

## 11 Proposatutako Sistemaren Deskribapena

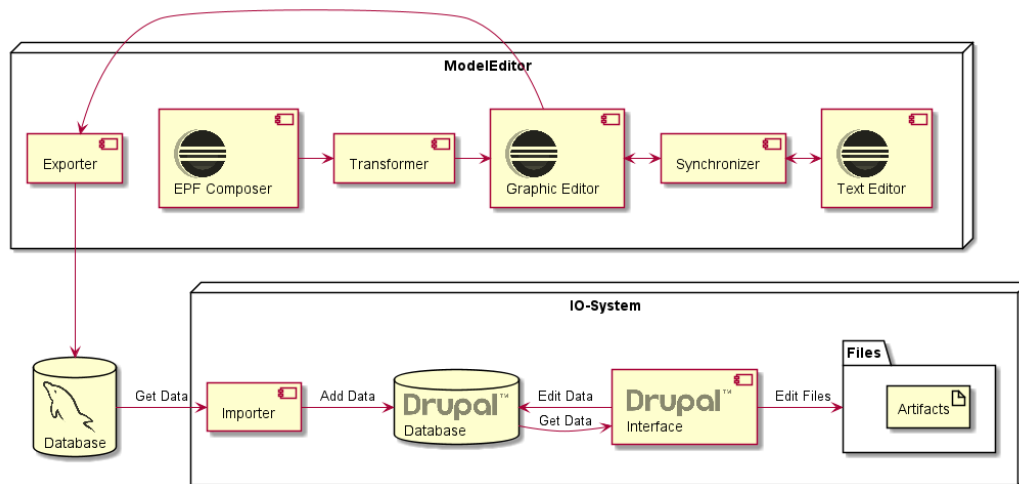
Kapitulu honetan planteatutako arazoa konpontzeko proposatzen den sistema, bere osagaiak eta bere ezaugarriak deskribatzen dira.

### 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 rolak 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

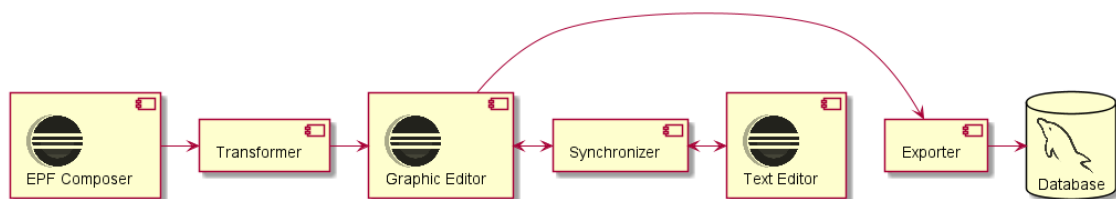
Sortutako editore grafikoa eta testu editorea erabiliz prozesuaren eredua editatzeko aukera emango du. Sistema honen ardura prozesu ingeniari rolak izango du.

#### 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.

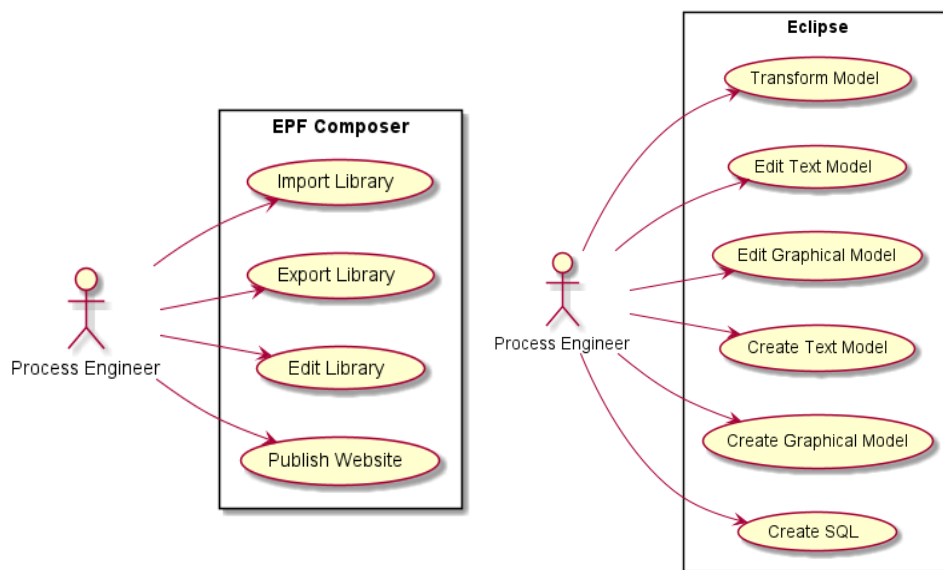
- **Formatu Aldatzailea:** Metodologiaren formatua aldatzen du XMItik UMara editore grafikoak erabili ahal izan dezan.
- **Editore Grafikoa:** Metodologiak modu grafikoan editatzeko aukera ematen du, metodologiaren jatorrizko ikonoak erabiliz.
- **Testu Editorea:** Metodologiak testu bidez editatzeko aukera ematen du, metaeredutik sortutako gramatika erabiliz.
- **Editore Sinkronizatzaila:** 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

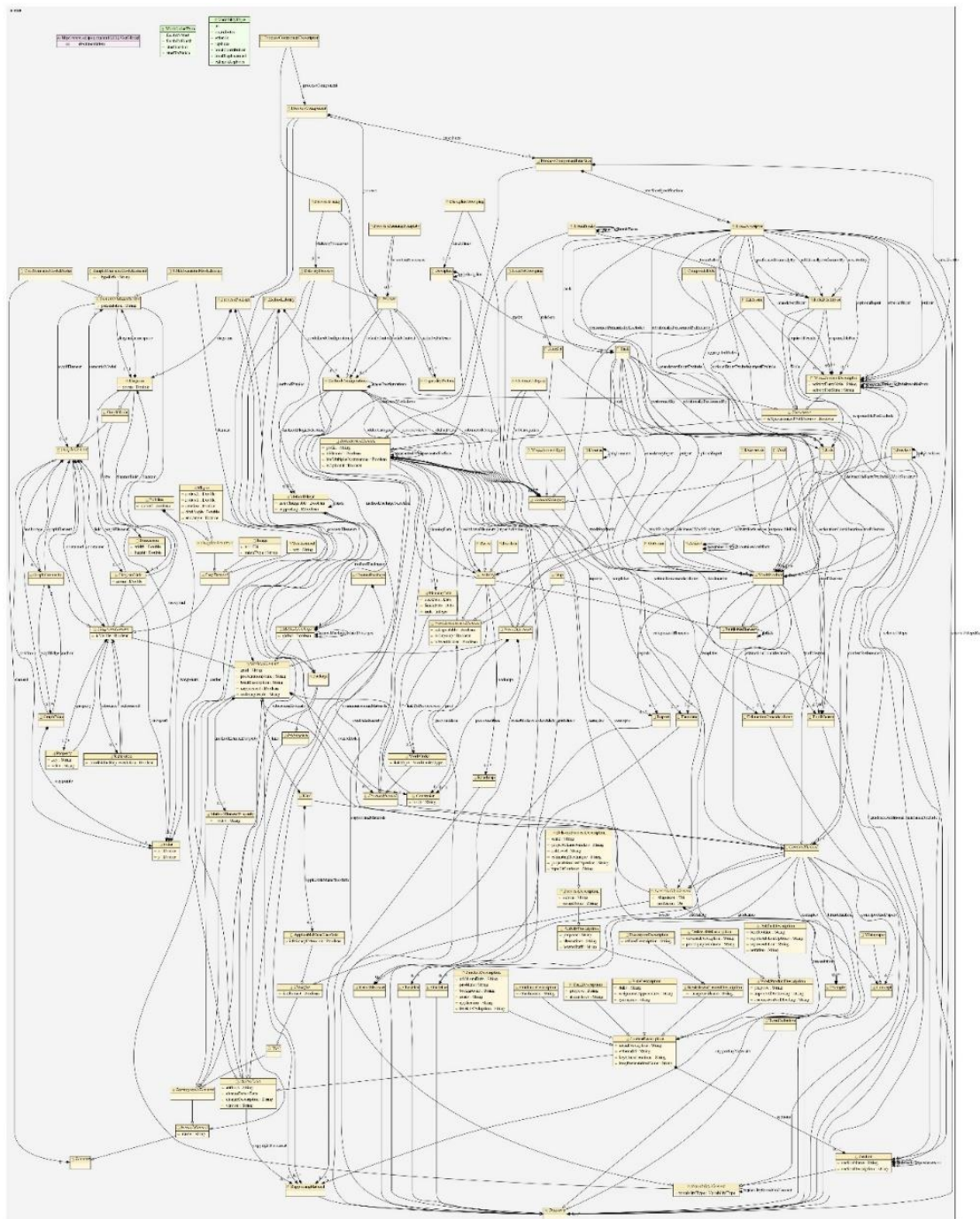
Sistemaren betekizunak zehazteko beharrezko da erabilpen kasuak eta aktoreak zein diren jakitea. Azpisistemaren erabilpen kasuak bi multzotan bana daitezke: EPF Composer-enak eta Eclipse-renak. EPF Composer-etik metodologiaren liburutegiak inportatu, esportatu eta aldatzeko aukera daukagu. Gainera, metodologiaren 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.

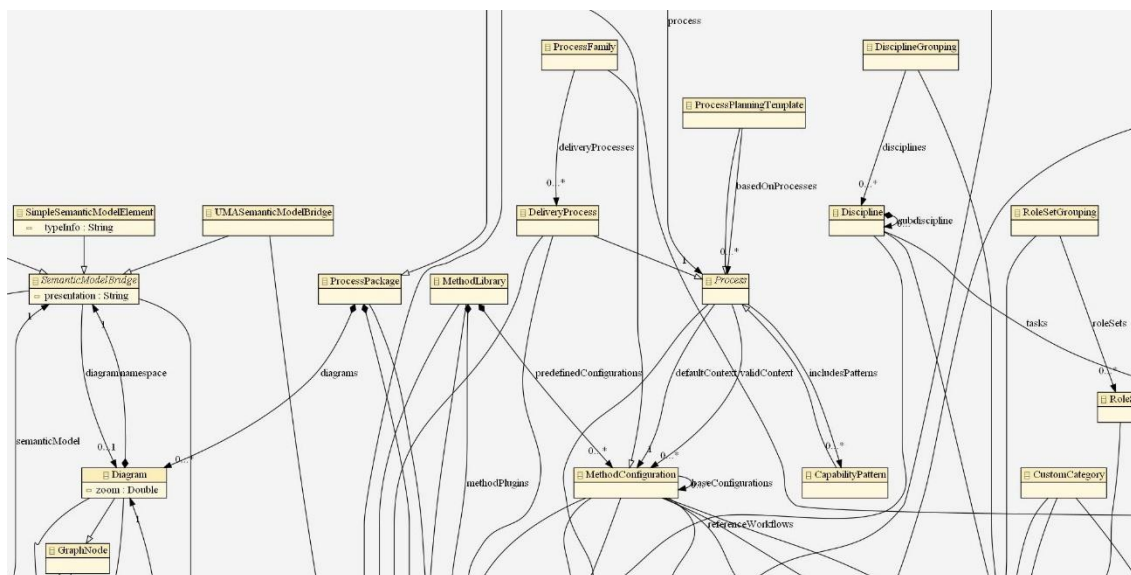
### 11.2.3 Diseinua

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... | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Inception Phase                             | 1     |                 |  | Phase       | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Inception Iteration [1..n]                  | 2     |                 | extends 'inception_phase_iteration, pro...   | Iteration   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initiate Project                            | 3     |                 | extends 'initiate_project, process.openu...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Plan and Manage Iteration                   | 5     |                 | extends 'plan_manage_iteration, process...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Identify and Refine Requirements            | 15    | 3               | extends 'identify_and_refine_requirement...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Agree on Technical Approach                 | 20    | 3               | extends 'agree_technical_approach, proc...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Lifecycle Objectives Milestone              | 22    | 2               |  | Milestone   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Elaboration Phase                           | 23    | 22,1            |  | Phase       | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Elaboration Iteration [1..n]                | 24    |                 | extends 'elaboration_phase_iteration, p...   | Iteration   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Plan and Manage Iteration                   | 25    |                 | extends 'plan_manage_iteration, process...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Identify and Refine Requirements            | 35    |                 | extends 'identify_and_refine_requirement...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Develop the Architecture                    | 40    |                 | extends 'develop_architecture, process.op... | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Develop Solution Increment                  | 48    |                 | extends 'develop_solution, process.openu...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Test Solution                               | 54    |                 | extends 'test_solution, process.openup.ba... | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Ongoing Tasks                               | 57    |                 | extends 'ongoing_tasks, process.openup.b...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Lifecycle Architecture Milestone            | 59    | 24              |  | Milestone   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Construction Phase                          | 60    | 59,23           |  | Phase       | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Construction Iteration [1..n]               | 61    |                 | extends 'construction_phase_iteration...     | Iteration   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Plan and Manage Iteration                   | 62    |                 | extends 'plan_manage_iteration, process...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Identify and Refine Requirements            | 72    |                 | extends 'identify_and_refine_requirement...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Develop Solution Increment                  | 77    |                 | extends 'develop_solution, process.openu...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Test Solution                               | 83    |                 | extends 'test_solution, process.openup.ba... | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Ongoing Tasks                               | 86    |                 | extends 'ongoing_tasks, process.openup.b...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Develop Product Documentation and Training  | 89    |                 | extends 'prep_dsc_tmng, process.openup.b...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial Operational Capability Milestone    | 93    | 61              |  | Milestone   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Transition Phase                            | 94    | 93,60           |  | Phase       | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Transition Iteration [1..n]                 | 95    |                 | extends 'transition_phase_iteration, pro...  | Iteration   | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Plan and Manage Iteration                   | 96    |                 | extends 'plan_manage_iteration, process...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Develop Solution Increment                  | 105   |                 | extends 'develop_solution, process.openu...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Test Solution                               | 111   |                 | extends 'test_solution, process.openup.ba... | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Finalize Product Documentation and Training | 114   |                 | extends 'prep_dsc_tmng, process.openup.b...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Prepare for Release                         | 119   |                 | extends 'prepare_for_release, process.op...  | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Provide Product Training                    | 124   | 105,111,96,1... | extends 'provide_product_training, proc...   | Activity    | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

Description Work Breakdown Structure Team Allocation Work Product Usage Consolidated View

Properties Problems

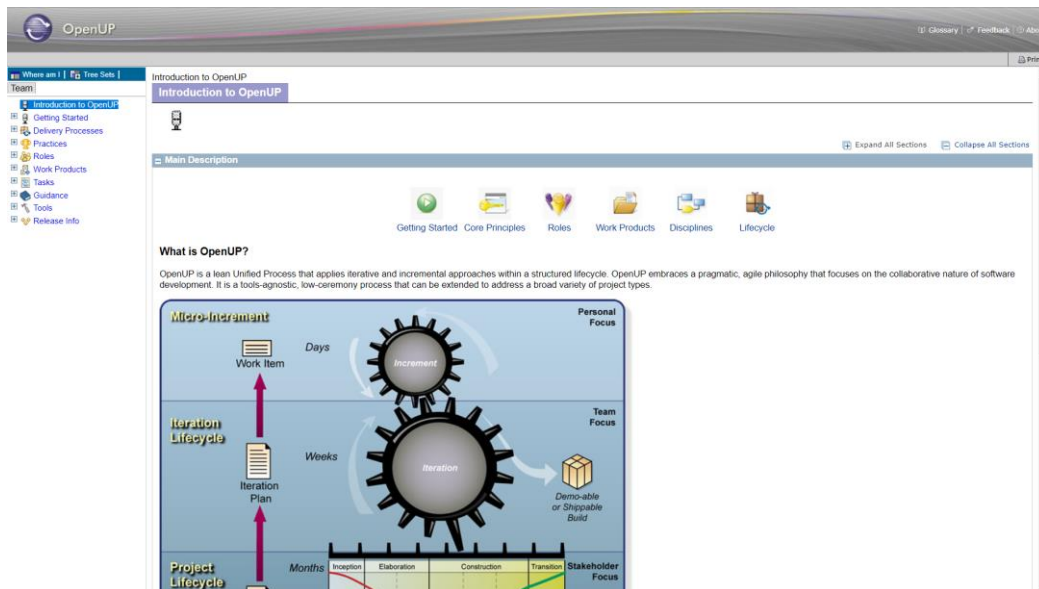
Property Value

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.



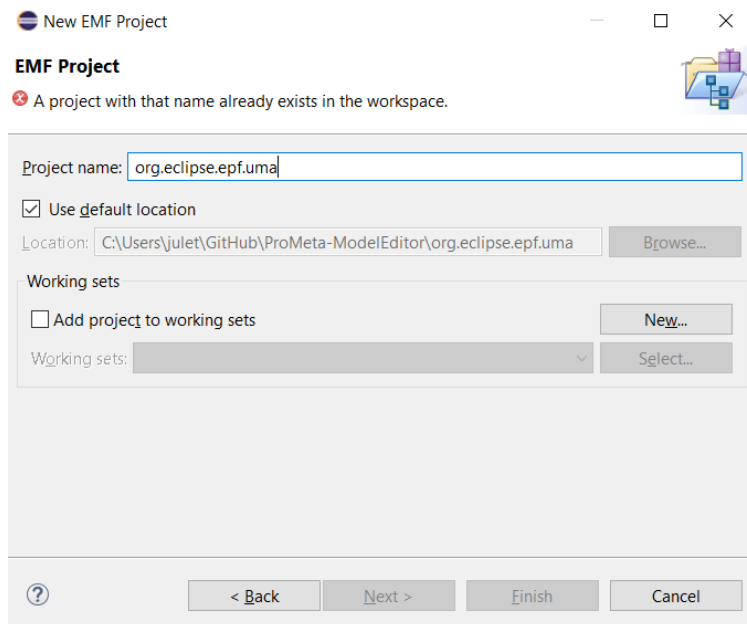
11.7. Irudia. OpenUP metodologiaren webgunea.

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 eredu hainbat eraldaketa egitea da. Izan ere, dagoen bezala uzten badugu, eredu 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 exekutatzeko 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>
```

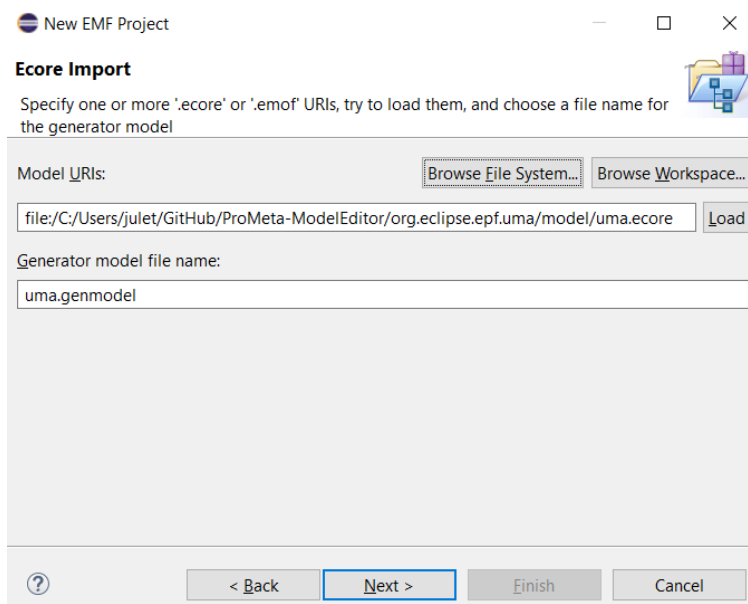
11.8. Irudia. Eraldaketa egiteko XSLT fitxategia.

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.



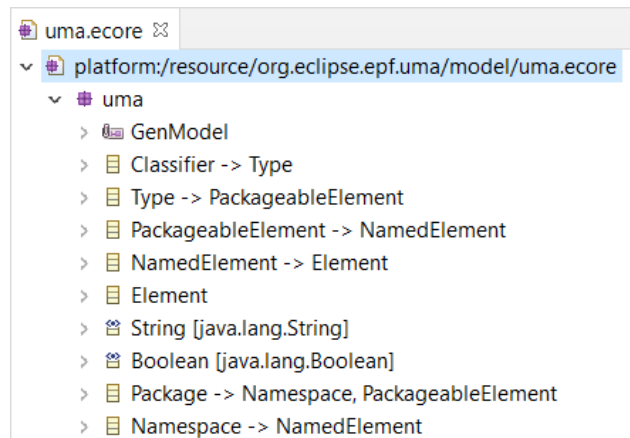
11.9. Irudia. EMF proiektuaren sorrera Eclipsen.

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.



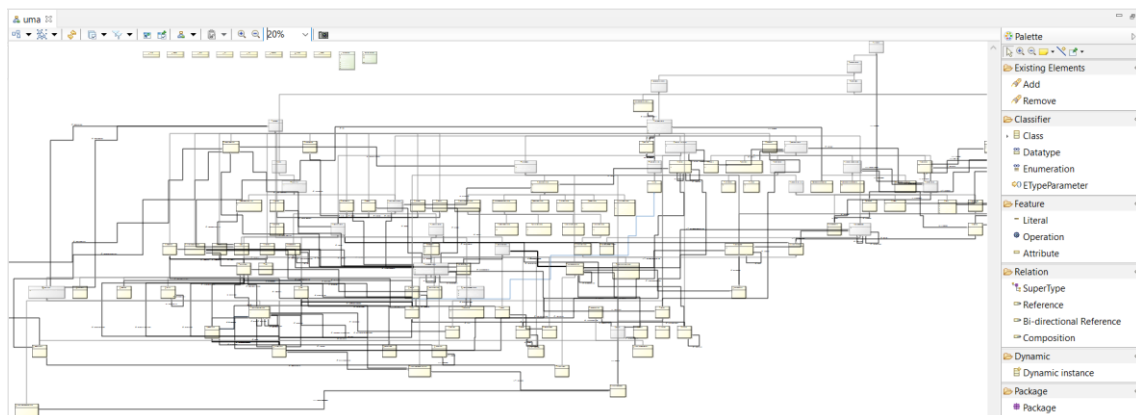
11.10. Irudia. Uma metaeredua inportatzen Eclipsen.

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.



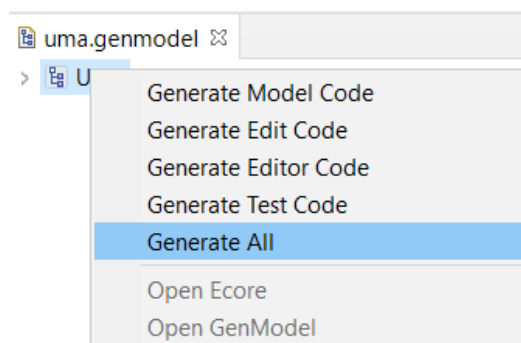
11.11. Irudia. Uma.ecore fitxategia Eclipsen.

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.

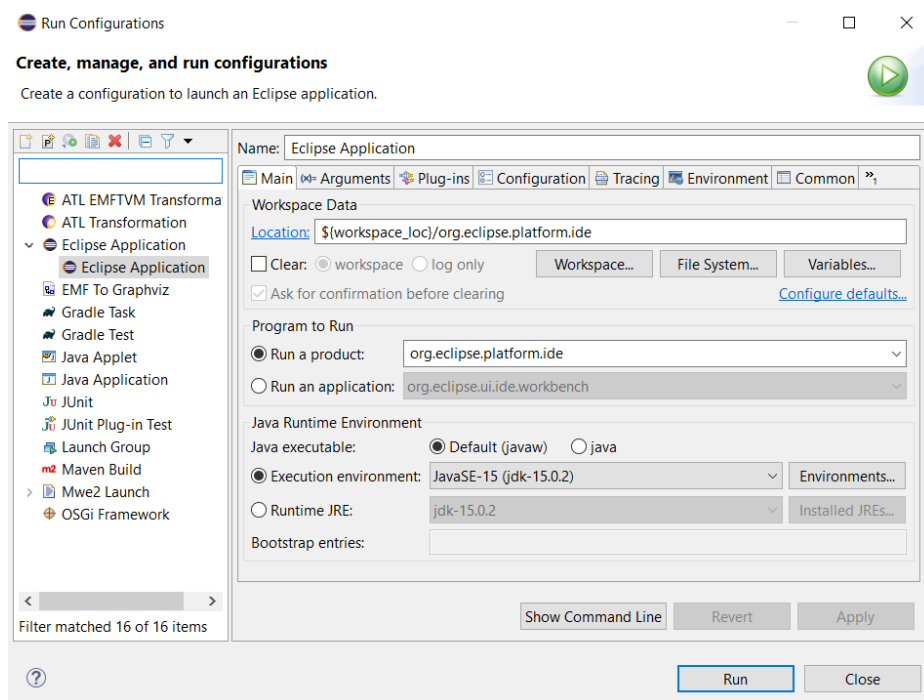
Behin metaereduaren proiektua prest daukagula, editore grafikoa sor dezakegu. Horretarako, `uma.genmodel` fitxategia ireki beharko dugu. Kodea sortzeko hainbat aukera daude, baina sinpleena *Generate All* aukeratzeko da. 11.13. Irudian ikusten den moduan. Izan ere, lehenengo 3 aukerak beharrezkoak dira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



11.13. Irudia. Editore grafikoaren kodearen sorrera.

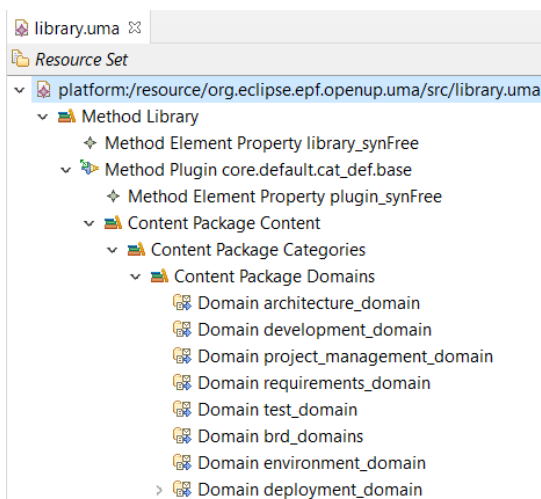
Lehenengo aukerak proiektuan bertan sortuko du ereduaren Java kodea. Beste aukera bakoitzerako Java proiektu berri bat sortuko da automatikoki. Honekin jadanik editore grafikoa daukagu, baina itxura hobea edukitzea nahi badugu ikonoak gehitu beharko ditugu `org.eclipse.uma.edit` proiektuan.

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.

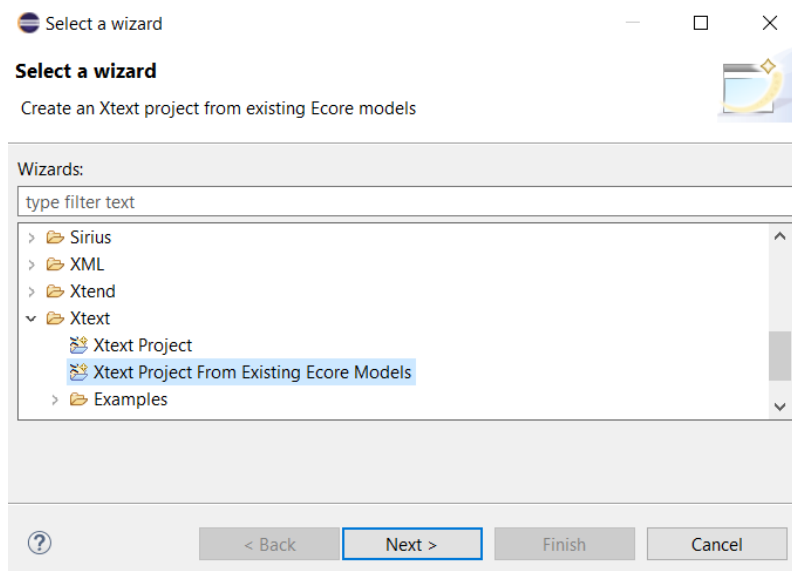
Martxan jartzen dugunean Eclipse aplikazio berri bat irekiko da. Bertan proiektu berri bat sortu beharko dugu eta XSLT eraldaketatik lortutako `.uma` fitxategiak kargatu. Edozein `.uma` fitxategi irekitzeko aukera izango dugu *Uma Model Editor* erabiliz. Editore honek EPF Composer-en antza du ikonoak 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.

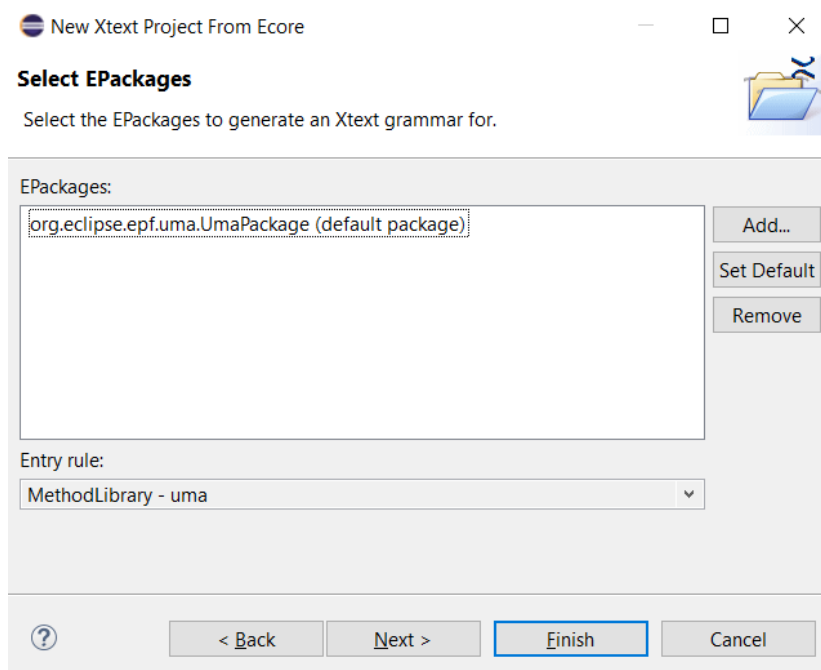


Editore grafikoa bukatuta dagoenez, testu editorearekin has gaitezke. Testu editorea sortzeko Xtext tresna erabiliko dugu. Helburua DSL bat sortzea denez, gramatika bat definitu beharko dugu. Metaeredua erabil dezakegu gramatika automatikoki sortzeko, 11.16. Irudian agertzen den *Xtext Project From Existing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



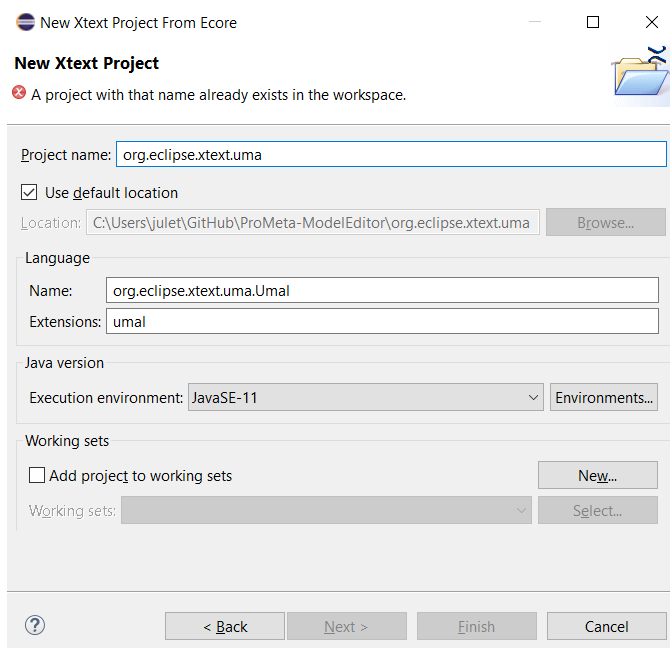
11.16. Irudia. Xtext proiektua sortzen Eclipsen.

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 aukeratzeko badugu, bi editoreak batera erabili 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 ereduak modu bisualean ikustea ahalbidetzen du eta testu editoreak, berriz, aldaketa errepikakorrak azkarrago egitea.

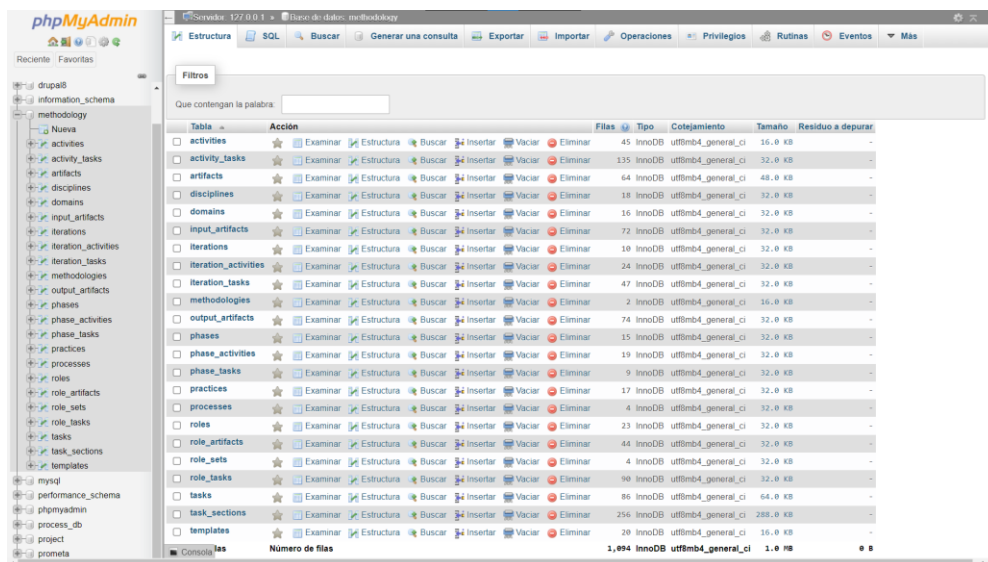
Beraz, bi itzultzaile behar ditugu, uma-tik umal-era itzultzen duena eta umal-etik uma-ra itzultzen duena. Bigarren itzulketa nahiko sinplea da, kode sorreraren antzekoa baita. Izan ere, Xtext-ek itzulketa hau errazten du serializatzaileak definituz Xtend programazio lengoia erabiliz. Lehenengoa zailagoa 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.



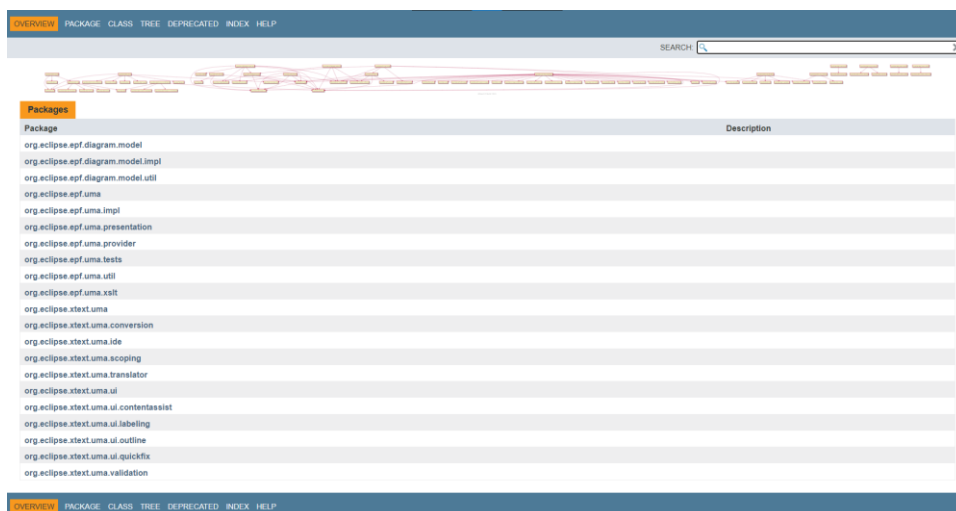
| Tabla                | Acción  | Filas        | Tipo          | Cotejamiento              | Tamaño        | Residuo a depurar |
|----------------------|---|--------------|---------------|---------------------------|---------------|-------------------|
| activities           | Examinar Estructura Buscar Insertar Vaciar Eliminar | 45           | InnoDB        | utf8mb4_general_ci        | 16.0 KB       | -                 |
| activity_tasks       | Examinar Estructura Buscar Insertar Vaciar Eliminar | 135          | InnoDB        | utf8mb4_general_ci        | 32.0 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 | 10           | 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 | 20           | InnoDB        | utf8mb4_general_ci        | 16.0 KB       | -                 |
| <b>Consultas</b>     | <b>Número de filas</b>                              | <b>1,894</b> | <b>InnoDB</b> | <b>utf8mb4_general_ci</b> | <b>1.0 MB</b> | <b>0 B</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 azpisisistema 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.

OVERVIEW

PACKAGE

CLASS

TREE

DEPRECATED

INDEX

HELP

SEARCH

Search

Package org.eclipse.epf.diagram.model.util

org.eclipse.epf.common.notify.impl

AdapterFactoryImpl

org.eclipse.epf.diagram.model.util

ModelSwitch

ModelAdapterFactory

UMLDoclet 2.9.12, PlantUML 1.2020.16

Class Summary

| Class               | Description                                       |
|---------------------|---|
| ModelAdapterFactory | The Adapter Factory for the model.                |
| ModelSwitch         | The Switch for the model's inheritance hierarchy. |

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.

OVERVIEW

PACKAGE

CLASS

TREE

DEPRECATED

INDEX

HELP

SUMMARY

NESTED

FIELD

CONSTR

METHOD

DETAIL

FIELD

CONSTR

METHOD

SEARCH

Search

Package org.eclipse.epf.diagram.model.util

Class ModelSwitch

java.lang.Object

org.eclipse.epf.diagram.model.util.ModelSwitch

```

public class ModelSwitch
extends java.lang.Object

```

The Switch for the model's inheritance hierarchy. It supports the call `doSwitch(object)` to invoke the `caseXXX` method for each class of the model, starting with the actual class of the object and proceeding up the inheritance hierarchy until a non-null result is returned, which is the result of the switch.

See Also:

ModelPackage

Constructor Summary

| Constructor                | Description                        |
|----------------------------|------------------------------------|
| <code>ModelSwitch()</code> | Creates an instance of the switch. |

Method Summary

All Methods

Instance Methods

Concrete Methods

| Modifier and Type             | Method   | Description  |
|-------------------------------|--|--|
| <code>java.lang.Object</code> | <code>caseActivityDetailDiagram(ActivityDetailDiagram object)</code> | Returns the result of interpreting the object as an instance of 'Activity Detail Diagram'. |
| <code>java.lang.Object</code> | <code>caseActivityDiagram(ActivityDiagram object)</code>             | Returns the result of interpreting the object as an instance of 'Activity Diagram'.        |

UMLDoclet 2.9.12, PlantUML 1.2020.16

11.26. Irudia. Klase diagrama adibidea.

### 11.3 IO-System

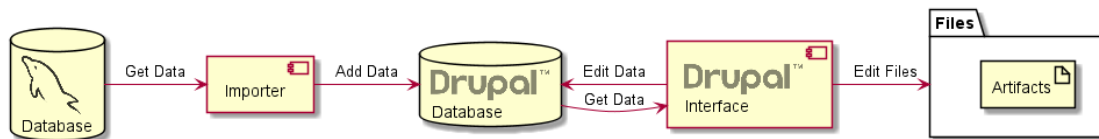
Drupal CMSaren bitartez kudeatutako web-aplikazioa izango da. Helburua metodologia jarraitzen duten proiektuen informazioa gordetzea da. Rol bakoitzak metodologian dituen ataza berdinak bete beharko ditu.



### 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 gordetzeko 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.

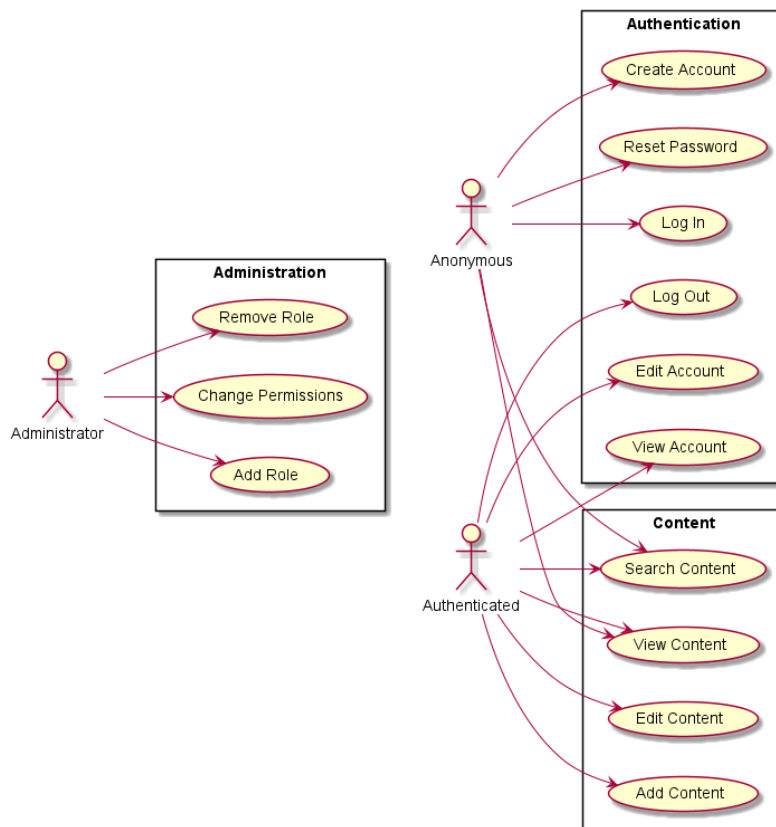
### 11.3.2 Erabilpen Kasuak

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.

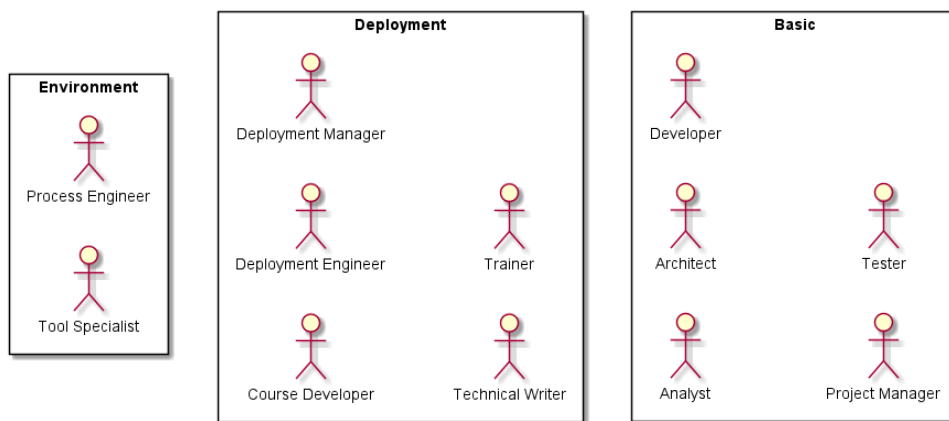
Erabiltzaile anonimoak autentikazioako ataza batzuk egiteko eta edukia ikusteko aukera bakarrik izango du. Erabiltzaile autentikatuak autentikazioako ataza batzuk egiteko eta edukia ikusi, editatu, gehitu eta bilatzeko aukerak izango ditu. Administrazioak erabiltzaile autentikatuak egin ditzakeen atazek gain administrazioako 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.



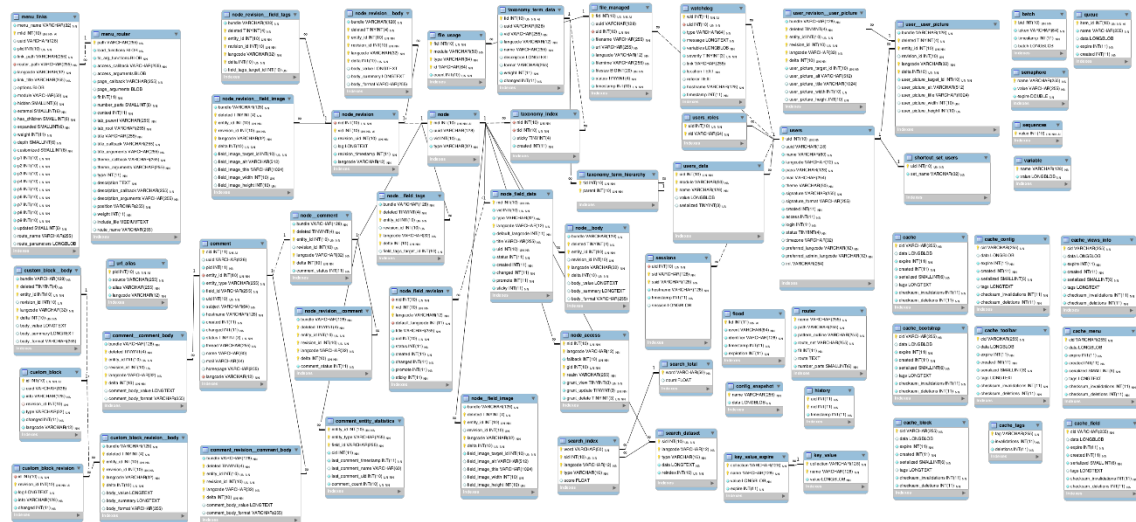
11.29. Irudia. OpenUP metodologiako rolak multzokatuta.



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



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

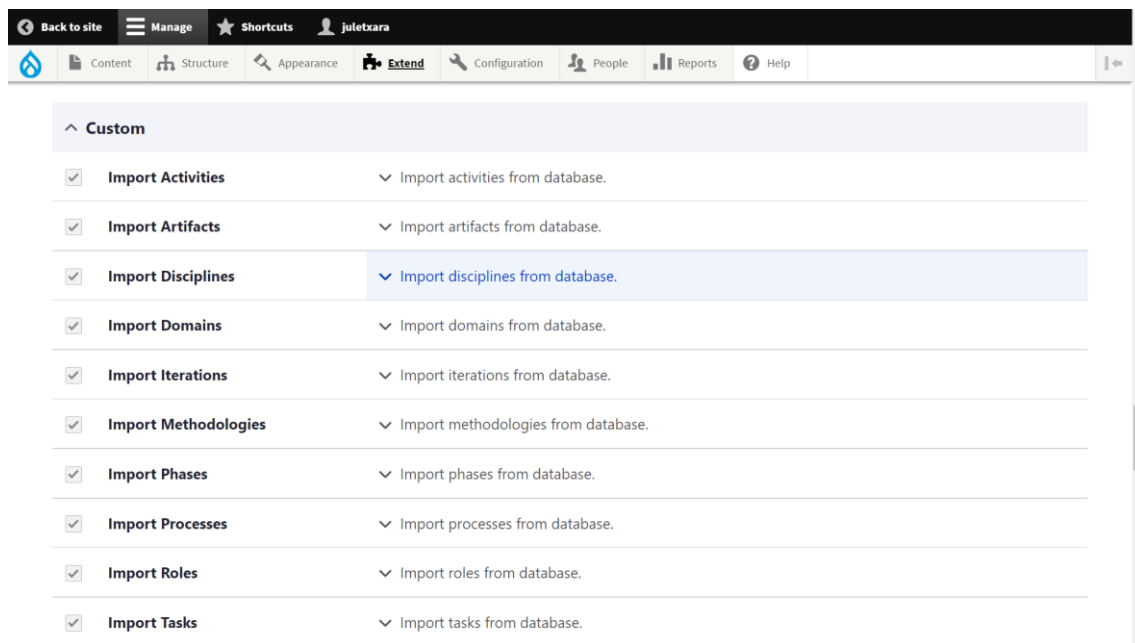
Esan bezala, azpisistemaren garapenerako Drupal 9 erabili da. Uneoro azkenengo bertsio egonkorra erabiltzea erabaki da. Horretarako, aktualizazioak zeuden bakoitzean Drupal eguneratu da *composer* tresna erabiliz. Programazio lengoia nagusia PHP izan da, Drupal lengoia horretan idatzita baitago.

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 aprobetxatzeko, 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.

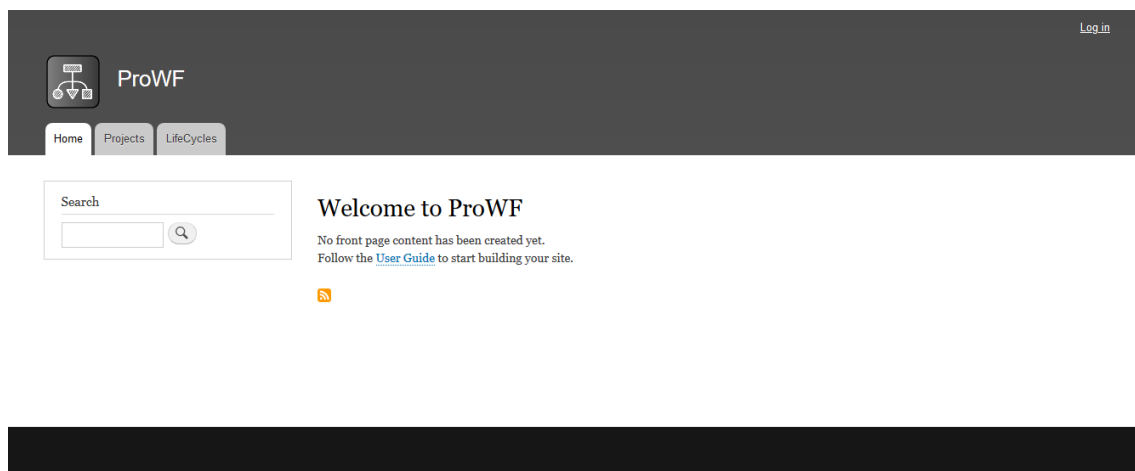
Eduki guztia eskuz gehitzeak zentzurik ez duenez, garatu beharreko lehenengo funtzionalitatea kanpoko datu-basetik edukia hartu eta automatikoki Drupal-era gehitzea da. Drupal-en funtzionalitateak gehitzeko modurik onena moduluak direnez, eduki mota bakoitza inportatzeko modulu bat garatu da, 11.33. 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*-ekin ordezkatu da. Aldaketa sinple honekin webguneak askoz itxura modernoagoa hartu du. Gainera, itxura ez ezik funtzionalitate gehigarriak ere eskaintzen dituzte, adibidez menu hedagarriak.

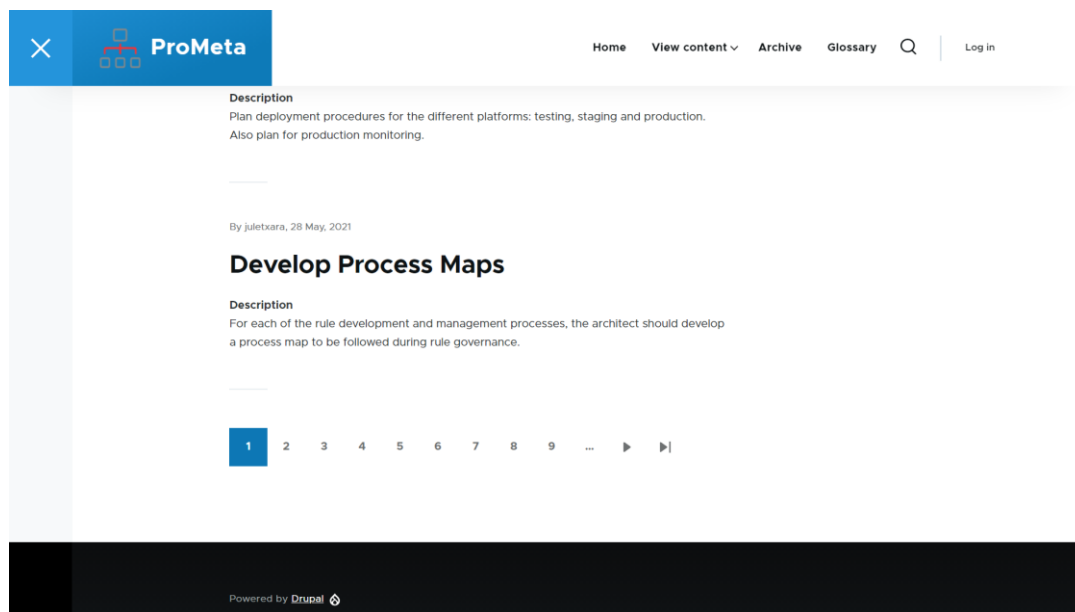


11.34. Irudia. ProWF sistemaren hasierako orria.



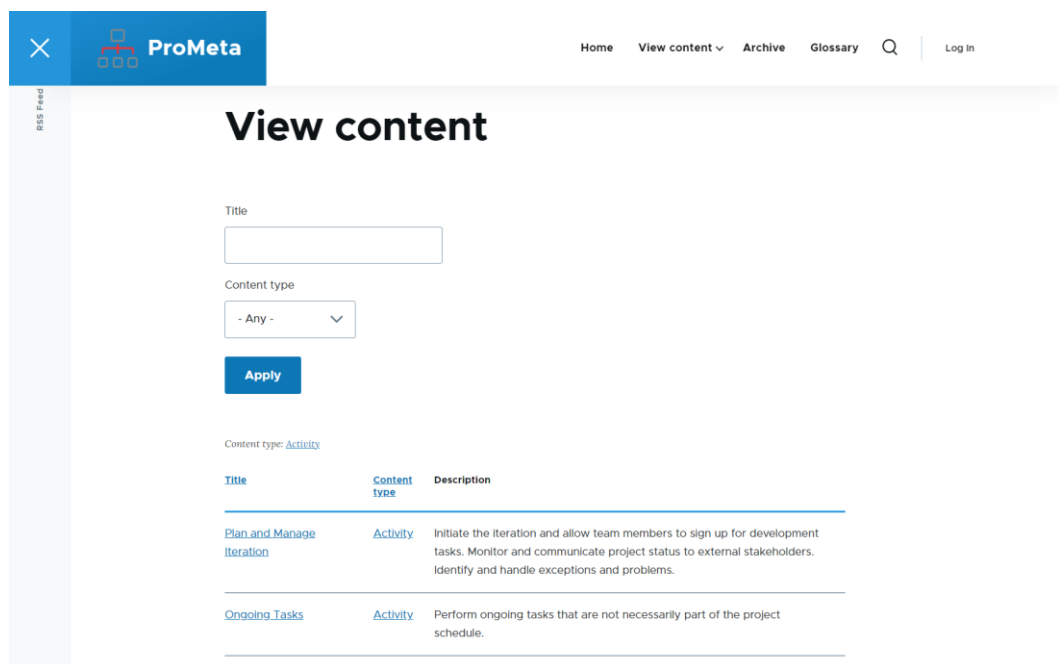
ProMeta webgunera sartutakoan hasierako orria bistaratzen da, 11.35. Irudian ikusten dena. Goiburuaren ezker aldean sistemaren logoa eta izena agertzen dira eta eskuinaldean menua. Erdialdean sistemara gehitutako edukia agertzen da ordena kronologikoan. Oinean webgunea Drupal-ekin eginda dagoela dioen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



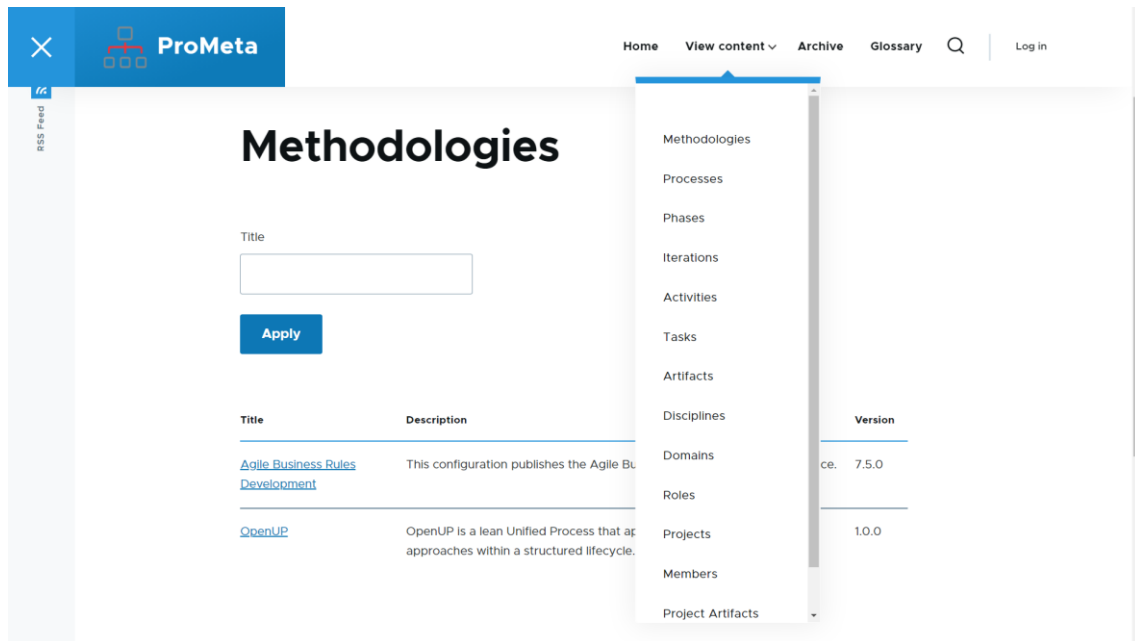
11.35. Irudia. Drupal webgunearen hasierako orria.

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.



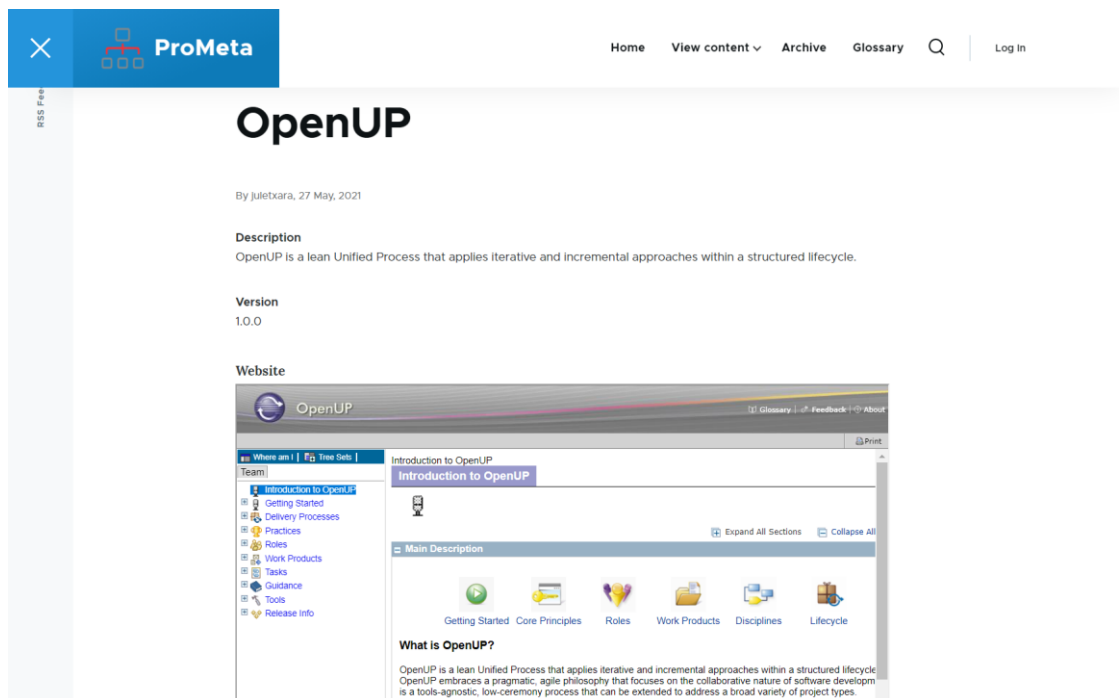
11.36. Irudia. Drupal webguneko edukia ikusteko orria.

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.



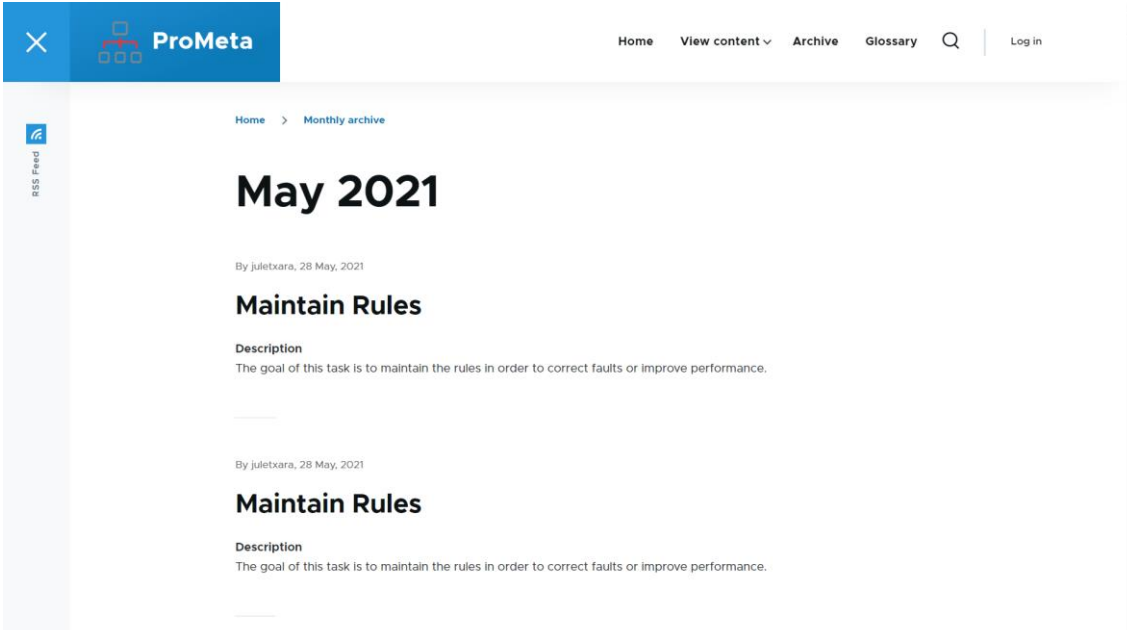
11.37. Irudia. Drupal webguneko metodologiak ikusteko orria.

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.



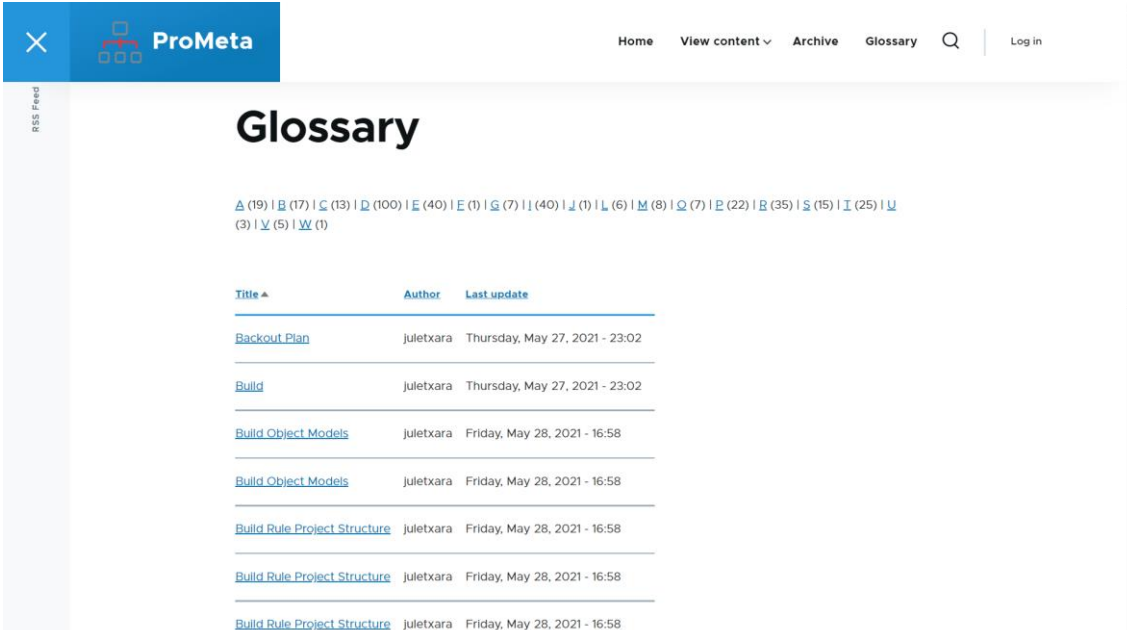
11.38. Irudia. Drupal webgunean OpenUP metodologia ikusteko orria.

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.

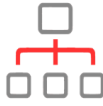
Edukia bilatzeko botoian klik egiten badugu, idazteko barra zabalduko da. Idatzitakoarekin bat egiten duten edukiak bakarrik azalduko dira. Adibidez, 11.41. Irudian OpenUP hitza duten edukiak bakarrik agertzen dira bilaketan.



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.



ProMeta

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

[Log in](#)



RSS Feed

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

[Home](#)

## Reset your password

Username or email address \*

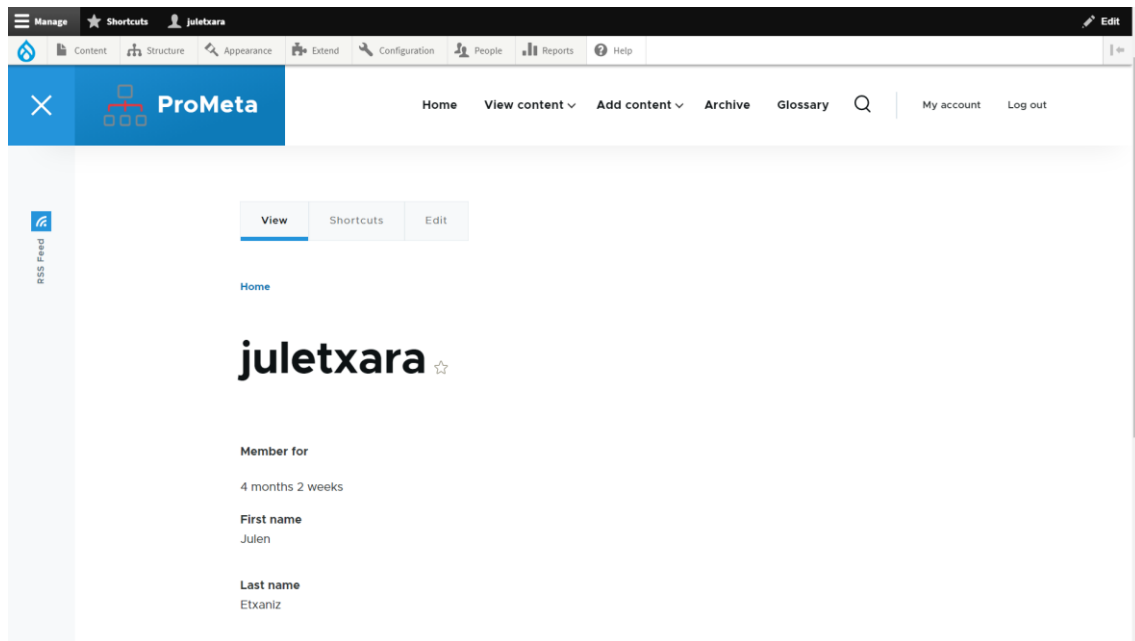
Password reset instructions will be sent to your registered email address.

[Submit](#)

11.44. Irudia. Drupal wenguneko pasahitza berrezartzeko orria.

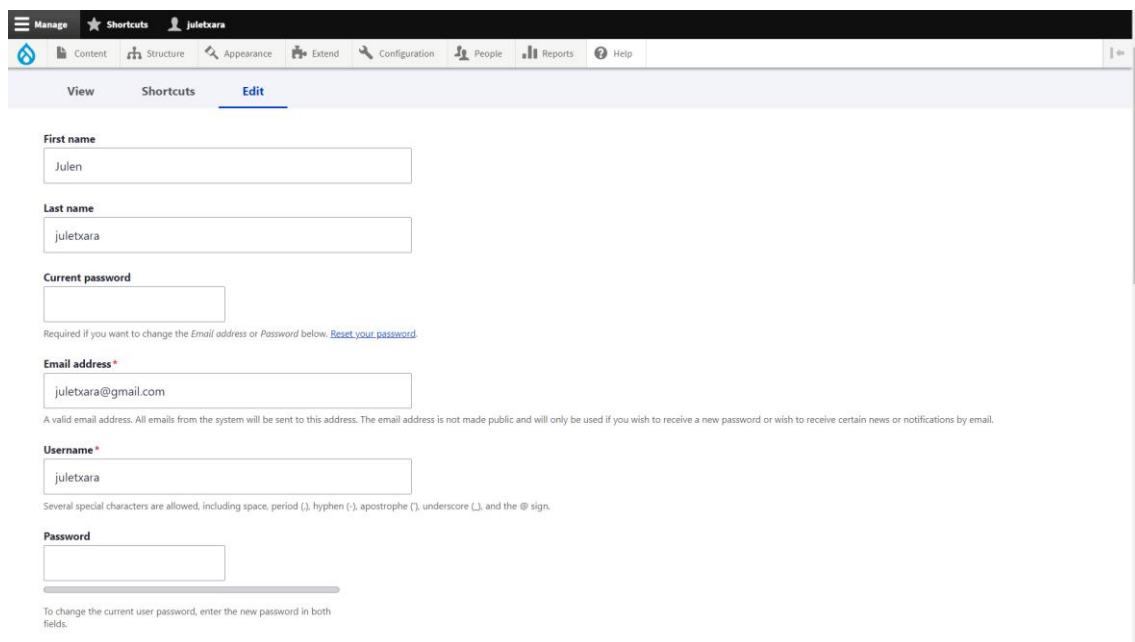


Saioa irekitzen badugu, gure kontura eramango gaitu. Goiko aldean bi menu berri daudela ikus dezakegu 11.45. Irudian, lehenengoa kontuari dagokiona eta bigarrena administrariaren menua. Gainera, menu nagusia edukia gehitzeko, kontua ikusteko eta saioa ixteko aukerak agertuko dira. Kontuari dagokionez,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



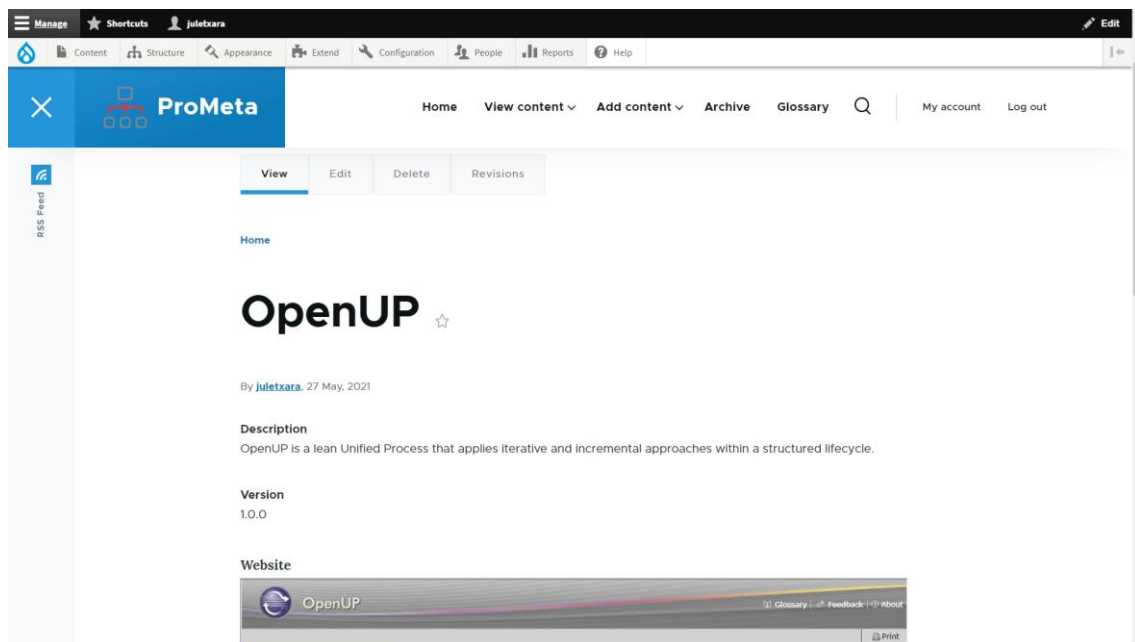
11.45. Irudia. Drupal webguneko erabiltzailearen kontuaren orria.

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.



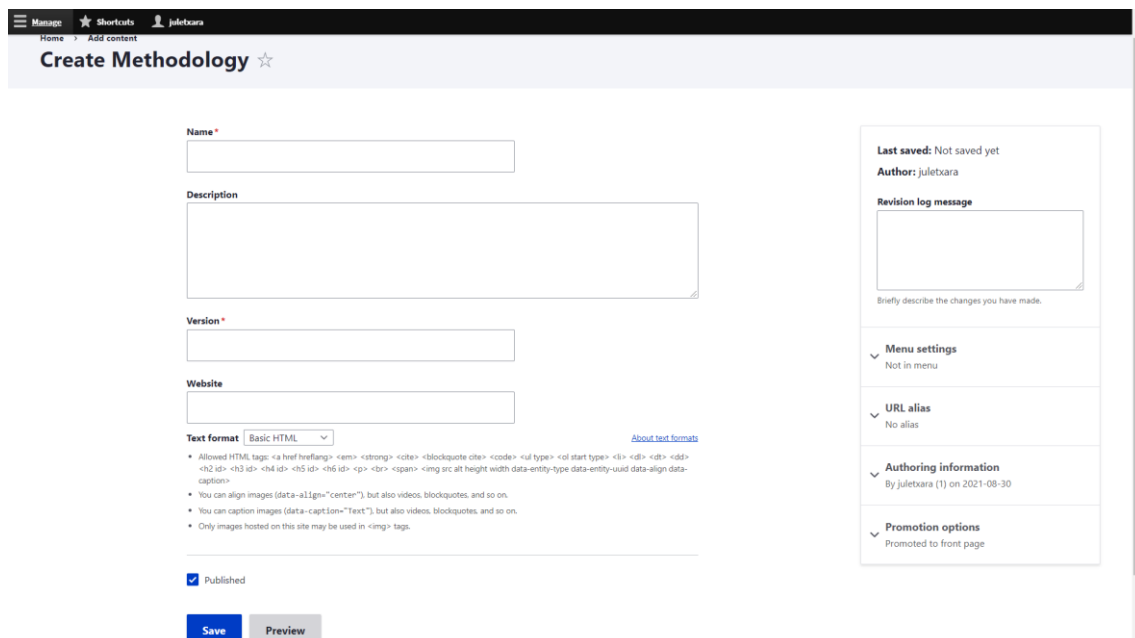
11.46. Irudia. Drupal webguneko kontua editatzeko orria.

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



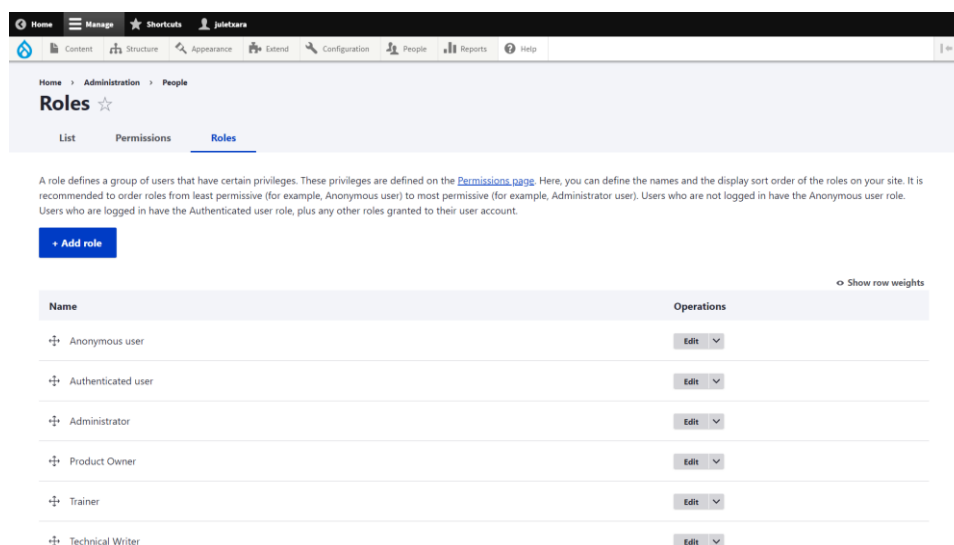
11.47. Irudia. Drupal webguneko OpenUP metodologiaren orria.

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.



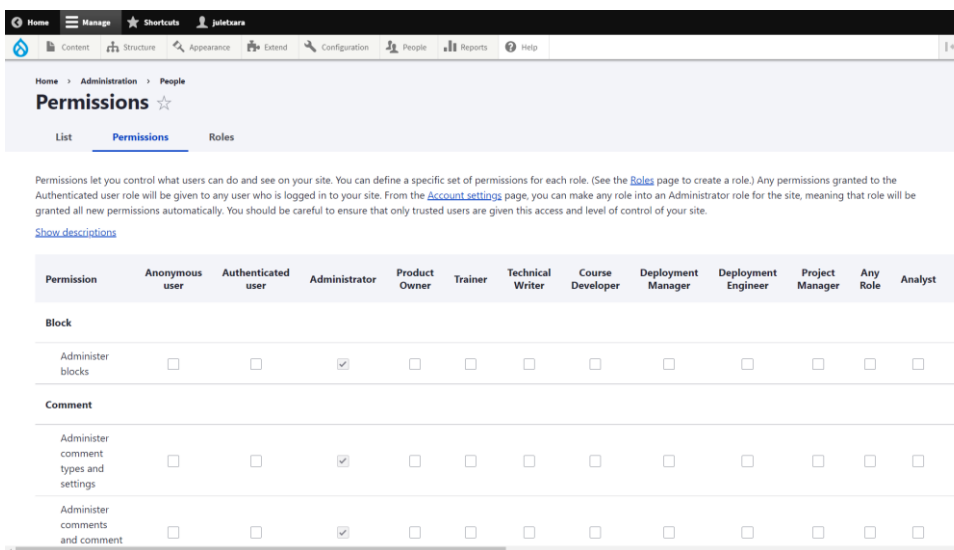
11.48. Irudia. Drupal webguneko metodologia sortzeko orria.

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 ereduan. 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](mailto: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 lengoia 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.