Arkitektura Koadernoa IO-System

1. Helburua

[Always address Sections 2 through 6 of this template. Other sections are recommended, depending on the amount of novel architecture, the amount of expected maintenance, the skills of the development team, and the importance of other architectural concerns.]

Dokumentu honek deskribatu egiten ditu proiektuan zehar, eta zehazki IO-System azpisistemaren atalean, filosofia, erabakiak, baldintzak, justifikazioak eta garrantzizkoa izan daitekeen edozer.

2. Arkitektura helburuak eta filosofia

[Describe the philosophy of the architecture. Identify issues that will drive the philosophy, such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues? Does it need to be robust for long-term maintenance?

Formulate a set of goals that the architecture needs to meet in its structure and behavior. Identify critical issues that must be addressed by the architecture, such as: Are there hardware dependencies that should be isolated from the rest of the system? Does the system need to function efficiently under unusual conditions?]

Azpisistemak hurrengo helburu nagusiak ditu:

- Erabiltzaileen autentikazioaren kudeaketa.
- Erabiltzaileen rolak eta baimenak kudeatzea.
- Bizi-zikloa definitzen duen metodologiaren informazioa gordetzea.
- Proiektuak sortzea eta bere informazioa gordetzea.
- Bizi-zikloan zehar sortutako dokumentuak gordetzea.
- Sortutako edukia guztia bistaratzea.

3. Hipotesiak eta menpekotasunak

[List the assumptions and dependencies that drive architectural decisions. This could include sensitive or critical areas, dependencies on legacy interfaces, the skill and experience of the team, the availability of important resources, and so forth]

CMS baten erabilera datuen sarrera/irteerarako irtenbide egokiena da. Webgune bat sortzeko aukera ematen duen tresna erabilerraza izateaz aparte, ez da baliabide tekniko aurreratuetara 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.

IO-System azpisistemaren arkitektura ModelEditor azpisistemarekin erlazionatuta dago. Izan ere, ModelEditor-ekin definitutako ereduaren datuak erabili beharko ditu. Hala ere, dependentzia ez da ProWF proiektuan bezain zuzena, ModelEditor sistemaren arkitektura abstraktuagoa delako.

Horrez gain, Drupal CMSarekiko dependentzia izango du, sistema horren gainean eraikita dagoelako. Beraz, Drupal-en bertsioak kudeatu beharko dira. Drupalek aldi berean dependentziak dituenez, horiek ere kudeatu beharko dira. Horregatik, Composer erabiliko da, dependentzien kudeaketa errazteko.

4. Betekizun arkitekturalki esanguratsuak

[Insert a reference or link to the requirements that must be implemented to realize the architecture.]

Sistemaren betekizunak Ikuspegia eta Betebeharren Espezifikazioa dokumentuetan definitu dira.

5. Erabakiak, mugak eta justifikazioak

[List the decisions that have been made regarding architectural approaches and the constraints being placed on the way that the developers build the system. These will serve as guidelines for defining architecturally significant parts of the system. Justify each decision or constraint so that developers understand the importance of building the system according to the context created by those decisions and constraints. This may include a list of DOs and DON'Ts to guide the developers in building the system.]

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 ziren.
- 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 erakundeen txostenetan, CMS atalean, liderra den Acquia enpresak Drupal erabiltzen du oinarri bezala.

Proiektu honetan saiakera bat egin da BPM eta workflow-lengoaiarik erabili gabe. Izan ere, eredutik sortutako datu-basea eta Drupal-eko moduluak bakarrik erabili dira.

6. Mekanismo arkitekturalak

[List the architectural mechanisms and describe the current state of each one. Initially, each mechanism may be only name and a brief description. They will evolve until the mechanism is a collaboration or pattern that can be directly applied to some aspect of the design.]

- **Prozesuak kudeatu:** Jarraitu beharreko prozesua bistaratuko da, bizi-zikloko zehaztasun guztiekin: faseak, iterazioak, jarduerak, atazak etab. Prozesu ingeniariak prozesuan aldaketak egiteko aukera izango du.
- **Proiektuak kudeatu:** Proiektu kudeatzaileak proiektuak sortu eta aldatzeko aukera izango du. Proiektuan kideak gehitu beharko ditu prozesuaren rolak betetzeko.
- **Artefaktuak idatzi:** Erabiltzaile bakoitzak bere rolari dagozkion artefaktuak bete beharko ditu txantiloietan oinarrituta. Artefaktu horiek DOC eta PDF dokumentu moduan igoko dira.

7. Funtsezko abstrakzioak

[List and briefly describe the key abstractions of the system. This should be a relatively short list of the critical concepts that define the system. The key abstractions will usually translate to the initial analysis classes and important patterns.]

8. Geruzak edo arkitektura frameworka

[Describe the architectural pattern that you will use or how the architecture will be consistent and uniform. This could be a simple reference to an existing or well-known architectural pattern, such as the Layer framework, a reference to a high-level model of the framework, or a description of how the major system components should be put together.]

- **Datu-basea:** Datu-base erlazionalak ereduaren informazio garrantzitsuena gordeko du, Drupal webgunerako beharrezkoa izan daitekeena. Hau da, OpenUP eta ABRD metodologien 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 lehenengodatu-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.

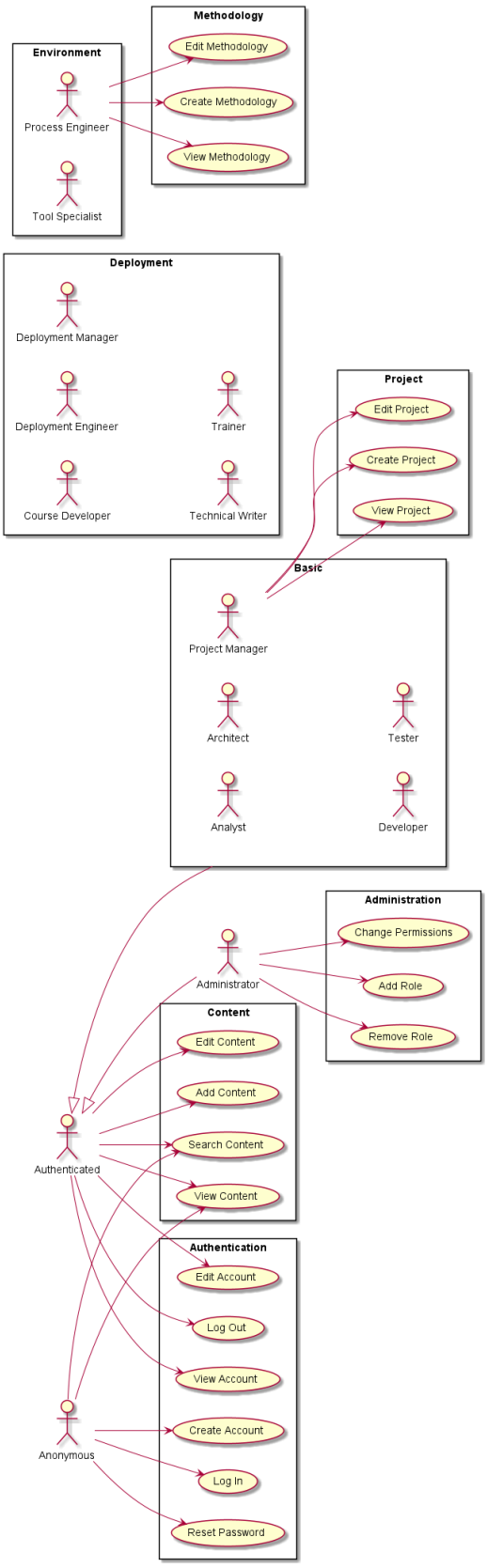
- **Drupal interfazea:** Interfazean edukia bistaratu eta aldatzeko aukera guztiak egongo dira. Esan bezala, edukia Drupal datu-basean gordeko da. Gainera, erabiltzailearen kontuekin zerikusia duten aukerak ere egongo dira. Horrez gain, administratzaileak aukera gehiagarri asko izango ditu webgunea kudeatzeko interfaze bidez.

9. Arkitektura ikuspegiak

[Describe the architectural views that you will use to describe the software architecture. This illustrates the different perspectives that you will make available to review and to document architectural decisions.]

Recommended views

- **Logical:** Describes the structure and behavior of architecturally significant portions of the system. This might include the package structure, critical interfaces, important classes and subsystems, and the relationships between these elements. It also includes physical and logical views of persistent data, if persistence will be built into the system. This is a documented subset of the design.
- **Operational:** Describes the physical nodes of the system and the processes, threads, and components that run on those physical nodes. This view isn't necessary if the system runs in a single process and thread.
- **Use case:** A list or diagram of the use cases that contain architecturally significant requirements.


Methodology
Edit Methodology
Create Methodology
View Methodology
Environment
Process Engineer
Tool Specialist
Deployment
Deployment Manager
Deployment Engineer
Trainer
Course Developer
Technical Writer
Project
Edit Project
Create Project
View Project
Basic
Project Manager
Architect
Tester
Analyst
Developer
Administration
Change Permissions
Add Role
Remove Role
Administrator
Content
Edit Content
Add Content
Search Content
View Content
Authenticated
Authentication
Edit Account
Log Out
View Account
Create Account
Log In
Reset Password
Anonymous