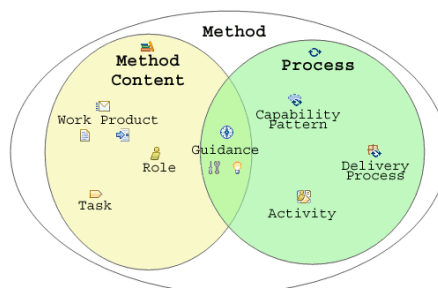
OpenUP metodologiaren eredu erabili da. Eredu horrek metodologiaren informazio guztia du. Metodologia hau jarraitzeko, bere webgunean artefaktu batzuen txantiloak daude eskuragarri. Txantilo horiek jarraituz OpenUP metodologiaren bitartez sortutako artefaktuak idatzi dira.

5.5.3 ABRD

Agile Business Rules Development metodologiaren eredu ere erabili da. Eredu horrek metodologiaren informazio guztia du. Metodologia hau jarraitzeko, bere webgunean artefaktu batzuen txantiloak daude eskuragarri.

5.5.4 UMA

Unified Method Architecture metaeredua erabili da OpenUP eta ABRD ereduak definitzeko. Metaeredu honen helburua edozein metodologia modelatu ahal izatea da. 5.12. Irudian ikusten den moduan, UMAN metodoaren edukia prozesutik bereizten da. Horrela, metodoaren edukia hainbat prozesutan berrerabil daiteke.



5.12. Irudia. UMA metodologiako metodoaren eta prozesuaren kontzeptuak.

5.5.5 Ecore

EMF-ko Ecore meta-metaeredua erabili da UMA metaeredua definitzeko. Meta-metaeredu honen helburua edozein metaeredu modelatu ahal izatea da. Ecore oinarri moduan hartuta definitzen dira metaeredu guztiak, baita Ecore bera ere.

5.6 Metrikak

Proiektuaren helburuekin erlazionaturako 4 metrika nagusi daude: irismena, denbora, kostua eta kalitatea.

5.6.1 Irismena

Proiektuaren irismena neurtzeko atazak definitu dira eta bakoitzaren denbora estimazioa egin da. Kontuan hartuta proiektua eta proiektuko taldea txikiak direla, OpenUP metodologia erabiltzea nahikoa da. Gainera, metodologiako lehenengo bi edo hiru fase egitearekin nahikoa izango da. Talde handiagoa

edo proiektu konplexuagoa izango balitz agian RUP bezalako metodologia konplexuagoa bat beharko genuke.

5.6.2 Denbora

Denboraren kontrola egiteko ataza bakoitzean pasatako denbora neurtu da, Toggle Track aplikazioaren kronometroa erabiliz. Ondoren, neurtutako denbora estimatutakoarekin konparatu da, eta horren arabera erabakiak hartu dira.

5.6.3 Kostua

Proiektuaren kostua neurtzeko aurrekontua egin da. Bertan, giza baliabideen eta erreminten kostuak zehaztu dira. Erreminta guztiak doakoak izatea espero denez, kostua kudeatzea erraza da. Giza baliabideen kostua jakiteko proiektuko kide bakoitzaren ordu kopurua eta orduko kostua biderkatzen dira.

5.6.4 Kalitatea

Proiektuaren kalitatea neurtzea garrantzitsua da onargarria dela ziurtatzeko. Kudeaketa planean zehazten den moduan zaintzeko hiru aspektu kontrolatu behar dira: inplementazioa, funtzionalitateak eta dokumentazioa.

1. **Inplementazioa.** Softwarearen ekoizpenerako azpiegitura teknologikoak arkitektura konplexua izango du BPM sistema arrakastatsua dutenen ideia, pausuz pausuko hurbilpen batean, jarraitzen delako. Sistematizazioa eta automatizazioa oinarri bezala hartuta. Ondorioz, menpekotasun teknologiko mugatua suposatu behar duten osagai asko izan behar ditu, denak irekiak azpiegituraren bilakaera errazteko.
2. **Funtzionalitateak.** Kalitatea, definizioz, bezeroen beharrak asebetetzearekin lotzen denez, azpiegitura aplikatuta kudeatzen diren proiektu errealak dagozkien bezeroen nahiak asebetetzeko besteko emaitzak (kode eta dokumentazioa) sortu behar dituzte. Azpiegitura teknologikoa bezero errealean beharren konplexutasun gorakorrera egokitu beharko da.
3. **Dokumentazioa.** Softwarearen ekoizpenerako azpiegitura teknologikoa proiektu askoren emaitza izango denez, ezagunak diren estandarrak erabilia eraiki behar da. Hau da, informazioaren konplexutasunera hobeto egokitzen den metodologia jarraituz adieraziko da hurrengo proiektuek beharko duten informazio guztia.

5.7 Prototipoak

Proiektuan bi prototipo nagusi garatu dira, azpisistema bakoitzari dagozkionak, ModelEditor eta IO-System.

5.7.1 ProMeta ModelEditor

Sortutako editore grafikoa eta testu editorea erabiliz prozesuaren eredua sortzeko eta editatzeko aukera emango du. Sistema honen ardura prozesu ingeniari rolak izango du.

5.7.2 ProMeta IO-System

CMS baten bitartez kudeatutako web-aplikazioa izango da. Helburua metodologia jarraitzen duten proiektuen informazioa gordetzea da. Rol bakoitzak metodologian dituen ataza berdinak bete beharko ditu.

6 Definizioak eta Laburdurak

Atal honetan memorian agertzen diren terminoen laburdurak eta definizioak daude.

6.1 Laburdurak

ABRD	Agile Business Rules Development
BETRADOK	Betekizunen Trazabilitatea eta Dokumentazio sorkuntza automatikoa
Bizagi	BPM hornitzailea
BPM	Business Process Management
BPMN	Business Process Model and Notation
CCII	Consejo de Colegios de Ingeniería Informática
CCII-N2016-01	Norma de Visado de Proyectos y Actuaciones Profesionales en Ingeniería Informática
CCII-N2016-02	Norma Técnica para la realización de la Documentación de Proyectos en Ingeniería Informática
CLIPS	C Language Integrated Production System
CMMI	Capability Maturity Model Integration
CMS	Content Management System
COOL	Classroom Object Oriented Language
CSS	Cascading Style Sheet
DOT	Grafoak definitzeko lengoaia
Drupal	CMS hornitzailea
DSL	Domain Specific Language
Ecore	Meta-metaeredua
EHSIS	Erabakiak Hartzen Laguntzeko Sistemen Sortzailea
EHU	Euskal Herriko Unibertsitatea
EMF	Eclipse Modeling Framework
EPF	Eclipse Process Framework
Freemium	Dohako oinarritzko zerbitzuak gehi ordaindu beharrekoak
GrAL	Gradu Amaierako Lana
HTML	HyperText Markup Language

IBM	International Business Machines
IDE	Integrated Development Environment
ISO	International Organization for Standardization
LDE	Lan Deskonposaketa Egitura
MDE	Model-Driven Engineering
NASA	National Aeronautics and Space Administration
OpenUP	Open Unified Process
PDF	Portable Document Format
PHP	Hypertext Preprocessor
PlantUML	Plant Unified Modeling Language
PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
ProMeta	Professional Metamodel
ProWF	Professional Workflow
RUP	Rational Unified Process
SOA	Service-oriented Architecture
SPEM	Software Process Engineering Metamodel
SPICE	Software Process Improvement and Capability dEtermination
SQL	Structured Query Language
SWEBOK	Software Engineering Body of Knowledge
UMA	Unified Method Architecture
UML	Unified Modeling Language
UNE	Una Norma Española
URPS	Usability, Reliability, Performance and Supportability
Workflow	Lan-fluxua
WYSIWYG	What You See Is What You Get
XAMPP	Cross-platform Apache, MariaDB, PHP and Perl
XMI	XML Metadata Interchange
XML	eXtensible Markup Language

XPDL	XML Process Definition Language
XSLT	eXtensible Stylesheet Language Transformations

6.2 Definizioak

6.2.1 Bizagi

Bi produktu osagarri dituen softwarea da, prozesuen modelatzaile (Bizagi Modeler) bat eta BPMren suite ofimaticoa bat (Bizagi Studio).

6.2.2 BPM

Sigla(ingelesez), Business Process Management. Enpresei prozesuak automatikoki modelatzeko, inplementatzeko eta exekutatzeak aukera ematen dien software teknologia da.

6.2.3 CCII

Sigla(gaztelaraz), *Consejo de Colegios de Ingeniería Informática*. Estatu-mailan informatika ingeniari guztiak errepresentatu eta bateratzen dituen antolakundea da. Ikus, gainera: [CCII, webgunea](#).

6.2.4 CCII-N2016-01

CCII erakundeak sortutako araua, Ingeniaritza informatikoko proiektuen ikuskaritza edo bisa egiteko estandarra. Estandarrak dokumentuen osotasuna berrikusteko zerbitzuen prozesua deskribatzen du [11].

6.2.5 CCII-N2016-02

CCII erakundeak sortutako araua, ingeniarietza informatikoko proiektuetarako dokumentazioaren estruktura eta beharrezkoak diren dokumentu eta sekzioak definitzen dituen [12]. Informazio gehiago: <https://www.cci.es/norma>.

6.2.6 CMMI

Capability Maturity Model Integration. Software-sistemak garatzeko, mantentzeko eta erabiltzeko, prozesuak hobetzeko eta ebaluatzeak eredu da, CMMi Institutuak administratutakoa.

6.2.7 CMS

Sigla(ingelesez), *Content Management System*. Dokumentuak eta bestelako edukiak antolatu eta kudeatzeko softwarea da, normalean web-aplikazioa.

6.2.8 DOT

Testu lauan idatzitako lengoia deskriptiboa da. Grafoak deskribatzeko modu sinple bat eskaintzen du, gizakiek eta konputagailuek ulertzeko modukoa.

6.2.9 Drupal

Edukiak kudeatzeko sistema edo CMS librea, modularra eta oso konfiguragarria. Ikus, gainera: [Drupal, webgunea](#).

6.2.10 IBM

Sigla(ingelesez), International Business Machines. Informatikarekin lotutako tresnak, programak eta zerbitzuak ekoiztu eta merkaturatzen dituen enpresa multinazionala da.

6.2.11 ISO

International Organization for Standardization edo Estandarizazio Nazioarteko Erakundea nazioarteko estandarrak ezartzen dituen erakunde bat da, hainbat estatutako estandarizazio erakundeek osatua.

6.2.12 OpenUP

Open Unified Process softwarea garatzeko metodo eta prozesu bat da, Eclipse Fundazioak garatua [2]. Rational Unified Process (RUP) metodologiaren azpimultzo minimoa da. Proiektua iteraziotan banatzen du eta eta proiektuaren bizi-zikloaren lau fasetan banatzen du: Hasiera, Elaborazioa, Eraikuntza eta Trantsizioa. Informazio gehiago: [OpenUp 1.0](#) edo [OpenUp 1.5](#).

6.2.13 PlantUML

Testu laua *UML* diagrametan bihurtzeko balio duen software irekia. Ikus, gainera: [PlantUML](#), [webgunea](#).

6.2.14 ProMeta

Softwarearen garapenerako prozesuen definizio eta ezarpenerako sistema metaereduetan oinarrituta. Proiektu honen izena ingeleseko hitzetatik eratutako hitz-jokoa da. “Pro” *professional* edo *process* hitzetik dator. “Meta” *metamodel* hitzaren laburdura da eta metaeredu esan nahi du.

6.2.15 ProWF

Software proiektuen elaboraziorako *workflow*etan oinarritutako sistemaren sorkuntza eta bizi-zikloa definitzeko metodologia baten ezarpena. Proiektu honen aurrekariaren izena ingeleseko hitzetatik eratutako hitz-jokoa da. “Pro” *professional* hitzetik dator eta profesionala esan nahi du, “WF” *workflow* hitzetik datorren laburdura da eta lan-fluxu esan nahi du.

6.2.16 RUP

Sigla(ingelesez), Rational Unified Process. Rational Software enpresak garatutako software-prozesu bat da [13]. Objektuetara bideratutako sistemak aztertu, diseinatu, inplementatu eta dokumentatzeko erabiltzen den metodologia estandarra.

6.2.17 SPICE

Software Process Improvement and Capability dEtermination. ISO/IEC 15504. Garapen-prozesuak hobetzeko, ebaluatzeko, informazio-sistemak eta software-produktuak mantentzeko eredua da.

6.2.18 UML

Unified Modeling Language (Modelaketarako lengoaia bateratua) sistemak zehaztu, diseinatu eta eraikitzeko lengoaia da, printzipioz objektuei orientatutako programaziorako prestatuta dagoena. UML aplikazio baten garapen fase guztiak modelatzeko lengoaia homogeneo bat definitzen saiatzen da, bezeroaren zehaztapenetatik hasita programatzailearen diseinu xehera arte.

6.2.19 UNE

Sigla(gaztelaraz), Una Norma Española. Comités Técnicos de Normalización (CTN) batzordeak sortutako arauen, arau esperimentalen eta txostenen (estandarrek) multzoak dira.

6.2.20 URPS

Sigla (Ingelesez). Usability, Reliability, Performance and Supportability. Softwarearen kalitate ezaugarriak dira: Erabilgarritasuna, Fidagarritasuna, Errendimendua eta Mantenigarritasuna.

6.2.21 Workflow

Aspektu operazionalekin lan-aktibitate bat deskribatzeko egiten den irudikapena. Irudikapen horretan atazak nola egituratzen diren, zein den atazen arteko ordena eta nola sinkronizatzen diren, nolakoa den atazen informazio-fluxua eta atazen betetzearen jarraipena nola egiten den grafikoki deskribatzen da.

6.2.22 WYSIWYG

Sigla (ingelesez), What You See Is What You Get. Testu-prozesadorei eta beste testu-editore batzuei aplikatutako esaldi bat da, azkenengo emaitza zuzenean erakutsiz dokumentu bat idazteko aukera ematen duena.

7 Hasierako Betekizunak

Kapitulu honetan proiektuaren hasierako betekizun funtzionalak eta ez-funtzionalak azaltzen dira. Gainera, sistemak izan behar dituen kalitate ezaugarriak eta sistemaren interfazeen ezaugarriak deskribatzen dira.

7.1 Betekizun Funtzionalak

Betekizun funtzionalek sistemaren beharrak eta horiek betetzeko ezaugarriak finkatzen dituzte. 7.1. Taulan datu horiek ikus daitezke lehentasunarekin batera. Betekizun hauek goi-mailan definitu dira, aurrerago zehaztuko da gehiago bakoitzaren inguruan.

Beharra	Ezaugarriak	Lehentasuna
Software garapeneko prozesuaren definizioa	Software garapeneko prozesuaren metaeredua definitu. Gutxienez OpenUP metodologiaren eredua definitu. Metodologia gehiagoren ereduak definitu.	Altua
Editore grafikoa eta testuala	Metaeredua erabiliz ereduak aldatzeko editore grafikoa sortu. Testu editorea sortu metaeredua erabiliz. Editoreen arteko bateragarritasuna bermatu.	Altua
Prozesuaren datu-basea	Prozesuaren informazioa gordeko duen datu-basea definitu. Datu-basean datuak gorde.	Altua
Prozesuaren webgunea	Web interfazea garatu. Web kodea garatu. Webgunea zerbitzari batean jarri.	Altua

7.1. Taula. Hasierako betekizun funtzionalak.

7.2 Betekizun Ez-Funtzionalak

Betekizun ez-funtzionalak sistemaren funtzionalitateekin zuzenean erlazionatuta ez dauden betebeharrak deskribatzen dituzte. 7.2. Taulan betekizun ez-funtzionalak eta horien lehentasuna ikus daiteke. Orokorrean, betekizun funtzionalek baina lehentasun txikiagoa dute, sistemaren funtzionamendurako ez baitira ezinbestekoak.

Betekizuna	Lehentasuna
Dokumentazioa eta proiektuaren webgunea CCII estandarren arabera.	Altua
Proiektuan OpenUP metodologia jarraitzea.	Altua
Garapen prozesua metodologia eta estandarrek esaten duten moduan eratzea.	Altua
Erreminta, metodologia eta ezagutzaren aldetik eman daitezkeen aldaketen aurrean, soluzioa malgua eta egokigarria izatea.	Ertaina
Proiektuaren garapenerako doakoak eta libreak diren tresnak erabiltzea.	Ertaina
Proiektuko osagaien dokumentazioa eta eskuliburuak	Ertaina

7.2. Taula. Hasierako betekizun ez-funtzionalak.

7.3 Sistemaren Ezaugarriak

Software kalitatearekin erlazionatutako URPS ezaugarriak deskribatuko dira. Garrantzitsua da argi edukitzea sistemak izan behar dituen kalitate ezaugarriak.

7.3.1 Erabilgarritasuna

Sistemak erabilgarritasun altua izango du, erabiltzeko, ikasteko eta memorizatzeko erraza izango da. Ez da ikastarorik beharko aplikazioa erabili ahal izateko, software talde bateko kideentzat intuitiboa izango denez erraz ikasiko baitute. Hala ere, sistemak erabiltzailea laguntzeko eskuliburuak izango ditu, erabiltzaileak zalantzak dituenean kontsultatu ahal izateko.

7.3.2 Fidagarritasuna

Sistemak fidagarritasun altua izango du. Honek esan nahi du ia beti eskuragarri egon behar duela, hutsegiteek eragin txikia izan behar dutela eta hauetatik azkar berreskuratuko dela. Fidagarritasuna bermatzeko, sistema monitorizatuko da arazo potentzialak azkar identifikatu eta ekiditeko.

7.3.3 Errendimendua

Sistemak errendimendu altua izango du. Honek esan nahi du erantzun denbora azkarra edukiko duela eta aldi berean hainbat konexio onartuko dituela. Hasieratze eta amaitze denbora ere azkarra izango da. Horretarako, garrantzitsua izango da zerbitzariak ahalmen nahikoa izatea.

7.3.4 Mantengarritasuna

Sistemak mantengarritasun altua izango du, hau da, instalatzeko, konfiguratze, eguneratzeko eta mantentzeko erraza izango dela. Erabiltzaileek web bidez erabiliko dute sistema, eta beraz ez dute ezer instalatu beharrik izango. Instalazioa, eguneratzeak eta mantenua zerbitzarian egingo dira.

7.4 Sistemaren Interfazeak

Sistemaren erabiltzaile interfazeen ezaugarri garrantzitsuak deskribatuko dira. Sistemaren interfazeak egokiak izatea ezinbestekoa da erabiltzailearen esperientzia ona izateko.

7.4.1 Itxura eta Sentsazioa

Orrialdean erabilitako kolore eta itxurari dagokionez, koloreak kontuz aukeratuko dira irakurgarritasuna bermatzeko. Gainera, kontzeptuak OpenUP prozesuan erabiltzen diren antzeko koloreekin adieraztea izango da helburua, lortura errazagoa izan dadin. Erabiliko diren menuak eta aukerak software proiektuetan aritzen direnentzat ulerterrazak eta ezagunak izan behar dira.

7.4.2 Diseinu eta Nabigazio Betekizunak

Nabigazio menuan funtzionalitate nagusiak bilduko dira, eskuragarri egon daitezen. Menuan agertzen diren aukerak erabiltzaile motaren arabera izango dira, eta menua atzigarria izan behar du edozein momentutan. Sistemaren atal desberdinak modu egokian antolatuta egongo dira. Sistema *responsive* izango da, pantailaren tamainaren arabera itxura automatikoki aldatuko da.

7.4.3 Trinkotasuna

Interfazeak trinkotasuna mantendu beharko du, itxuraz berdinak diren elementuek funtzionalitate antzekoa izan behar dute. Gainera, interfazeetako elementuek ohiko portaera izango dute, erabiltzaile gehienentzat beste sistemetatik ezaguna dena.

7.4.4 Erabiltzailearen Pertsonalizazio Betekizunak

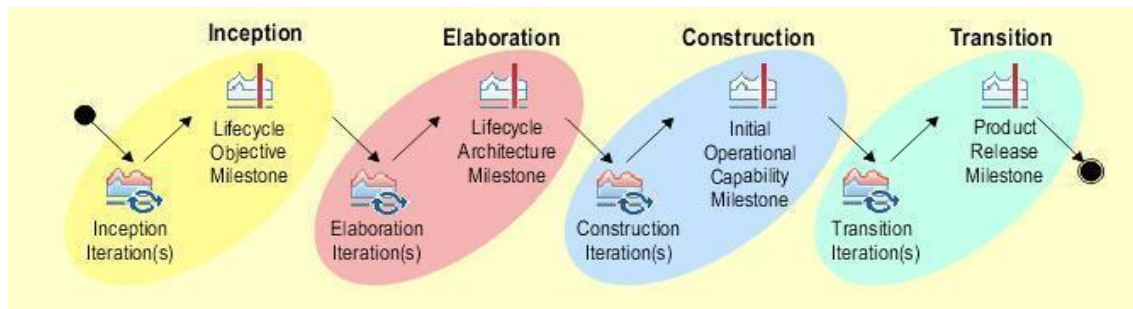
Ez da aurreikusten erabiltzailean pertsonalizazio aukerarik izatea. Baliteke, interfazearen itxura edo hizkuntza aldatzeko aukeraren bat gehitzea.

8 Irismena

Kapitulu honek proiektuaren irismena definitzea eta proiektuko artefaktuak zerrendatzea du helburu.

8.1 Bizi-zikloa

Proiektu honen irismena finkatzeko, OpenUP metodologiaren bizi-zikloa jarraitu da. 8.1. Irudian ikusten den moduan bizi-ziklo hori lau fasez osatuta dago: hasiera, elaborazioa, eraikuntza eta trantsizioa.



8.1. Irudia. OpenUP metodologiako bizi-zikloaren faseak.

Proiektuaren kasuan **elaborazio** faseraino iritsiko da, **produktuaren lehenengo prototipoa ateraz eta dokumentazioa sortuz**. Beste proiektu batzuen bidez hobekuntzak egingo dira. Denbora mugatuko proiektua izanik, eraikuntza eta trantsizio faseak kanpoan geratu dira. Hurrengo zerrendan deskribatzen dira irismena definitzen duten ezaugarriak:

- OpenUP metodologiak eskatutako dokumentuak betetzea. Horretarako OpenUP metodologiak bere webgunean eskaintzen dituen txantiloiak jarraituz.
- CCII-N2016-02 arauak eskatzen dituen dokumentuak betetzea. Ingeniaritza informatikako proiektu profesional baten dokumentazioa ere profesionala izan dadin, arau estandar bat erabiltzea oso garrantzitsua da.
- Proiektuaren webgunea osatu. Webgune honetan jarritako dokumentuak CCII-N2016-02 arauak eskatzen duen dokumentazio egitura jarraituz. Bertan, proiektuaren memoria, memoriaren eranskinak, OpenUP metodologiarekin sortutako dokumentu guztiak eta proiektuarekin zerikusia duten hainbat aspektu agertuko dira.
- ProMeta ModelEditor azpisistemaren garapena.
- ProMeta IO-System azpisistemaren garapena.
- Proiektuaren memoria idaztea.
- Proiektuaren defentsa prestatzea. Horretarako, memorian idatzitako aspektu guztiak laburbiltzen dituen aurkezpen bat prestatuz.
- Proiektuaren posterra egitea.

8.2 Artefaktuak

OpenUP metodologiari dagokionez, 8.1. Taulan ikusten dira bete behar diren dokumentuak, domeinuaren arabera sailkatuta. Esan bezala, hauek hasiera, elaborazio eta eraikuntza faseei dagozkien dokumentuak dira. Hala ere, baliteke denbora faltagatik dokumentu gutxi batzuk ez betetzea.

Domeinua	Artefaktua
Arkitektura	Arkitektura Koadernoa
Hedapena	Produktuaren Dokumentazioa Laguntza Dokumentazioa Erabiltzaile Dokumentazioa Trebatzeko Materialak
Garapena	Inplementazioa Eraikuntza Diseinua
Ingurunea	Garapen Kasua Tresnak
Proiektu Kudeaketa	Iterazio Plana Proiektu Plana Lan-atazen zerrenda Arriskuen zerrenda
Betekizunak	Glosarioa Ikuspegia Betebeharren Espezifikazioa Erabilpen Kasuak Erabilpen Kasuen Eredua
Proba	Proba Kasuak Proba Log-ak Proba Script-ak

8.1. Taula. OpenUP metodologiako bete diren artefaktuak domeinutan sailkatuta.

9 Hipotesiak eta Murritzapenak

Kapitulu honetan, proiektuaren hasierako hipotesiak eta proiektuaren garapenerako ezarritako murritzapenak deskribatuko dira.

9.1 Hipotesiak

Proiektuarekin lanean hasteko hipotesi batzuk atera ditugu aurreko lan eta datuetatik. Hurrengoak dira proiektuaren hasierako hipotesiak:

- Metaereduen erabilerak malgutasuna ematen du proiektuaren hurbilpena aldatzeko edo etorkizunean beste hurbilpen batzuk probatzeko.
- Definitutako garapen prozesuak software proiektuen elaborazioa gidatu eta kontrolatuko du. Prozesua aldatzeak sistemaren portaera eta datu-basea automatikoki aldatzea ekarriko du.
- CMS baten erabilera datuen sarrera/irteerarako irtenbide egokiena da. Webgune bat sortzeko aukera ematen duen tresna erabilerraza izateaz gain, ez da baliabide tekniko aurreratueta etengabe jo behar. Kudeaketa, administrazioa eta mantentze-lanak egiteko laguntza ematen du kanpoko baliabiderik erabili gabe.
- Datu-base erlazionalak prozesu baten ezagutza gordetzeko modurik egokiena da, datuen independentzia, emaitzen koherentzia eta datu-basearen produktibitatea handitzea lortuz.
- Sistema iteratiboki hobetzen joango denez, estandarretan oinarritutako garapenak bere mantenua eta hedapena errazten ditu.

9.2 Murritzapenak

Proiektu informatikoen bezeroen eskakizunen ondorioz, neurri batean mugatu egin dira proiektuarekin lotutako elementu batzuk egiteko kontuan hartu beharreko aukerak. Hurrengoak dira proiektuaren hasierako murritzapenak:

- Kostu ekonomikoa ez da batere aldatu hasierako planteamendutik. Izan ere, proiektuan zehar ez da kostu gehigarriarik sortu behar, erabilitako teknologia guztiak doakoak direlako.
- Denborari dagokionez, ekainaren 20rako proiektua bukatzeko murritzapena bete behar da.
- Kalitateari dagokionez, proiektuak denbora eta kontu murritzapenen barruan kalitate onargarria izan behar du, hau da, bezeroari aurkezteko modukoa.
- Betekizunen ingeniariaritz eta bizi-zikloa definitzen duen metodologia bat jarraitzea proiektuaren elaborazio eta garapenerako: OpenUP. Bezeroak egiten duen exigentzia da metodologia bat erabiltzea.
- Proiektuaren dokumentuen antolaketarako CCII-2016N-02 estandarra erabiltzea.
- Sortuko den software proiektuen elaboraziorako sistema web bidez atzigarria izan behar du.
- Beste proiektu batzuen oinarria izango denez, informazio transmisio eraginkorra ahalbideratzen duen dokumentazioa eraman behar du.

10 Aukeren Azterketa eta Egingarritasuna

Kapitulu honetan proiektuko soluzioaren alternatibak, balorazio-irizpideak eta hautatutako aukeren justifikazioak agertzen dira.

10.1 Arkitektura

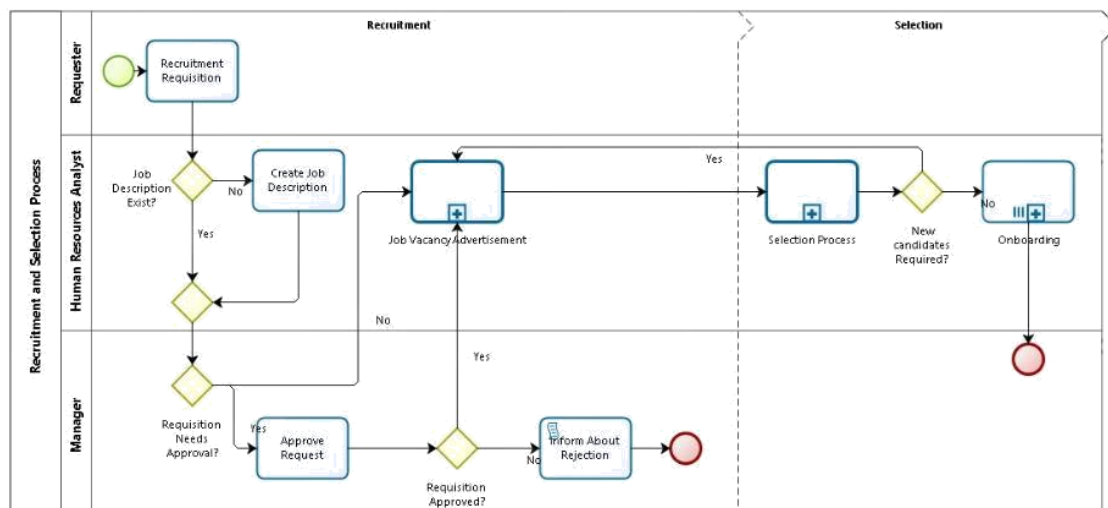
Soluzioaren arkitekturari dagokionez, ProWF proiektuan bi aukera aztertu ziren: *Bizagi* bezalako softwarearen bitartez prozesuetan (BPM) oinarritutako web-aplikazioa eraikitzea edo arkitektura propioa sortzea. Proiektu honetan aukera berri bat gehitu da.

10.1.1 Bizagi

Bizagi Softwarearen Kalitatea irakasgaia erabilitako softwarea da. Bertan, *Bizagi Modeler* editorearen bitartez prozesuak sortu eta ondoren, *Bizagi Studio* tresnan negozio-erregelak adieraziz, datu-basea konfiguratuz, formularioak definituz, web-zerbitzuak integratuz eta beste hainbat aspektu ukituz, sortutako prozesuan oinarritutako web-aplikazio bat sortu zen. Sortutako web-aplikazioak itxura oso profesionala zuen eta bere sorkuntza ez zen izan batere zaila.

Proiektu honetarako bideragarria izango litzateke software hau erabiltzea, hurrengo bi arrazoiengatik:

- Lan-fluxua definitzeko *Bizagi Modeler* editorea erabili daiteke, bizi-zikloa definitzen duten metodologiek faseak eta rolak erabiltzen dituztelako, 10.1. Irudian ikus daitekeen bezala. Beraz, *Bizagi Modeler* erabiliz, bere lengoaiaren (BPMN) arauak errespetatuz, *workflow* azpi-lengoaia bat sortu izango litzateke.
- Soluzioaren sarrera/irteerak kudeatzeko *Bizagi Studioren* bitartez sortutako web-aplikazioa erabili daiteke, bere datu-baseen kudeaketa, negozio-erregelak, formularioak eta web-zerbitzuak baliatuz.



10.1. Irudia. Bizagi Modeler erabiliz sortutako prozesua.

Arkitekturaren soluzio honek, ostera, bi desabantaila nagusi ditu:

- Arkitektura guztia kanpoko tresna baten bitartez eraikitzeak izugarrizko menpekotasun teknologikoa sortuko du. Garapen-prozesuaren konplexutasuna gorakorra izango denez, sistema osoa, hasieratik, *Bizagiren* menpe jartzea arriskutsua izan daiteke eta ez da batere komenigarria produktuaren aldaketa eta hobekuntzarako.

- *Bizagi Modeler freemium* bat da, hau da, zerbitzu basikoak dohain eskaintzen ditu eta zerbitzu aurreratuago edo bereziengatik zerbait ordaindu behar da. *Bizagi Studio*, ordea, *suite* ofimatikoa da, enpresen erabiltzenerako paketeak eskaintzen ditu bere produktua erosiz. Hori dela eta, *Bizagirekin* arkitekturaren kostu ekonomikoa handia izango litzateke.

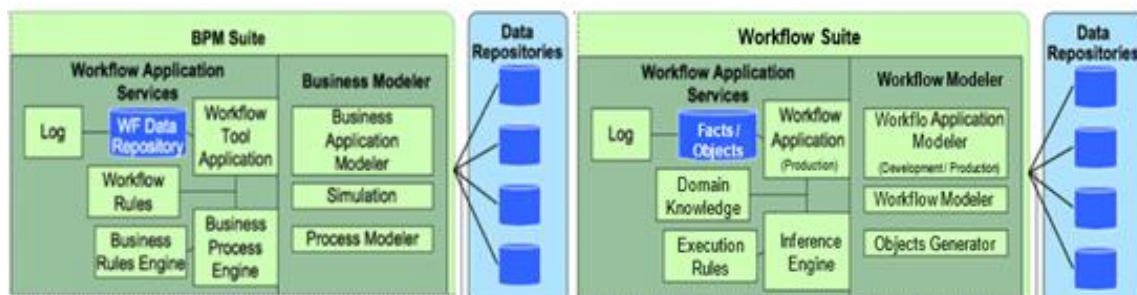
10.1.2 ProWF

Esan bezala, ProWF proiektuan erabakia **arkitektura zerotik eraikitzea** izan zen. Hurrengo kapituluaren sartuko gara soluzioaren deskribapenean.

Workflowetan oinarritutako web-aplikazio bat zerotik eraikitzea apustu handia zen, denbora gehiago eman beharko litzateke arkitekturaren garapenean, baina, ordea, pisuzko arazoak zituen:

- *Bizagiren* desabantailak oso kaltegarriak ziran. Honela, menpekotasun teknologikoa saihestu eta kostu ekonomikoa asko gutxitzen da.
- *Workflow*-lengoaia guztiz propioa sortzea komenigarria zen, lengoaia hori exekutatzeko motor propioa ere, bere erregela eta berezitasunekin.
- Gaur egun, sarrera/irteeren kudeaketarako, *CMS* baten erabilerarekin web-aplikazioen sormena ez da hain zaila eta soluzio profesionalak lortu daitezke.

Hala ere, workflowetan oinarritutako arkitektura hau zerotik eraikitzea ez da lan erraza eta sistema konplexua bat sortu behar da, gainera, *Bizagik* eta *BPMN*¹⁸ estandarrak eskaintzen dituzten funtzionalitate asko galduko ditugu. 10.2. Irudian BPMn eta workflowetan oinarritutako arkitekturak ikus daitezke.



10.2. Irudia. BPMn eta workflowetan oinarritutako arkitekturak.

10.1.3 ProMeta

ProMeta proiektuan sistemaren **abstrakzio-maila igotzea** erabaki da. Metodologiaren informazioa workflow lengoia edo BPMNn eskuz definitu ordez, metaeredu eta ereduen bidez gordetzen da. Horrela, ereduaren informazioa berrerrabil daiteke eta beste ereduera eraldaketak egin. Ereduen informazioa zuzenean kodea sortzeko ere erabil daiteke, beste metaereduren beharrik gabe.

Hurbilpen honek ziurtatzen du etorkizunean aurreko aukeretak edozein edo beste batzuk aukeratu ahal izatea. Izan ere, BPMNren eta workflow-lengoiaaren metaereduak izanik, eraldaketa egitea ez litzateke horrenbeste lan izango. Gainera, eredutik datu-baserako SQL kodea edo webgunerako HTML kodea sortzeko aukera ere egongo litzateke.

Proiektu honetan saiakera bat egin da BPM eta workflow-lengoia erabili gabe. Izan ere, eredutik sortutako datu-basea eta Drupal-eko moduluak bakarrik erabili dira. Drupal CMSaren funtzionalitateak aprobetxatzeko ahal den guztia bere interfazea erabiliz eraiki da. Adibidez, edukia kanpoko datu-basean gorde ordez, Drupal-eko eduki motetan gordetzea erabaki da.

10.2 CMS

ProWF proiektun, soluzioaren datu zein informazioaren sarrera/irteerak kudeatzeko web-aplikazio bat sortzea erabaki zenez, CMS bat erabiltzea adostu zen. CMS baten bitartez web-aplikazioaren administrazioa eta kudeaketa ahalbidetzen da eta itxura profesionala duen emaitza lortu daiteke.

Hasieratik *Drupal* erabiltzea gomendatu zuen proiektuaren tutoreak, Juan Manuel Pikatzak, baina *Drupal* erabiltzen hasi baino lehen merkatuan zeuden beste CMSak aztertu behar ziren ere. Hiru CMS aztertu ziren nagusiki: *Wordpress*, *Joomla* eta *Drupal*.

Taulan ProWF proiektuan egindako konparaketa bat ikus daiteke erabakia hartzeko gehien nabarmentzen diren puntuekin.

Ezaugarria	Wordpress	Joomla	Drupal	Oharrak
Kode irekia	BAI	BAI	BAI	-
Dokumentazio sinple eta ondo egituratuta	BAI	BAI	BAI	-
Komunitate aktiboa eta foroak	BAI	EZ	BAI	Hemen Wordpress da nagusia.
Estentsio gehigarri eta moduluen hedapena	BAI	EZ	BAI	Joomlak estentsio gehigarriak ditu ere, baina ez askorik.
Erabiltzaile berrientzako erabilerraza	BAI	EZ	EZ	Joomla eta Drupalekin zaila izan daiteke hasieran bere konfigurazioa edo gunearen itxura aldatzen jakitea edo
Erabiltzaileen kudeaketa erraza	EZ	EZ	BAI	Drupalen bitartez rolak sortu/esleitu eta baimen espezifikoak eman daitezke
Programazio-lengoaia	PHP	PHP	PHP	-

10.1.Taula. CMS ezberdinen ezaugarrien konparaketa.

Azkenik, azterketa sakon bat egin eta aukera bakoitza ebaluatu ostean, *Drupal* CMSa erabiltzea izan zen erabakia, hurrengo arrazoiengatik:

- Drupalen erraza da edukia gehitzea/sortzea. Eduki pertsonalizatu motak malguak dira eta aukera asko eskaintzen dituzte.
- Guneari gehitzeko hainbat modulu eskuragarri daude bere webgunean eta proiektu honetarako oso erabilgarriak diren moduluak aurkitu dira.
- Erabiltzaileak administratzea erraza da, rol berriak sortu eta baimenak zehaztu ditzakeen sistema integratu batekin. Funtzionalitate hori oso komenigarria zen proiektu honentzat.
- Mundu mailan garrantzitsuenak diren teknologia saltzaileen sailkapenak argitaratzen dituzten Gartner eta Forrester erakundearen txostenetan, CMS atalean, liderra den Acquia enpresak *Drupal* erabiltzen du oinarri bezala.

ProMeta proiektuak CMSari dagokionez antzeko helburuak dituenaz, aurreko arrazoi guztiak mantentzen dira. Gainera, *Drupal* erabiltzeak orain beste abantaila bat du, aurreko proiektuaren zati batzuk berrerabiltzeko aukera.

10.2.1 Wordpress

2003ko maiatzaren 27an jarri zen abian, edozein motatako web orrialdeak sortzera bideratuta. Jatorrian blogen sorkuntzan arrakasta handia lortu zuen, baina geroago web orrialde komertzialak sortzeko tresna nagusietako bat bilakatu zen.

WordPress PHP hizkuntzan garatzen da MySQL eta Apache exekutatzeko inguruneetarako, GPL lizentziapean eta software librea da.

Helburu orokorreko CMS ezagunena da. 2019ko martxoan Interneteko gune guztien % 33,4k eta eduki kudeatzaileetan oinarritutako gune guztien %60,3k erabiltzen zuten.

Arrakastaren arrazoietakoa bat garatzaile eta diseinatzaileen komunitate izugarria da, bere muinean programatzeaz edo komunitatearentzako pluginak eta txantiloak sortzeaz arduratzen dena.

10.2.2 Drupal

Doakoa, modularra, erabilera anitzekoa eta oso konfiguragarria da. Artikuluak, irudiak, fitxategiak argitaratzea ahalbidetzen du eta beste zerbitzu gehigarri batzuk ere eskaintzen ditu, hala nola foroak, inkestak, bozketak, blogak, erabiltzaileen administrazioa eta baimenak.

Drupal sistema dinamikoa da: bere edukia zerbitzariaren fitxategi estatikoetan gorde beharrean, orrien testu edukia eta bestelako ezarpenak datu base batean gordetzen dira eta web ingurunea erabiliz editatzen dira.

Doako programa da, GNU/GPL lizentziarekin, PHP-n idatzia eta MySQL-rekin bateragarria. Erabiltzaileen komunitate aktibo batek garatu eta mantentzen du. Aipagarria da kodearen eta sortutako orrien kalitatea, web estandarrak errespetatzea eta sistema osoaren erabilgarritasuna eta koherentzia.

Drupal-en diseinua bereziki egokia da Interneteko komunitateak eraikitzeke eta kudeatzeko. Malgutasun eta moldagarritasunagatik nabarmentzen da, baita eskuragarri dauden modulu osagarrien kopuru handiagatik ere, webgune mota ugari egiteko egokia da.

10.2.3 Joomla

Webgune dinamikoa eta interaktiboak garatzeko aukera ematen du. Webgune bateko edukia modu errazean sortu, aldatu edo ezabatzeko aukera ematen du administrazio panel baten bidez. Kode irekiko softwarea da, PHP-n programatua edo garatua eta GNU General Public License (GPL) lizentziapean argitaratua.

Joomla-k bere funtzionamendurako datu-base kudeatzailearekin sortutako datu-basea behar du (MySQL da ohikoena), baita Apache HTTP zerbitzaria ere.

Ondo eratutako HTML kodea sortzea, blogen kudeaketa, artikuluak inprimatzeko ikuspegiak, albisteen flash-a, foroak, inkestak (inkestak), egutegiak, gune bilaketak integratuak eta hizkuntza anitzeko laguntza dira Joomla-rekin sor daitezkeen tresnetako batzuk. Gaur egungo joerak direla eta apustu handia egiten ari da merkataritza elektronikoaren alde.

10.3 Dokumentazioa *Hostinga*

Proiektuaren dokumentazioaren webgunea interneten publikoki eskuragarri egon daitezen hosting zerbitzu bat erabili behar da. Doako aukerak bakarrik aztertu dira eta GitHub Pages erabiltzea erabaki da.

10.3.1 GitHub Pages

GitHubekin integratuta, automatikoki eraikitzen da webgunea kodea GitHub-era igotakoan. GitHub Pages-en konfigurazioa oso erraza, errepositoriotik bertatik egiten da. Webgune estatikoak bakarrik onartzen ditu, beraz, drupal-erako ez du balio, PHP erabiltzen baitu. Webguneak HTML, CSS, eta JavaScript-en idatzita egon daitezke edo Jekyll webgune estatiko sortzailea erabiliz.

10.3.2 GitLab Pages

GitHub Pages-en antzekoa da, baina aukera gehiago eskaintzen ditu. Webgune estatiko sortzaile guztiak onartzen ditu, Gastby, Jekyll, Hugo, etab. Hori bai, GitLab Pages-en webgune guztiak eraikitzeke Continuous Integration fitxategi bat behar da. GitHub-en ez da beharrezkoa, aukeretan aktibatzea nahikoa

da. Gure webguneak ez duenez erabiltzen webgune estatiko sortzailerik, GitHub Pages erabiltzea hobe da sinpleagoa delako.

10.3.3 Netlify

GitHub-ekin integratu daiteke, automatikoki eraikitzen da webgunea kodea GitHub-era igotakoan. Aurrekoek bezala, webgune estatikoak bakarrik onartzen ditu baina aukera zabalagoak ditu. Netlify-k edozein webgune estatikorako hosting-a eskaintzen du, funtzionalitate zabalagoekin. Kasu honetan webgune sinple bat eraikitzea bakarrik interesatzen zaigunez, aurreko aukerekin nahikoa daukagu.

10.4 Drupal *Hostinga*

Drupal webgunea interneten publikoki eskuragarri egon daitezen hosting zerbitzu bat erabili behar da. Doako aukerak bakarrik aztertu dira eta Pantheon erabiltzea erabaki da.

10.4.1 000webhost

Ez dauka GitHubekin integraziorik eta beraz kodea eskuz igo beharko litzateke aldi bakoitzean. PHP kodea onartzen du, eta beraz Drupalerako balio du. 000webhost-ek MySQL datu-basea eskaintzen du, xampp-ekin bateragarria. ProWF proiektuan aukera hau erabili da proiektuaren webgunerako. Drupal webgunea ez da eskuragarri jarri.

10.4.2 Heroku

GitHubekin integratu daiteke, automatikoki eraikitzen da webgunea kodea GitHub-era igotakoan. Herokuren konfigurazioa GitHub Pages-ena baino zailagoa da. PHP kodea onartzen du eta beraz, Drupalerako balio du. Defektuz PostgreSQL datu-basea eskaintzen du. Xampp-ek, berriz, MySQL eskaintzen du. Drupal modu sinplean instalatzeko ez dago prestatuta.

10.4.3 Pantheon

GitHub-ekin ondo integra daiteke, nahiz eta ez den horren sinplea. Drupal modu sinplean instalatzeko prestatuta dago. Drupal-erako honek ematen du aukera onena.

10.5 Datu-base Kudeaketa Sistema

Metodologiaren informazioa gordetzen duen datu-baserako DBKS egokia aukeratzea garrantzitsua da. Erabiliko ditugun tresnen eta *hosting*-aren menpekoea da, ez baitaude guztiak eskuragarri. MySQL erabiltzea erabaki da.

10.5.1 MySQL

Drupal-ekin bateragarria da, ematen dituen aukeretako bat da. Xampp-ek eskaintzen duen datu-basea da, eta beraz garapen lokalerako egokia. Xampp-eko phpMyAdmin bidez kontrola daiteke.

10.5.2 PostgreSQL

Drupal-ekin bateragarria da, ematen dituen aukeretako bat da. Xampp-ek defektuz ez du eskaintzen, baina gehigarri moduan aktibatzeak aukera ematen du. Xampp-eko phpPgAdmin gehigarriaren bidez kontrola daiteke.

10.6 Datu-basearen Sorrera

Metodologiaren informazioa datu-basean gordetzeko prozesua automatizatzea ezinbestekoa da. Hainbat aukera aztertu ondoren egokiena Xtext dela erabaki da.

10.6.1 Inferentzia Motorra

ProWF proiektuan inferentzia motorra erabiltzen da datu-basearen sorrerarako. Inferentzia motorra erabiltzen ez badugu bide honek ez dauka zentzurik.

10.6.2 Teneo

Teneo-k EMF eredu eta datu-base erlazionalen arteko mapaketa eskaintzen du <https://wiki.eclipse.org/Teneo>. Hibernate-rekin bateragarria da, objektu eta datu-base erlazional arteko mapaketarako tresna <https://hibernate.org/>. Aukera honek datu-basea automatikoki sortuko luke. Baina, datu-basea oso handia izango litzateke eta beharrezkoa baina elementu gehiago izango litzateke. Gainera, Teneo zaharkitua dago eta Eclipsen eta EMF-ren bertsio berriei ez da bateragarria.

10.6.3 Xtext

Aukera onena Xtext eta Xtend erabiliz ereduko datuekin SQL INSERT-ak sortzea da. Honek lan dezente eskatuko luke. Baina, flexibilitate asko emango luke, datu-basea nahi dugun bezala diseina dezakegu. Horrela, behar dugun informazioa bakarrik izango dugu datu-basean, eta errazagoa izango da honekin lan egitea.

10.7 Lanerako Ingurunea

Lanerako ingurune egokia aukeratzea garrantzitsua da. Baita ere ingurunearekin arazoren bat badago alternatibak izatea.

10.7.1 Makina Birtuala

Juanmak lanerako makina birtuala eskaini dit. Ordenagailu horretan aurreko proiektuak daude eta erabilitako softwarea instalatuta. Honetara VPN bidez konektatu beharko nintzateke. Honen arazoa da lokalean lan egitean baino makalago joango dela.

10.7.2 Ordenagailu Pertsonala

Ordenagailu pertsonala nire gustura konfiguratuta daukat eta erabiltzen dutan softwarea instalatuta. Makina birtualean softwarea instala daiteke baina errazagoa da nire konputagailuan falta den softwarea instalatzea. Izan ere, Drupal eta EHSIS bakarrik falta dira. Hori bai, bukaeran sistema zerbitzariko makina birtualera pasatuko da, besteek eskuragarri eduki dezaten.

10.8 Bertsio Kontrola

Bertsio kontrola oso garrantzitsua da, batik bat horrelako proiektuetan. Bertsio ezberdinak gordetzeko modu egokiena da, eta arazoren bat badago kodea berreskuratzeko aukera ematen du. GitHub erabiltzea erabaki da.

10.8.1 GitHub

Bertsio kontrolerako Git eta GitHub erabiltzea da aukera onena. Informazio guztia GitHub-eko errepositorio batean egongo da eta webgunea publikatuta egongo da. Horrela, beharrezkoa denean Juanmari erakutsi ahal izango diot egiten ari naizena. Gainera, bertsio kontrolak segurtasuna eta trazabilitatea ematen du.

10.8.2 GitLab

GitLab erabiltzea ere aukera ona izan daiteke, antzeko aukerak eskaintzen ditu. GitHub-ekin praktika gehiago daukat eta beraz ez dauka zentzurik GitLab-era aldatzeak. GitLab-ek bakarrik eskaintzen duen zerbait beharko bagenu orduan bai.

10.9 Metaeredua

Garrantzitsua da prozesuak definitzeko metaeredu egokia aukeratzea, horrek mugatuko baititu defini ditzakegun prozesuak.

10.9.1 SPEM

Software Process Engineering Metamodel CMOF formatuan dagoen metaeredu estandarra da, OMG erakundeak sortua.

10.9.2 UMA

Unified Method Architecture metaereduaren helburua edozein metodologia modelatu ahal izatea da. SPEM metaereduan oinarrituta dago, baina ecore formatuan dago. Eclipse tresnarekin lan egiteko formatu hau behar da, EMF formatu horretarako prestatuta dagoelako

10.10 Metodologia

Prozesua definitzeko erabiliko ditugun metodologiak ongi aukeratzea garrantzitsua da. OpenUP eta ABRD metodologiak erabiltzea erabaki da.

10.10.1 OpenUP

OpenUP softwarea garatzeko metodo eta prozesu bat da, teknologien sektoreko enpresa multzo batek proposatutakoa, zeintzuk 2007an Eclipse Fundazioari dohaintzan eman zioten. Fundazioak lizentzia libre bezala argitaratu du eta eredu gisa mantentzen du Eclipse Process Framework (EPF) proiektuaren barruan.

Metodologia honek garrantzi handia izan du proiektu osoan zehar. Batetik, proiektuaren helburuetako bat metodologia baten definizioa eta ezarpena izan da eta OpenUP izan da aukeratutako metodologia. Bestetik, proiektuaren elaborazio prozesurako OpenUP metodologia jarraitu da, dokumentazioa bilduz eta proiektuaren kontrola eramanez.

10.10.2 RUP

Kontuan hartuta proiektua eta proiektuko taldea txikiak direla, OpenUP metodologia erabiltzea nahikoa da. Gainera, metodologiako lehenengo bi edo hiru fase egitearekin nahikoa izango da. Talde handiagoa edo proiektu konplexuagoa izango balitz agian RUP bezalako metodologia konplexuagoa bat beharko genuke.

Metodologiaren ezarpenari dagokionez, sinpleagoa da OpenUP metodologia ezartzea txikiagoa delako. Gainera, OpenUP metodologiarekin lan egiteko Eclipse-ren tresnak erabil daitezke. Tresna horiek libreak dira, eta egileak esperientzia du horiek erabiltzen.

10.10.3 ABRD

Agile Business Rules Development metodologiaren eredua ere erabili da. Horrela, bi metodologia erabiliz ziurtatzen da sistemaren egitura egokia dela. ABRD negozio erregela aplikazioa garatzeko metodologia da, erregela motorra eta erregelak kudeatzeko sistema erabiliz.

10.11 Metodologia Editorea

Behin metodologiak aukeratuta, metodologia horiek nola editatuko ditugun erabaki behar da. Kasu honetan 3 aukera ematea erabaki da: EPF Composer, editoe grafikoa eta testu editorea.

10.11.1 EPF Composer

Metodologia definitzeko aukera argiena da, horretarako egindako tresna baita. Jadanik hainbat metodologia definituta daude tresna honekin: OpenUp, ABRD, Scrum... Metodologia horietako bat erabili

nahi badugu ez daukagu ezer diseinatu beharrik. Horiek oinarri bezala hartuta ere errazagoa da beste metodologia bat definitzea. Metodologiaren webgunea sortzeko aukera ematen du, dokumentazio moduan erabili ahal izateko. Nahiko zaharkitua dago, Java 8 32 biteko bertsioa eskatzen du.

10.11.2 Rational Method Composer

EPF Composer-en antzeko tresna, baina IBMrena da eta ordainpekoa. RUP metodologia du oinarrian eta horrekin lana egin nahi badugu aukera interesgarria izan daiteke.

10.11.3 Editore Grafikoa

EPF Composer baino sinpleagoa den editorea edukitzea ondo etorriko litzateke. Ikono berdinak partekatuko lituzke, zuhaitz egitura sinpleagoa edukiko luke.

10.11.4 Testu Editorea

Editore grafikoa baino sinpleagoa den testu editore bat edukitzea ere komenigarria da. Batzuetan, editore grafikoa baino erosoagoa da testu editorea erabiltzea. Onena eredu bi editoreekin aldatzeko aukera izatea izango litzateke, eta edozein momentutan bien artean aldatzeko aukera izatea.

10.12 Prozesua Bistaratu

Prozesua bistartzeko modua izatea ezinbesteko da erabiltzaileek prozesua kontsultatu ahal izateko eta prozesuan hobekuntzak egiteko. EPF Composer-ekin sortutako webgunea nahiko dela erabaki da.

10.12.1 DOT

ProWF proiektuan prozesua definitzeko eta bistartzeko DOT lengoaia erabili da. Beraz, aukera ona izan daiteke prozesua bistartzeko. Hori bai, honek lan dezente gehituko luke. Izan ere, eredu DOT lengoiara pasatzea eskatuko luke metaereduak erabiliz.

10.12.2 XPD

XPD ere aukera egokia izan daiteke estandarra delako. Baina, oraingo DOT aukera hobe izango litzateke, sinpleagoa delako.

10.12.3 Webgunea

Prozesua bistartzeko EPF Composer-ekin publikatzen den webgunea erabiltzea da aukera onena. Izan ere, webguneak metodologiaren informazio guztia dauka, eta honen sorrera automatikoa da. Beraz, prozesua EPF Composer-ekin definitzea komenigarria da.

10.13 Denboraren Kontrola

Denboraren kontrola modu zehatz eta antolatuan egiteko tresna egokia aukeratzeak asko laguntzen du. Hainbat aukera aztertu dira eta Toggl Track aukeratu da.

10.13.1 Toggl Track

Tenporizadore baten bidez ataza bakoitzean pasatako denbora neurtu daiteke. Eskuz ere aldatu daiteke denbora ahaztu egin bazaigu. Abisatu egiten du inaktibo bagaude eta denbora neurtzen ari bagara. Aspaldian denborarik ez badugu neurtu ere abisatzen du, ez ahazteko. Denbora proiektutan, atazatan eta kategoriatan sailkatu daiteke. Edozein momentuko denboraren estatistikak ikusi eta deskargatu daitezke. Gailu mota guztietarako aplikazioak daude, mugikorrerako ordenagailurako, nabigatzaileerako, etab.

10.13.2 Clockify

Toggl Track-en antzeko funtzionamendua dauka, denbora neurtzeko tresna batek eduki beharreko oinarritzko funtzionalitatea dauka. Doako planean aukera batzuk gutxiago eskaintzen ditu.

10.13.3 WakaTime

Plugin moduan instalatu daiteke editore askotan. Gure kasuan VSCode eta Eclipse editoreetan instalatu da, horiek izan baitira erabili diren editoreak. Denbora automatikoki neurtzen du eta metrikak sortzen ditu. Adibidez, programazio-lengoaia eta proiektu bakoitzean pasatako denbora erakusten du. Ez du balio editoreetatik kanpo pasatako denbora neurtzeko. Hala ere, Toggl Track-en osagarri ona da, estatistika zehatzagoak eskaintzen dituelako. Doako planean 2 asteko historia bakarrik erakusten du.

10.14 Diagramak

Diagramak beharrezkoak dira hainbat kontzeptu modu grafikoan adierazteko. Hainbat aukera aztertu ondore, diagrama guztiak egiteko PlantUML erabiltzea erabaki da.

10.14.1 PlantUML

Hainbat motatako diagramak sortzeko aukera ematen duen softwarea da. Eskuz sortzen dira diagramak eta bi aukera daude aplikazioa erabiltzeko: online edo offline. Proiektuan zehar aplikazio hau erabili izan da dokumentazioan agertzen diren diagrama gehienak egiteko, bezeroak dokumentazio ulergarria jaso dezan.

10.14.2 Draw.io

Hainbat motatako diagramak sortzeko aukera ematen duen softwarea da. Eskuz sortzen dira diagramak eta bi aukera daude aplikazioa erabiltzeko: online edo offline. Azkenean diagrama guztiak egiteko PlantUML tresna erabiltzea erabaki da.

10.14.3 UMLDoclet

Javadoc metadatuak erabiltzen ditu PlantUML diagramak automatikoki sortzeko eta HTML dokumentazioan txertatzeko. Diagramak irudi klikagarri gisa txertatzen dira eta pakete eta klaseko dokumentaziora estekatzen dira eskuragarri dagoenean. Pakete dependentzia, pakete eta klase diagramak sortzen ditu elementu guztietarako. ModelEditor azpisistemaren dokumentazioan diagramak gehitzeko aukera egokia da.

11 Proposatutako Sistemaren Deskribapena

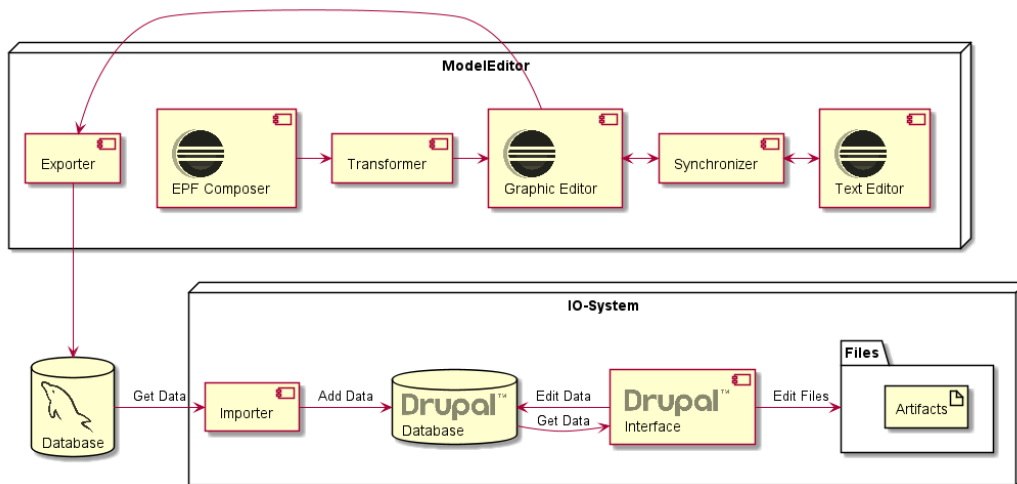
Kapitulu honetan planteatutako arazoa konpontzeko proposatzen den sistema, bere osagaiak eta bere ezaugarriak deskribatzen dira.

11.1 Azpisistemak

ProWF sistema bi azpisistema ezberdinetan bananduta egongo da: **ModelEditor** eta **IO-System**.

- **ModelEditor:** Sortutako editore grafikoa eta testu editorea erabiliz prozesuaren eredua editatzeko aukera emango du. Sistema honen ardura prozesu ingeniari rola izango du.
- **IO-System:** CMS baten bitartez kudeatutako web-aplikazioa izango da. Helburua metodologia jarraitzen duten proiektuen informazioa gordetzea da. Rol bakoitzak metodologian dituen ataza berdinak bete beharko ditu.

Idea orokor bat egiteko, 11.1. Irudian sistema osoaren arkitektura ikus daiteke, azpisistematatan banatuta.



11.1. Irudia. ModelEditor eta IO-System azpisistemen arkitektura.

Ondoren, azpisistema bakoitzaren hainbat aspektu azalduko ditugu: arkitektura, erabilpen kasuak, diseinua eta dokumentazioa.

Bukatzeko, sistemaren proba eta hedapena nola egingo diren azalduko da eta etorkizunerako hobekuntza posibleak aurkeztuko dira. Horrela, proiektu honi jarraipena ematea erraztuko da.

11.2 ModelEditor

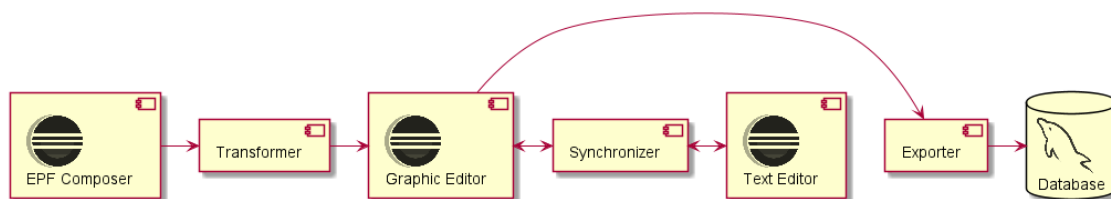
Sortutako editore grafikoa eta testu editorea erabiliz prozesuaren eredua editatzeko aukera emango du. Sistema honen ardura prozesu ingeniari rola izango du.

11.2.1 Arkitektura

ModelEditor azpisistemak honako osagaiak izango ditu. 11.2. Irudian ikus daitezke osagaia hauen arteko erlazioak.

- **EPF Composer:** Metodologiak modu grafikoan editatzeko aukera ematen du. Metodologiaren webgunea sortu daiteke bertatik.

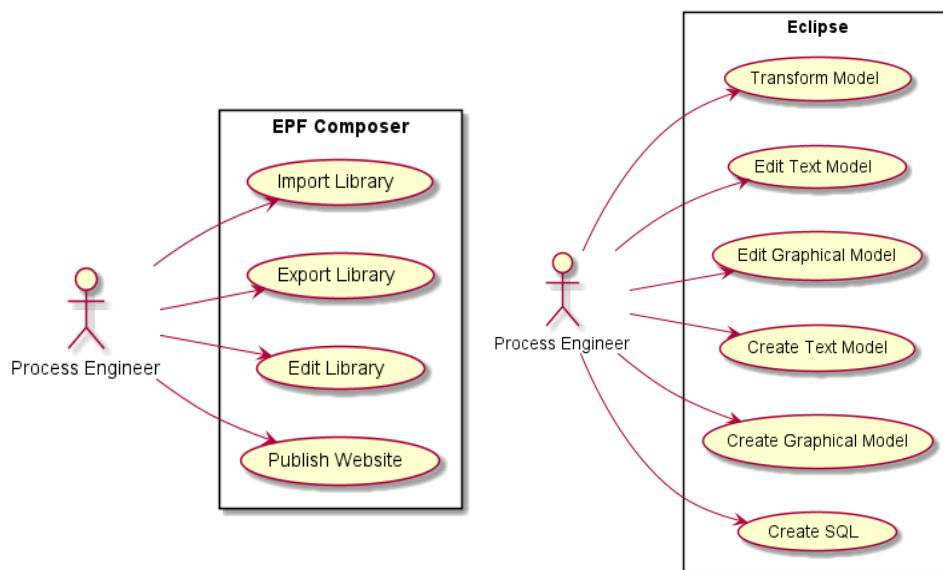
- **Formatu Aldatzailea:** Metodologiaren formatua aldatzen du XMItik UMara editore grafikoak erabili ahal izan dezan.
- **Editore Grafikoa:** Metodologiak modu grafikoan editatzeko aukera ematen du, metodologiaren jatorrizko ikonoak erabiliz.
- **Testu Editorea:** Metodologiak testu bidez editatzeko aukera ematen du, metaeredutik sortutako gramatika erabiliz.
- **Editore Sinkronizatzailea:** Metodologiaren informazioa editore batekin aldatutakoan bestea ere aldatzeaz arduratzen da.
- **Kode Sortzailea:** Metodologiaren informazioa datu-basean gorde ahal izateko SQL kodea sortzen du.



11.2. Irudia. ModelEditor azpisistemaren arkitektura.

11.2.2 Erabilpen Kasuak

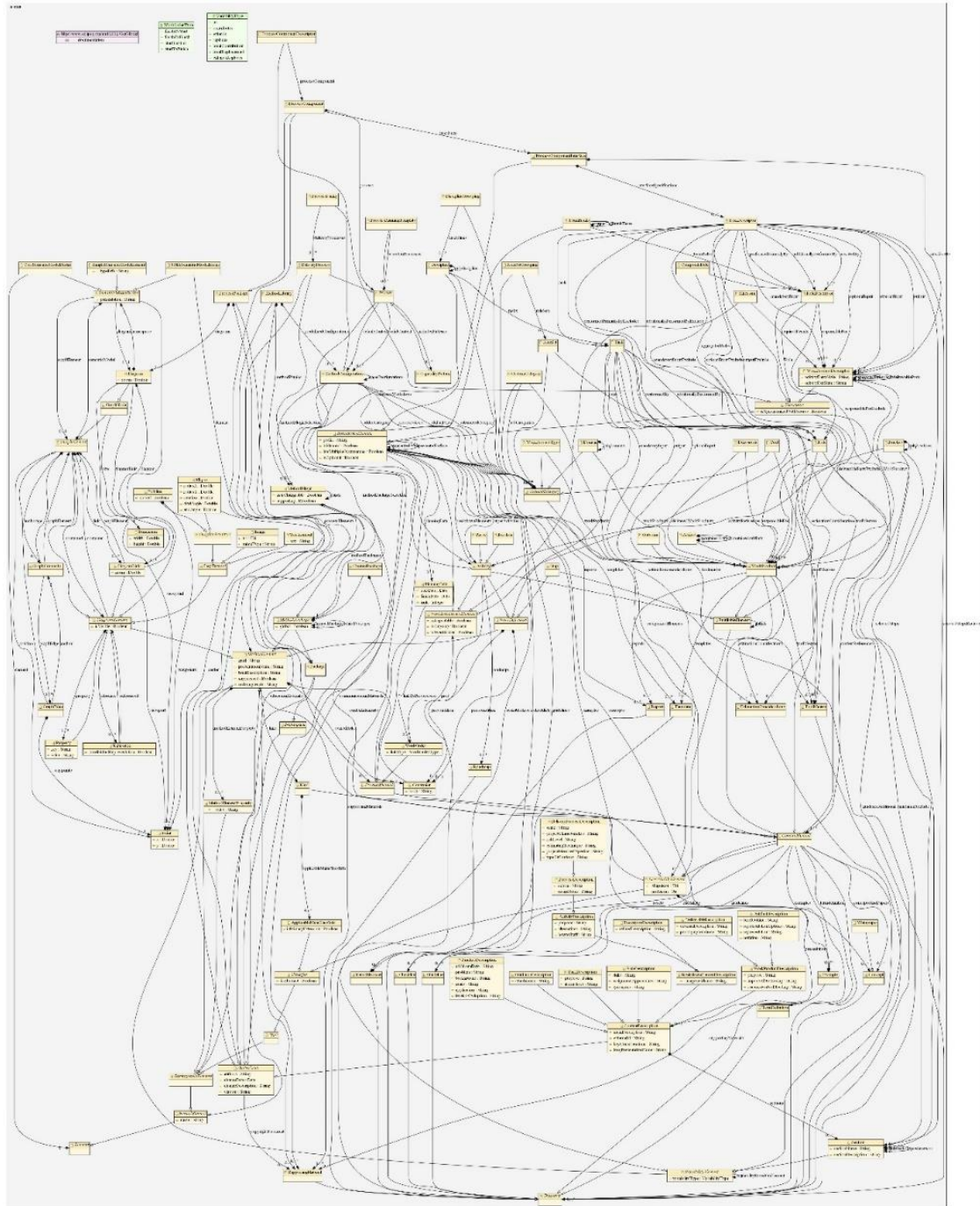
Sistemaren betekizunak zehazteko beharrezko da erabilpen kasuak eta aktoreak zein diren jakitea. Azpisistemaren erabilpen kasuak bi multzotan bana daitezke: EPF Composer-enak eta Eclipse-renak. EPF Composer-etik metodologiaren liburutegiak inportatu, esportatu eta aldatzeko aukera daukagu. Gainera, metodologiaren webgunea ere sor daiteke dokumentazio moduan erabiltzeko. Eclipsen ereduarekin zerikusia duten erabilpen kasuak daukagu: eredua eraldatu, sortu eta editatu. Horrez gain, SQL kodea sortzeko aukera ere badago. Erabilpen kasu hauek guztiak prozesu ingeniariak edo antzeko rola duen pertsonak egin beharko ditu. Hala ere, metodologia bat adosterakoan erabiltzaile mota askok hartu beharko dute parte, guztien beharretara egokitu behar baita. 11.3. Irudian ModelEditor azpisistemako erabilpen kasuen eredua ikus daiteke.



11.3. Irudia. ModelEditor azpisistemako erabilpen kasuen eredua.

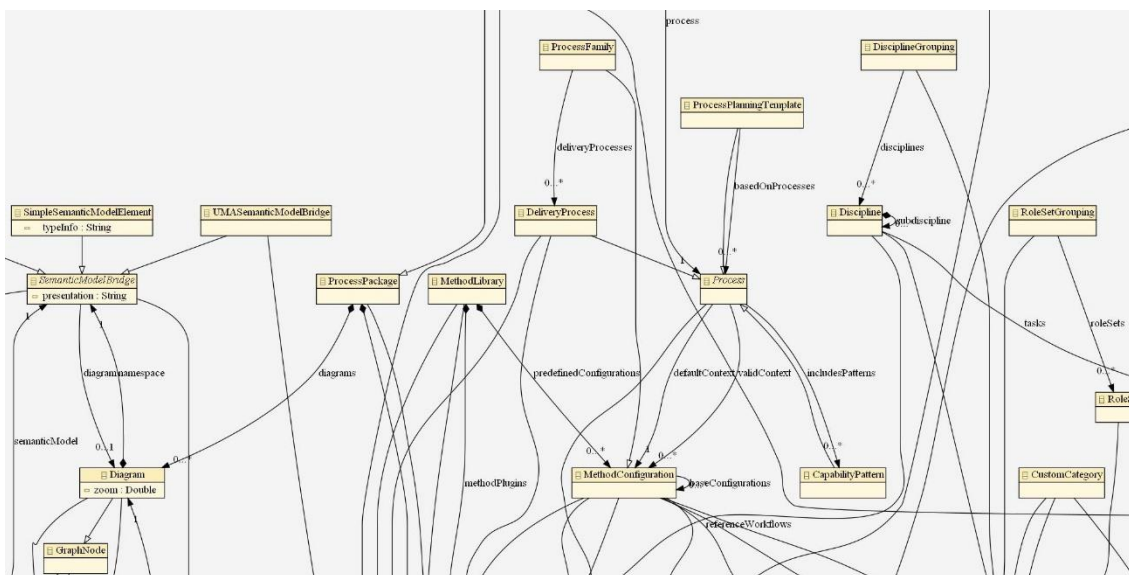
11.2.3 Diseinua

Metodologientzako metaeredu moduan UMA metaeredua erabiltzea erabaki da, SPEM estandarren bertsio hobetua. Metaeredu hau oso konplexua da, edozein metodologia definitzeko beharrezko klase eta atributuak baititu. Metaereduaren tamainaren ideia bat egiteko 11.4. Irudia ikus daiteke.



11.4. Irudia. UMA metaeredua graphviz formatuan.

Diagrama handiegia denez, ez dira klaseen izenak ikusten. Horregatik, klaseak eta atributuak nolakoak diren ikusteko, 11.5. Irudian zati bat gertuagotik agertzen da. Bertan, ezagunak diren kontzeptu batzuk aurki ditzakegu: metodologia, prozesua, diagrama, disziplina, rola, etab. Klaseen arteko hainbat erlazio mota ere ikus daitezke.



11.5. Irudia. UMA metaereduaren zati bat graphviz formatuan.

11.2.4 Dokumentazioa

ModelEditor azpisistemaren funtzionamendua dokumentatuko da hobeto uler dadin. Garapenerako Eclipse-ren tresnak erabili dira orokorrean. EPF Composer eta Eclipse IDE izan dira tresna nagusiak baina Visual Studio Code ere erabili da editore moduan. Programazio lengoia nagusiak Java eta Xtend izan dira.

Hasteko, metodologia aukeratu behar dugu eta EPF Composer tresnan kargatu. Metodologiak XMLi formatuan egon behar du eta UMA metaereduarekin bat etorri behar du. Metodologia propioa ere defini daiteke tresna horrekin existitzen den metodologia baten aldatetarik eginez. Proiektu honetarako EPF Practices liburutegi osoa erabiltzea erabaki da. Bertan OpenUP eta ABRD metodologiak daude, 11.6. Irudian ikus daitezkeen moduan. Metodologia gehiago ere deskargatu daitezke Eclipseren webgunetik, Scrum eta XP adibidez. Etorkizunean RUP metodologia definitzeko aukera ere badago.

Eclipse Process Framework Composer - C:\Users\juleti\GitHub\ProMeta-ModelEditor\org.eclipse.epf.library

File Edit Search Configuration Window Help

Select a configuration

Library

- core
 - practice
 - process
 - abrd
 - openup
 - base
 - Method Content
 - Processes
 - Capability Patterns
 - General
 - deploy_release_to_production
 - prep_dsc_tmng
 - prepare_for_release
 - provide_product_training
 - Management
 - initiate_project
 - plan_manage_iteration
 - Phase Iteration Templates
 - construction_phase_iteration
 - elaboration_phase_iteration
 - inception_phase_iteration
 - transition_phase_iteration
 - Technical
 - agree_technical_approach
 - develop_architecture
 - develop_solution
 - identify_and_refine_requirements
 - ongoing_tasks
 - test_solution
 - publish
 - all_epf_practices
 - openup
 - tech.abrd

openup_lifecycle

| Presentation Name | Index | Predecessors | Model Info | Type | Plann... | Repe... | Multi... | Ongo... | Event... | Optio... |
|---|-------|-----------------|--|-------------|-------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| OpenUP Lifecycle | 0 | | | Delivery... | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Inception Phase | 1 | | | Phase | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Inception Iteration [1..n] | 2 | | extends 'inception_phase_iteration, pro... | Iteration | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Initiate Project | 3 | | extends 'initiate_project, process.openu... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Plan and Manage Iteration | 5 | | extends 'plan_manage_iteration, process... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Identify and Refine Requirements | 15 | 3 | extends 'identify_and_refine_requirement... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Agree on Technical Approach | 20 | 3 | extends 'agree_technical_approach, proc... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Lifecycle Objectives Milestone | 22 | 2 | | Milestone | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Elaboration Phase | 23 | 22,1 | | Phase | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Elaboration Iteration [1..n] | 24 | | extends 'elaboration_phase_iteration, p... | Iteration | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Plan and Manage Iteration | 25 | | extends 'plan_manage_iteration, process... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Identify and Refine Requirements | 35 | | extends 'identify_and_refine_requirement... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Develop the Architecture | 40 | | extends 'develop_architecture, process.op... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Develop Solution Increment | 48 | | extends 'develop_solution, process.openu... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Test Solution | 54 | | extends 'test_solution, process.openup.ba... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Ongoing Tasks | 57 | | extends 'ongoing_tasks, process.openup... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Lifecycle Architecture Milestone | 59 | 24 | | Milestone | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Construction Phase | 60 | 59,23 | | Phase | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Construction Iteration [1..n] | 61 | | extends 'construction_phase_iteration... | Iteration | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Plan and Manage Iteration | 62 | | extends 'plan_manage_iteration, process... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Identify and Refine Requirements | 72 | | extends 'identify_and_refine_requirement... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Develop Solution Increment | 77 | | extends 'develop_solution, process.openu... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Test Solution | 83 | | extends 'test_solution, process.openup.ba... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Ongoing Tasks | 86 | | extends 'ongoing_tasks, process.openup.b... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Develop Product Documentation and Training | 89 | | extends 'prep_dsc_tmng, process.openup.b... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Initial Operational Capability Milestone | 93 | 61 | | Milestone | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Transition Phase | 94 | 93,60 | | Phase | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Transition Iteration [1..n] | 95 | | extends 'transition_phase_iteration, pro... | Iteration | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Plan and Manage Iteration | 96 | | extends 'plan_manage_iteration, process... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Develop Solution Increment | 105 | | extends 'develop_solution, process.openu... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Test Solution | 111 | | extends 'test_solution, process.openup.ba... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Finalize Product Documentation and Training | 114 | | extends 'prep_dsc_tmng, process.openup.b... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Prepare for Release | 119 | | extends 'prepare_for_release, process.op... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |
| Provide Product Training | 124 | 105,111,96,1... | extends 'provide_product_training, proc... | Activity | ☒ | ☐ | ☐ | ☐ | ☐ | ☐ |

Description | Work Breakdown Structure | Team Allocation | Work Product Usage | Consolidated View

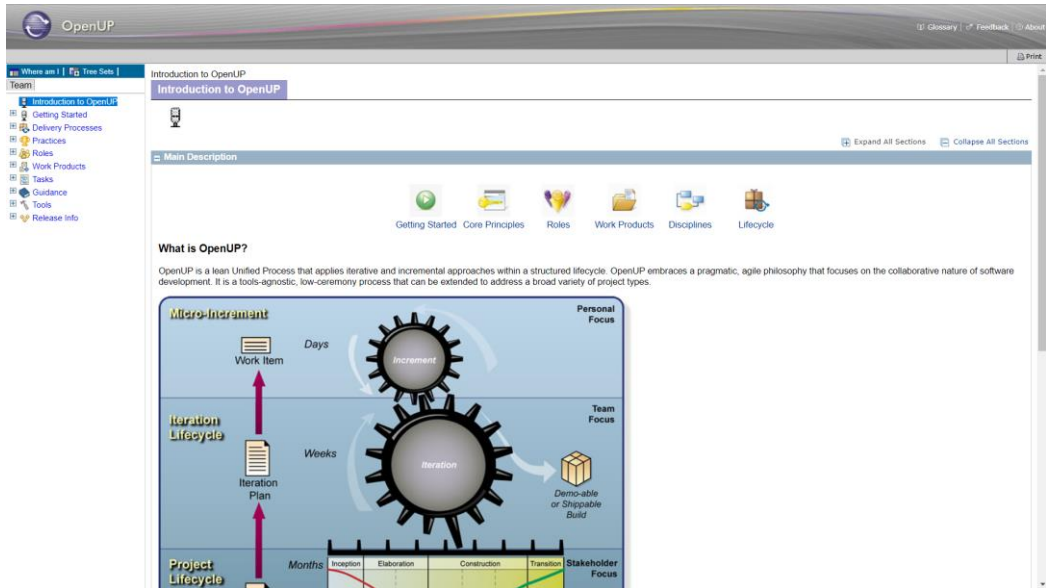
Properties Problems

Property Value

C:\Users\juleti\GitHub\ProMeta-ModelEditor\org.eclipse.epf.library\process.openup.base\deliveryprocesses\openup_lifecycle\model.xml

11.6. Irudia. OpenUP bizi-zikloa EPF Composer tresnan.

Behin metodologia kargatuta dugula hainbat aukera ditugu. Esan bezala, EPF Composer tresna bera erabil daiteke metodologian aldaketak egin nahi baditugu. Beste aukera garrantzitsu bat metodologiaren webgunea sortzea da, dokumentazio moduan oso erabilgarria dena. Adibidez, OpenUP metodologiaren webgunea ikus daiteke 11.7. Irudian.



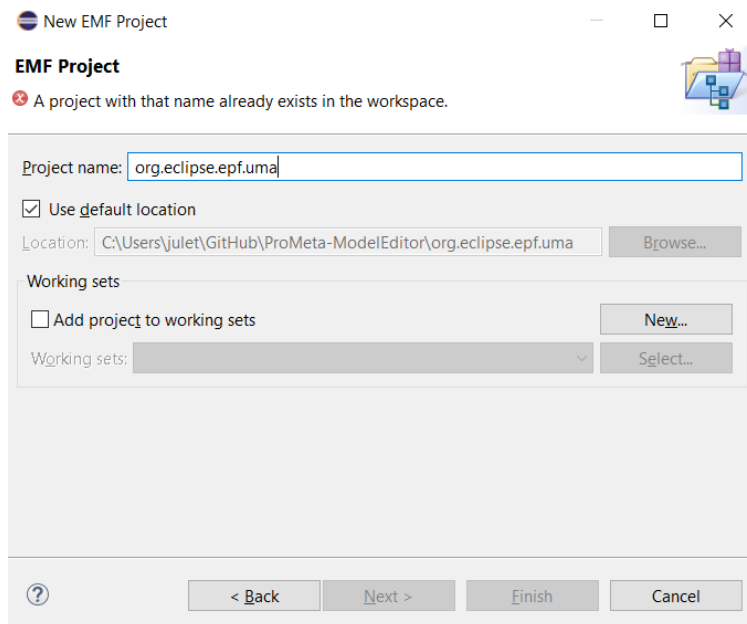
11.7. Irudia. OpenUP metodologiaren webgunea.

Metodologia nahi dugun moduan dagoenean hurrengo pausura pasa gaitezke. Aurrerago ere edukiko dugu aldaketak egiteko aukera beharrezkoa bada. Arkitekturan ikus daitekeen moduan, hurrengo pausua XSLT erabiliz ereduan hainbat eraldaketa egitea da. Izan ere, dagoen bezala uzten badugu, eredia ez dator bat UMA metaereduarekin eta errore pila bat ematen ditu. 11.8. Irudian agertzen den XSLT fitxategiak 600 lerro baino gehiago dituen arren, nahikoa da Java fitxategi bat exekutatzea eta segundu gutxi batzutan eraldatutako .uma fitxategiak lortzen ditugu.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:transform version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:org.eclipse.epf.uma="http://www.eclipse.org/epf/uma/1.0.6/uma.ecore" xmlns:org.eclipse.epf.uma.resourcemanager="http://org.eclipse/epf/uma/resourcemanager.ecore">
  <!--Identity-->
  <xsl:template match="@*|node()">
    <xsl:copy>
      <xsl:apply-templates select="@*|node()" />
    </xsl:copy>
  </xsl:template>
  <!--Root-->
  <xsl:variable name="root" select="/org.eclipse.epf.openup.uma/src/*"/>
  <!--ResourceManager-->
  <xsl:template match="org.eclipse.epf.uma.resourcemanager:ResourceManager"/>
  <!--MethodLibrary-->
  <xsl:template match="org.eclipse.epf.uma:MethodLibrary">
    <xsl:copy>
      <xsl:apply-templates select="@*|node()" />
      <xsl:element name="predefinedConfigurations">
        <xsl:attribute name="href">
          <xsl:value-of select="'configurations/all_epf_practices.uma#_QD87wFRHed2CWscN8Mx6rg'"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:copy>
  </xsl:template>
</xsl:transform>
```

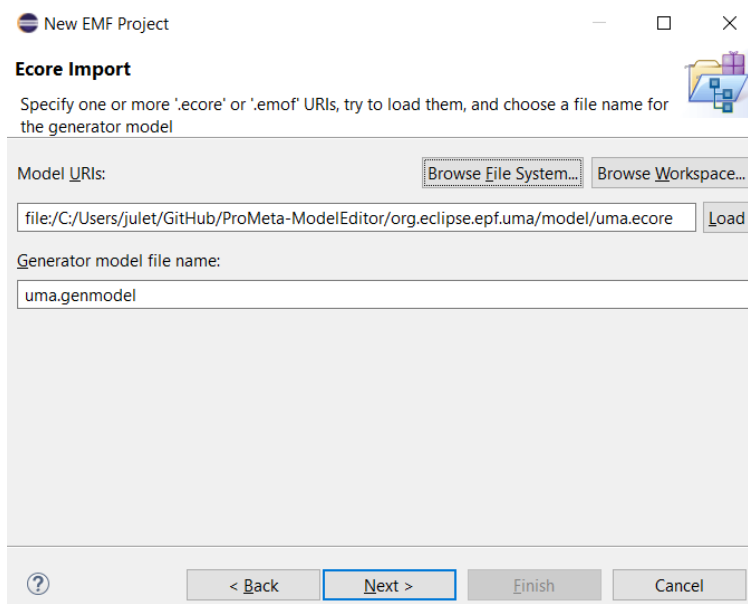
11.8. Irudia. Eraldaketa egiteko XSLT fitxategia.

Esan bezala, erabiliko dugun metaeredua UMA da, jadanik definituta dagoena. Beraz, lehenik metaeredu hori deskargatu behar da Eclipse-ren webgunetik. Metaereduarekin lan egiteko EMF proiektu bat sortuko dugu, 11.9. Irudian ikusten den bezala.



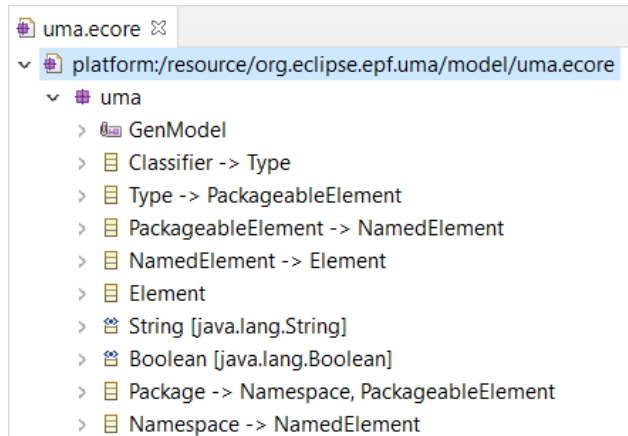
11.9. Irudia. EMF proiektuaren sorrera Eclipsen.

Proiektuaren izena aukeratu ondoren inportatzeko *Ecore model* aukeratuko dugu eta 11.10. Irudian dagoen leihora eramango gaitu. Bertan `uma.ecore` metaeredua aukeratu beharko dugu eta *Next* botoia sakatu. Hurrengo lehioan *Finish* sakatzen badugu proiektua sortuko da.



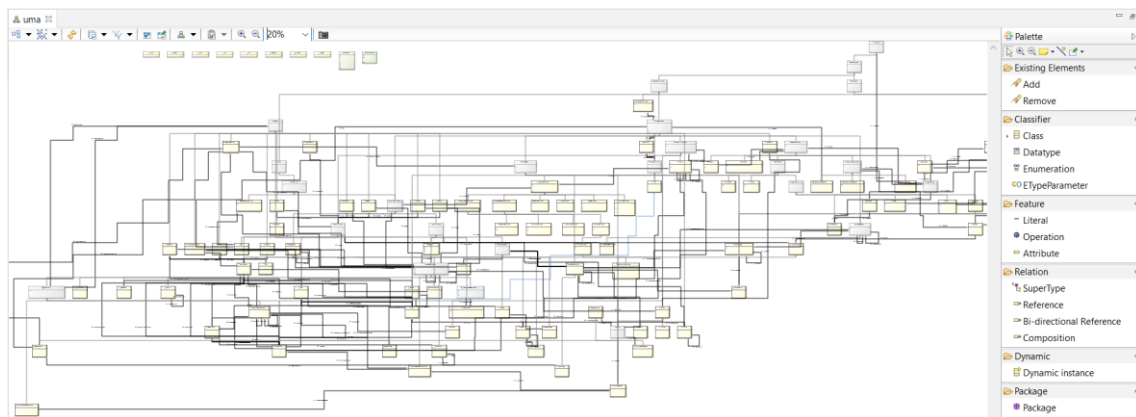
11.10. Irudia. Uma metaeredua inportatzen Eclipsen.

Proiektu horrek `uma.ecore` eta `uma.genmodel` fitxategiak izango ditu. Fitxategi horiek *Sample Reflective Ecore Model Editor* erabiliz ireki eta editatu daitezke. Adibidez, `uma.ecore` fitxategiaren zati bat ikus daiteke 11.11. Irudian.



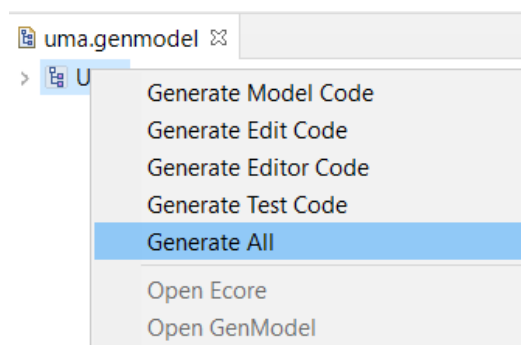
11.11. Irudia. Uma.ecore fitxategia Eclipsen.

Hurrengo pausua `uma.aird` fitxategia sortzea izango da, metaereduaren diagrama bat sortzeko klase eta atributu guztiekin. Horretarako, `uma.ecore` fitxategian *Initialize Ecore Diagram* aukeratu dugu eta ondoren *Entities in a Class Diagram*. Diagramaren izena aukeratu eta *Finish* botoia sakatutakoan klase diagrama sortuko da. 11.12. Irudian ikus daitekeen moduan, metaeredua osoa handia da eta klase pila bat daude. Aldaketaren bat egitea nahiko bagenu eskuineko paleta erabili beharko genuke.



11.12. Irudia. Uma metaereduaren klase diagrama.

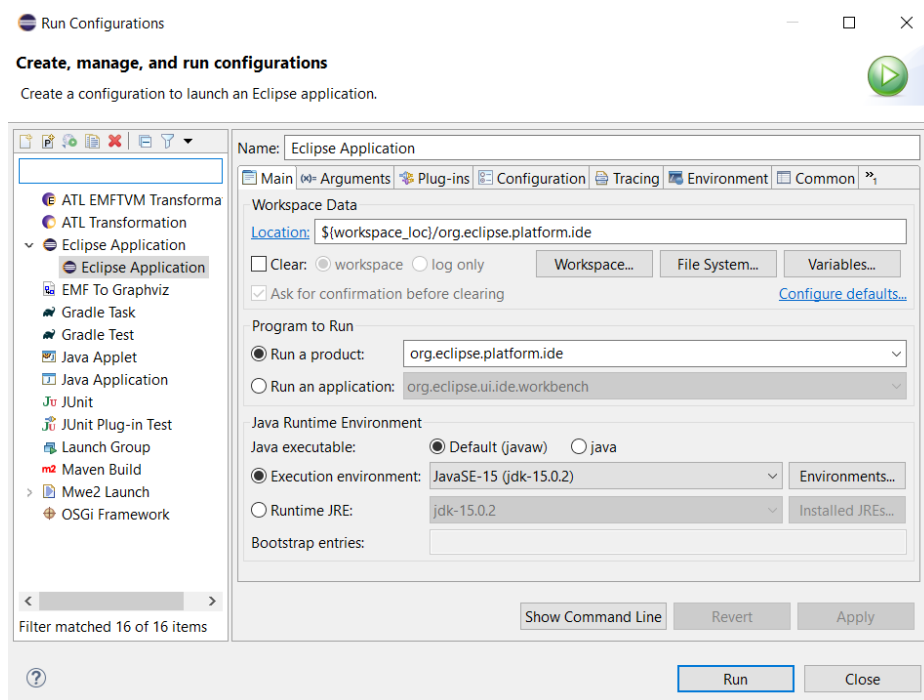
Behin metaereduaren proiektua prest daukagula, editore grafikoa sor dezakegu. Horretarako, `uma.genmodel` fitxategia ireki beharko dugu. Kodea sortzeko hainbat aukera daude, baina sinpleena *Generate All* aukeratzeko da. 11.13. Irudian ikusten den moduan. Izan ere, lehenengo 3 aukerak beharrezkoak dira editorea sortzeko eta 4. aukera probentzako da.



11.13. Irudia. Editore grafikoaren kodearen sorrera.

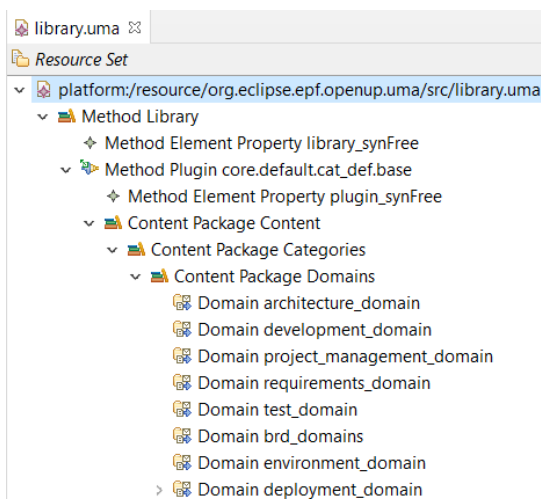
Lehenengo aukerak proiektuan bertan sortuko du ereduaren Java kodea. Beste aukera bakoitzerako Java proiektu berri bat sortuko da automatikoki. Honekin jadanik editore grafikoa daukagu, baina itxura hobea edukitzea nahi badugu ikonoak gehitu beharko ditugu `org.eclipse.uma.edit` proiektuan.

Dena prest daukagunean editorea martxan jar dezakegu *Run As > Eclipse Application* aukeratuz. Irekitzen dugun lehenengo aldia bada, konfiguratu egin beharko dugu. Adibidez, 11.14. Irudian agertzen den konfigurazioa aukera daiteke.



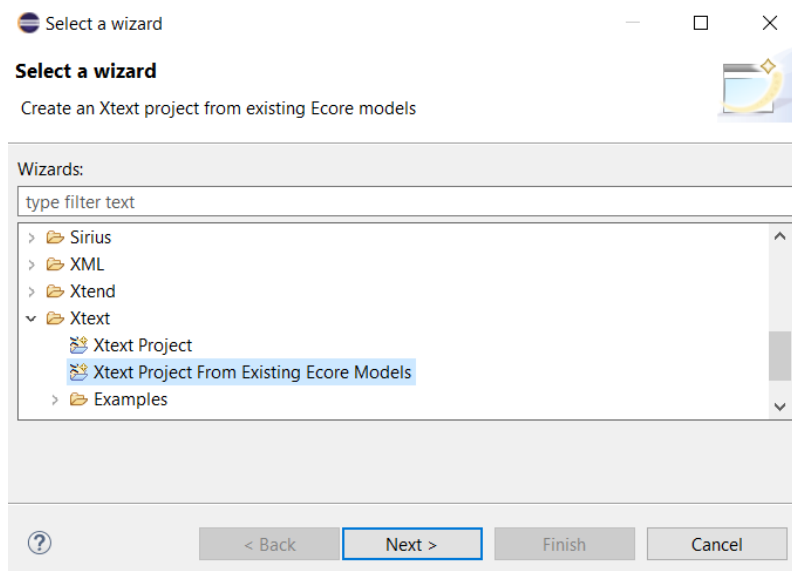
11.14. Irudia. Editore grafikoa martxan jartzeko konfigurazioa.

Martxan jartzen dugunean Eclipse aplikazio berri bat irekiko da. Bertan proiektu berri bat sortu beharko dugu eta XSLT eraldaketatik lortutako `.uma` fitxategiak kargatu. Edozein `.uma` fitxategi irekitzeko aukera izango dugu *Uma Model Editor* erabiliz. Editore honek EPF Composer-en antza du ikonoak eta zuhaitz egitura partekatzen dituelako. Adibidez, `library.uma` fitxategia irekitzen badugu metodologiaren eduki osoa bistaratuko da, 11.15. Irudian ikus daitekeen bezala.



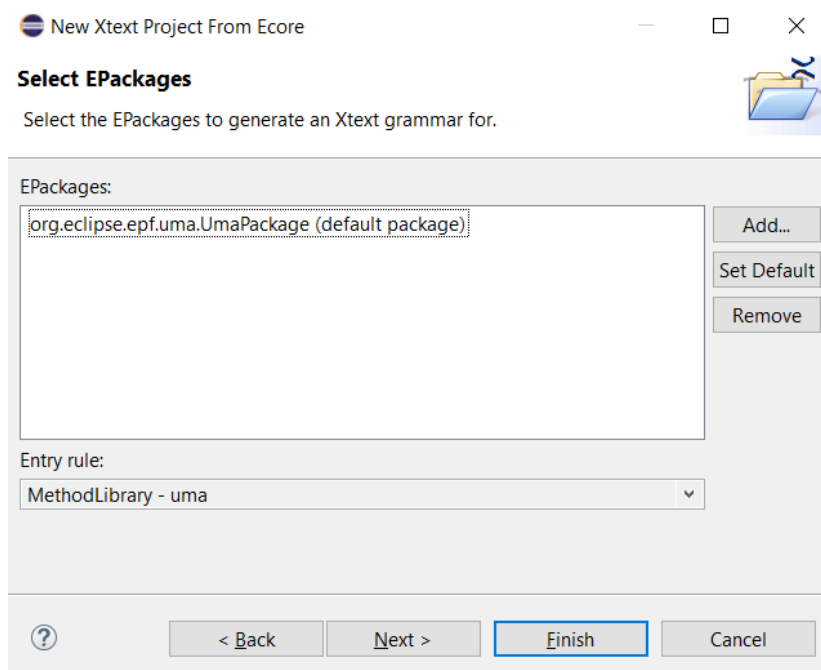
11.15. Irudia. Editore grafikoarekin `library.uma` fitxategia bistaratzen.

Editore grafikoa bukatuta dagoenez, testu editorearekin has gaitezke. Testu editorea sortzeko Xtext tresna erabiliko dugu. Helburua DSL bat sortzea denez, gramatika bat definitu beharko dugu. Metaeredua erabil dezakegu gramatika automatikoki sortzeko, 11.16. Irudian agertzen den *Xtext Project From Existing Ecore Models* aukera erabilita.



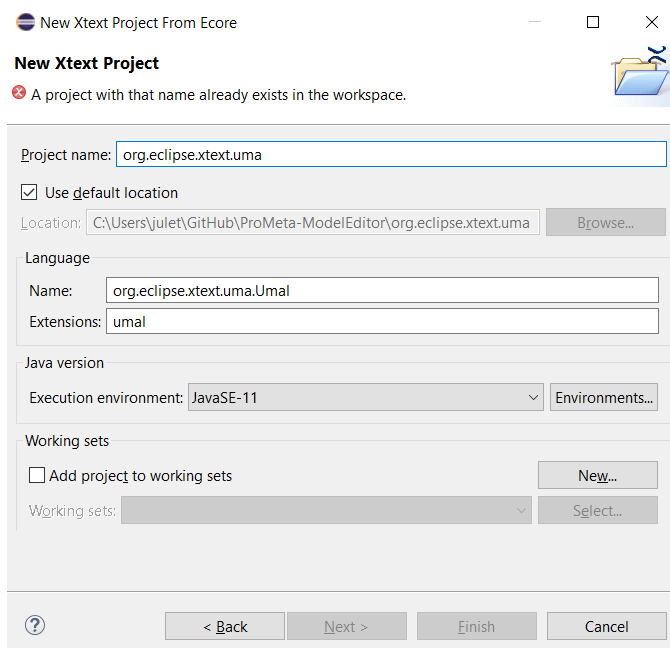
11.16. Irudia. Xtext proiektua sortzen Eclipsen.

Hurrengo lehioan *Add* botoia sakatu eta `uma.genmodel` fitxategia aukeratuko dugu, 11.17. Irudian ikus daitekeen bezala. Beheko aukeretan metaereduaren erroa aukeratu behar da, kasu honetan *MethodLibrary* dena.



11.17. Irudia. Uma metaeredutik Xtext proiektua sortzen.

Hurrengo leihoan proiektuaren izena, lengoariaren izena eta fitxategien luzapena aukeratu beharko ditugu, 11.18. Irudian ikusten den moduan.



11.18. Irudia. Xtext proiektuaren eta lengoaiaren izenak aukeratzen.

Amaitutakoan 11.19. Irudian agertzen diren 5 Xtext proiektuak sortuko dira. Lehenengoa proiektu nagusia da, gramatika eta fitxategi garrantzitsuenak izango dituen. Beste bi proiektuek editorearen kodea dute eta probarako bi proiektu ere badaude.

- >  org.eclipse.xtext.uma
- >  org.eclipse.xtext.uma.ide
- >  org.eclipse.xtext.uma.tests
- >  org.eclipse.xtext.uma.ui
- >  org.eclipse.xtext.uma.ui.tests

11.19. Irudia. Testu editorearen proiektuak Eclipsen.

Gramatika lehenengo proiektuan dago eta Umal.xtext izena du. Metaeredua oso handia denez, sortutako gramatikak 2600 lerro inguru ditu eta errore batzuekin sortzen da. Adibidez, 11.20. Irudian gramatikaren erroaren zatia ikus daiteke.

```

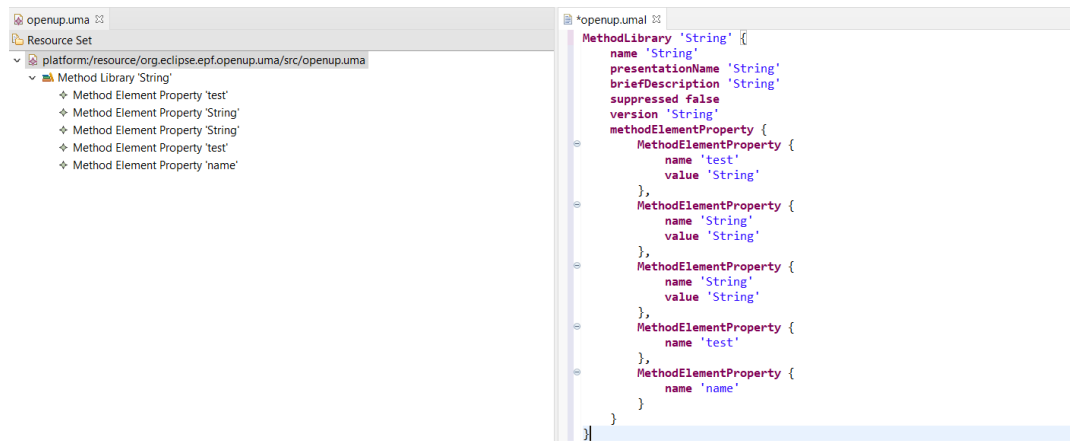
1 // automatically generated by Xtext
2 grammar org.eclipse.xtext.uma.Umal with org.eclipse.xtext.common.Terminals
3
4 import "http://www.eclipse.org/epf/uma/1.0.6/uma.ecore"
5 import "http://www.eclipse.org/emf/2002/Ecore" as.ecore
6
7 @MethodLibrary returns MethodLibrary:
8   'MethodLibrary'
9   guid=String()
10   '{'
11     ('name' name=String())?
12     ('presentationName' presentationName=String())?
13     ('briefDescription' briefDescription=String())?
14     ('suppressed' suppressed=Boolean)
15     ('orderingGuide' orderingGuide=String())?
16     ('authors' authors=String())?
17     ('changeDate' changeDate=Date)?
18     ('changeDescription' changeDescription=String())?
19     ('version' version=String())?
20     ('kind' ('kind'=[Kind] ( ',' kind=[Kind])* ' ' )?)
21     ('copyrightStatement' copyrightStatement=[SupportingMaterial])?
22     ('ownedRules' ('ownedRules+=Constraint ( ',' ownedRules+=Constraint)* ' ' )?)
23     ('methodElementProperty' ('methodElementProperty+=MethodElementProperty ( ',' methodElementProperty+=MethodElementProperty)* ' ' )?)
24     ('methodPlugins' ('methodPlugins+=MethodPlugin ( ',' methodPlugins+=MethodPlugin)* ' ' )?)
25     ('predefinedConfigurations' ('predefinedConfigurations+=MethodConfiguration ( ',' predefinedConfigurations+=MethodConfiguration)* ' ' )?)
26   '}'

```

11.20. Irudia. Xtext proiektuko lengoaiaren gramatika.

Behin gramatika konpondutakoan `GenerateUmal.mwe2` fitxategia *Rus As > MWE2 Workflow* aukeratu beharko dugu editorearen kodea sortzeko. Gainerako proiektuetan editorearen Java eta Xtend kodea sortuko da.

Editorea exekutatzeko modua editore grafikoaren berdina da eta konfigurazio bera aukeratzen badugu, bi editoreak batera erabil ahal izango ditugu. Eredu osoa erabili ordez, adibide simple batekin hobeto ikus ditzakegu bi editoreak 11.21. Irudian.

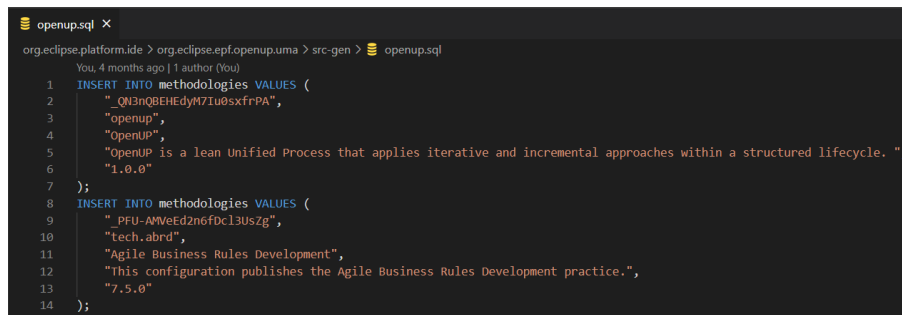


11.21. Irudia. Editore grafikoa ezkerrean eta testu editorea eskuinean.

Hala ere, oraindik aspektu garrantzitsu bat falta da, editoreak ez baitira elkarren artean sinkronizatzen. Helburua da bietako bat aldatzen badugu automatikoki segundu gutxi batzutan bestearekin sinkronizatzea. Horrela, nahiz eta bi fitxategi desberdin izan, edozein momentutan alda daiteke editorez. Hau garrantzitsua da editore bakoitzak bere abantailak dituelako. Editore grafikoak eredu modu bisualean ikustea ahalbidetzen du eta testu editoreak, berriz, aldaketa errepikakorrak azkarrago egitea.

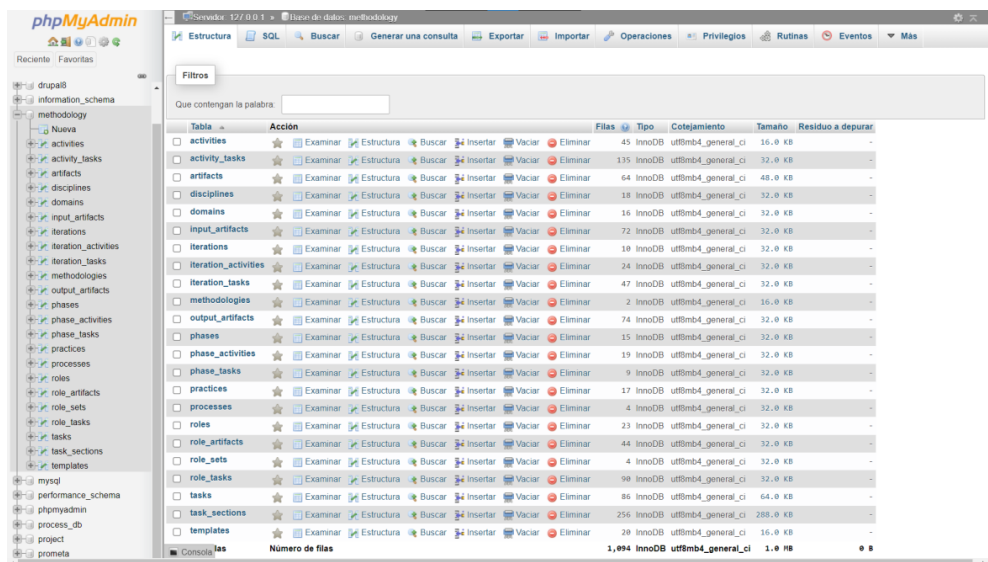
Beraz, bi itzultzaile behar ditugu, uma-tik umal-era itzultzen duena eta umal-etik uma-ra itzultzen duena. Bigarren itzulketa nahiko sinplea da, kode sorreraren antzekoa baita. Izan ere, Xtext-ek itzulketa hau errazten du serializatzaileak definituz Xtend programazio lengoia erabiliz. Lehenengoa zailagoa da, Xtext-ek uma fitxategien lengoia detektatu behar baitu, eta umal lengoariaren antzeko tratamendua eman. Horretarako, Xtext-ek umal-erako automatikoki sortu dituen fitxategietako batzuk eskuz sortu behar dira uma-rako.

Bukatzeko, aldaketak dauden bakoitzean metodologiaren informazio garrantzitsuena datu-base batean gordetzeko balio duen SQL fitxategia eguneratzen da. Datu-basearen tamainaren ideia bat egiteko, sortutako fitxategiak 9400 lerro 1100 INSERT inguru dauzka. Argi dago, beraz, prozesu hau automatizatzearen erabilgarritasuna, informazio hori guztia eskuz gehitzeko denbora pila bat beharko genukeelako. 11.22. Irudian sortutako kodearen lehenengo 3 INSERT aginduak ikus daitezke.



11.22. Irudia. Datu-basearen datuak dituen SQL fitxategia.

Editatzea bukatutakoan, fitxategi hori phpMyAdmin-etik inportatuko dugu, datu-basean metodologiaren informazio guztia gordetzeko. 11.23. Irudian metodologiaren datu-baseko taulak agertzen dira phpMyAdmin-en.



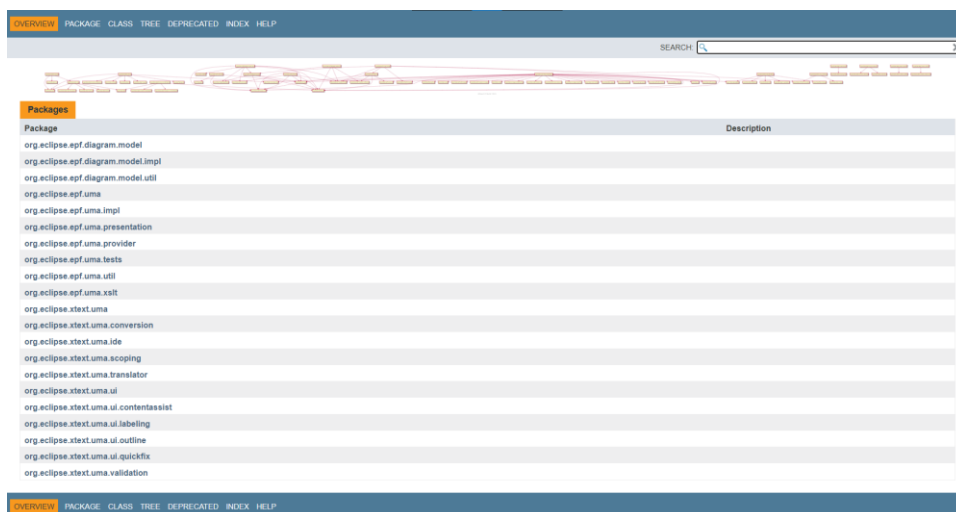
| Tabla | Acción | Filas | Tipo | Cotejamiento | Tamaño | Residuo a depurar |
|----------------------|---|--------------|---------------|---------------------------|---------------|-------------------|
| activities | Examinar Estructura Buscar Insertar Vaciar Eliminar | 45 | InnoDB | utf8mb4_general_ci | 16.0 KB | - |
| activity_tasks | Examinar Estructura Buscar Insertar Vaciar Eliminar | 135 | InnoDB | utf8mb4_general_ci | 72.0 KB | - |
| artifacts | Examinar Estructura Buscar Insertar Vaciar Eliminar | 64 | InnoDB | utf8mb4_general_ci | 48.0 KB | - |
| disciplines | Examinar Estructura Buscar Insertar Vaciar Eliminar | 18 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| domains | Examinar Estructura Buscar Insertar Vaciar Eliminar | 16 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| input_artifacts | Examinar Estructura Buscar Insertar Vaciar Eliminar | 72 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| iterations | Examinar Estructura Buscar Insertar Vaciar Eliminar | 18 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| iteration_activities | Examinar Estructura Buscar Insertar Vaciar Eliminar | 24 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| iteration_tasks | Examinar Estructura Buscar Insertar Vaciar Eliminar | 47 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| methodologies | Examinar Estructura Buscar Insertar Vaciar Eliminar | 2 | InnoDB | utf8mb4_general_ci | 16.0 KB | - |
| output_artifacts | Examinar Estructura Buscar Insertar Vaciar Eliminar | 74 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| phases | Examinar Estructura Buscar Insertar Vaciar Eliminar | 15 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| phase_activities | Examinar Estructura Buscar Insertar Vaciar Eliminar | 19 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| phase_tasks | Examinar Estructura Buscar Insertar Vaciar Eliminar | 9 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| practices | Examinar Estructura Buscar Insertar Vaciar Eliminar | 17 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| processes | Examinar Estructura Buscar Insertar Vaciar Eliminar | 4 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| roles | Examinar Estructura Buscar Insertar Vaciar Eliminar | 23 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| role_artifacts | Examinar Estructura Buscar Insertar Vaciar Eliminar | 44 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| role_sets | Examinar Estructura Buscar Insertar Vaciar Eliminar | 4 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| role_tasks | Examinar Estructura Buscar Insertar Vaciar Eliminar | 98 | InnoDB | utf8mb4_general_ci | 32.0 KB | - |
| tasks | Examinar Estructura Buscar Insertar Vaciar Eliminar | 86 | InnoDB | utf8mb4_general_ci | 64.0 KB | - |
| task_sections | Examinar Estructura Buscar Insertar Vaciar Eliminar | 256 | InnoDB | utf8mb4_general_ci | 288.0 KB | - |
| templates | Examinar Estructura Buscar Insertar Vaciar Eliminar | 28 | InnoDB | utf8mb4_general_ci | 16.0 KB | - |
| Consolas | Numero de filas | 1,094 | InnoDB | utf8mb4_general_ci | 1.0 MB | 0 B |

11.23. Irudia. Metodologiaren datu-basea phpMyAdmin-en.

Kodea gordetzeko eta bertsio kontrolerako Git eta GitHub tresnak erabili dira. Dokumentazio webgunearen hosting-erako GitHub Pages erabili da. Java proiektuen dokumentazioa sortzeko modu ohikoa Javadoc erabiltzea da. Kasu honetan, plugin bat gehitu da, dokumentazioarekin batera diagramak automatikoki sortzeko.

UMLDoclet tresnak Javadoc metadatuak erabiltzen ditu PlantUML diagramak automatikoki sortzeko eta HTML dokumentazioan txertatzeko. Diagramak irudi klikagarri gisa txertatzen dira eta pakete eta klaseko dokumentaziara estekatzen dira eskuragarri dagoenean. Pakete dependentzia, pakete eta klase diagramak sortzen ditu elementu guztietarako.

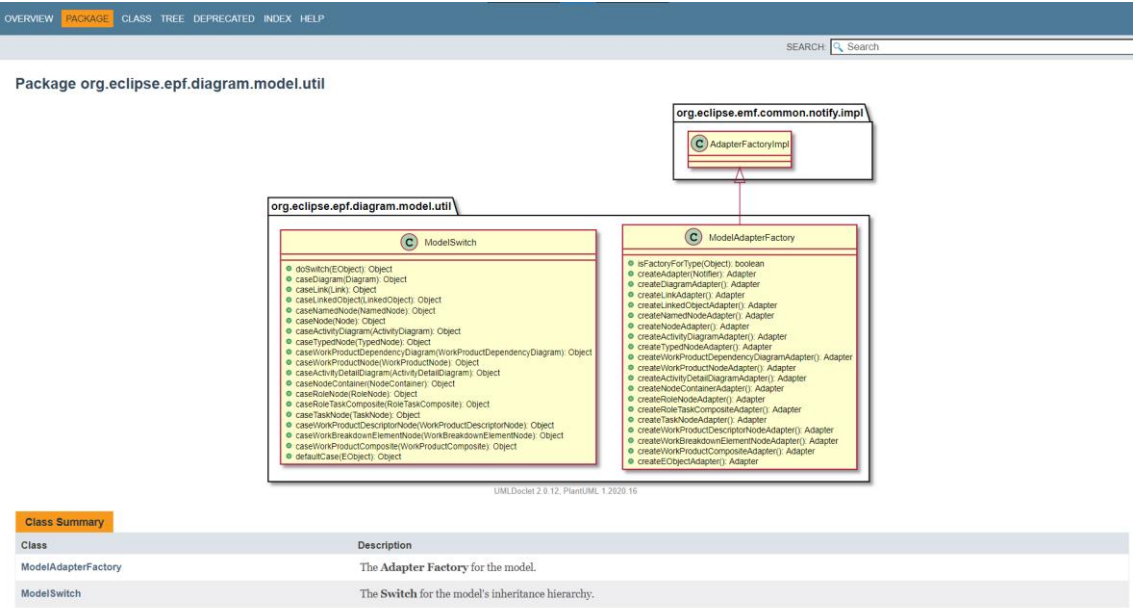
Dokumentazio webguneak Javadoc webgune baten itxura du, baina hainbat lekutan diagramak daude. 11.24. Irudian ModelEditor azpisisemako dokumentazio webgunearen hasierako orria agertzen da. Bertan paketeak zerrendatzen dira eta goiko aldean paketeen arteko dependentzia diagrama ikus daiteke.



| Package | Description |
|--|-------------|
| org.eclipse.epf.diagram.model | |
| org.eclipse.epf.diagram.model.impl | |
| org.eclipse.epf.diagram.model.util | |
| org.eclipse.epf.uma | |
| org.eclipse.epf.uma.impl | |
| org.eclipse.epf.uma.presentation | |
| org.eclipse.epf.uma.provider | |
| org.eclipse.epf.uma.tests | |
| org.eclipse.epf.uma.util | |
| org.eclipse.epf.uma.xslt | |
| org.eclipse.xtext.uma | |
| org.eclipse.xtext.uma.conversion | |
| org.eclipse.xtext.uma.ide | |
| org.eclipse.xtext.uma.scoping | |
| org.eclipse.xtext.uma.translator | |
| org.eclipse.xtext.uma.ui | |
| org.eclipse.xtext.uma.ui.contentassist | |
| org.eclipse.xtext.uma.ui.labeling | |
| org.eclipse.xtext.uma.ui.outline | |
| org.eclipse.xtext.uma.ui.quickfix | |
| org.eclipse.xtext.uma.validation | |

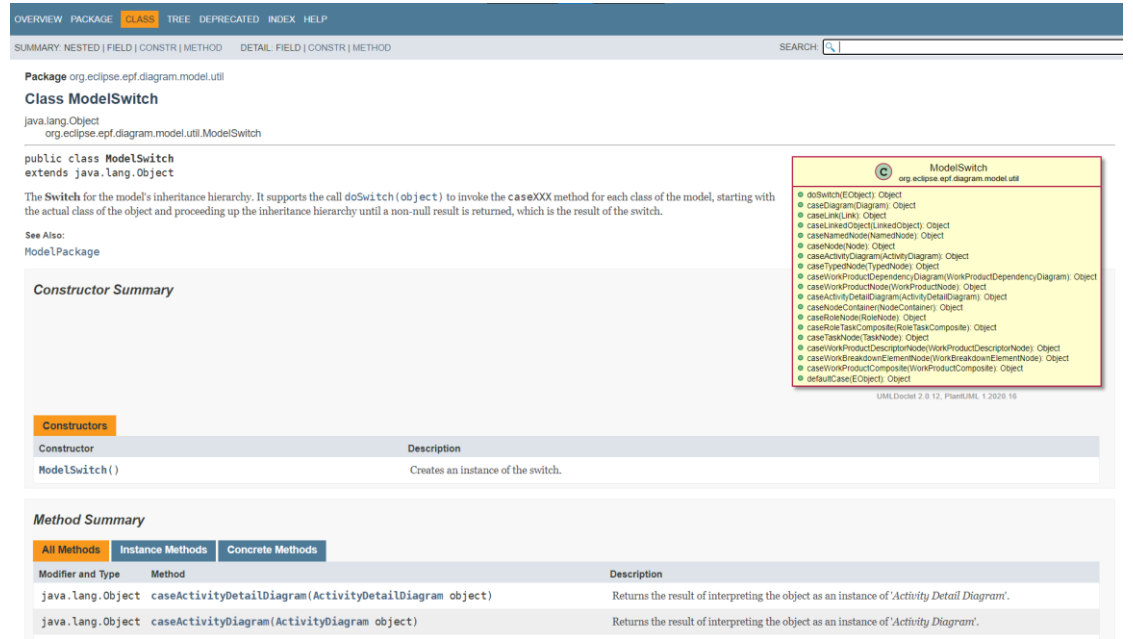
11.24. Irudia. ModelEditor-en paketeak eta paketeen arteko dependentziak.

Pakete bat aukeratzen badugu, diagrama eta klaseak bistaratuko dira. 11.25. Irudian adibide bat ikus daiteke.



11.25. Irudia. Pakete diagrama adibidea.

Klase bat aukeratzen badugu, bere diagrama eta metodoak bistaratuko dira. 11.26. Irudian ikus daiteke klase diagrama baten adibidea.



11.26. Irudia. Klase diagrama adibidea.

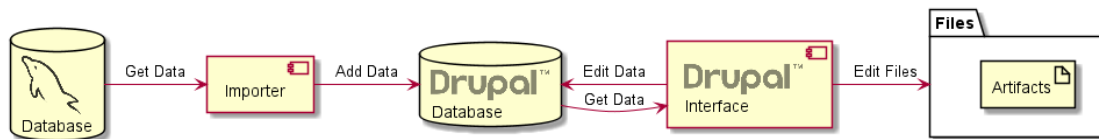
11.3 IO-System

Drupal CMSaren bitartez kudeatutako web-aplikazioa izango da. Helburua metodologia jarraitzen duten proiektuen informazioa gordetzea da. Rol bakoitzak metodologian dituen ataza berdinak bete beharko ditu.

11.3.1 Arkitektura

IO-System azpisistemak honako osagaiak izango ditu. 11.27. Irudian osagaien arteko loturak ikus daitezke.

- **Datu-basea:** Datu-base erlazionalak ereduaren informazio garrantzitsuen gordeko du, Drupal webgunerako beharrezkoa izan daitekeena. Hau da, OpenUP eta ABRD metodologiaren faseak, iterazioak, jarduerak, atazak, artefaktuak, rolak, etab.
- **Drupal datu-basea:** Drupal sistemaren datu-basea eduki guztia gordetzeaz arduratzen da, fitxategiak izan ezik. Adibidez, erabiltzaileak, rolak, baimenak eta eduki motak gordetzean dira.
- **Drupal datu inportatzailea:** Datu-inportatzailea Drupal modulu multzo bat izango da. Hauen ardura aurretik aipatutako lehenengo datu-basetik Drupal datu-basera edukia inportatzea da. Horretarako, nodoak sortu beharko dira, eta Drupal arduratuko da edukia gordetzeaz.
- **Drupal fitxategiak:** Fitxategiak datu-basetik kanpo gordeko dira. Gure kasuan fitxategi gehienak artefaktuei dagozkienak izango dira, DOC eta PDF dokumentuak.
- **Drupal interfazea:** Interfazean edukia bistaratu eta aldatzeko aukera guztiak egongo dira. Esan bezala, edukia Drupal datu-basera gordeko da. Gainera, erabiltzailearen kontuekin zerikusia duten aukerak ere egongo dira. Horrez gain, administratzaileak aukera gehiagarri asko izango ditu webgunea kudeatzeko interfaze bidez.



11.27. Irudia. IO-System azpisistemaren arkitektura.

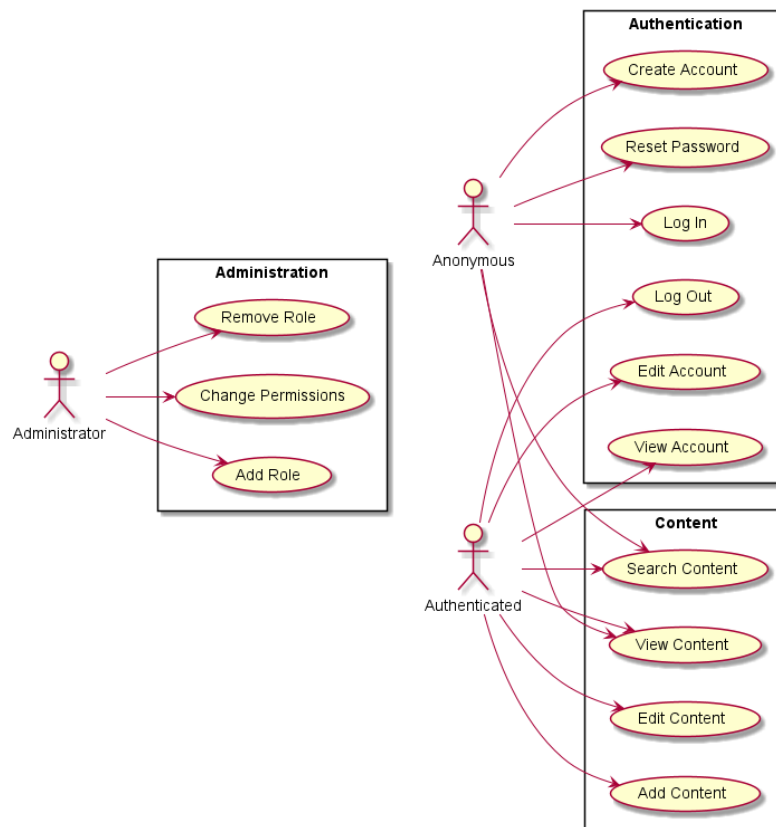
11.3.2 Erabilpen Kasuak

Sistemaren betekizunak zehazteko beharrezko da erabilpen kasuak eta aktoreak zein diren jakitea. Erabilpen kasuak 3 multzotan banatu daitezke: administrazioa, autentikazioa eta edukia. Aldi berean 3 erabiltzaile mota nagusi daude, anonimoa, autentikatu eta administraria.

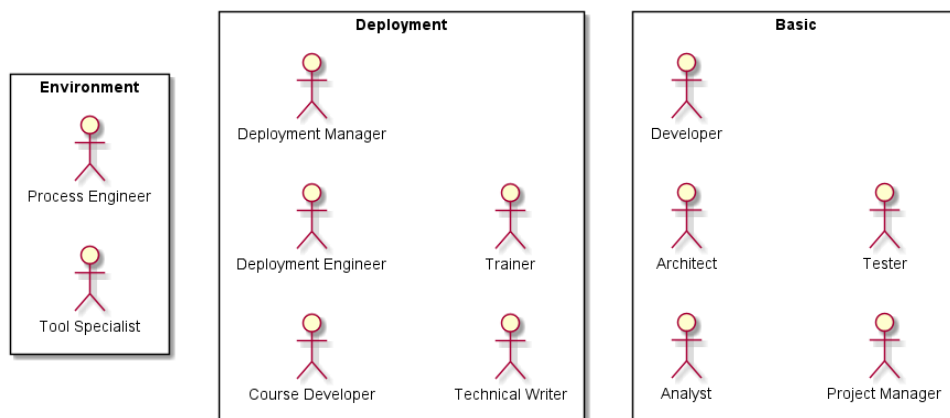
Erabiltzaile anonimoak autentikazioako ataza batzuk egiteko eta edukia ikusteko aukera bakarrik izango du. Erabiltzaile autentikatuak autentikazioko ataza batzuk egiteko eta edukia ikusi, editatu, gehitu eta bilatzeko aukerak izango ditu. Administrazioak erabiltzaile autentikatuak egin ditzakeen atazek gain administrazioko atazak izango ditu, rolei eta baimenei dagozkienak. 11.28. Irudian IO-System azpisistemako erabilpen kasuen eredua agertzen da.

Sinplifikatzeko 3 rol nagusi jarri diren arren, erabiltzaile autentikatu mota asko daude, sistemako metodologietan dauden erabiltzaile adina. Adibidez, 11.29. Irudian ikusten da OpenUP metodologiako erabiltzaileak 3 multzotan banatu daitezkeela: ingurunea, hedapena eta oinarritzakoa.

Erabiltzaile horietako bakoitzak OpenUP metodologian dagozkion ataza eta artefaktuak bete beharko ditu. Adibidez, proiektu kudeatzaileak 11.30. Irudian ikusten diren ataza eta artefaktuen ardura dauka. Sistema ideal batean rol bakoitzak beharrezkoak diren baimenak bakarrik eduki beharko lituzke. Hala ere, baimen guztiak kudeatzea oso konplexua denez, etorkizuneko hobekuntza moduan utzi da. Beraz, praktikan aurretik aipatutako 3 rol nagusiak bakarrik daudela esan dezakegu.



11.28. Irudia. IO-System azpisistemako erabilpen kasuen eredua.



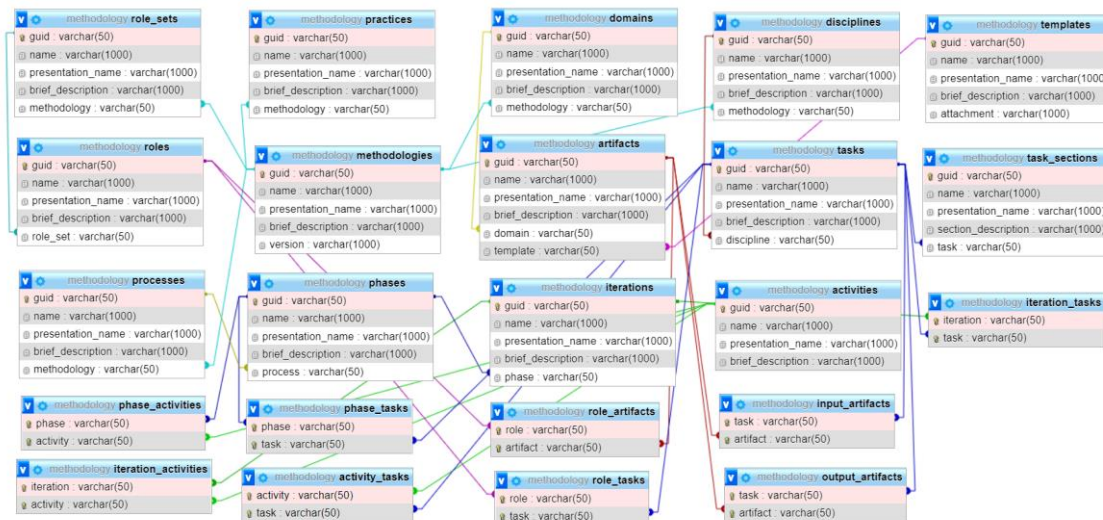
11.29. Irudia. OpenUP metodologiako rola multzokatuta.



11.30. Irudia. Proiektu kudeatzaileak bete beharreko ataza eta artefaktuak.

11.3.3 Diseinua

Metodologiaren informazio garrantzitsuenak gordetzeko datu-base erlazionala erabili da. 11.1. Irudian ikusten den moduan, datu-basearen diseinua eragina du bi azpisistematan. Atal honetan azaltzea erabaki da Drupal datu-basearekin lotura zuzenagoa duelako. Datu base honen diseinua nahiko konplexua da, 23 taula eta taulen arteko hainbat erlazio baititu, 11.31. Irudian ikus daitezkeenak. Taula nagusiek identifikadore bat, izena, aurkezpen izen eta deskribapena dituzte. Gainera, atributu gehigarriak dituzte beste taulekin erlazioak adierazteko. Gainerako taulak beste taulen arteko erlazioak adierazteko dira, bi taulen identifikatzaileak bakarrik dituzte.

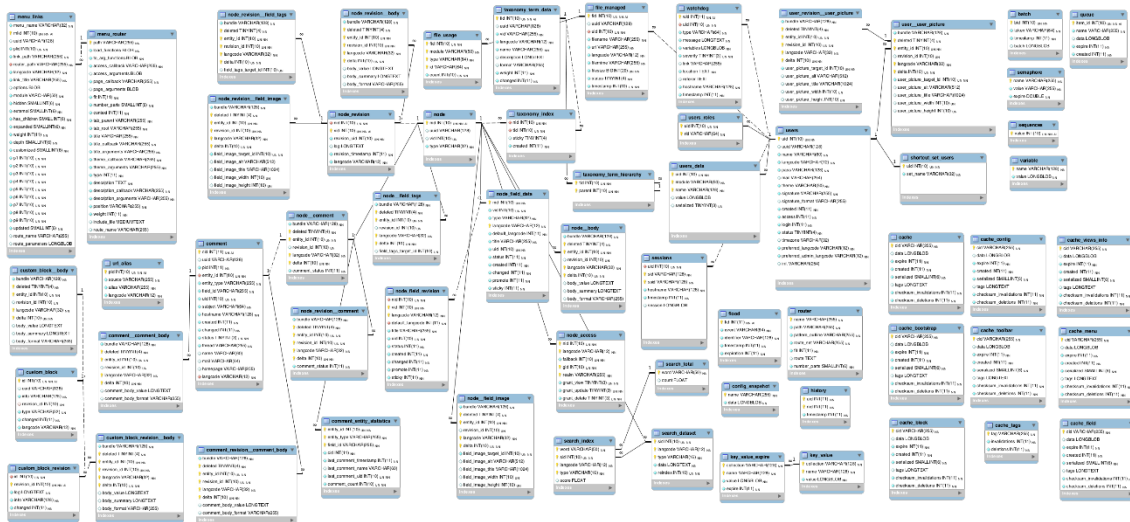


11.31. Irudia. Datu-base erlazionalaren diseinua.

Jarraian, taula nagusien deskribapen labur bat egingo da diseinua hobeto ulertzeko:

- **methodologies:** metodologiaren datuak gordetzen ditu.
- **practices:** metodologiari dagozkion praktiken datuak gordetzen ditu.
- **roles:** prozesuko rolak gordetzen ditu.
- **role_sets:** metodologiari dagozkion rolen multzoak gordetzen ditu.
- **processes:** metodologiak definitzen duen prozesuaren datuak gordetzen dira.
- **phases:** prozesuko faseen datuak gordetzen ditu.
- **iterations:** prozesuko iterazioen datuak gordetzen ditu.
- **activities:** prozesuko jardueren datuak gordetzen ditu.
- **tasks:** prozesuko atazen datuak gordetzen ditu.
- **disciplines:** prozesuko atazen multzoak gordetzen ditu.
- **task_sections:** prozesuko atazen sekzioak gordetzen ditu.
- **artifacts:** prozesuko artefaktuen datuak gordetzen ditu.
- **domains:** prozesuko artefaktuen multzoak gordetzen ditu.
- **templates:** prozesuko artefaktuen txantiloak gordetzen ditu.

Datu horiek Drupal sistemara inportatu direnez eduki moduan, Drupal-ek bere datu-basean gorde ditu. Drupal-ek atzetik erabiltzen duen datu-basearen diseinua jakitea beharrezkoa ez den arren, ideia bat egiteko sistema simple baten datu-basea ikus dezakegu 11.32. Irudian.



11.32. Irudia. Drupal datu-base baten diseinua.

Gure datu-basearekin konparatuz ikus daiteke askoz konplexuagoa dela, Drupal-ek informazio guztia gordetzen du datu-basean gordetzen baitu. Gainera, IO-System sistemaren datu-basea oraindik handiago da, 113 taula ditu guztira. Izan ere, eduki mota asko definitu ditugu eta bakoitzarentzat hainbat taula sortzen dira. Gainera, kontuan hartu behar da metodologiez gain proiektuen informazioa ere gordetzen duela.

11.3.4 Dokumentazioa

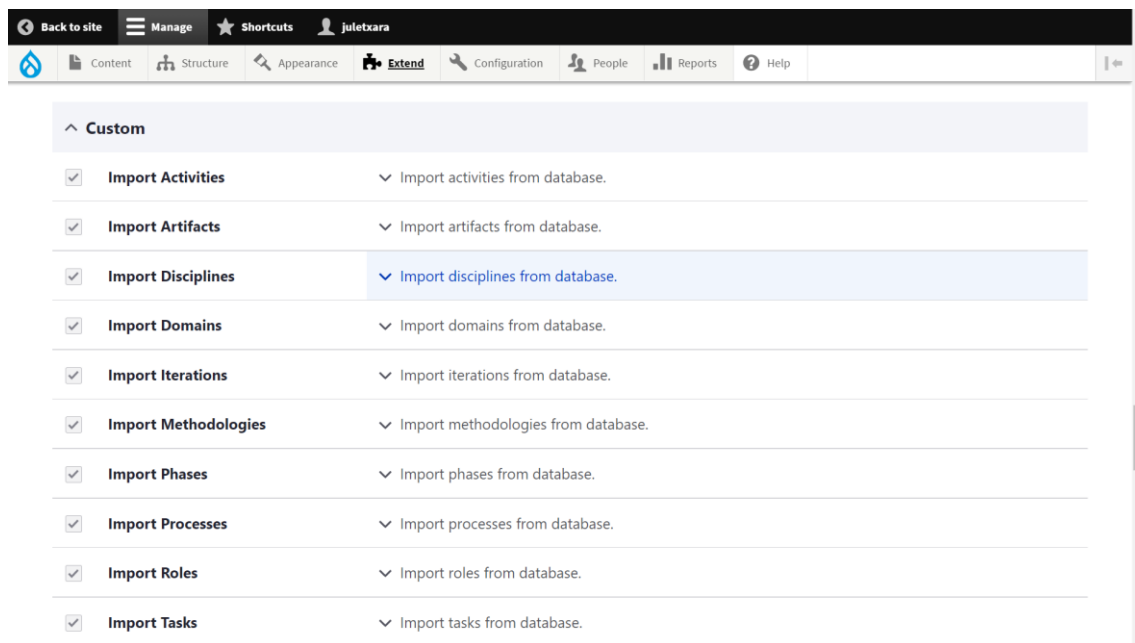
Esan bezala, azpististemaren garapenerako Drupal 9 erabili da. Uneoro azkenengo bertsio egonkorra erabiltzea erabaki da. Horretarako, aktualizazioak zeuden bakoitzean Drupal eguneratu da *composer* tresna erabiliz. Programazio lengoaia nagusia PHP izan da, Drupal lengoaia horretan idatzita baitago.

Webgunearen lokalean garatzeko XAMPP software libreko paketea erabili da. Honek MariaDB datu-baseen kudeaketa sistema eta Apache web zerbitzaria integratzen ditu. Gainera, datu-basearen kudeaketa errazteko *phpMyAdmin* eskaintzen du, web nabigatzaile bidez erabiltzen dena. Editore moduan *Visual Studio Code* erabili da.

Kodea gordetzeko eta bertsio kontrolerako *Git* eta *GitHub* tresnak erabili dira. *Hosting*-erako *Pantheon* erabili da, Drupal webguneentzako doako *hosting*-a eskaintzen duen plataforma. Aldaketak lokalean egin eta probatu ondoren, *GitHub*-era eta *Pantheon*-era igo dira, webgunea eta kodea publikoki eskuragarri egoteko. *Pantheon*-en webgunearen edukia eguneratuta mantentzeko, beharrezko da fitxategiak eta datu-basea ere igotzea.

Drupal bezalako CMS batek eskaintzen dituen funtzionalitateak aprobeatzeko, ahal den guztia bertako aukerak erabiliz egitea erabaki da. Hau da, edukia gordetzeko kanpoko datu-base bat erabili ordez Drupal-eko eduki motak erabiltzea erabaki da. Guztira 13 eduki mota sortu dira: metodologia, prozesua, fasea, iterazioa, jarduera, ataza, artefaktua, disziplina, domeinua, rola, proiektua, kidea eta proiektuko artefaktua. Eduki motak sortzeaz gain, bakoitzarentzat bista bat definitu da, eta menuko aukerak gehitu dira edukia errazago gehitu eta ikusteko.

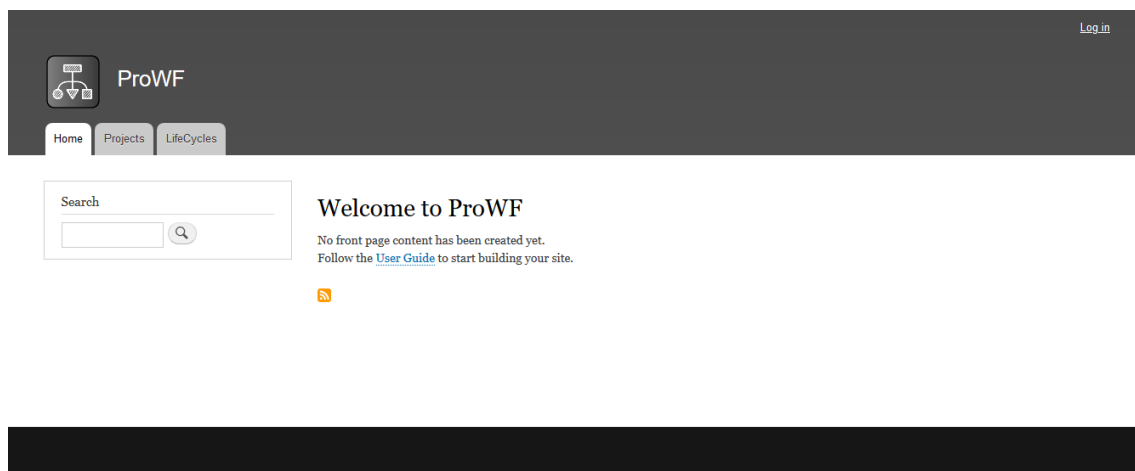
Eduki guztia eskuz gehitzeak zentzurik ez duenez, garatu beharreko lehenengo funtzionalitatea kanpoko datu-basetik edukia hartu eta automatikoki Drupal-era gehitzea da. Drupal-en funtzionalitateak gehitzeko modurik onena moduluak direnez, eduki mota bakoitza inportatzeko modulu bat garatu da, 11.33. Irudian ikus daitekeen bezala.



11.33. Irudia. Drupal webgunera datu-basetik edukia inportatzeko moduluak.

Gainera, Drupal-ekin batera etortzen diren modulu batzuk aktibatu dira funtzionalitateak gehitzeko. Bukatzeko, kanpoko bi modulu ere instalatu dira, [Admin Toolbar](#) administrazio menua hobetzeko eta [Menu Item Role Access](#) menuko baimenak kudeatzeko.

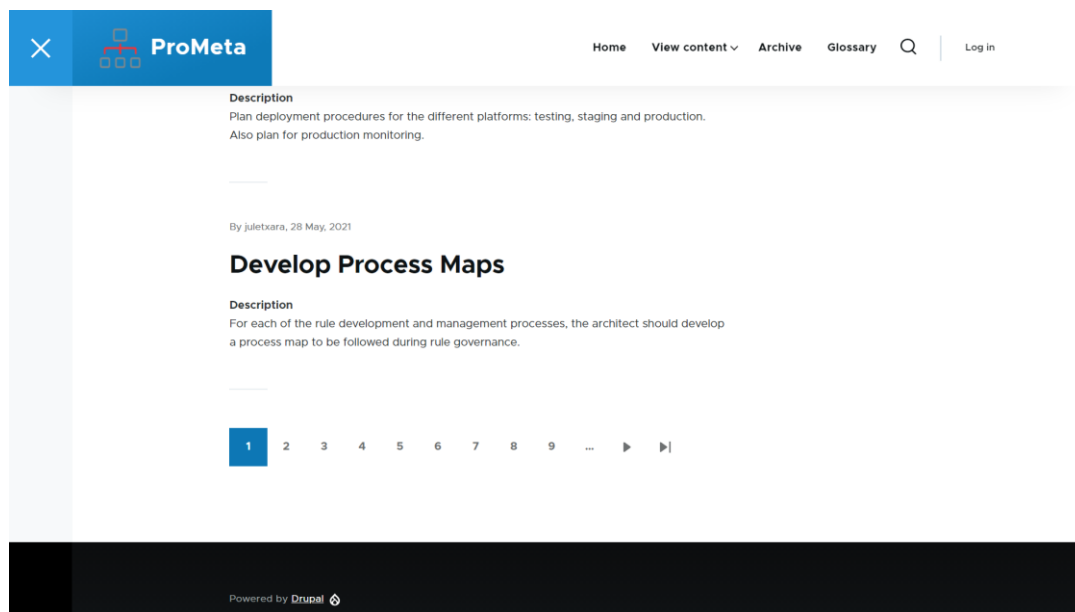
Itxurari dagokionez, Drupal-ek defektuz eskaintzen dituen gaiak oso zaharkituak daudenez, aldatzea erabaki da. Adibide moduan, ProWF sistemaren hasierako orria ikus daiteke *Bartik* gaiarekin. Beraz, webgune nagusirako *Bartik*-en ordeztu *Olivero* aukeratu da. Administrazio webgunerako, berriz, *Seven* gaia *Claro*-rekin ordezkatu da. Aldaketa sinple honekin webguneak askoz itxura modernoagoa hartu du. Gainera, itxura ez ezik funtzionalitate gehigarriak ere eskaintzen dituzte, adibidez menu hedagarriak.



11.34. Irudia. ProWF sistemaren hasierako orria.

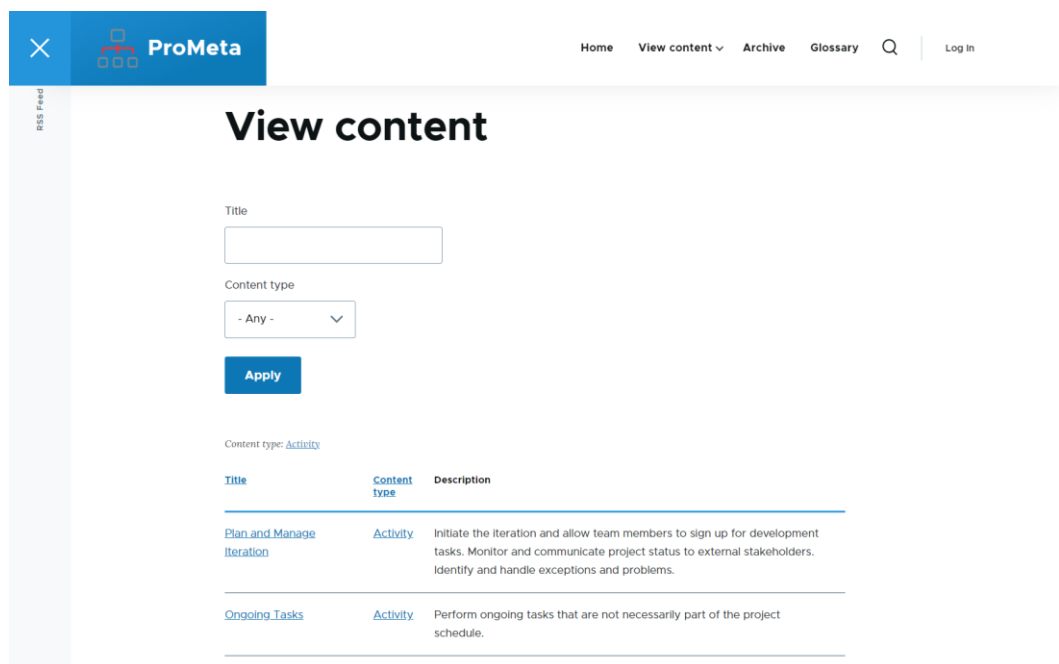
ProMeta webgunera sartutakoan hasierako orria bistaratzen da, 11.35. Irudian ikusten dena. Goiburuaren ezker aldean sistemaren logoa eta izena agertzen dira eta eskuinaldean menua. Erdialdean sistemara gehitutako edukia agertzen da ordena kronologikoan. Oinean webgunea Drupal-ekin eginda dagoela dioen mezu bat bakarrik dago.

Menuaren ezker aldean hainbat esteka daude: hasierako orrira joateko esteka, edukia ikusteko estekak, erregistroa eta glosarioa. Erdian bilaketak egiteko botoia dago eta eskuinean saioa irekitzekoa.



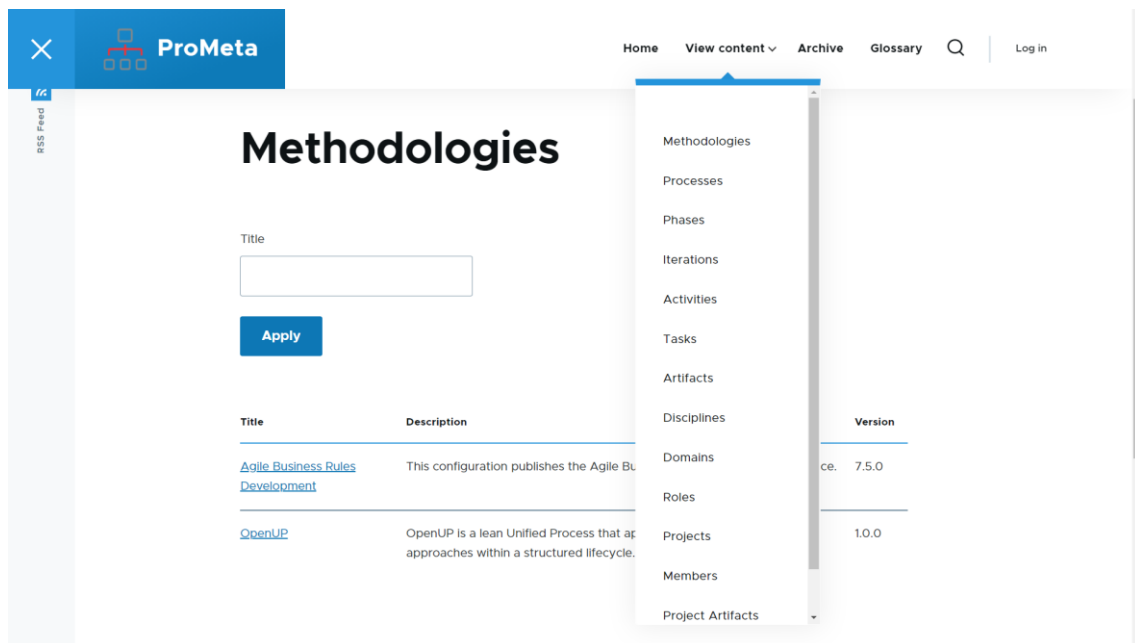
11.35. Irudia. Drupal webgunearen hasierako orria.

Edukia ikusteko aukerari ematen badiogu, edukia bistaratuko da taula batean motaren arabera multzokatuta. Izenburu eta eduki motarekin bilaketak egiteko aukera ere badago, 11.36. Irudian ikusten den moduan.



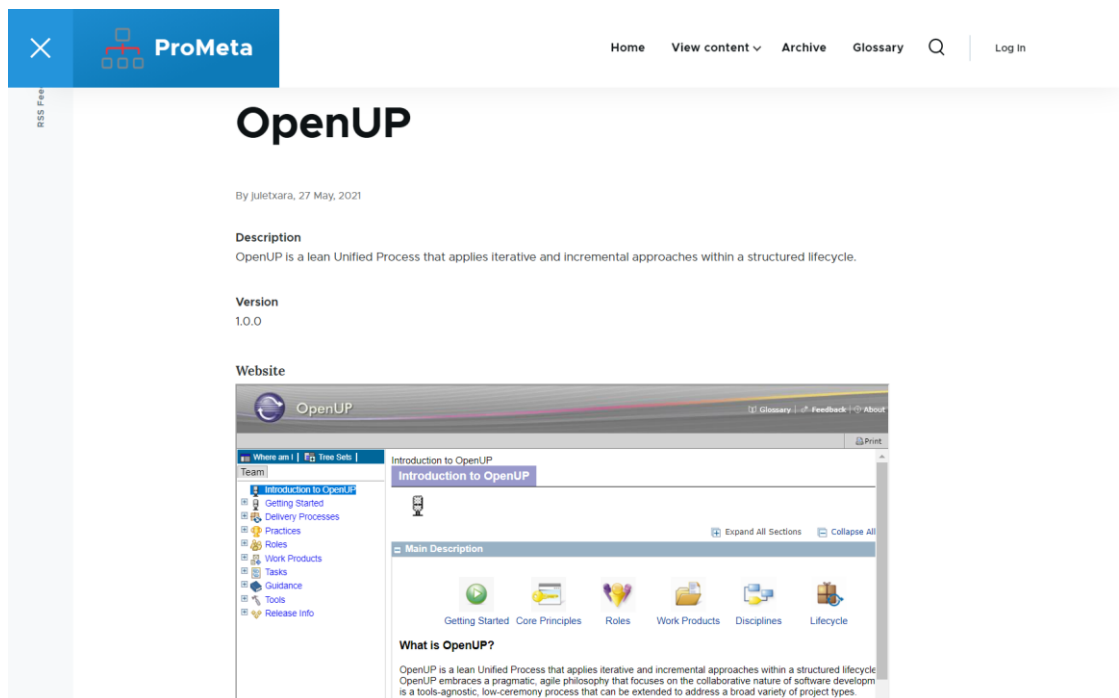
11.36. Irudia. Drupal webguneko edukia ikusteko orria.

Eduki mota zehatz bat bakarrik ikusi nahi badugu zehaztasun gehiagorekin, menu hedagarria erabil dezakegu. Guztira 13 eduki mota daude: metodologia, prozesua, fasea, iterazioa, jarduera, ataza, artefaktua, disziplina, domeinua, rola, proiektua, kidea eta proiektuko artefaktua. Adibidez, 11.37. Irudian metodologiak bistartzen dira eta izenburuaren arabera bilaketak egin daitezke.



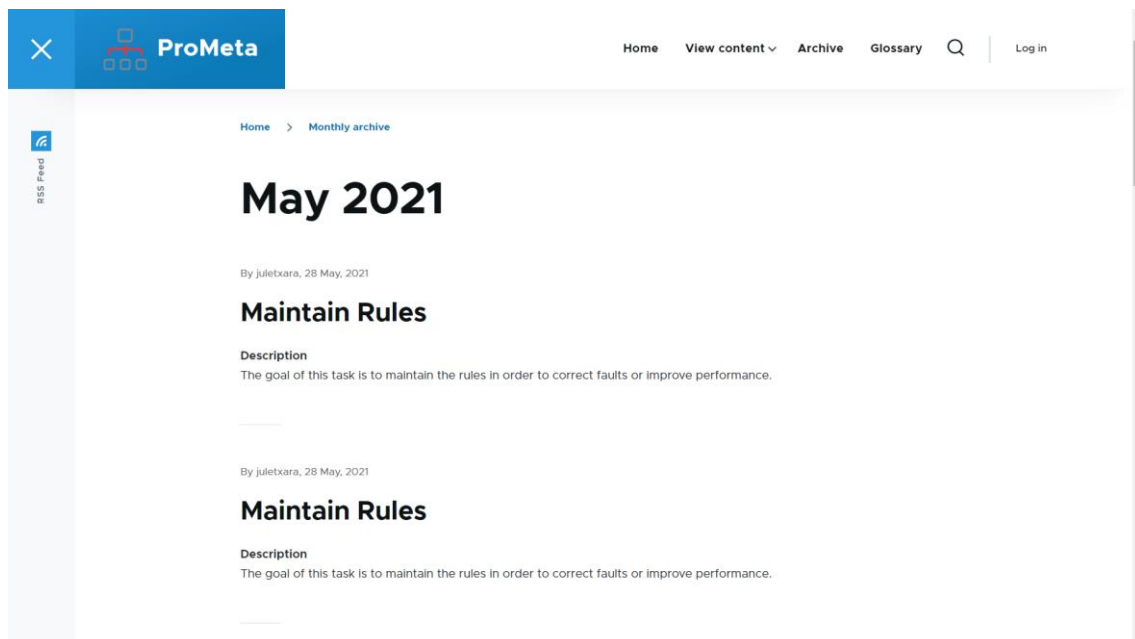
11.37. Irudia. Drupal webguneko metodologiak ikusteko orria.

Eduki bat zehaztasun gehiagorekin ikusi nahi badugu, izenburuan klik egin behar dugu. Adibidez, OpenUP metodologia aukeratzen badugu, bere webgunea bistaratuko da, 11.38. Irudian ikusten den bezala.



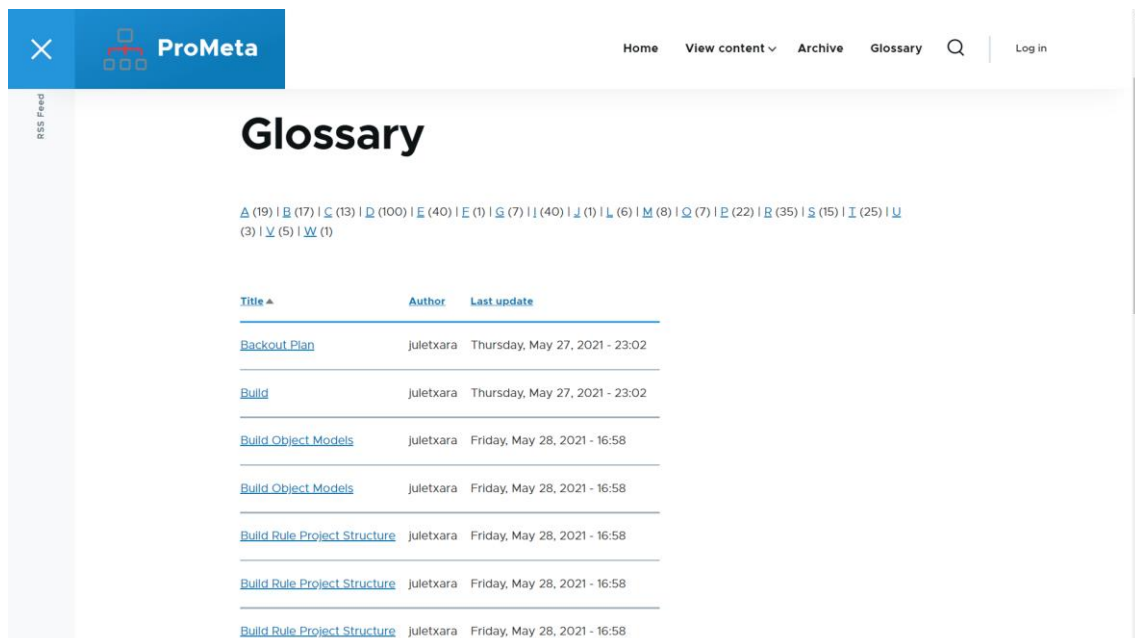
11.38. Irudia. Drupal webgunean OpenUP metodologia ikusteko orria.

Artxiboa aukeratzen badugu, edukia hilabeteka sailkatuta bistaratuko da. Adibidez, maiatza aukeratzen badugu, hilabete horretan gehitutako edukia bakarrik agertuko da, 11.39. Irudian agertzen den bezala.



11.39. Irudia. Drupal webgunean maiatzean gehitutako edukia.

Glosarioan klik egiten badugu, berriz, edukia hasierako letraren arabera sailkatuta agertuko da. Adibidez, B hizkian klik egiten badugu, letra horrekin hasten den edukia bakarrik bistaratuko da, 11.40. Irudian ikus daitekeen bezala.



11.40. Irudia. Drupal webguneko glosario orria.

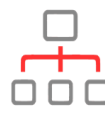
Edukia bilatzeko botoian klik egiten badugu, idazteko barra zabalduko da. Idatzitakoarekin bat egiten duten edukiak bakarrik azalduko dira. Adibidez, 11.41. Irudian OpenUP hitza duten edukiak bakarrik agertzen dira bilaketan.



Proiektua: ProMeta

Egilea: Julen Etxaniz Aragoneses

Tutorea: Juan Manuel Pikatza Atxa



Manage

Shortcuts

juletxara

Content

Structure

Appearance

Extend

Configuration

People

Reports

Help

×



ProMeta

Home

View content ▾

Add content ▾

Archive

Glossary

×

My account

Log out

OpenUP

×

Q

Search for OpenUP ☆

Enter your keywords

OpenUP

Search

[About searching](#)

> Advanced search

Search results

By [juletxara](#), 27 May, 2021

OpenUP Lifecycle

OpenUP Lifecycle Description ... development lifecycle that supports the core principles of OpenUP. It is designed to support small, co-located teams in ... Methodology OpenUP

11.41. Irudia. Drupal webgunean edukia bilatzeko aukera.

Orain arteko funtzionalitateak erabiltzeko ez zen beharrezkoa webgunean saioa hastea. Baina, edukia gehitu edo aldatu nahi badugu, saioa hasi beharko dugu. Saioa irekitzeko botoiari ematen badiogu beste orri batera bideratuko gaitu. Bertan 3 aukera izango ditugu: saioa hasi, kontu berria sortu eta pasahitza berrezarri. 11.42. Irudian ikusten den bezala, saioa hasteko erabiltzaile izena eta pasahitza eskatzen dira.

×



ProMeta

Home

View content ▾

Archive

Glossary

Q

Log in

Log in

Create new account

Reset your password

Home

Log in

Username *

Enter your ProMeta username.

Password *

Enter the password that accompanies your username.

Log in

11.42. Irudia. Drupal webguneko saioa hasteko orria.

Kontu berria sortzeko izena, abizena, posta, erabiltzaile izena eta argazkia eskatzen dira, 11.43. Irudian ikusten den moduan. Erabiltzaileari mezu elektronikoa bidaliko zaio posta baieztatzeko eta ondoren pasahitza aukeratuko du.




ProMeta

[Home](#)
[View content](#)
[Archive](#)
[Glossary](#)

[Log in](#)

Create new account

First name

Last name

Email address *

A valid email address. All emails from the system will be sent to this address. The email address is not made public and will only be used if you wish to receive a new password or wish to receive certain news or notifications by email.

Username *

Several special characters are allowed, including space, period (.), hyphen (-), apostrophe ('), underscore (_), and the @ sign.

Picture

Ningún archivo seleccionado

Your virtual face or picture.
One file only.
64 MB limit.
Allowed types: png gif jpg jpeg.

> Contact settings

Create new account

11.43. Irudia. Drupal webguneko kontua sortzeko orria.

Erabiltzaileari pasahitza ahazten bazaio berrezartzeko aukera izango du posta elektroniko bidez. Horretarako, bere erabiltzaile izena edo posta jarri beharko du, 11.44. Irudian ikus daitekeen moduan.



RSS Feed



ProMeta

[Home](#)
[View content](#)
[Archive](#)
[Glossary](#)

[Log in](#)

[Log in](#)
[Create new account](#)
[Reset your password](#)

Home

Reset your password

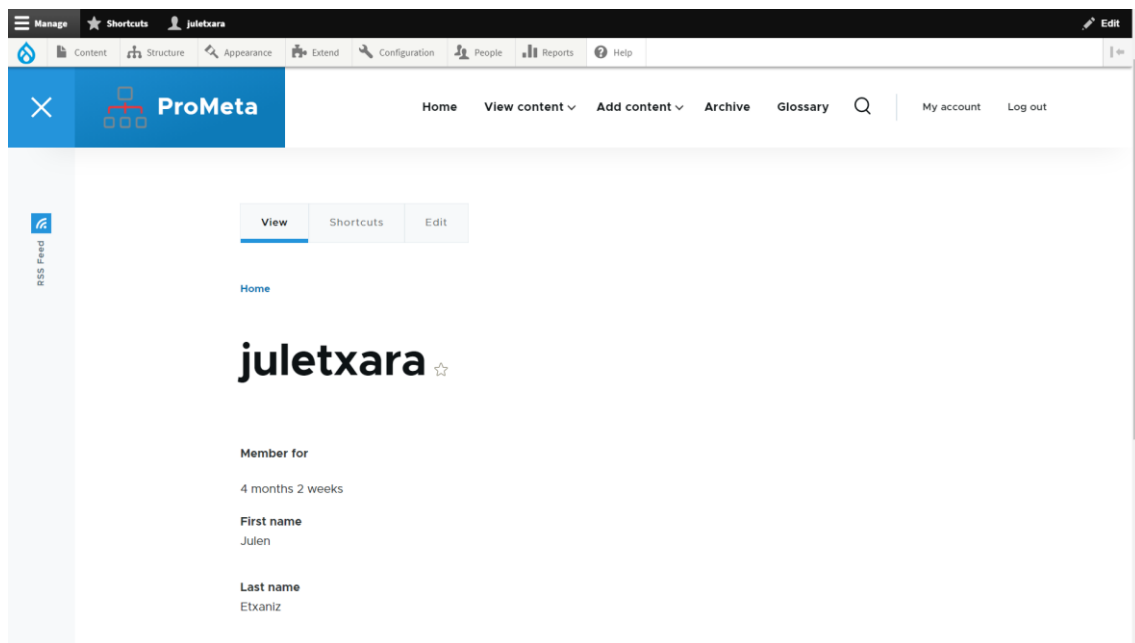
Username or email address *

Password reset instructions will be sent to your registered email address.

Submit

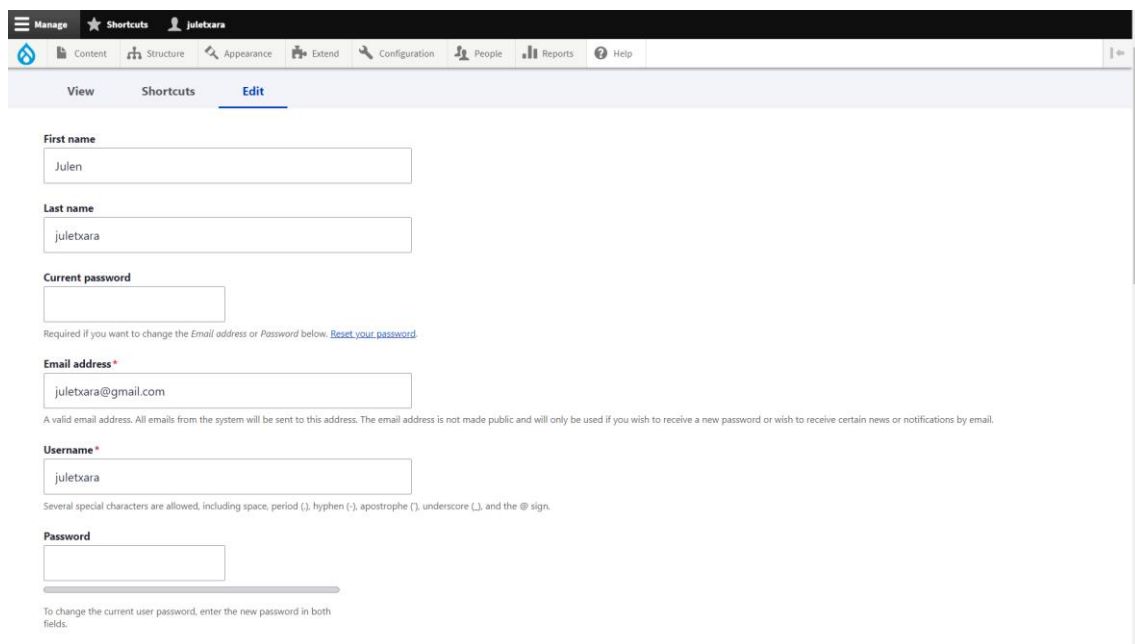
11.44. Irudia. Drupal wenguneko pasahitza berrezartzeko orria.

Saioa irekitzen badugu, gure kontura eramango gaitu. Goiko aldean bi menu berri daudela ikus dezakegu 11.45. Irudian, lehenengoa kontuari dagokiona eta bigarrena administrariaren menua. Gainera, menu nagusian edukia gehitzeko, kontua ikusteko eta saioa ixteko aukerak agertuko dira. Kontuari dagokionez, lasterbideak definitzeko eta kontua aldatzeko aukerak izango ditugu.



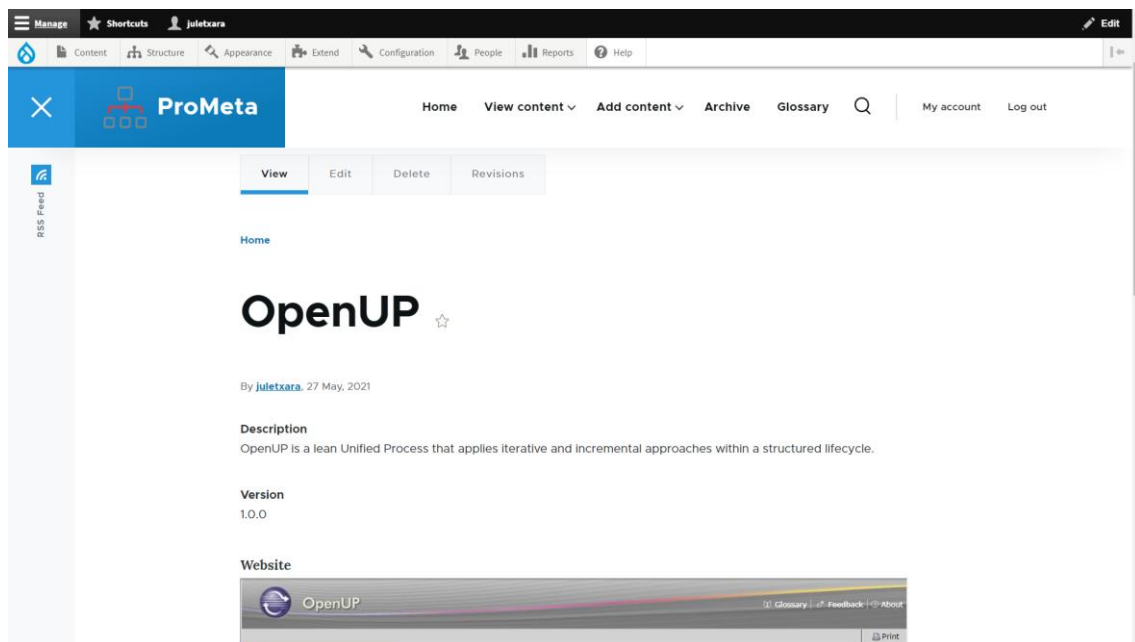
11.45. Irudia. Drupal webguneko erabiltzailearen kontuaren orria.

Kontua aldatzeko estekan sakatzen badugu itxura desberdina duen orrialde batera eramango gaitu, 11.46. Irudian azaltzen dena. Bertan kontuari dagozkion zehaztasunak aldatzeko aukera izango dugu: izena, posta, erabiltzailea, pasahitza eta rolak.



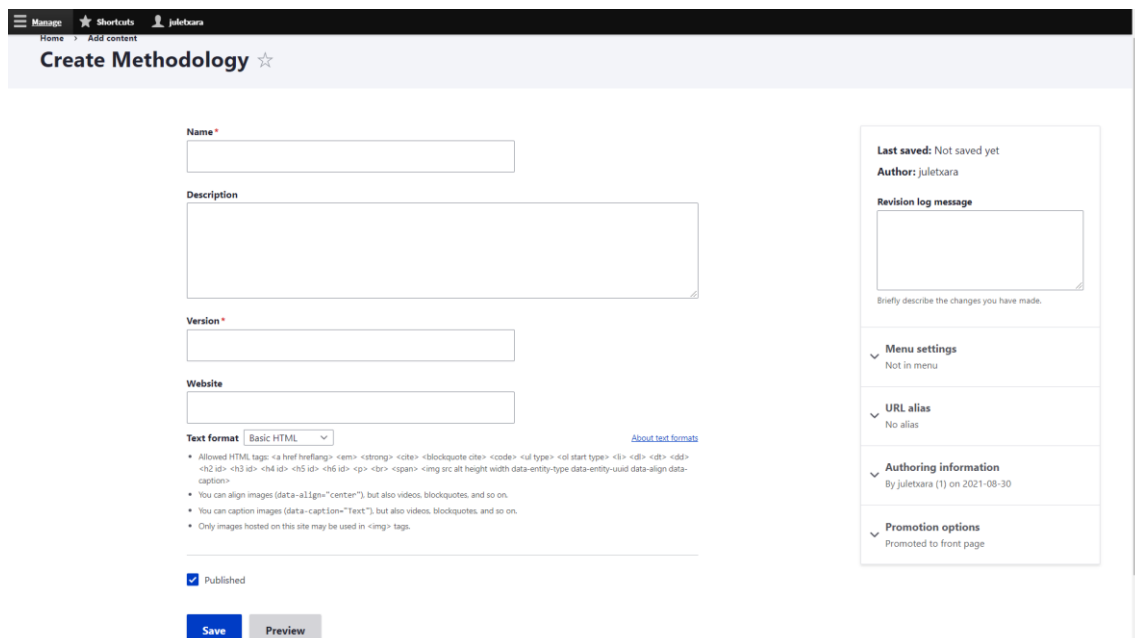
11.46. Irudia. Drupal webguneko kontua editatzeko orria.

Lehen bezala OpenUP metodologia bistaratzen badugu, aukera gehigarriak ditugula ikus daiteke 11.47. Irudian. Edukia ikustez gain, editatzeko, ezabatzeko eta bertsioak ikusteko aukerak dauzkagu.



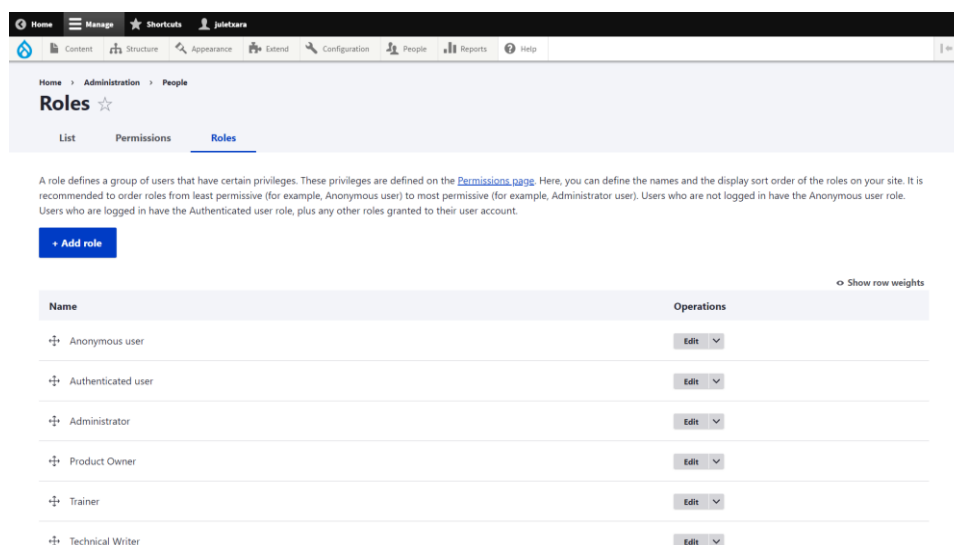
11.47. Irudia. Drupal webguneko OpenUP metodologiaren orria.

Edukia gehitzeko menu hedagarriaren bidez nahi dugun motako edukia gehitu dezakegu. Adibidez, metodologia aukeratzen badugu, metodologi sortzeko orria bistaratuko da, 11.48. Irudian ikusten den bezala. Edukia gorde aurretik ikusteko aukera ematen du, nola geratuko den jakiteko.



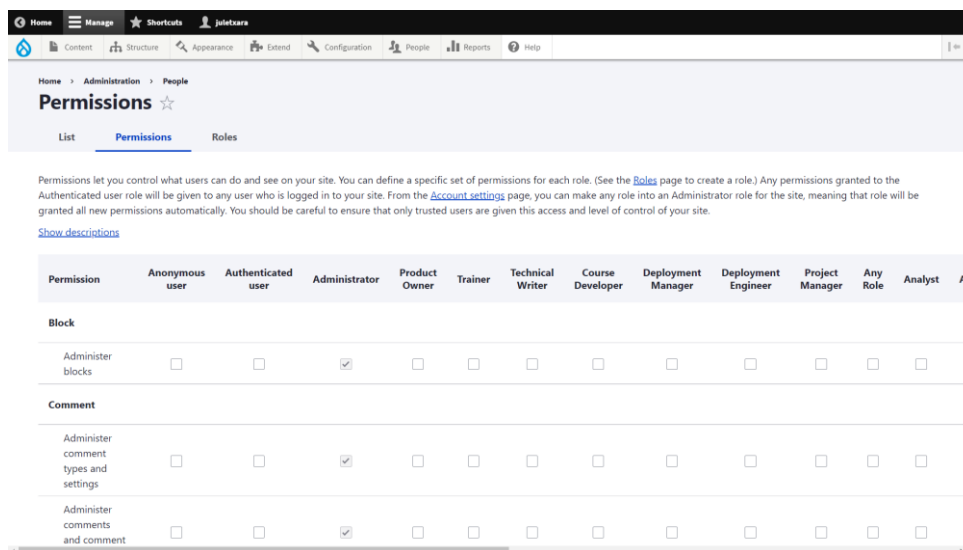
11.48. Irudia. Drupal webguneko metodologia sortzeko orria.

Orain arte aipatutako funtzionalitateak erabiltzaile anonimoak edo arrutak egin ditzakeenak dira. Erabiltzaile administrariak horiez gain webguneko edozein aspektu aldatzeko aukera dauka. Esaterako, rolak sortu ditzake eta erabiltzaileei rolak esleitu, 11.49. Irudian ikus daitezkeen bezala.



11.49. Irudia. Drupal webgunean rola kudeatzeko orria.

Rol bakoitzak dituen baimenak ere administrariak kudeatzen ditu, 11.50. Irudian ikusten den moduan. Horrela, nahikoa da rolaren baimenak behin zehaztea eta rol hori hainbat erabiltzaileri eslei dakioko.



11.50. Irudia. Drupal webgunean baimenak kudeatzeko orria.

11.4 Proba

Sistemaren probari dagokionez, proba kasuak zehaztu ez diren arren, sistemak funtzionamendu zuzena duela baieztatzeko hainbat egiaztapen egin dira.

XSLT eraldaketa egin aurretik errore pila bat zeuden ereduak. Pixkanaka eraldaketa agindu berriak gehitu dira erroreak gutxitu, bukaeran errore guztiak konpondu arte. XSLT eraldaketak ondo funtzionatzen ez badu, editore grafikoak erroreak ematen ditu. Gainera, erroredun atalak ez dira bistaratzen, zuzen daudenak bakarrik. Beraz, errorerik ez badago eredu osoa bistaratuko da editorean. Horrek esan nahi du eraldaketa ongi egin dela eta eredu metaereduarekin bat datorrela. Gainera, eraldaketa bi metodologia erabiliz egin denez, ziurtasun handiagoz esan dezakegu ongi dagoela.

Editoreen sinkronizazioari dagokionez, antzeko zerbait gertatzen da. Editoreen arteko sinkronizazioak huts egiten badu, beste editoreak errorea emango du. Kasu honetan, editoreen arteko sinkronizazioak ongi funtzionatzen du eredua fitxategi bakarrean dagoenean. Hori bai, eredua hainbat fitxategitan banatuta badago erreferentziak erabiliz, huts egiten du. Erabilitako eredua fitxategi pila batean banatuta dagoenez, sinkronizazioa ez dabil ondo eta ez dira testu editorearen fitxategiak sortzen. Fitxategi berri batean eredu sinple bat definitzen badugu, arazorik gabe sinkronizatzen dira editoreak.

Datu-basearen SQL kodea sortzerakoan, konprobatu behar dugu nahi ditugun datu guztien INSERT aginduak sortu direla. Aginduak zuzenak diren edo ez jakiteko, nahikoa da *phpMyAdmin* tresnan inportatzen dugunean erroreak ematen ez duela jakitea. Gainera, datu-basearen taulak definitzerakoan murriztapenak ongi zehazten baditugu, ez digu utziko betetzen ez dituzten datuak gehitzen. Adibidez, PRIMARY KEY erabilita ziurtatzen dugu taulen gakoak desberdinak direla eta FOREIGN KEY erabilita taulen arteko erreferentziak zuzenak direla.

Metodologiaren informazioa aurreko datu-basetik Drupal datu-basera pasatzeko, moduluen bidez mota bakoitzeko edukia gehitu behar da. Beraz, eduki motak ongi definitzen badira, sistemak ez du utziko edukia gehitzen ez baditu beharrezko eremu guztiak. Adibidez, datuak ez baditugu ordena egokian inportatzen erroreak agertuko dira, erreferentziatutako elementuak ez direlako existitzen. Ez daukagu konprobatu beharrik datu-basean gorde al diren, webguneko interfazeaz agertzen badira datu-basean baitaude.

Drupal webguneak proiektuen informazioa gordetzeko balio duela konprobatzeko, proiektu honen informazioa gehitu da. ProMeta izeneko proiektu bat sortu da eta proiektuko kideak eta rolak gehitu dira. Ondoren, ProMeta proiektuaren dokumentazio webgunean dauden OpenUP artefaktu guztiak gehitu dira, erabiltzaile bakoitzak berari dagozkionak.

11.5 Hedapena

Proiektuaren dokumentazioa eta inplementazioa publikoki eskuragarri egongo dira GitHub bidez eta webguneetan. Printzipioz, lana bukatu ondoren ere eskuragarri jarraituko dute, edozeinek kontsultatu ahal izan ditzan.

Proiektuaren dokumentazioaren kodea GitHub-en egongo da eskuragarri: <https://github.com/juletx/ProMeta>. Webgune hori automatikoki eraikiko da aldaketa bakoitzarekin <https://juletx.github.io/ProMeta> GitHub Pages erabiliz. GitHub Pages aukera ona da kasu honetan webgunea estatikoa delako.

Aurreko bi proiektuen webguneekin ere berdina egin dut, ProWF [1] eta BETRADOK [14]. ProWF proiektu honen aurrekaria denez kontsultatzeko behar dut. Eta BETRADOK proiektua antzekoa denez ongi etorriko zait ideiak hartzeko. ProWF proiektuaren errepositorioa <https://github.com/juletx/BETRADOK> eta webgunea <https://juletx.github.io/ProWF/>. BETRADOK proiektuaren GitHub errepositorioa <https://github.com/juletx/BETRADOK> eta GitHub Pages webgunea <https://juletx.github.io/BETRADOK/>.

Proiektuaren metaereduen atalaren inplementazioaren kodea ere GitHub-eko errepositorio batean dago: <https://github.com/juletx/ProMeta-ModelEditor>. Kodearen dokumentaziorako webgune bat erabiliko da, aurreko kasuetan bezala GitHub Pages erabiliz <https://juletx.github.io/ProMeta-ModelEditor>.

Prozesuaren webguneak ere aparteko GitHub errepositorioa edukiko du: <https://github.com/juletx/ProMeta-IO-System>. ProWF proiektuaren IO-System ere errepositorio batean jarriko da: <https://github.com/juletx/ProWF-IO-System>.

Dokumentazioarekin egiten den bezala, ondo egongo litzateke webgunea aldaketa bakoitzarekin automatikoki eraikitzea. Edo gutxienez Git-en bidez kontrolatu ahal izatea kode lokala eta zerbitzarikoa. Kasu honetan webgunea dinamikoa denez, beste hosting bat aurkitu beharko da, Drupal-erako balio duena.

Aukeren azterketa sakona egin eta gero, [Pantheon](https://dev-prometa.pantheonsite.io/) erabiltzea erabaki dut. Honek 3 webgune sortzeko aukera ematen du garapena errazteko: *Development* <https://dev-prometa.pantheonsite.io/>, *Test* <https://test-prometa.pantheonsite.io/> eta *Live* <https://live-prometa.pantheonsite.io/>. *Development* webgunea garapenerako erabiltzen da. *Live* webgunea erabiltzaileek edukia gehitzeko da. *Test* webgunea *Development*-eko hobekuntzak probatzeko erabiltzen da, *Live* webguneko edukiarekin. Webguneak *Test*-en funtzionatzen badu, *Live*-n ere funtzionatuko du.

Estrategia honekin ziurtatzen da nik eta tutoreak uneoro atal bakoitzaren azkenengo bertsioa kontsultatu dezakegula. Honek tutorearekin errebisioak egitea errazten du. Gainera, *git* bertsio kontrolari esker egindako aldaketa guztiak ikus daitezke. Horrez gain, webguneak automatikoki eraikitzeak lana errazten du, ez baitaukat zerbitzari batera igotzen ibili beharrik aldaketak dauden bakoitzean.

Proiektua amaitutakoan, lana [GAUR](https://gaurl.com/)en matrikulatu behar da eta zuzendariak oniritzia eman behar du. Ondoren, ikasleak lana [ADDI](https://addi.com/) plataformara igoko du. Horrez gain, ikasleak bere lanaren posterra bidali behar du dif.gral@ehu.eus helbidera. Gainera, zuzendariak emandako makinara igoko da lana, proiektu honi jarraipena emateko eskuragarri egon dadin.

Proiektu berriekin domeinu honetan sakondu eta emailtza hobeak lortu ahal izateko, orain arte bezala, proiektu honen emailtzen **jabetza intelektual**a partekatua izango da egile eta tutorearen artean.

11.6 Etorkizunerako Hobekuntzak

Proposatutako sistema prototipo bat denez, oraindik bere kalitate maila igo daiteke. Prototipoaren bidez lortutako emailtzak oinarri bezala hartuta, sistemaren eta proiektuaren hurrengo hobekuntzak proposatzen dira etorkizunerako:

- Editore grafikoaren eta testu editorearen arteko sinkronizazioa hobetzea. Horrela, eredu osoa editore grafiko eta testu editorearekin aldatzeko aukera egongo litzateke.
- Testu editorean edukia automatikoki formateatzea irakurgarritasuna hobetzeko.
- Garapen-prozesua BPMN lengoaia estandarra erabiliz definitzea, partekatze eta adoste lanak erraztu eta azkartzeko. Horretarako, metaereduak erabiliz ereduaren arteko eraldaketa egin daiteke automatikoki.
- Bezero ezberdin gehiagoren eskakizunak asetzeko gaitasuna izateko metodologia gehiagoren garapen-prozesuak definitzea, adibidez RUP eta Scrum metodologiak.
- Metodologiez gain proiektuak aurkezteko arauak ere garapen-prozesuan barneratzea, adibidez CCII-N2016-02 araua.
- IO-System azpisisteman rola eta baimenak hobeto kudeatzea. Rol bakoitzari beharrezko baimenak esleitzea falta da, dagozkion atazak egiteko aukera bakarrik izan dezan.
- IO-System azpisisteman erabiltzaile bakoitzari beharrezkoa den edukia bakarrik bistaratzea. Adibidez, egin beharreko atazak eta bete beharreko artefaktuak.
- Drupal webgunean artefaktu bakoitzerako eduki mota desberdin bat sortzea. Horrela, artefaktu bakoitzaren sekzioak Drupalen bertan defini daitezke eta errazagoa da rola eta baimenak kudeatzea.
- Drupal webgunean proiektuen diagnostikoak egitea eta bistaratzea, egiteko falta dena jakiteko eta proiektuaren egoera ezagutzeko.
- CMMI 2.0 kalitate-ereduaren 2. maila lortzeko garapen-prozesua osatzea.
- CMMI 2.0 kalitate-ereduaren 3. maila lortzeko garapen-prozesua osatzea.

12 Arriskuen Analisia

Atal honetan proiektuan zehar identifikatutako arriskuen inguruko analisia egiten da. Arriskuak zehazteko OpenUP metodologiako Arriskuen Zerrenda dokumentuan oinarritu gara. Atal honetan proiektuaren hurrengo faseetarako esanguratsuak diren arriskuak bakarrik agertzen dira. Hau da, bezeroak eraikuntza eta trantsizio faseekin aurrera jarraitzea erabakiko balu, kontuan hartu beharko lituzkeenak. Proiektu honen kudeaketarekin zerikusia duten arriskuak ikusi nahi badira, IV.1 eranskina kontsultatu daiteke.

12.1 Arriskuak

12.1. Taulan identifikatutako arriskuen zehaztasunak agertzen dira. Proiektuaren fase guztietan identifikatutako arriskuak zerrendatzen dira eta horien probabilitatea eta inpaktua zehazten da, arazoaren magnitudea neurtzeko. Gainera, arriskuei aurre egiteko mitigazio estrategiak definitzen dira.

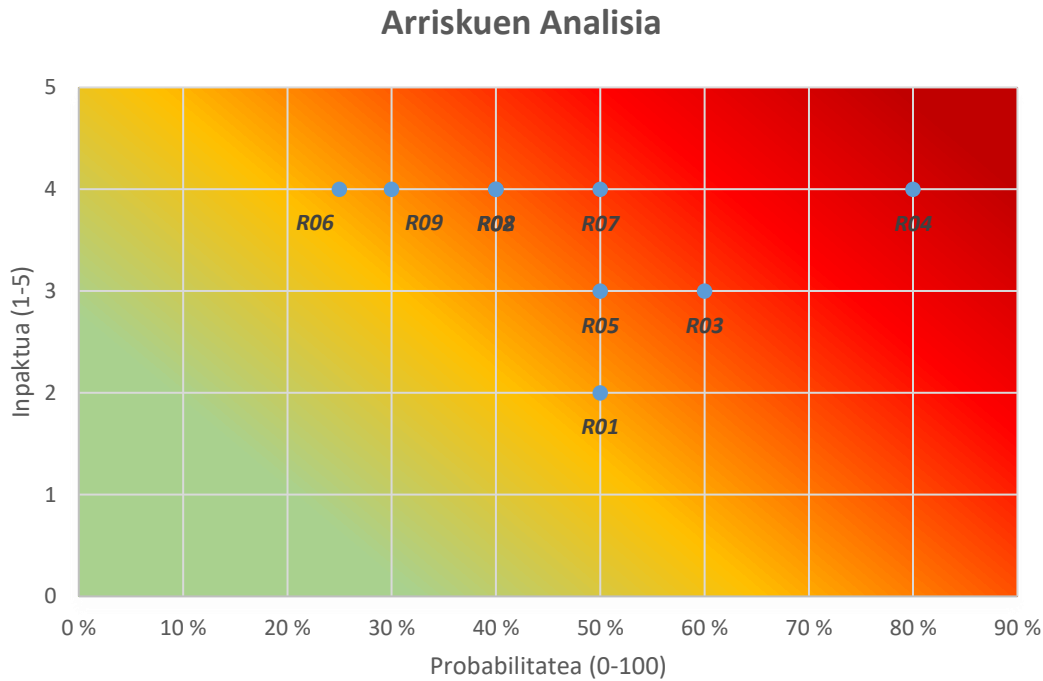
| ID | Izena | Deskribapena | Inpaktua | Probabilitatea (%) | Magnitudea | Mitigazio Estrategia |
|----|--------------------------------|--|----------|--------------------|------------|--|
| 01 | Lan ingurunearen prestakuntza | Lan ingurunearen instalazioan eta prestakuntzan agertu daitezken arazoak eta denbora galerak. | 2 | 50 | 1,0 | Instalazioan zehar egindakoa dokumentu batean idatzi, instalatuko dudako softwarearen espezifikazioak ondo irakurri. |
| 02 | Lan tresnekin arazoak | Lan tresnak erabiltzerakoan ager daitezkeen arazoak: bertsio bateraezintasunak, pluginak, erroreak... | 4 | 40 | 1,6 | Instalatzeko ongi ziurtatu bertsioak bateragarriak direla. Beharrezkoak diren pluginak bakarrik instalatu. |
| 03 | OpenUP ereduarekin arazoak | OpenUP eredu zaharra denez, baliteke guztiz osatua ez egotea eta bateraezintasunak egotea UMA metaereduarekin. | 3 | 60 | 1,8 | Ahalik eta bateragarrienak diren OpenUp eta UMA bertsioak aukeratu, ahal bada berrienak. |
| 04 | Xtext-ekin arazoak | Xtext tresnarekin arazoak. Xtext tresnak askotan erroreak sortzen ditu. Normalean erraz konpontzen dira, baina gure eredu oso handia denez, zailagoa izan daiteke. | 4 | 80 | 3,2 | Xtext-ekin lan egitean aldaketak murriztu. Errorerik ez dagoenean lana aurreratzeko aprobetxatu. |
| 05 | Tresnen aukeraketa okerra | Tresna okerra aukeratzeak lana asko atzeratu dezake. Izan ere, honek tresna aldatu beharra edo lana okerrago egitea eragin dezake. | 3 | 50 | 1,5 | Aukeren azterketa sakona egin tresnarentzako. Beharrezkoa bada hainbat tresna probatu, egokiena zein den jakiteko. |
| 06 | Metodologia gehiegi aukeratzea | Metodologia denak edo gehiegi aukeratzen baditugu ezarpena ez bukatzeko aukera daukagu. | 4 | 25 | 1,0 | Hasteko metodologia garrantzitsuenak bakarrik aukeratu. Ondoren |

| | | | | | | |
|----|---|---|---|----|-----|--|
| | | | | | | beharrezkoa bada metodologia gehiago gehitu. |
| 07 | Metodologia konplexu bakarra aukeratzea | Metodologia konplexu bakarra aukeratzen badugu, lan handia hartuko dugu eta azpiegitura teknologikoa konplexua izango da. Gainera, gerta daiteke sinpleagoa behar izatea. | 4 | 50 | 2,0 | Metodologia garrantzitsu bat baino gehiago aukeratu, sinpleak eta konplexuak. |
| 08 | Metodologia sinple bakarra aukeratzea | Metodologia sinpleegia aukeratzen bada, azkar bukatuko dugu, baina seguruena aurrerago konplexuagoak beharko ditugu. | 4 | 40 | 1,6 | Metodologia garrantzitsu bat baino gehiago aukeratu, sinpleak eta konplexuak. |
| 09 | CMMI mailak azkarregi lortzen saiatzea | Hasieratik CMMI kalitate-ereduaren 2. edo 3. mailak lortzea handinahia izan daiteke eta arazoak ekar ditzake. | 4 | 30 | 1,2 | CMMI mailak lortzearen erabakia seguru gaudenean bakarrik hartu, metodologiekin esperientzia lortu denean. |

12.1. Taula. Identifikatutako arriskuen zehaztasunak.

12.2 Arriskuen Analisia

Aurreko atalean arriskuen analisia egiteko arriskuen inpaktua eta probabilitatea bidertzen dira, eta magnitudea kalkulatzen da. Horrela, arrisku garrantzitsuenak zein diren argiago izango dugu uneoro. 12.1. Irudian, berriz, arriskuen inpaktu eta probabilitate grafika batean ikus daiteke. Grafika honek arriskuen ikuspegi orokorra ematen digu, arrisku bakoitzaren larritasuna erakutsiz. Adibidez, arrisku garrantzitsuenak eskuinean eta goian agertzen dira gorriz.



12.1. Irudia. Arriskuen inpaktu eta probabilitate analisia.

13 Proiektuaren Antolamendua eta Kudeaketa

Kapitulu honek proposatutako sistemaren proiektuaren antolaketa eta kudeaketa egiteko jarraitu beharreko plana deskribatzea du helburu. Honetarako, *Project Management Insitutek* gomendatutako *PMBOK* gida jarraitu da.

13.1 Proiektuaren Antolamendua

Proiektuaren antolaketari dagokionez, garrantzitsua da hasieratik hainbat aspektu argi edukitzea: taldekideen rola, informazio-sistema eta komunikazio-kanalak.

13.1.1 Taldekideak eta rola

Proiektu hau Gradu Amaierako Lana denez, ni naiz proiektuaren kide bakarra, Julen Etxaniz. Beraz, nik hartu beharko ditut OpenUp metodologian beharrezkoak diren rol guztiak: Proiektu Kudeatzailea, Analista, Arkitektoa, Probatzailea, Garatzailea, etab. Proiektuaren tutorea Juan Manuel Pikatza izango da.

13.1.2 Informazio-sistema

Sistemaren hedapenean komentatu den moduan, proiektuaren dokumentazioa eta inplementazioa publikoki eskuragarri egongo dira GitHub bidez eta webgune bidez. Printzipioz, lana bukatu ondoren ere eskuragarri jarraituko dute, edozeinek kontsultatu ahal izan ditzan. Gainera, amaitutakoan zuzendariak emandako makinara igoko da lana, proiektu honi jarraipena emateko eskuragarri egon dadin.

Estrategia honekin ziurtatzen da nik eta tutoreak uneoro atal bakoitzaren azkenengo bertsioa kontsultatu dezakegula. Honek tutorearekin errebisioak egitea errazten du. Gainera, git bertsio kontrolari esker egindako aldaketa guztiak ikus daitezke, eta egindako lana galtzea saihesten da. Horrez gain, webguneak automatikoki eraikitzeak lana errazten du, ez baitaukat zerbitzari batera igotzen ibili beharrik aldaketak dauden bakoitzean.

13.1.3 Komunikazio-kanalak

Gehienetan etxetik egingo dut lana. Batzuetan EHUKo liburutegian ere arituko naiz lanean. Bilerak gehienak BBC bidez online egingo ditugu eta beste batzuk presentzialki Informatika Fakultatean. Printzipioz, bilerak astero egingo ditugu aurretik adostutako ordu batean. Beharrezkoa bada eguna edo ordua aldatu dezakegu. Bilerekin gain posta elektronikoa erabiliko dugu elkarrekin komunikatzeko.

13.2 Proiektuaren Kudeaketa

Aurretik aipatu den bezala, proiektuaren kudeaketa PMBOK gida-liburuaren arabera egin da. Hori dela eta, liburu horrek aipatzen dituen kudeaketa-motak deskribatzen dira atal honetan.

13.2.1 Integrazioaren Kudeaketa

Atal honen helburua proiektuan zehar ematen diren prozesu desberdinak identifikatu, definitu, konbinatu, bateratu eta koordinatzeko eman beharreko pausuak kudeatzea da.

Proiektuaren hasiera tutorearekin egin beharreko lana adostutakoan izan zen. Lan eskaintza publikoan aukeratutakoa izan zen. Enpresarik ez dagoenez, ez dago bestelako akordiorik sinatu beharrik.

Proiektuaren plangintza eramateko OpenUP metodologiaren Proiektu Plana erabili da, helburuak eta irismena definitzeko. Gainera, CCII-N2016-02 estandarraren arabera, proiektuko kudeaketako beste esparru batzuetako kudeaketa planak ere egin dira, hurrengo ataletan daudenak.

Proiektua zuzentzeko eta kudeatzeko ardura bi aktore garrantzitsuenena izango da: Julen Etxaniz egilea eta Juan Manuel Pikatza tutorea. Proiektua epe motzetan zuzentzeko eta kudeatzeko OpenUP metodologia erabili da, hilabete bateko iterazioekin. Iterazioak definitzeko Iterazio Planak erabili dira.

13.2.2 Irismenaren Kudeaketa

Atal honen zeregina da proiektuaren irismena kudeatzea, hots, proiektuaren espezifikazioak eta denbora zeintzuk diren kontuan hartuta, proiektua zenbateraino garatu daitekeen estimatzea eta kudeatzea.

Proiektuaren Ikuspegia bete den lehenengo dokumentua izan da. Izan ere, dokumentu horretan konpondu nahi dugun arazoa eta proposatutako produktua deskribatzen dira. Gainera, proiektuko interesatuen erantzukizunak ere zehazten dira.

Proiektuaren betekizunak definitzeko OpenUP metodologiako Ikuspegia eta Betebeharren Espezifikazioa dokumentuak erabili dira. Ikuspegian betekizun funtzional eta ez-funtzionalak zehaztu dira. Betebeharren Espezifikazioan beste betekizunak definitu dira: kalitatea, interfazeak, konplimendua etab. Erabilpen Kasuen dokumentuek eta ereduak ere betekizunak ulertzen laguntzen dute.

Proiektuaren irismena zehazteko, LDE diagrama definitu da lan-pakete eta ataza nagusiekin. Proiektua lan-paketetan zatitzeko ondorengo atazak egin dira:

1. Emangarriak eta erlazionatutako lana identifikatu eta aztertu.
2. LDE diagrama egituratu eta antolatu.
3. LDE diagramako goi mailako elementuak, zehatzagoak diren maila txikiagoko osagarrietan zatitu.
4. Identifikazio kodea garatu eta LDE diagramako elementuei kode hori esleitu.
5. Emangarrien deskonposaketa maila egokia dela ziurtatu.

13.2.3 Epeen Kudeaketa

Dokumentu honen xedea da garrantzitsuak diren mugarriak identifikatzea, eta proiektuan zehar egon daitezkeen arazoak direla eta epeak aldatu behar badira, gertakizun hori kudeatzeko plana zehaztea.

Atazak zehaztu eta sekuentziazteko OpenUPeko Lan Atazen Zerrenda erabili da, atazen kudeaketa egiteko aproposak diren datuak zehazten dituen taula. Adibidez, ataza bakoitzaren lehentasuna, tamaina, iterazioak eta denbora estimazioa zehazten da.

Atazen plangintza egiteko, Gantt diagrama erabili da. Atazen plangintza zehatza egitea oso zaila denez, diagrama honek iterazioko zehaztasuna izango du, hau da, hilabete bakoitzean egingo diren atazak zehaztuko ditu.

Proiektuaren mugarriei dagokionez, iterazio hasiera eta amaierekin eta proiektuaren entregarekin zerikusi duten mugarriak zehaztu dira. Gerta daiteke proiektua epe horietan bukatzeko denborarik ez izatea eta epea atzeratu behar izatea. Horrek proiektua uztailan aurkeztu ordez irailean aurkeztu beharra eragingo luke. Kontuan hartu behar da behin proiektuaren defentsa eskatzen denean derrigorrez aurkeztu behar dela, eta beraz ekainaren 11rako argi eduki behar dela noiz aurkeztu behar den.

13.2.4 Kostuen Kudeaketa

Atal honetan definitzen dira proiektua aurrera eramateko behar diren kostu ekonomikoak eta horiek nola kontrolatu.

Proiektuan egon daitezkeen kostu ekonomikoak aurreikusiak daudela onartzen da. Hori dela eta, pentsatzen da kostuak ez dutela plangintza eta kontrol berezirik beharko. Ahal den heinean, erabili beharreko softwarea librea izan behar da.

Beraz, estimatzen da tresna aldetik proiektuak ez duela kosturik izango. Hori bai, proiektuko kideen lan orduak kontuan hartuz, proiektuaren kostua zenbatekoa izango litzatekeen estimatzen da. Kostu hauek guztiak Aurrekontuan definitzen dira, doakoak diren tresnak ere zerrendatuz.

13.2.5 Kalitate Kudeaketa

Atal honetan proiektuan zehar helburu diren kalitate betekizunak ahalik eta hoberen gauzatzeko hurrengo jarraitu beharreko prozesua deskribatzen da.

Proiektuaren tamaina eta bereziki, egile bakar batek egindako proiektua dela kontuan hartuta, kalitatea zaintzeko hiru aspektu kontrolatu behar dira:

- **Implementazioaren kalitatea:** kode-mailako implementazioa egokia dela eta software arruntren patroiak jarraitzen direla bermatzeko.
- **Funtzionalitate eta ezaugarrien kalitatea:** produktuak bete behar dituen ezaugarri eta espezifikazioen kontrola.
- **Memoria, dokumentazioa eta aurkezpenaren kalitatea:** proiektuak atxikituta izan behar duen dokumentazio eta memoriaren kalitatea. Adostutako estandarrak betetzen diren (CCII eta OpenUP) ziurtatu behar da, bai eta artefaktu guztiak atal edo sekzio guztiekin beteta daudela. Gainera, webgunearen kalitate kontrola egin behar da.

Proiektuko rolen eta kalitate kudeaketako arduren banaketa 13.1. Taulan ikus daiteke. Proiektuaren egileak etengabe hiru aspektuak kontrolatu beharko ditu proiektuaren bukaerara arte. Proiektuaren tutoreari dagokionez, memoria, dokumentazioa eta aurkezpenaren kalitate-kontrola egiten lagunduko du. Horretarako, astean behin egiten diren bileretan jasotako feedbacka erabiliko da.

| Rolak | Ardurak |
|------------------------------|--|
| Egilea: Julen Etxaniz | Implementazioaren kalitatea
Funtzionalitate eta ezaugarrien kalitatea
Memoria, dokumentazioa eta aurkezpenaren kalitatea |
| Tutorea: Juan Manuel Pikatza | Memoria, dokumentazioa eta aurkezpenaren kalitatea |

13.1. Taula. Kalitate kudeaketako rolak eta ardurak.

13.2.6 Giza Baliabideen Kudeaketa

Atal honen helburua da proiektuan parte hartuko duten aktore guztien kudeaketa deskribatzea da.

Proiektu hau Gradu Amaierako Lana denez, ni naiz proiektuaren kide bakarra, Julen Etxaniz. Beraz, nik hartu beharko ditut OpenUp metodologian beharrezkoak diren rol guztiak: Proiektu Kudeatzailea, Analista, Arkitektoa, Probatzailea, Garatzailea, etab. Proiektuaren tutorea Juan Manuel Pikatza izango da.

Beraz, giza baliabideen kudeaketa nahiko sinplea izango da. Egileak rol bakoitzarekin pasatako denbora kudeatu beharko da. Tutorearekin komunikazioa kudeatuko da, postaz eta bileretan.

13.2.7 Komunikazioen Kudeaketa

Komunikazioen kudeaketari begira, atal honetan definitzen dira zein pausu jarraitzen diren kideen artean komunikatzeko.

Ahalik eta komunikazio hoberena edukitzeko lehenengo interesatuak identifikatu dira. Ondoren, komunikazioak planifikatu dira. Komunikaziorako bi kanal nagusi erabiliko dira, korreoa eta bilerak.

Korreo bidezko komunikazioa bilera baino lehenago komentatu beharreko aspektuetarako erabiliko da. Komunikazioa EHUKo posta edo Gmail erabiliz gauzatuko da. Beraz, nahiko irregularra izango da, baliteke aste batzuetan hainbat aldiz idaztea eta beste batzuetan ezer ez.

Bilerei dagokionez, astean behin egingo dira adostutako orduan BBC bidez. Hala ere, tutorearekin adostuta eguna eta ordua alda daitezke beharrezkoa bada. Bilerak online egitea erabaki dugu Covid19-ren egoeratik eta ordutegia aukeratzea errazagoa delako. Gainera, pantaila partekatzeko aukera izateak bileretan asko lagundu digu.

13.2.8 Arriskuen Kudeaketa

Atal honen zeregina proiektuan zehar gerta daitezkeen arriskuen identifikatzea, arrisku horiek ebazteko prozesua definitzea eta kontrolatzea da.

Lehenik eta behin, proiektua planteatutakoan arrisku-zerrenda bat egin da OpenUP metodologia erabiliz. Arrisku bakoitzarentzat analisi kualitatibo eta kuantitatiboa egin da, inpaktua eta probabilitatea estimatuz. Horrez gain, mitigazio estrategiak definituko dira arriskuei aurre egiteko. Gainera, gerta daiteke proiektua garatu bitartean arrisku gehiago agertzea. Kasu horretan, arriskua zerrendatu eta analisisa egingo da.

13.2.9 Erosketen Kudeaketa

Atal honen helburua proiektuko erosketak kudeatzeko plana zehaztea da.

Kostuen kudeaketan azaldu den moduan, ez da aurreikusten erosketarik egin beharra. Izan ere, erabiliko diren tresna guztiak libreak izatea da helburua. Gainera, ordainpekoak diren software batzuk eskuragarri daude tutoreak eskainitako makina birtualean beharko balira. Hala ere, zerbait erosi beharko balitz egileak kudeatuko luke tutorearen laguntzarekin.

13.2.10 Interesatuena Kudeaketa

Atal honen helburua proiektuko interesatuak identifikatzeko, kudeatzeko eta kontrolatzeko pausuak zehaztea da.

Interesatuak eta horien erantzukizunak proiektuaren hasieran finkatuko dira Ikuspegia dokumentuan. Interesatu garrantzitsuenak kontuan hartzen direla ziurtatu beharko da proiektuaren garapenean. Esan bezala, interesatuena arteko komunikaziorako korreoa eta bilerak erabiliko dira.

14 Denbora Planifikazioa

Atal honen helburua da proiektuaren hurrengo faseetarako denbora planifikazioa azaltzea da. Hau da, bezeroak proiektua aurrera eramatea erabakiko balu baliagarria izango litzatekeen planifikazioa. Proiektu honen denbora planifikazioa IV.2 eranskinean azaltzen da. Bertan mugarri garrantzitsuak, atazen denborak, iterazioak eta desbiderapenak zehazten dira.

Proiektu honetan lehenengo bi iterazioak Hasiera fasekoak izan dira eta beste bostak Elaborazio fasekoak. Bezeroak proiektuari jarraipena ematea erabakiko balu, Eraikuntza eta Trantsizio faseak egin beharko lituzke. OpenUP metodologian gutxi gorabehera hauek dira fase bakoitzerako aurreikusitako iterazio kopuruak: Hasiera fasean 1, Elaborazio fasean 2, Eraikuntza fasean 3 eta Trantsizio fasean 1. Hasiera eta Elaborazio faseetan behar izan dugun denbora kontuan hartuta, aurreikusi daiteke gutxienez Eraikuntza faserako 6 iterazio beharko direla eta Trantsizio faserako 2.

Egin beharreko atazei dagokionez, OpenUP metodologiak Eraikuntza eta Trantsizio faseetarako zehazten dituenak izango lirateke. Fase horietan azpisistemen prototipoak osatu eta hobetu egin behar dira, bukaerako sistema osatu arte. Proiektu honetan egin den moduan, fase eta iterazio bakoitzean OpenUP metodologiako artefaktuak betetzea garrantzitsua da.

15 Aurrekontuaren Laburpena

Aurrekontua sortzeko *ALI (Asociación de Titulados Universitarios Oficiales en Informática)* elkarteak banatutako irizpide batzuk jarraitu dira. Batetik, giza baliabideen barne-kostuak eta kanpo-kostuak ateratzea bere ordu kopuruekin batera. Gure kasuen ez dago kanpo-kosturik. Bestetik, proiektua garatzeko behar izan diren erreminten kostua kalkulatzea.

Erabilitako erreminta guztiak doakoak izan dira, beraz, arkitektura propioa eraikitzearen erabakia egokia izan da. Bizagiren arkitektura erabiliz bere lizentziak eta urteroko mantenuak proiektuaren kostua handituko lukete. Beste alde batetik, ez dira aurkitu erabilitako erreminten premium lizentziarik, baina egotekotan aurrekontuan sartu daitezke, sistemaren kalitatea handitzeko asmoz. Proiektuko kide bakoitzak beharko lituzkeen orduak estimazioak direnez, aurrekontua eta kostu errealaren artean alde handia egon daiteke. 15.1. Taulan proposatutako sistema bideragarria izateko aurrekontua ikus daiteke:

| PARTIDA | | PARAMETROAK | | TOTALA BEZ GABE | TOTALA BEZ BARNE |
|------------------|------------------|-------------|----------|-----------------|------------------|
| Giza Baliabideak | | Orduak | Kostua | | |
| 1 | Zuzendaria | 200 | 100 | 20.000 | 24.200 |
| 2 | Arkitektoa | 150 | 70 | 10.500 | 12.705 |
| 3 | Analista | 200 | 70 | 14.000 | 16.940 |
| 4 | Garatzailea | 350 | 50 | 17.500 | 21.175 |
| 5 | Probatzailea | 100 | 50 | 5.000 | 6.050 |
| Totala | | | | 67.000 | 81.070 |
| Erremintak | | Lizentzia | Mantenua | | |
| 1 | XAMPP | 0 | 0 | 0 | 0 |
| 2 | Drupal | 0 | 0 | 0 | 0 |
| 3 | VSCode | 0 | 0 | 0 | 0 |
| 4 | PlantUML | 0 | 0 | 0 | 0 |
| 5 | Eclipse | 0 | 0 | 0 | 0 |
| 6 | Git | 0 | 0 | 0 | 0 |
| 7 | GitHub | 0 | 0 | 0 | 0 |
| 8 | Java | 0 | 0 | 0 | 0 |
| 9 | Microsoft Office | 0 | 0 | 0 | 0 |
| 10 | Pantheon | 0 | 0 | 0 | 0 |
| 11 | Toggl Track | 0 | 0 | 0 | 0 |
| 12 | EPF Composer | 0 | 0 | 0 | 0 |
| 13 | Xtext | 0 | 0 | 0 | 0 |
| 14 | PHP | 0 | 0 | 0 | 0 |
| 15 | GitHub Pages | 0 | 0 | 0 | 0 |
| Totala | | | | 0 | 0 |
| TOTALA | | | | 67.000 | 81.070 |

15.1. Taula. Aurrekontua giza baliabideen eta erreminten kostuekin.

Giza baliabideen kostua Ekonomia eta Ogasun Ministerioaren 26/2010 Esparru Akordioan oinarrituta dago. Bezeroari begira software proiektu baten kide bakoitzarentzako erabiltzen dugun baremoa 15.2. Taulan ikus daiteke.

| Kidea | Kostua (€/ordu) |
|-----------------------|-----------------|
| Proiektuko zuzendaria | 100 |
| Arkitektoa | 70 |
| Analista | 70 |
| Garatzailea | 50 |
| Probatzailea | 50 |

15.2. Taula. Software taldeko kideen kostu baremoa.

16 Dokumentuen Lehentasun Ordena

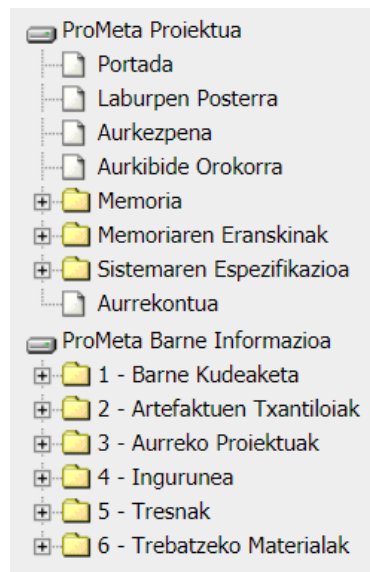
Dokumentazio luze honetan inkoherentziak egotea posible izan daiteke. Proiektuaren garapena luzea izan da, dokumentu asko idatzi dira eta gerta daiteke dokumenturen batean agertzen den baieztapen bat kontrajartzea beste dokumentu batean agertzen den baieztapen batekin edo dokumentu batean agertutako datu bat beste batean ezberdina izatea.

Hori dela eta, memoria izango da kontuan hartu beharreko informazioa inkoherentzien kasuan. Memoria dokumentu askoren bilketa da azken finean, baita proiektuaren azkenekoz idatzitako dokumentua. Horregatik, irakurleak memoria kontsultatu beharko du zalantzarik izanez gero.

Hala ere, beti prest egongo naiz edozein zalantza edo arazo argitzeko. Nirekin kontaktuan jartzeko posta elektronikoa erabil daiteke: juletxara@gmail.com.

ERANSKINAK

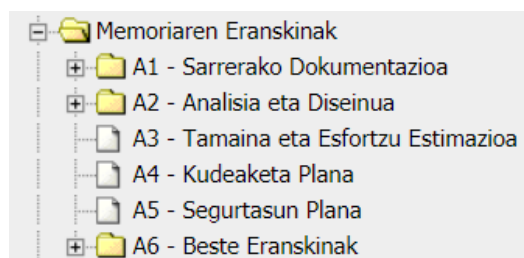
Eranskin guztiak proiektuaren webgunean daude, <https://juletx.github.io/ProMeta/>. Webgune honen helburua ProMeta proiektuaren dokumentazio guztia biltzea eta proiektuko bezero zein interesdunek eskura izatea da. Eranskin batzuetan dokumentuak daude eta besteetan berriz estekak bakarrik. 0.1. Irudian ikus daitezkeen moduan, eranskinak 3 multzotan sailkatzen dira: Memoriaren Eranskinak, Sistemaren Espezifikazioa eta Barne Informazioa. Atal honetan eranskinen azalpen orokorra egingo da irudiekin, eranskin bakoitza ikusteko webgunea kontsultatu daiteke. IV. atalean arriskuen analisia eta denbora planifikazioa osorik agertuko dira.



0.1. Irudia. Eranskinak webgunekeo menuan.

I. Memoriaren Eranskinak

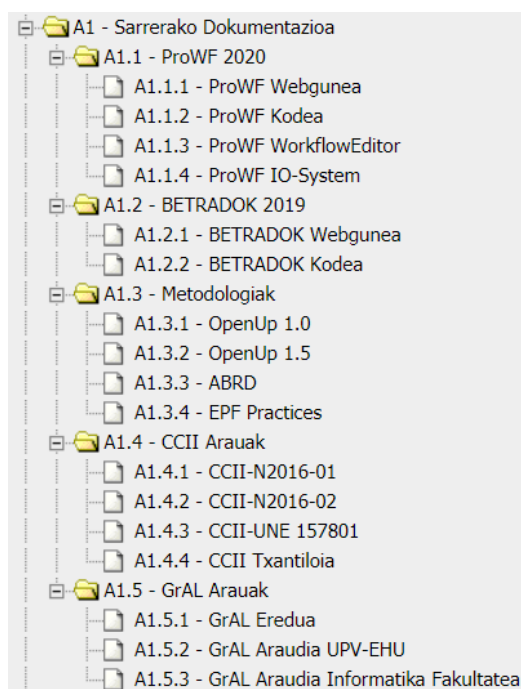
Memoriaren Eranskinak 0.2. Irudian ikus daitezkeen 6 multzotan sailkatzen dira: Sarrerako Dokumentazioa, Analisia eta Diseinua, Tamaina eta Esfortzu Estimazioa, Kudeaketa Plana, Segurtasun Plana eta Beste Eranskinak.



0.2. Irudia. Memoriaren Eranskinak webgunekeo menuan.

A1 - Sarrerako Dokumentazioa

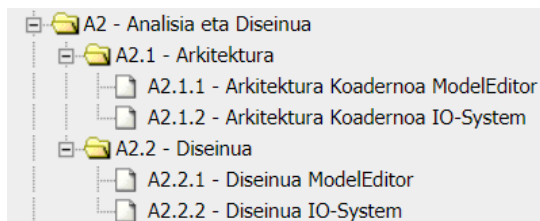
Sarrerako Dokumentazioaren helburua proiektua eta dokumentazioa gauzatzeko egileak kontuan hartu beharreko dokumentazioa biltzea da. Atal honetan hainbat dokumentu eta webgune esteka daude, 0.3. Irudian ikusten den bezala. Alde batetik, proiektu honekin erlazionatuta dauden aurreko proiektuak daude, ProWF eta BETRADOK. Bestetik, proiektua gauzatzeko erabilitako metodologia eta arauak biltzen dira, OpenUP, CCII eta GrALarekin zerikusia dutenak.



0.3. Irudia. Sarrerako Dokumentazioa webguneko menuan.

A2 - Analisia eta Diseinua

Analisia eta Diseinua eranskinak sistemaren Analisia, Arkitektura eta Diseinua biltzen ditu. Kasu honetan, azpisistema bakoitzarentzat dokumentu bana egitea erabaki da, 0.4. Irudian ikusten den moduan.



0.4. Irudia. Analisia eta Diseinua webguneko menuan.

A3 - Tamaina eta Esfortzu Estimazioa

Tamaina eta Esfortzu Estimazioaren helburua da proiektuaren tamaina eta esfortzuaren estimazioa azaltzea da. Beraz, mugarriak, lan-atazak, iterazioak, LDE eta Gantt diagramak agertzen dira eranskin honetan.

A4 - Kudeaketa Plana

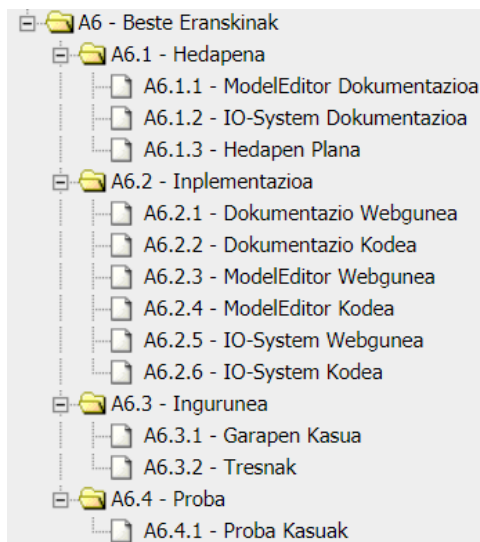
Kudeaketa Plana PMBOK gida-liburuaren arabera egin da. Hori dela eta, bertan aipatzen diren kudeaketa motak deskribatzen dira eranskin honetan: Integrazioaren, Irismenaren, Epeen, Kostuen Kalitateren, Giza Baliabideen, Komunikazioen, Arriskuen, Erosketen eta Interesatuen kudeaketa.

A5 - Segurtasun Plana

Segurtasun Planaren helburua da proiektuarekin erlazionatutako elementu eta entitateen segurtasuna definitzea eta proiektuan segurtasuna integratzeko neurriak definitzea. Horretarako, hedapen plana kontuan hartzea nahikoa da, bertan definitzen baitira proiektuaren segurtasunari eta konfidentziasunari buruzko aspektu garrantzitsuenak.

A6 - Beste Eranskinak

Beste Eranskinetan proiektuan zehar sortu diren beste eranskin batzuk agertzen dira, proiektua gehiago ulertzeko beharrezkoak direnak baina aurreko atalei ez dagozkienak. 0.5. Irudian ikusten bezala, OpenUP metodologiaren artefaktuak daude, 5 multzotan sailkatuta: Hedapena. Garapena, Implementazioa, Ingurunea eta Proba.

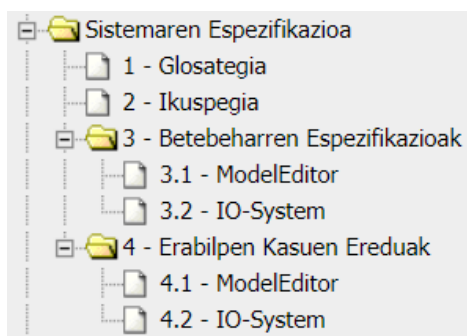


0.5. Irudia. Beste Eranskinak webguneko menuan.

II. Sistemaren Espezifikazioa

Atal honen helburua sistema osoaren espezifikazioak definitzea da, ahalik eta zehaztapen gehien emanaz. Proiektuan software arina garatzeko OpenUp metodologia erabiltzen denez, honek zehazten ditu atal honetan egon behar diren dokumentuak, baita dokumentu horiek eduki behar dituzten sekzioak ere. 0.6. Irudian ikusten den moduan, honako dokumentuak daude:

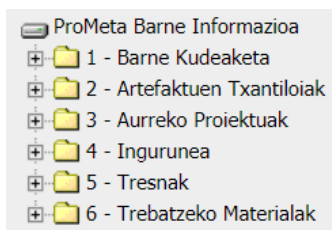
- **Glosategia:** proiektuan egon daitezkeen akronimo, laburdura edo hitz-berezien definizioak dituen bilduma da.
- **Ikuspegia:** proiektuaren ikuspegi orokorra definitzen du eta proiektuaren arazoa kokatzen du.
- **Betebeharren Espezifikazioak:** azpisistema bakoitzaren betebehrrak espezifikatzen ditu.
- **Erabilpen Kasuak:** proiektuan dauden erabilpen kasu garrantzitsuenen deskripzio bat biltzen duen dokumentua.
- **Erabilpen Kasuen Ereduak:** proiektuko erabilpen kasuen ereduko diagrama azpisistema bakoitzerako.



0.6. Irudia. Sistemaren Espezifikazioa webguneko menuan.

III. Barne Informazioa

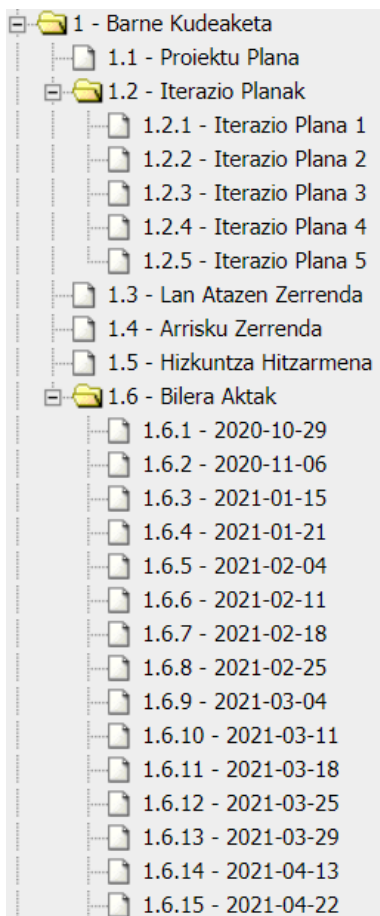
Barne Informazioan proiektuaren garapenerako garrantzitsuak diren baina bezeroari bidaliko ez zaizkion dokumentuak gordetzen dira. 0.7. Irudian ikusten den bezala, Barne Kudeaketa, Artefaktuen Txantiloiak, Aurreko Proiektuak, Ingurunea, Tresnak eta Trebatzeko Materialak daude.



0.7. Irudia. Barne Informazioa webgunearen menuan.

III.1 Barne Kudeaketa

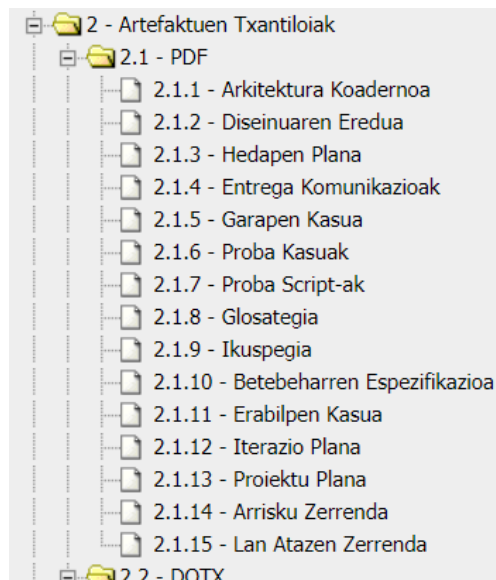
Barne Kudeaketan proiektuan zehar kudeaketarako erabilitako dokumentuak biltzen dira, 0.8. Irudian agertzen den moduan. Alde batetik, OpenUP metodologiak zehazten dituen artefaktuak daude: Proiektu Plana, Iterazio Plana, Arrisku Zerrenda eta Lan Atazen. Bestetik, hizkuntza hitzarmena eta proiektuan zehar egindako bilera guztien aktak daude.



0.8. Irudia. Barne Kudeaketa webguneko menuan.

III.2 Artefaktuen Txantiloiak

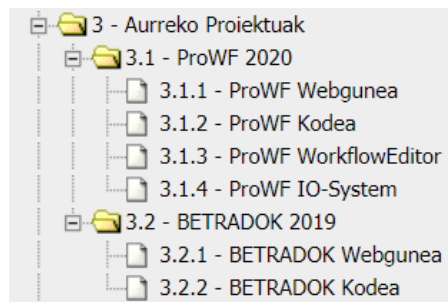
Artefaktuen Txantiloiak atalean OpenUP metodologiak eskaintzen dituen txantiloiz guztiak biltzen dira PDF eta DOTX formatuetan, beharrezkoa denean eskuragarri edukitzeko. 0.9. Irudian ikusten den moduan, 15 txantiloia daude baina ez dira guztiak erabili.



0.9. Irudia. Artefaktuen Txantiloiak webguneko menuan.

III.3 Aurreko Proiektuak

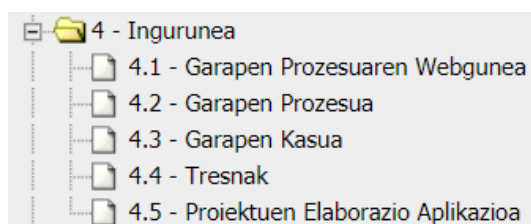
Aurreko Proiektuak atalean erlazionatutako proiektuen kode eta webguneetako estekak daude, 0.10. Irudian ikusten den moduan.



0.10. Irudia. Aurreko Proiektuak webguneko menuan.

III.4 Ingurunea

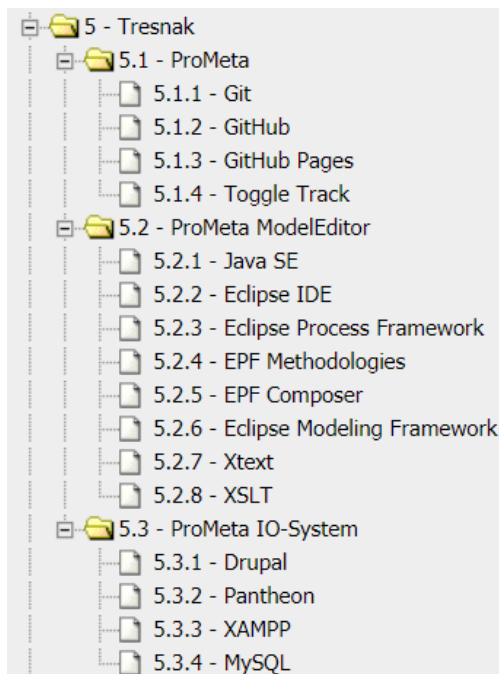
Ingurunean proiektuan jarraitutako garapen prozesuaren inguruko zehaztasunak agertzen dira, 0.11. Irudian agertzen direnak.



0.11. Irudia. Ingurunea webguneko menuan.

III.5 Tresnak

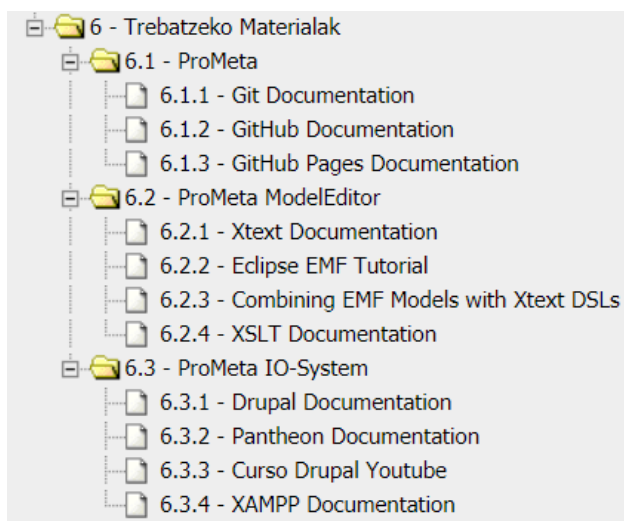
Tresnak atalean proiektua garatzeko erabilitako tresna garrantzitsuen webguneetara eramaten duten estekak daude. 0.12. Irudian ikusten den bezala, 3 multzotan daude, dokumentazio webguneari eta bi azpisisistemei dagozkienak.



0.12. Irudia. Tresnak webgunekeo menuan.

III.6 Trebatzeko Materialak

Trebatzeko Materialak atalean erabilitako dokumentazio eta tutorial garrantzitsuetara estekak daude. 0.13. Irudian ikusten den bezala, 3 multzotan daude, dokumentazio webguneari eta bi azpisisistemei dagozkienak.



0.13. Irudia. Trebatzeko Materialak webgunekeo menuan.

IV.1 Arriskuen Analisia

Atal honetan proiektuan zehar identifikatutako arriskuen inguruko analisia egiten da. Arriskuak zehazteko OpenUP metodologiako Arriskuen Zerrenda dokumentuan oinarritu gara. Eranskin honetan proiektuan identifikatutako arrisku guztiak agertzen dira. Alde batetik, hurrengo faseetarako esanguratsuak diren arriskuak bakarrik agertzen dira. Hau da, bezeroak eraikuntza eta trantsizio faseekin aurrera jarraitzea erabakiko balu, kontuan hartu beharko lituzkeenak. Bestetik, proiektu honen kudeaketan bakarrik eragina dutenak.

IV.1.1 Arriskuak

12.1. Taulan identifikatutako arriskuen zehaztasunak agertzen dira. Proiektuaren fase guztietan identifikatutako arriskuak zerrendatzen dira eta horien probabilitatea eta inpaktua zehazten da, arazoen magnitudea neurtzeko. Gainera, arriskuei aurre egiteko mitigazio estrategiak definitzen dira.

| ID | Izena | Deskribapena | Inpaktua | Probabilitatea (%) | Magnitudea | Mitigazio Estrategia |
|----|--------------------------------|---|----------|--------------------|------------|---|
| 01 | Lan ingurunearen prestakuntza | Lan ingurunearen instalazioan eta prestakuntzan agertu daitezken arazoak eta denbora galerak. | 2 | 50 | 1,0 | Instalazioan zehar egindakoa dokumentu batean idatzi, instalatuko dudan softwarearen espezifikazioak ondo irakurri. |
| 02 | Lan tresnekin arazoak | Lan tresnak erabiltzerakoan ager daitezkeen arazoak: bertsio bateraezintasunak, pluginak, erroreak... | 4 | 40 | 1,6 | Instalatzerakoan ongi ziurtatu bertsioak bateragarriak direla. Beharrezkoak diren pluginak bakarrik instalatu. |
| 03 | OpenUP ereduarekin arazoak | OpenUP eredua zaharra denez, baliteke guztiz osatua ez egotea eta bateraezintasunak egotea UMA metaereduarekin. | 3 | 60 | 1,8 | Ahalik eta bateragarrienak diren OpenUp eta UMA bertsioak aukeratu, ahal bada berrienak. |
| 04 | Xtext-ekin arazoak | Xtext tresnarekin arazoak. Xtext tresnak askotan erroreak sortzen ditu. Normalean erraz konpontzen dira, baina gure eredua oso handia denez, zailagoa izan daiteke. | 4 | 80 | 3,2 | Xtext-ekin lan egitean aldaketak murriztu. Errorerik ez dagoenean lana aurreratzeko aprobetxatu. |
| 05 | Tresnen aukeraketa okerra | Tresna okerra aukeratzeak lana asko atzeratu dezake. Izan ere, honek tresna aldatu beharra edo lana okerrago egitea eragin dezake. | 3 | 50 | 1,5 | Aukeren azterketa sakona egin tresnentzako. Beharrezkoa bada hainbat tresna probatu, egokiena zein den jakiteko. |
| 06 | Metodologia gehiegi aukeratzea | Metodologia denak edo gehiegi aukeratzen baditugu ezarpena ez bukatzeko aukera daukagu. | 4 | 25 | 1,0 | Hasteko metodologia garrantzitsuenak bakarrik aukeratu. Ondoren |

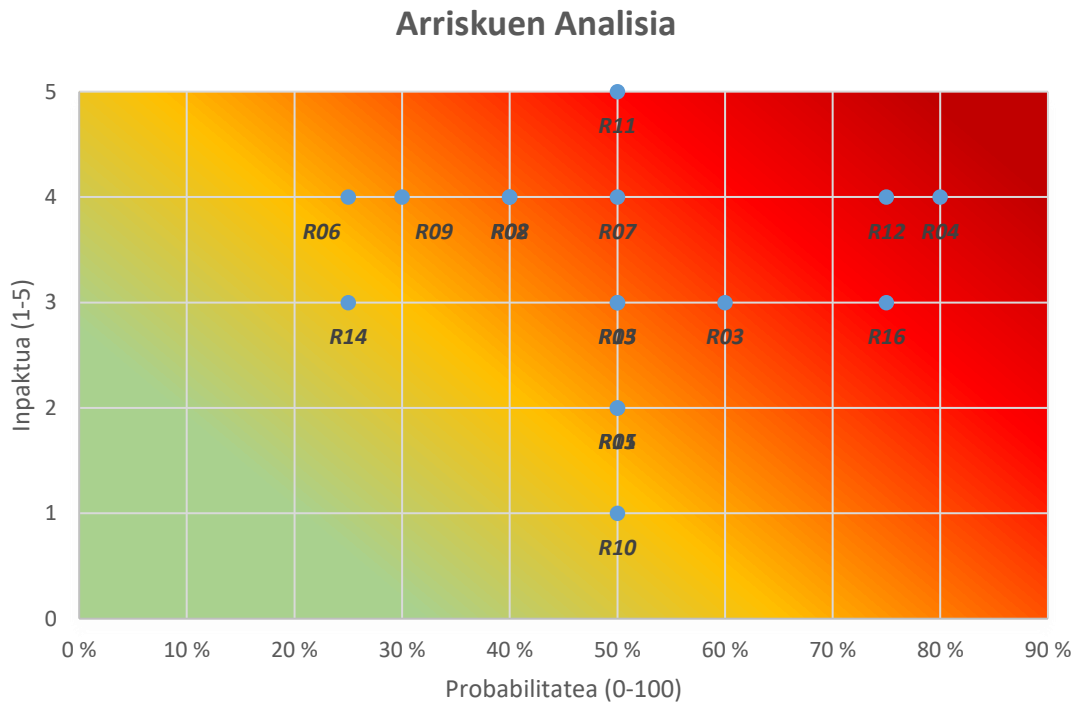
| | | | | | | |
|----|--|--|---|----|-----|--|
| | | | | | | beharrezkoa bada metodologia gehiago gehitu. |
| 07 | Metodologia konplexu bakarria aukeratzea | Metodologia konplexu bakarria aukeratzen badugu, lan handia hartuko dugu eta azpiegitura teknologikoa konplexua izango da. Gainera, gerta daiteke sinpleagoa behar izatea. | 4 | 50 | 2,0 | Metodologia garrantzitsu bat baino gehiago aukeratu, sinpleak eta konplexuak. |
| 08 | Metodologia sinple bakarria aukeratzea | Metodologia sinpleegia aukeratzen bada, azkar bukatuko dugu, baina seguruena aurrerago konplexuagoak beharko ditugu. | 4 | 40 | 1,6 | Metodologia garrantzitsu bat baino gehiago aukeratu, sinpleak eta konplexuak. |
| 09 | CMMI mailak azkarregi lortzen saiatzea | Hasieratik CMMI kalitate-ereduaren 2. edo 3. mailak lortzea handinahia izan daiteke eta arazoak ekar ditzake. | 4 | 30 | 1,2 | CMMI mailak lortzearen erabakia seguru gaudenean bakarrik hartu, metodologiekin esperientzia lortu denean. |
| 10 | Memoriaren eta eranskinen arteko koherentzia falta | CCII-2016N-02 eta OpenUP metodologia jarraitu denez, eranskinetan dokumentu asko daude eta koherentzi faltak agertu daitezke dokumentu guztien artean. | 1 | 50 | 0,5 | Eranskin guztiak berrikusi, batez ere, proiektuaren hasieran idatzitakoak. Memoriak eranskinekiko lehentasuna duela zehaztu. |
| 11 | Memoria idazteko denbora falta | Gerta daiteke memoria idazteko denbora nahikoa ez izatea, bukaerarako uzten bada. Oso garrantzitsua da memoria ondo idaztea. | 5 | 50 | 2,5 | Memoria osoa ez utzi bukerarako, pixkanaka kapitulu batzuk idazten joan nahiz eta proiektua bukatu gabe egon. |
| 12 | Proiektua amaitzeko denbora falta | Baliteke proiektua amaitzeko denboraz juxtu ibiltzea. Horrek kalitatea jaitea eragin dezake, lana presaka egiteagatik. | 4 | 75 | 3,0 | Lana modu egokian antolatu eta konstantea izaten saiatu. Denbora aprobetxatu eta gauza garrantzitsuenetan zentratu. Ez dago dena bukatu beharrik, etorkizuneko lan bezala utz daiteke. |
| 13 | Egindako lanaren galera | Egindako lana galtzeak lana berregin behar izatea eragin dezake. Honek denbora galtzeaz gain frustrazioa eragiten du. | 3 | 50 | 1,5 | Bertsio kontrola erabili proiektuaren informazio guztia gordetzeko. Datu guztien segurtasun kopiak egin egunero. |
| 14 | Ordenagailuarekin arazoak | Nire ordenagailuarekin arazoak izateak denbora galtzeak eragin ditzake. Ordenagailua konpondu bitartean lana egin ahal ez izatea eragin dezake. | 3 | 25 | 0,8 | Alternatiba moduan lanerako tutoreak emandako makina birtuala edukitzea. |

| | | | | | | |
|----|------------------------------------|---|---|----|-----|---|
| 15 | Makina birtualarekin arazoak | Makina birtualekin arazoak izateak lanerako ingurune hori eskuragarri ez izatea eragin dezake. Nire kasuan ordenagailu pertsonala erabili dudanez ez dauka eragin handirik. | 2 | 50 | 1,0 | Ordenagailu pertsonala erabili garapenerako ingurune nagusi moduan. Makina birtuala alternatiba moduan eduki arazoren bat badago erabiltzeko. |
| 16 | Proiektuaren planteamendu aldaketa | Proiektuan ziurgabetasun handia dagoenez, hurbilpen aldaketak egon daitezke. Honek atzerapenak eragin ditzake. | 3 | 75 | 2,3 | Proiektuaren osagaiak ahalik eta flexibleenak egiten saiatu. Horrela, hurbilpena aldatu arren egindako lana aprobetxatu daiteke. |

0.1. Taula. Identifikatutako arriskuen zehaztasunak.

IV.1.2 Arriskuen Analisia

Aurreko atalean arriskuen analisia egiteko arriskuen inpaktua eta probabilitatea bidertzen dira, eta magnitudea kalkulatzten da. Horrela, arrisku garrantzitsuenak zein diren argiago izango dugu uneoro. 12.1. Irudian, berriz, arriskuen inpaktu eta probabilitate grafika batean ikus daiteke. Grafika honek arriskuen ikuspegi orokorra ematen digu, arrisku bakoitzaren larritasuna erakutsiz. Adibidez, arrisku garrantzitsuenak eskuinean eta goian agertzen dira gorriz.



0.1. Irudia. Arriskuen inpaktu eta probabilitate analisia.

IV.2 Denbora Planifikazioa

Atal honen helburua da proiektuaren denbora planifikazioa azaltzea eta mugarri garrantzitsuak, atazen denborak, iterazioak eta desbiderapenak zehaztea da. Hasieran proiektua uztailean aurkezte aurreikusten zen baina azkenean irailean aurkeztu da. Horren ondorioz birplangintza egin behar izan da eta plangintzarekiko dituen desberdintasunak azalduko dira. Denboraren jarraipena egiteko Toggle Track tresna erabili da. Tresna honekin proiektuko denbora guztiak kontrolatu dira, tenporizadore bat erabiliz.

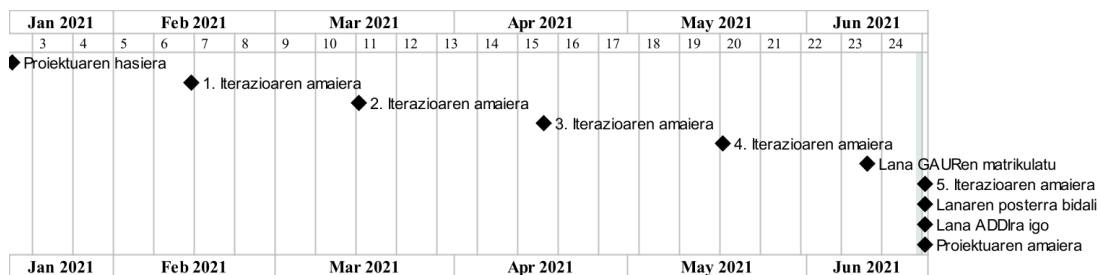
Azpiatazak iterazioka edo beharrezko momentuan sortu dira. Izan ere, oso zaila da aurreikustea ezjakintasun handia duen proiektu batean zein atazetan egingo den lan hemendik hilabete batzuetara. Ordu kantitatea kontuan hartuko da, baina oso zaila da ataza bakoitzaren denbora eta epeak estimatzea. Horregatik, iterazioen planak eguneratzen joatea garrantzitsua da.

IV.2.1 Mugarriak

Proiektua planifikatzeko garrantzitsua da mugarri garrantzitsuak biltzea eta momentu guztietan argi edukitzea. Mugarri gehienek iterazio amaierekin edo entregekin dute zerikusia. 0.1. Taulan proiektuaren plangintzako eta birplangintzako mugarri garrantzitsuenen datak azaltzen dira. Ikus daiteke birplangintzan iterazio gehigo daudela eta proiektuaren entrega data atzeratzen dela. 0.1. Irudian plangintzako mugarriak diagrama agertzen da eta 0.2. Irudian birplangintzakoa.

| Mugarria | Plangintza | Birplangintza |
|-------------------------|------------|---------------|
| Proiektuaren hasiera | 2021/01/14 | 2021/01/14 |
| 1. Iterazioaren amaiera | 2021/02/14 | 2021/02/14 |
| 2. Iterazioaren amaiera | 2021/03/15 | 2021/03/15 |
| 3. Iterazioaren amaiera | 2021/04/16 | 2021/04/16 |
| 4. Iterazioaren amaiera | 2021/05/17 | 2021/05/17 |
| 5. Iterazioaren amaiera | 2021/06/20 | 2021/06/20 |
| 6. Iterazioaren amaiera | - | 2021/07/23 |
| Lana GAURen matrikulatu | 2021/06/11 | 2021/07/23 |
| 7. Iterazioaren amaiera | - | 2021/09/05 |
| Lanaren posterra bidali | 2021/06/20 | 2021/09/05 |
| Lana ADDIra igo | 2021/06/20 | 2021/09/05 |
| Proiektuaren amaiera | 2021/06/20 | 2021/09/05 |

0.1. Taula. Plangintza eta birplangintzako mugarri garrantzitsuak.



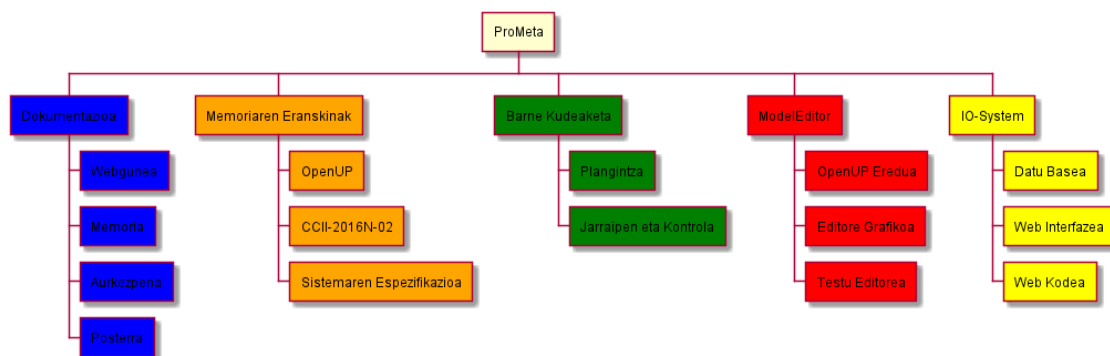
0.1. Irudia. Plangintzako mugarri garrantzitsuen diagrama.



0.2. Irudia. Birplangintzako mugarri garrantzitsuen diagrama.

IV.2.2 LDE Diagrama

LDE diagramaren bitartez lan-pakete eta ataza nagusiak modu argian adieraz daitezke. 0.3. Irudian ikus daitezke LDE diagrama.



0.3. Irudia. LDE diagrama.

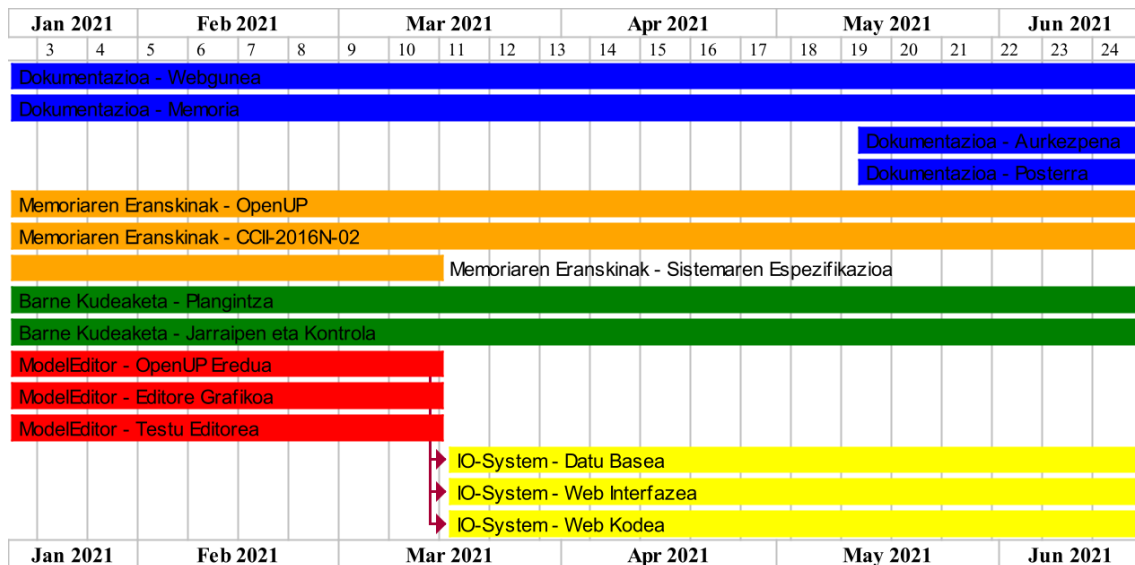
Hurrengo zerrendan LDE diagraman agertzen diren lan-pakete eta ataza nagusiak deskribatzen dira.

- **Dokumentazioa**
 - **Webgunea:** proiektuaren dokumentu guztiak biltzen dituen eta CCII-2016N-02 araua betetzen duen webgunea.
 - **Memoria:** proiektuaren memoriaren txantiloia prestatu eta memoria idatzi.
 - **Aurkezpena:** proiektuaren aurkezpena prestatu.
 - **Posterra:** proiektuaren posterra egin.
- **Memoriaren Eranskinak**
 - **OpenUP:** metodologiak zehazten dituen artefaktuak, hurrengo atalean zerrendatzen direnak.
 - **CCII-2016N-02:** arauak zehazten dituen memoriaren eranskinak.
 - **Sistemaren Espezifikazioa:** OpenUP metodologiak zehazten dituen artefaktuak.
- **Barne Kudeaketa**
 - **Plangintza:** OpenUP metodologiaren plangintza dokumentuak.
 - **Jarraipen eta Kontrola:** OpenUP metodologiaren jarraipen eta kontroleko artefaktuak.

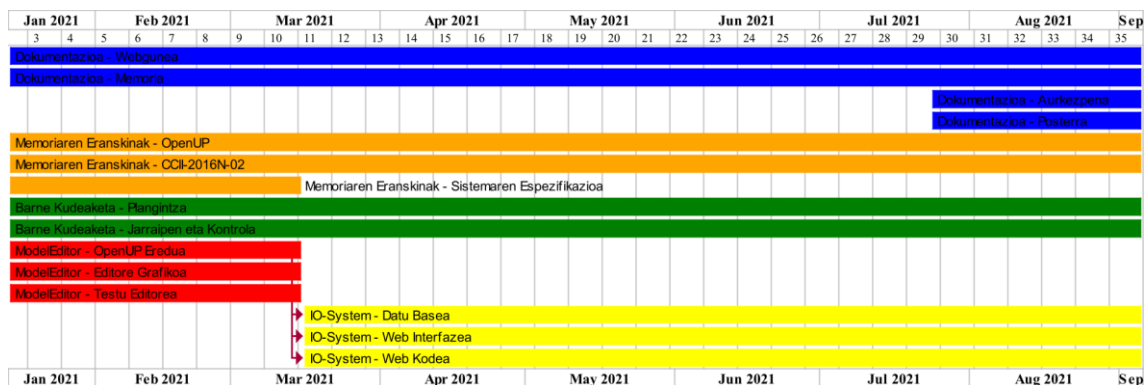
- **ModelEditor**
 - **OpenUP Eredua:** OpenUP metodologiaren informazio guztia duen eredua.
 - **Editore Grafikoa:** metodologiaren ereduak grafikoki aldatzeko editorea.
 - **Testu Editorea:** metodologiaren ereduak testu bidez aldatzeko editorea.
- **IO-System**
 - **Datu Basea:** metodologiaren informazioa gordetzen duen datu-basea sortu eredutik.
 - **Web Interfazea:** webgunearen interfazea aldatu.
 - **Web Kodea:** webgunearen kodea programatu.

IV.2.3 Gantt Diagrama

Aurreko atalean definitutako paketeak eta atazak kontuan hartuz, Gantt diagramaren bitartez atazen garapen denborak zehaztu daitezke. Atazen garapen denbora zehatza estimatzea oso zaila denez, iterazio bakoitzean egingo diren atazak zehaztu dira. Beraz, atazen data iterazioen hasiera eta bukaera datekin kointziditzen dute. Kasu honetan ere birplangintzan aldaketak egon direnez, 0.4. Irudian agertzen den plangintzako diagrama eta 0.5. Irudian ikusten dena desberdinak dira.



0.4. Irudia. Plangintzako Gantt diagrama.



0.5. Irudia. Birplangintzako Gantt diagrama.

IV.2.4 Iterazioak

Atal honetan, OpenUP metodologiaren bitartez jaso diren atazak iterazioetan zehar nolako banaketa izango duten deskribatuko da. Lan guztia iteraziotan banatu da, bakoitzak hilabete inguruko iraupena izanik. Esan bezala, plangintzan proiektuaren iraupena laburragoa izatea aurreikusten zen. Ondorioz, iterazio kopurua 5etik 7ra handitu da birplangintza egin denean. 0.2. Taulan iterazio bakoitzaren fasea, helburuak eta datak agertzen dira. Lehenengo bi iterazioak Hasiera fasekoak dira eta beste bostak Elaborazio fasekoak.

Bezeroak proiektuari jarraipena ematea erabakiko balu, Eraikuntza eta Trantsizio faseak egin beharko lituzke. OpenUP metodologian gutxi gorabehera hauek dira fase bakoitzerako aurreikusitako iterazio kopuruak: Hasiera fasean 1, Elaborazio fasean 2, Eraikuntza fasean 3 eta Trantsizio fasean 1. Hasiera eta Elaborazio faseetan behar izan dugun denbora kontuan hartuta, aurreikusi daiteke gutxienez Eraikuntza faserako 6 iterazio beharko direla eta Trantsizio faserako 2.

| Fasea | Iterazioa | Helburuak | Hasiera | Amaiera |
|---------|-----------|--|------------|------------|
| Hasiera | I1 | <p>Iterazio Plana.</p> <p>OpenUp Hasiera faseko artefaktuak.</p> <p>Aurreko proiektuen informazioa ulertu.</p> <p>OpenUp txantiloia eskuratu.</p> <p>Memoriaren txantiloia sortu gradu amaierako lanaren eredua eta CCII araua kontuan hartuz.</p> <p>CCII arauaren informazioa jaso.</p> <p>CCII araua betetzen duen webgunea eraiki.</p> <p>OpenUp metodologiaren informazioa bildu.</p> <p>Lan egiteko tresnak eskuratu (Eclipse, EPF Composer, EHSIS, Drupal, etab.).</p> <p>ProMeta proiektuak jarraituko duen garapen prozesuaren OpenUP eredua definitu.</p> <p>Metaereduan oinarritutako eredu editore grafiko eta testu editorearen prototipoa.</p> | 2021/01/14 | 2021/02/14 |
| Hasiera | I2 | <p>Iterazio Plana.</p> <p>OpenUp Hasiera faseko artefaktuak.</p> <p>CCII araua betetzen duen webgunea eraiki.</p> <p>ProMeta proiektuak jarraituko duen garapen prozesuaren OpenUp eredua definitu.</p> <p>Metaereduan oinarritutako eredu editore grafiko eta testu editorearen prototipoa.</p> <p>Editore grafiko eta testu editorearen arteko sinkronizazioa. Eredu bera bi modutan editatzeko aukera.</p> <p>Drupal webguneak erabiliko duen datu-basea sortu eredutik.</p> | 2021/02/15 | 2021/03/15 |

| | | | | |
|-------------|----|--|------------|------------|
| Elaborazioa | I3 | Iterazio Plana.

Drupal webguneak erabiliko duen datu-basea sortu eredutik.

OpenUp Elaborazio faseko artefaktuak.

Web interfazea eraiki (Drupal).

Web kodea garatu (Drupal).

Webgunea.

Memoria. | 2021/03/16 | 2021/04/16 |
| Elaborazioa | I4 | Iterazio Plana.

OpenUp Elaborazio faseko artefaktuak.

Web interfazea eraiki (Drupal).

Web kodea garatu (Drupal).

Webgunea.

Memoria. | 2021/04/17 | 2021/05/17 |
| Elaborazioa | I5 | Iterazio Plana.

OpenUp Elaborazio faseko artefaktuak.

Web interfazea eraiki (Drupal).

Web kodea garatu (Drupal).

Webgunea.

Memoria. | 2021/05/18 | 2021/06/20 |
| Elaborazioa | I6 | Iterazio Plana.

OpenUp Elaborazio faseko artefaktuak.

Web interfazea eraiki (Drupal).

Web kodea garatu (Drupal).

Webgunea.

Memoria. | 2021/06/21 | 2021/07/23 |
| Elaborazioa | I7 | Iterazio Plana.

OpenUp Elaborazio faseko artefaktuak.

Web interfazea eraiki (Drupal).

Web kodea garatu (Drupal).

Webgunea.

Memoria.

Posterra.

Aurkezpena. | 2021/07/24 | 2021/09/05 |

0.2. Taula. Proiektuko iterazioen fasea, helburuak eta datak.

IV.2.5 Atazen Denborak

Lana modu egokian antolatu eta kontrolatzeko, ezinbestekoa da lana ataza eta azpiatazetan banatzea. Hala ere, ez da gomendagarria azpiataza gehiegi definitzea, kontrolatzeko eta ulertzeko zailagoa baita. Ataza hauek ez dira estatikoak, proiektua aurrera joan ahala berriak ager daitezke.

0.3. Taulak proiektuko lan-ataza guztien zehaztasunak adierazten ditu: lehentasuna, tamaina, iterazioak, estimatutako orduak eta benetako ordua. Ataza batzuetan desbiderapen handiak egon dira estimatutako eta benetako orduen artean. Hala ere, orokorrean orduen estimazioa nahiko zehatza izan da. Guztira 415 ordu estimatu ziren eta benetan 430 inguru egin dira. Hala ere, benetako orduei oraindik aurkezpenaren denbora gehitu behar zaienez, 450 ingurura iritsiko da.

| Izena | Lehentasuna | Tamaina | Iterazioak | Estimatutako orduak | Benetako orduak |
|---|-------------|---------|------------|---------------------|-----------------|
| Dokumentazioa - Webgunea | 1 | 3 | I1-I7 | 20 | 18 |
| Dokumentazioa - Posterra | 2 | 2 | I5 | 5 | 5 |
| Dokumentazioa - Aurkezpena | 3 | 3 | I5 | 20 | - |
| Dokumentazioa - Memoria | 3 | 5 | I1-I7 | 100 | 97 |
| Memoriaren Eranskinak - OpenUP | 1 | 3 | I1-I7 | 10 | 9 |
| Memoriaren Eranskinak - CCII-2016N-02 | 1 | 3 | I1-I7 | 10 | 4 |
| Memoriaren Eranskinak - Sistemaren Espezifikazioa | 1 | 2 | I1-I3 | 20 | 14 |
| Barne Kudeaketa - Plangintza | 3 | 2 | I1 | 20 | 22 |
| Barne Kudeaketa - Jarraipen eta Kontrola | 1 | 3 | I1-I7 | 30 | 48 |
| ModelEditor - OpenUP Eredua | 1 | 5 | I1-I2 | 20 | 49 |
| ModelEditor - Editore Grafikoa | 1 | 5 | I1-I2 | 25 | 26 |
| ModelEditor - Testu Editorea | 1 | 5 | I1-I2 | 30 | 31 |
| IOSystem - Datu Basea | 1 | 3 | I3-I7 | 40 | 38 |
| IOSystem - Web Interfazea | 1 | 5 | I3-I7 | 35 | 43 |
| IOSystem - Web Kodea | 1 | 5 | I3-I7 | 30 | 23 |

0.3. Taula. Lan-atazen zehaztasunak.

IV.2.6 Denboraren Neurketa

Toggl Track aplikazioa erabiliz neurtu dira proiektuko denborak. Ataza bakoitzaren denbora neurtu denez, ataza bakoitzeko denbora jakiteko ez daukagu kalkulurik egin beharrik, aplikazioak zuzenean esaten digu. Gainera, nahi dugun denborak bistaratu ditzakegu, adibidez hilabete bakoitzeko denbora edo egun bakoitzekoa.

Guztira hilabete bakoitzean pasatako denbora ikus dezakegu 0.6. Irudian. Ikus daiteke lehenengo hilabeteetan denbora hasten doala, otsailean maximora iristen dela eta hortik aurrera jaisten doala. Hau proiektu baten ohiko bilakaera izaten da.

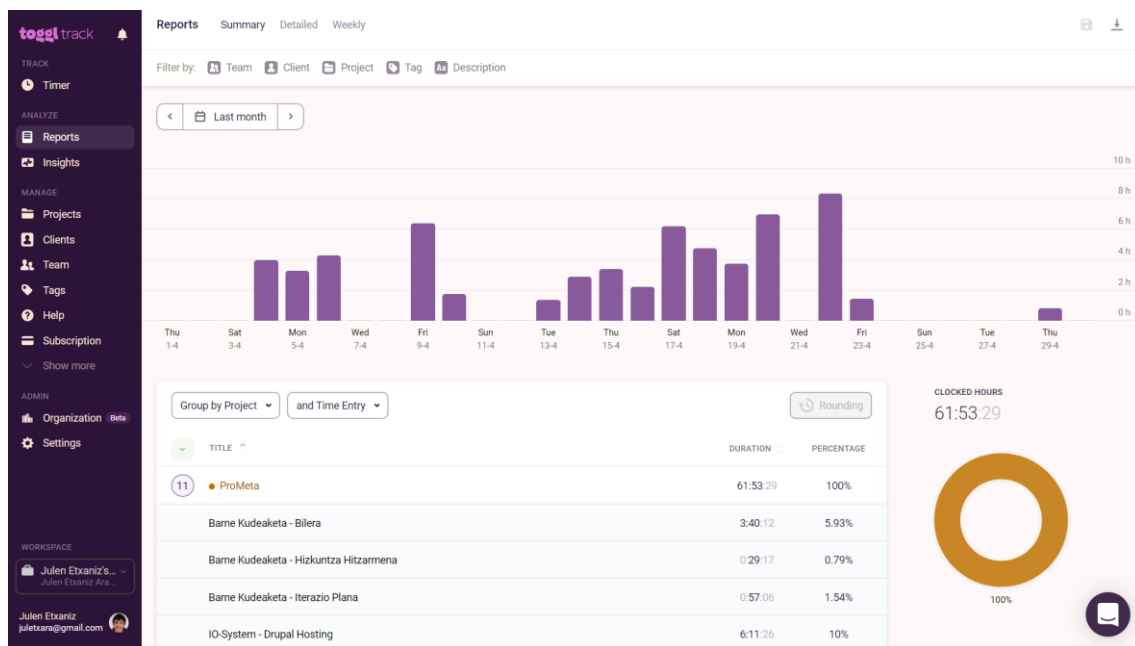
29-10-2020 – 05-09-2021

TOTAL HOURS: 415:32:00



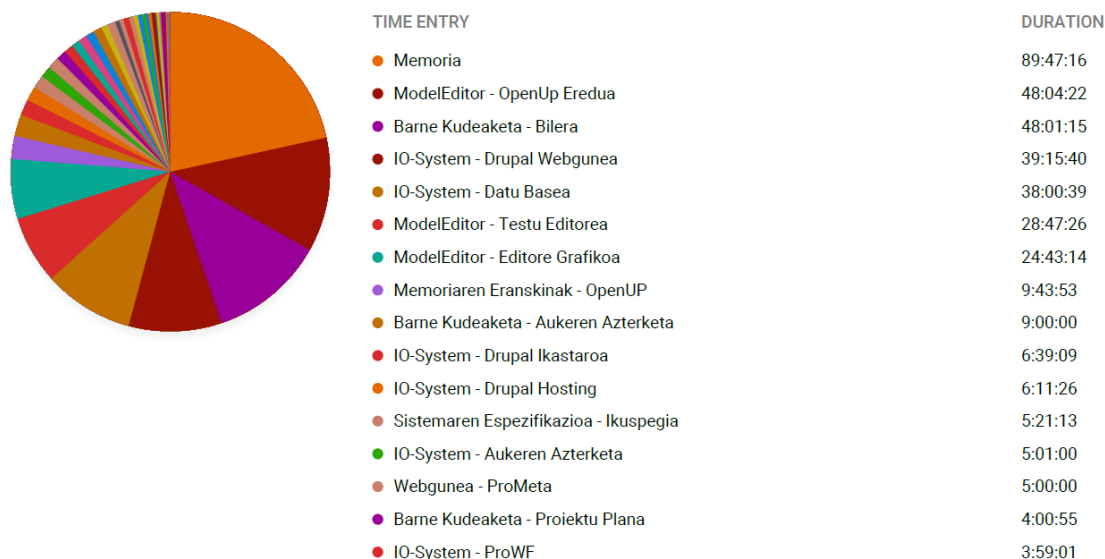
0.6. Irudia. Proiektuaren hilabete bakoitzeko denbora.

Hilabete bateko egun bakoitzeko denbora ikus dezakegu, adibidez 0.7. Irudian apirileko egunak agertzen dira. Gainera, azpian ataza bakoitzari eskainitako denbora eta denbora totala ikus ditzakegu. Hau baliagarria izan daiteke hilabete baten inguruko ikuspegi orokorra eduki nahi badugu.



0.7. Irudia. Apirileko egun bakoitzeko denbora.

Ataza garrantzitsuenei guztira eskainitako denbora ikus dezakegu 0.8. Irudian. Denbora gehien eskainitako atazak bakarrik agertzen dira irudi horretan. Kontuan hartu behar da kasu batzuetan azpiatazak definitu direla, eta orduan atazaren denbora hainbat azpiatazatan banatuta agertzen dela.



0.8. Irudia. Proiektuaren denbora atazaka.

IV.2.7 Desbiderapenak

Proiektuan zehar hainbat desbiderapen egon dira plangintzarekiko. Esan bezala, desbiderapen garrantzitsuena proiektuaren iraupenean egon dena izan da. Izan ere, horrekin erlazionatuta daude beste desbiderapen gehienak. Proiektuaren iraupena luzatu denez, hasieran planifikatutakoak baina bi iterazio gehiago behar izan dira.

Atazen denbora estimazio batzuetan ere desbiderapen handiak egon dira. Hala ere, proiektuarentzat estimatutako denbora totala nahiko gertu ibili da. Aurretik esan den bezala, 415 ordu estimatu ziren eta azkenean 450 inguru behar izan dira. Lan-pakete bakoitzean zentratzen bagara, honakoa ikus dezakegu:

- Dokumentazioa lan-paketeko benetako denborak estimatutakoetatik oso gertu ibili dira, 125 estimatu ziren eta 120 behar izan dira. Memorian, webgunean eta posterrean gutxi gorabehera aurreikusitako denbora behar izan da.
- Memoriaren Eranskinak lan-paketeko atazetan estimatutako denbora baina gutxiago behar izan da orokorrean, 27 ordu estimatutako 40 orduren orde. Honen arrazoiak izan daitezke memoriako hainbat atal berrerabiltzea eta eranskin batzuk bete ez izana.
- Barne Kudeaketa lan-paketean estimatutakoa baino askoz denbora gehiago behar izan da, 70 ordu 50en orde. Izan ere, Jarraipen eta Kontrola atazan bilerak egiten pentsatutakoa baina denbora gehiago behar izan da.
- ModelEditor lan-paketean estimatutakoa baino denbora gehiago behar izan da, 105 ordu 75 orduren orde. Izan ere, OpenUP eredua atazarekin askoz denbora gehiago behar izan da.
- IOSystem lan-paketean gutxi gorabehera estimatutako denbora behar izan da, 104 ordu 105en orde. Web interfazearekin denbora gehiago behar izan da, baina alde hori web kodearekin konpentsatu da.