1
1
1

Desenvolupament d'un quadre d'indicadors de qualitat de les dades obertes publicades a CKAN

Ana Maria Mendoza Guasch
Màster en Enginyeria Informàtica
Desenvolupament d'aplicacions web

Ignasi Lorente Puchades
César Pablo Córcoles Briongos

07/01/2019

FITXA DEL TREBALL FINAL

Títol del treball:	*Desenvolupament d'un quadre d'indicadors de qualitat de les dades obertes publicades a CKAN*
Nom de l'autor:	*Ana Maria Mendoza Guasch*
Nom del consultor/a:	*Ignasi Lorente Puchades*
Nom del PRA:	*César Pablo Córcoles Briongos*
Data de lliurament (mm/aaaa):	*01/2019*
Titulació o programa:	*Màster en Enginyeria Informàtica*
Àrea del Treball Final:	*Desenvolupament d'aplicacions web*
Idioma del treball:	*Català*
Paraules clau	*open data, CKAN, data quality*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

La finalitat d'aquest treball és el desenvolupament d'una ferramenta per a conscienciar i millorar la qualitat de les dades que s'ofereixen als portals de dades obertes, especialment, aquells que corresponen a administracions públiques.

El context principal d'aplicació són tots aquells organismes interessats en la publicació de dades obertes (administració pública, organismes sense ànim de lucre, etc.) i l'augment de la qualitat de les seves dades.

Els objectius perseguits per aquest treball es volen aconseguir mitjançant el desenvolupament d'una extensió de codi obert, flexible i modular per al software CKAN que permeti als publicadors de dades obertes tenir una visió global de la qualitat de les seves dades i com es comparen amb la resta de dades que es publiquen dins del mateix catàleg per tal de fomentar la competitivitat entre diferents departaments.

S'ha aconseguit implementar totes les funcionalitats del producte viable mínim plantejat a més a més d'alguna funcionalitat addicional complint la planificació proposta inicialment. El projecte resultant està disponible a Github: https://github.com/ammendoza/ckanext-qadashboard

Una de les principals dificultats d'afrontar un projecte com aquest que es basa en un software ja existent és que requereix conèixer les tecnologies que utilitza, que sovent son moltes i que no sempre estan documentades adientment.

Abstract (in English, 250 words or less):

The purpose of this thesis is the development of a tool to raise awareness and improve the quality of the data offered on open data portals, especially those that belong to public administrations.

The results could be applied to any organization interested in publishing open data (public administrations, non-profit organizations, companies...) and improving the quality of the data they offer.

These objectives are to be achieved by developing an open, flexible and modular extension for CKAN software that allows open data publishers to have a global vision of their data quality and how they compare with the rest of publishers within the same catalog to promote competitiveness between different departments.

The final product implements all functionalities included on the minimum viable product and an additional functionality in accordance with the initial project plan. The resulting project is available at Github: https://github.com/ammendoza/ckanext-qadashboard

One of the main difficulties while facing a project which is based on an existing software such as this one is that it requires previous knowledge on the technologies it uses, which may not be documented properly.

Índex

Llista de figures

1. Introducció

1.1. Context i justificació del Treball

En aquest treball final de màster es vol aconseguir desenvolupar una ferramenta per a conscienciar i millorar la qualitat de les dades que s'ofereixen als portals de dades obertes, especialment, aquells que corresponen a administracions públiques. Aquest és un tema rellevant, ja que, com a ciutadans, hem d'exigir que les nostres administracions públiques ofereixin les dades de les que disposen de forma adequada per a que puguin revertir positivament en la societat.

A hores d'ara, el software de publicació de dades obertes de codi obert més utilitzat per diferents administracions de tot el món és CKAN, basat en Python. Alguns exemples de portals de dades obertes que utilitzen CKAN són el portal de dades obertes del govern federal dels Estats Units (data.gov), l'European Data Portal, el portal de dades obertes del govern d'Espanya (datos.gob.es), o el portal de dades obertes de l'Ajuntament de Barcelona. Disposa d'una comunitat de desenvolupament activa, ja que diferents administracions desenvolupen extensions amb funcionalitats addicionals sobre el core de CKAN que publiquen a Github.

Actualment existeixen algunes extensions per a CKAN d'avaluació de la qualitat de les dades o generació d'informes, com ckanext-qa o ckanext-report, però no s'ha trobat cap extensió de CKAN, dins de les disponibles públicament a repositoris de Github, que processi les dades de tot el conjunt de dades per a oferir una visió general.

Per altra banda, moltes d'aquestes extensions ofereixen dades del catàleg global, sense acotar la informació per als administradors o editors, que només poden actuar sobre una part del conjunt de dades; el grup de *datasets* sobre els que té permisos d'edició. D'aquesta manera, no es dilueix la responsabilitat dels editors en mantenir la qualitat de les seves dades entre la totalitat del catàleg, si no que se'ls ofereix dades concretes per a que puguin decidir, de manera autònoma, quines millores poden realitzar sobre les dades que publiquen.

Tal i com ja s'ha indicat anteriorment, el principal interessat en millorar les dades és la pròpia ciutadania, però en contextos com el de l'administració pública, aquest recull d'informació habitualment s'ha de canalitzar mitjançant un grup de treball responsable del portal de dades obertes i la publicació de dades en aquest portal; són ells els que vetllaran per que les dades publicades siguin de qualitat. Per a alleugerar aquesta tasca, es vol proveir també d'un mecanisme de notificació de problemes per a cadascun dels *datasets* que pugui servir per a marcar els datasets que requereixen atenció per part dels seus publicadors per tal de que puguin emmenar els problemes detectats.

1.2. Objectius del Treball

L'objectiu principal d'aquest treball final de màster és el desenvolupament d'una solució per a CKAN que sigui de codi obert, flexible i modular, que serveixi a la millora de les dades oferides per diferents organismes per tal d'enriquir la societat al seu voltant.

De **codi obert**, per a que es pugui auditar el seu codi i adaptar-lo. Es vol distribuir el codi desenvolupat a Github amb una llicència copyleft, GNU Affero GPL, per tant, qualsevol llibreria utilitzada en aquest projecte ha de ser compatible amb aquesta llicència.

Flexible, en quant a que el producte desenvolupat ha de poder adaptar-se en diferents situacions en funció del que es vol aconseguir a la plataforma on s'instal·la, dins de les limitacions del propi projecte.

Modular per a que la seva instal·lació no afecti al funcionament d'altres extensions instal·lades dins de la mateixa instància de CKAN.

Finalment, es tractarà de desenvolupar el projecte fent ús de bones pràctiques en el desenvolupament d'aplicacions i la utilització de codi llegible i de fàcil manteniment.

Per a aconseguir la millora de les dades oferides, es farà ús de conceptes ja existents al món de les dades obertes com, per exemple, el nivell de qualitat de les dades dels recursos i datasets del catàleg segons l'escala de valoració de Tim Berners-Lee:

- **1 estrella**: dades disponibles al web en qualsevol format amb una llicència oberta (per exemple: HTML).
- **2 estrelles**: dades disponibles amb un format estructurat que es pugui llegir per part d'una màquina (per exemple: excel).
- **3 estrelles**: que a més a més del punt anterior, sigui un format obert (per exemple: CSV).
- **4 estrelles**: tot l'anterior i que a més a més utilitzi estàndards del W3C (URIs) per a identificar elements
- **5 estrelles**: que a més a més d'utilitzar URIs, aquests elements s'enllacin amb altres per a donar-li context, el que es coneix com a linked data.

1.3. Enfocament i mètode seguit

Per tal d'aconseguir els objectius del treball, es desenvoluparà una extensió per a un producte existent, CKAN, que ja proporciona les funcionalitats principals per a la publicació de dades obertes. CKAN, junt amb l'extensió ckanext-qa també proporciona les dades necessàries per als indicadors que s'implementaran al quadre de comandaments.

Per tal d'aconseguir que l'extensió siga el més modular possible, es crearà un plugin diferent per a cada subsistema dins del mateix projecte d'extensió per tal de que les organitzacions que puguin reutilitzar aquesta extensió puguin decidir quines parts volen habilitar a la seva instància.

1.4. Planificació del Treball

Les fites principals del projecte corresponen a les dates d'entrega dels lliuraments indicats a l'aula de l'assignatura del TFM.

Fita	Data	Descripció
Definició del projecte (PAC1)	02/10/2018	Definició i planificació del projecte.
Disseny del projecte (PAC2)	31/10/2018	Definició del disseny del projecte: diagrames UML, wireframes i diagrames UML. Instal·lació inicial de CKAN i creació de l'estructura del projecte.
Desenvolupament del projecte (PAC3)	09/12/2018	Implementació del projecte de desenvolupament, realització de proves funcionals, documentació técnica i manual d'usuari.
Lliurament final	07/01/2019	Finalització del desenvolupament, de la memòria i realització de la presentació

Per tal d'assegurar el correcte compliment d'aquestes fites i balancejar el treball a realitzar fins a la finalització del TFM, s'estableixen les següents tasques amb una mida més petita que permeti valorar-les.

1 Definició del projecte (PAC1)

   1.1 Primera versió de la memòria

   1.2 Definició de les funcionalitats del projecte

2 Disseny del projecte (PAC2)

   2.1 Disseny de diagrames UML

   2.2 Disseny centrat en l'usuari

   2.3 Descripció de la infraestructura

   2.4 Instal·lació del software i creació de l'estructura del projecte

3 Desenvolupament del projecte (PAC3)

   3.1 Desenvolupament del subsistema de notificació d'errors

   3.2 Desenvolupament del subsistema de quadre de comandaments

3.3 Proves funcionals

3.4 Documentació tècnica

3.5 Manual d'usuari

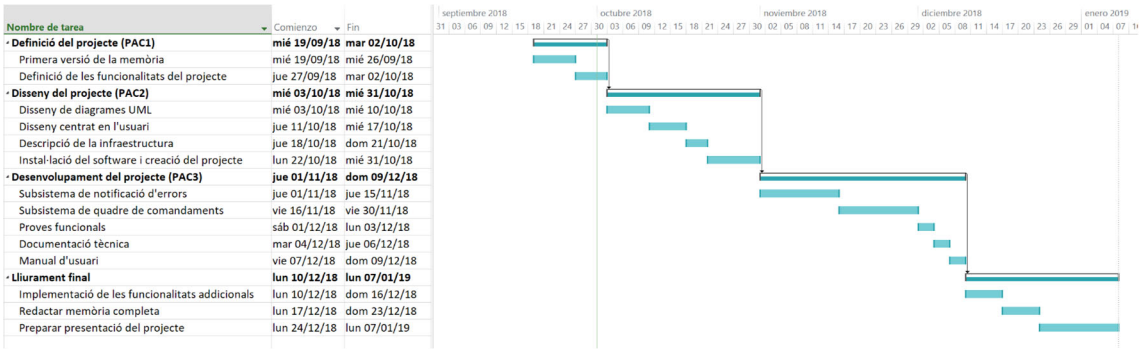
3.6 Implementació de funcionalitats addicionals

4 Lliurament final

4.1 Redactar memòria completa

4.2 Preparar presentació del projecte

La planificació temporal d'aquestes tasques agrupades per fites s'inclou al següent diagrama de Gantt.



Nombre de tarea	Comienzo	Fin
Definició del projecte (PAC1)	**mié 19/09/18**	**mar 02/10/18**
Primera versió de la memòria	mié 19/09/18	mié 26/09/18
Definició de les funcionalitats del projecte	jue 27/09/18	mar 02/10/18
Disseny del projecte (PAC2)	**mié 03/10/18**	**mié 31/10/18**
Disseny de diagrames UML	mié 03/10/18	mié 10/10/18
Disseny centrat en l'usuari	jue 11/10/18	mié 17/10/18
Descripció de la infraestructura	jue 18/10/18	dom 21/10/18
Instal·lació del software i creació del projecte	lun 22/10/18	mié 31/10/18
Desenvolupament del projecte (PAC3)	**jue 01/11/18**	**dom 09/12/18**
Subsistema de notificació d'errors	jue 01/11/18	jue 15/11/18
Subsistema de quadre de comandaments	vie 16/11/18	vie 30/11/18
Proves funcionals	sáb 01/12/18	lun 03/12/18
Documentació tècnica	mar 04/12/18	jue 06/12/18
Manual d'usuari	vie 07/12/18	dom 09/12/18
Lliurament final	**lun 10/12/18**	**lun 07/01/19**
Implementació de les funcionalitats addicionals	lun 10/12/18	dom 16/12/18
Redactar memòria completa	lun 17/12/18	dom 23/12/18
Preparar presentació del projecte	lun 24/12/18	lun 07/01/19

Figura 1: Diagrama de Gantt

1.5. Breu sumari de productes obtinguts

Els productes que s'obtindran a la finalització d'aquest projecte són:

- Codi font de l'extensió CKAN
- Documentació tècnica del projecte
- Manual d'usuari
- Memòria del treball
- Presentació del treball en vídeo

1.6. Breu descripció dels altres capítols de la memòria

El capítol 2 aprofundeix en el disseny de l'aplicació, començant amb la descripció dels perfils d'usuari que es consideren per a la seva implementació. S'indiquen totes les funcionalitats del projecte mitjançant l'enumeració de les històries

d'usuari i com es relacionen entre elles i amb els perfils indicats amb els diagrames de casos d'ús. A continuació es presenta el model i l'arquitectura de l'aplicació a nivell de servidor i de implementació.

El capítol 3 es centra en la fase de desenvolupament del projecte, que es considera que comença amb la instal·lació de tot el software base amb el que es va a treballar i la creació del projecte de desenvolupament.

2. Disseny de l'aplicació

2.1. Usabilitat/UX

2.1.1. Perfils d'usuari

Per al desenvolupament del projecte s'han de tenir en compte els següents perfils d'usuari:

- Visitant: usuari que accedeix al catàleg sense identificar. Aquest perfil no podrà veure cap de les funcionalitats que es desenvoluparan.
- Editor: usuari amb permisos d'administració en certs grups del catàleg. Pot editar la informació dels datasets dels grups per al que té aquest rol i afegir nous datasets dins dels mateixos grups. Per simplicitat, es consideren dins d'aquest perfil els usuaris amb rol «administrador» dins de grups específics, que no s'ha de confondre amb el perfil administrador indicat a continuació.
- Administrador: usuari amb permisos d'administració a tot el CKAN (sysadmin).

2.1.2. Històries d'usuari

Les funcionalitats del projecte de desenvolupament es poden separar en dos subsistemes que es desenvoluparan com a *plugins* independents l'un de l'altre per a augmentar la flexibilitat de l'extensió desenvolupada:

- Subsistema de quadre de comandaments (dashboard): s'encarregarà d'oferir una visió global a l'usuari dels datasets dels que és editor (en endavant, els seus datasets).
- Subsistema de notificació de problemes (notify problems): permetrà notificar errors detectats a un dataset concret per a perfils concrets de la plataforma.

La codificació de les històries d'usuari comença amb el subsistema al que pertany (01 per al subsistema de quadre de comandaments i 02 per al

subsistema de) i a continuació l'identificador de la història. Aquesta codificació resulta d'utilitat per a referenciar les funcionalitats al llarg del projecte.

[H01.00]: quadre de comandaments

Com a editor o administrador, vull veure una pàgina amb el resum de qualitat dels meus datasets.

Criteris d'acceptació:

- La pàgina només ha de ser visible a usuaris autenticats.

[H01.01]: gràfica de nivell de qualitat

Com a editor o administrador, vull veure una gràfica amb el número de datasets per a cada nivell de l'escala de qualitat de les dades de Tim Berners-Lee.

Criteris d'acceptació:

- Només he de veure els totals dels datasets que puc editar.
- Els resultats s'han de mostrar a una gràfica de barres.
- Cada nivell ha d'aparèixer com una barra.
- L'interval de valors ha d'anar de 0 a 5 i per unitats.

[H01.02]: datasets amb baixa qualitat

Com a editor o administrador, vull veure un llistat amb els datasets que tenen un nivell més baix dins de l'escala de qualitat de les dades de Tim Berners-Lee de entre els meus datasets.

Criteris d'acceptació:

- Només he de veure datasets que puc editar.
- He de veure el nom del dataset amb un enllaç al seu detall i el seu nivell de qualitat.

[H01.03]: problemes reportats als meus datasets

Com a editor o administrador, vull veure un llistat amb els últims problemes reportats als meus datasets.

Criteris d'acceptació:

- Només he de veure problemes dels datasets que puc editar.
- He de veure el seu títol, que ha d'enllaçar al detall del problema.
- Els problemes han d'estar en estat «Obert»
- S'ha de mostrar un llistat amb un màxim de 5 problemes.

[H01.04]: gràfica de vistes

Com a editor o administrador, vull veure una gràfica amb la progressió de visites per dies als meus datasets.

Criteris d'acceptació:

- Només he de veure les visites dels datasets que puc editar.
- S'han de mostrar les visites per a cada dia.
- S'han de mostrar les visites de l'últim mes.

[H01.05]: mitjana de qualitat

Com a editor o administrador, vull veure la mitjana de la qualitat dels meus datasets i els datasets del catàleg.

Criteris d'acceptació:

- La mitjana dels meus datasets ha de correspondre a la mitjana dels datasets que puc editar.
- La mitjana dels datasets del catàleg ha d'incloure tots els datasets.
- S'han de mostrar els punts de diferència entre les mitjanes (-0,5 o +1,1, per exemple), només si sóc editor.

[H01.06]: mitjana de vistes

Com a editor o administrador, vull veure la mitjana de la vistes dels meus datasets i els datasets del catàleg.

Criteris d'acceptació:

- La mitjana dels meus datasets ha de correspondre a la mitjana dels datasets que puc editar.
- La mitjana dels datasets del catàleg ha d'incloure tots els datasets.
- S'han de mostrar els punts de diferència entre les mitjanes només si sóc editor.

[H01.07]: mitjana de problemes actius

Com a editor o administrador, vull veure la mitjana de problemes en estat obert dels meus datasets i els datasets del catàleg.

Criteris d'acceptació:

- La mitjana de problemes dels meus datasets ha de correspondre a la mitjana del problemes oberts dels datasets que puc editar.
- La mitjana de problemes del catàleg ha d'incloure tots els datasets.
- S'han de mostrar els punts de diferència entre les mitjanes, només si sóc editor.

[H01.08]: personalitzar dashboard

Com a administrador, vull canviar l'ordre dels blocs que es veuen al quadre de comandaments.

Criteris d'acceptació:

- Quan canvio l'ordre dels blocs, tots els usuaris veuen el mateix ordre.

[H01.09]: quadre de format lliure

Com a administrador, vull poder afegir un nou bloc al quadre de comandaments amb un contingut lliure.

Criteris d'acceptació:

- Quan afegeixo el bloc, tots els usuaris el veuen al quadre de comandaments.
- El bloc em permet afegir, al menys, text, imatges i enllaços.

[H01.10]: filtrar vistes per dates

Com a editor, vull poder seleccionar el rang de dades sobre el que es mostren les visites als meus datasets.

Criteris d'acceptació:

- El formulari em permet seleccionar una data d'inici.
- El formulari em permet seleccionar una data de fi.
- El gràfic s'actualitza per a mostrar les visites entre les dades seleccionades.

[H02.00]: afegir problema

Com a usuari identificat, vull poder reportar un problema a un dataset.

Criteris d'acceptació:

- El problema s'obre amb estat «obert».
- El problema conté: data de creació, dataset, usuari que crea el problema, tipus i una nota de l'usuari que crea el problema.
- No puc crear un problema sense informar totes les dades.

[H02.01]: notificar problema per correu

Com a editor, vull rebre per correu un missatge quan es reporta un problema a un dels datasets que puc editar.

Criteris d'acceptació:

- El missatge s'envia en el moment que un usuari reporta un problema.
- Només reben el missatge els editors del dataset.
- Els administradors globals no reben aquest missatge.

[H02.02]: afegir informació a un problema

Com a editor, vull poder afegir informació a un problema per tal de canviar el seu estat o afegir notes addicionals.

Criteris d'acceptació:

- He de poder canviar el seu estat a: resolt, no reproduïble, requereix informació addicional.
- He de poder deixar el mateix estat que ja tenia el problema.
- Quan canvio l'estat s'emmagatzema: usuari, data, estat, nota de l'usuari que fa el canvi.

[H02.03]: canviar estats dels problemes en bloc

Com a editor, vull canviar l'estat de diversos problemes a l'hora.

Criteris d'acceptació:

- Em permet seleccionar més d'un problema a l'hora.
- S'aplica el mateix estat a tots els problemes seleccionats.

[H02.04]: veure problema

Com a usuari identificat, vull veure la informació d'un problema reportat a un dataset.

Criteris d'acceptació:

- He de veure la següent informació: títol, data de creació, dataset, usuari que reporta, tipus, nota, estat, llistat amb els canvis d'estat, notes afegides i data del canvi.

[H02.05]: llistar problemes

Com a usuari identificat, vull veure el llistat de problemes d'un dataset.

Criteris d'acceptació:

- He de poder veure la següent informació de cada problema: títol, estat actual.

[H02.06]: llistar problemes dels meus datasets

Com a editor, vull veure el llistat de problemes dels meus datasets.

Criteris d'acceptació:

- He de veure tots els problemes que s'han reportat als meus datasets.
- Per a cada problema, s'ha de mostrar el seu títol amb un enllaç al seu detall i el seu estat actual.

[H02.07]: filtrar problemes per tipus

Com a usuari, vull filtrar els problemes per tipus de problema.

Criteris d'acceptació:

- Em permet seleccionar qualsevol estat.
- Els problemes que es mostren corresponen a datasets que puc editar.
- Només em mostra els problemes del tipus que he seleccionat.

[H02.08]: ordenar per data de creació

Com a usuari, vull ordenar els problemes per data de creació.

Criteris d'acceptació:

- Em permet seleccionar data de creació descendent o ascendent.
- Els problemes del llistat s'ordenen segons el criteri indicat.

[H02.09]: ordenar per data de modificació

Com a usuari, vull ordenar els problemes per data d'última modificació.

Criteris d'acceptació:

- Em permet seleccionar data de creació descendent o ascendent.
- Els problemes del llistat s'ordenen segons el criteri indicat.

[H02.10]: llistar els tipus de problema

Com a administrador, vull poder llistar els tipus de problemes creats i que els usuaris podran seleccionar al reportar.

Criteris d'acceptació:

- Em permet indicar el nom del nou tipus.

[H02.11]: afegir tipus de problema

Com a administrador, vull poder afegir nous tipus de problemes que els usuaris podran seleccionar al reportar.

Criteris d'acceptació:

- El nou tipus de problema apareix en el seleccionable al formulari per afegir un nou problema.
- El formulari em permet indicar el seu nom.
- El formulari em permet seleccionar els possibles estats anteriors.

[H02.12]: modificar tipus de problema

Com a administrador, vull poder modificar un tipus de problema.

Criteris d'acceptació:

- El tipus de problema es modifica amb les dades indicades.
- El tipus de problema apareix modificat al seleccionable de tipus del formulari per afegir un nou problema.

[H02.13]: eliminar tipus de problema

Com a administrador, vull poder eliminar un tipus de problema per a que ja no aparegui al reportar un problema.

Criteris d'acceptació:

- El tipus de problema ja no apareix al seleccionable de tipus del formulari per afegir un nou problema.

2.1.3. Diagrama de casos d'ús

Una vegada descrites les històries d'ús, es presenta el diagrama de casos d'ús per a cadascun dels subsistemes.

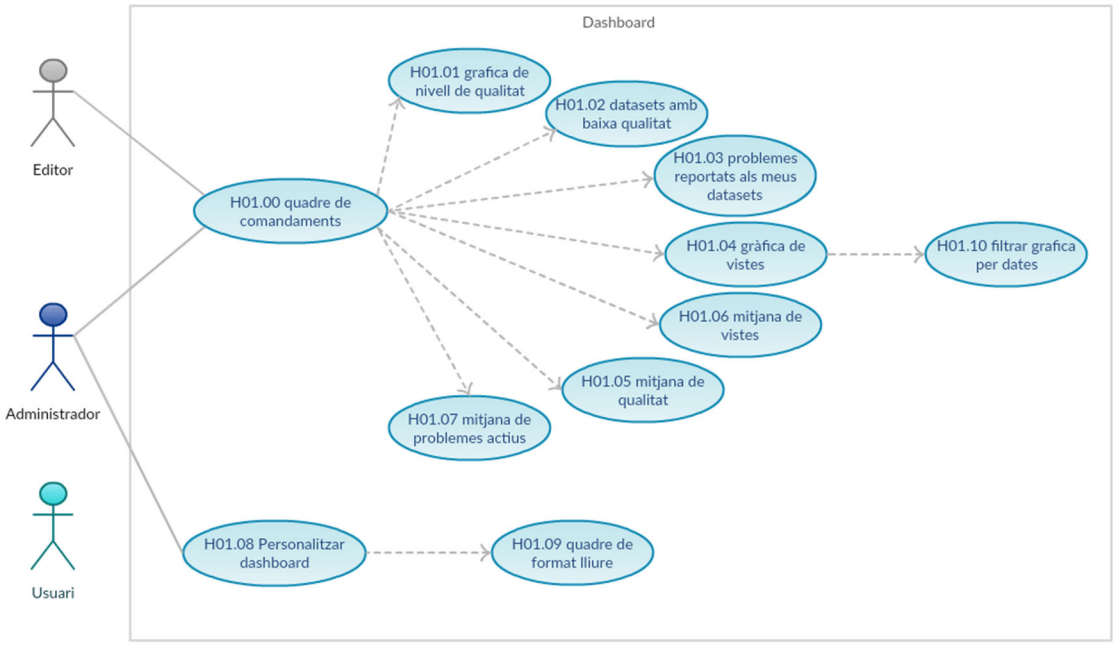


Figura 2: diagrama de casos d'us, subsistema quadre de comandaments

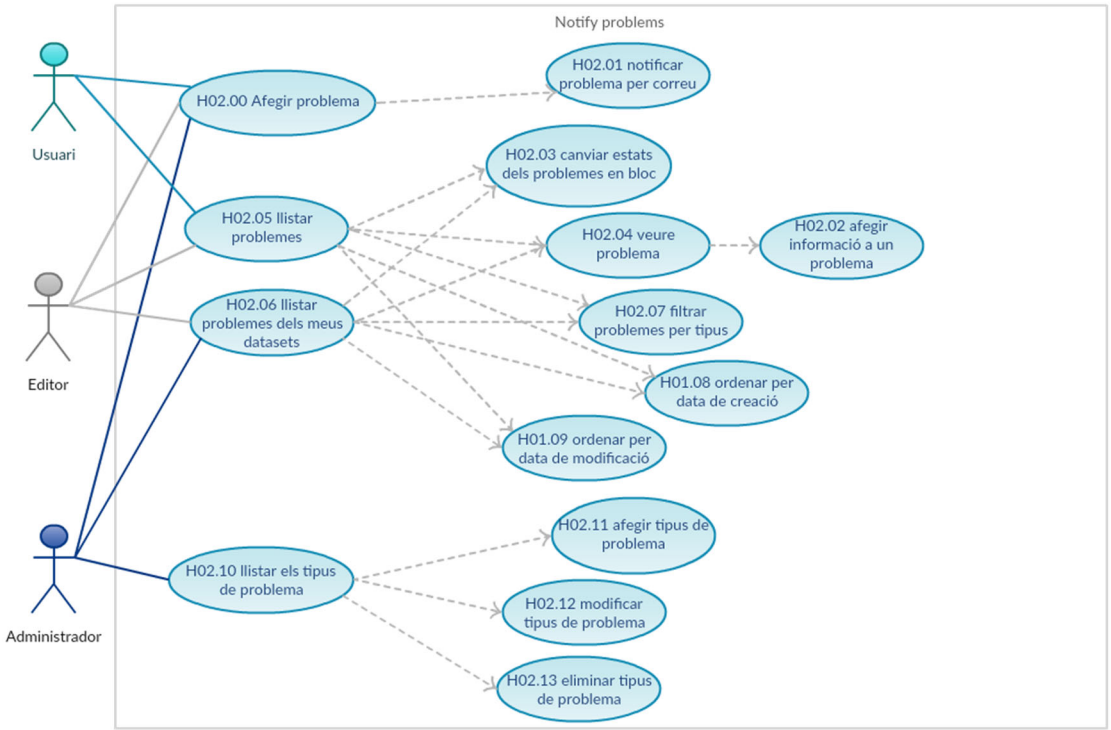


Figura 3: diagrama de casos d'ús, subsistema notificació de problemes

2.2. Producte viable mínim (MVP)

De les funcionalitats indicades al llistat d'històries d'usuari, es consideren les següents com a necessàries per a desenvolupar un producte viable mínim (MVP) i per tant, s'han de prioritzar al desenvolupament:

- [H01.00]: quadre de comandaments
- [H01.01]: gràfica de nivell de qualitat
- [H01.02]: datasets amb baixa qualitat
- [H01.03]: problemes reportats als meus datasets
- [H01.04]: gràfica de vistes
- [H01.05]: mitjana de qualitat
- [H01.06]: mitjana de vistes
- [H01.07]: mitjana de problemes actius
- [H02.00]: afegir problema
- [H02.02]: canviar estat d'un problema
- [H02.04]: veure problema
- [H02.05]: llistar problemes
- [H02.06]: llistar problemes dels meus datasets

La resta de funcionalitats es tractaran d'incloure al desenvolupament en quant a que es considera que afegeixen valor al producte resultant.

A més a més de les funcionalitats ja descrites, al subsistema de quadre de comandaments es pot valorar la inclusió de d'indicadors addicionals si la seva inclusió resultaria molt beneficiosa per a l'avaluació de la qualitat de les dades.

2.3. Diagrama d'estats

A continuació es presenta el diagrama d'estats inicial per als errors reportats del subsistema de notificació d'errors.

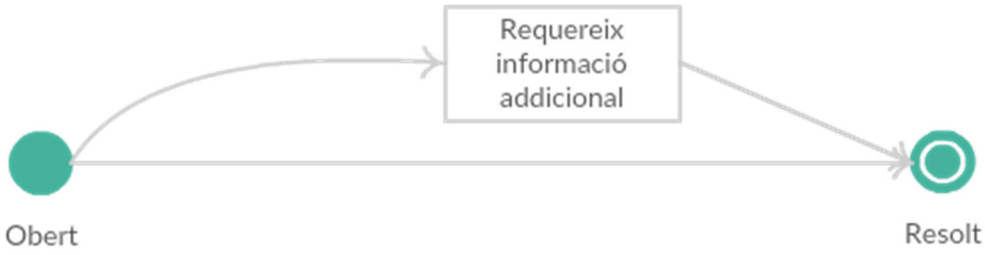


Figura 4: Diagrama d'estats inicial per als problemes

Com es pot observar, es tracta d'un flux molt simple, que es podria ampliar en futures versions del projecte per tal d'augmentar les possibilitats a l'hora de gestionar els problemes notificats.

2.4. Model

El model del projecte depèn en gran mesura del model del core de CKAN, ja que el que fa és estendre les funcionalitats d'aquest software. A continuació es presenten els diagrames UML de les entitats del projecte i les seves relacions.

El subsistema de quadre de comandaments utilitzarà principalment les dades proporcionades per l'API core de CKAN i les afegides per l'extensió ckanext-qa a l'entitat *Dataset*. Només en cas de que s'implementin les històries d'usuari H01.08 i H01.09 caldrà generar les taules de BBDD i les classes necessàries per a gestionar el seu model.

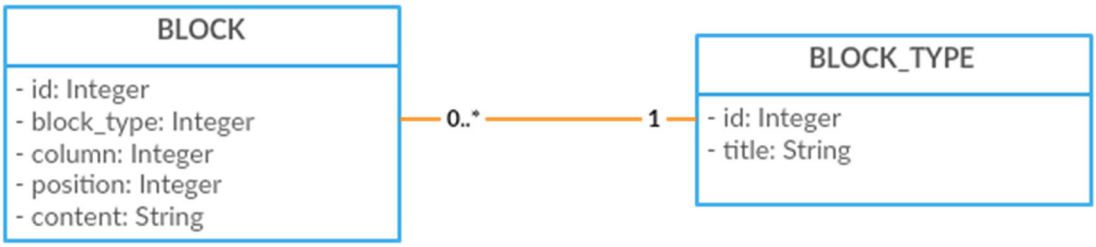


Figura 5: Diagrama UML del subsistema de quadre de comandaments

Per altra banda, el subsistema de notificació de problemes es compon de tres entitats que es relacionen amb les entitats *Dataset* i *User* del core de CKAN.

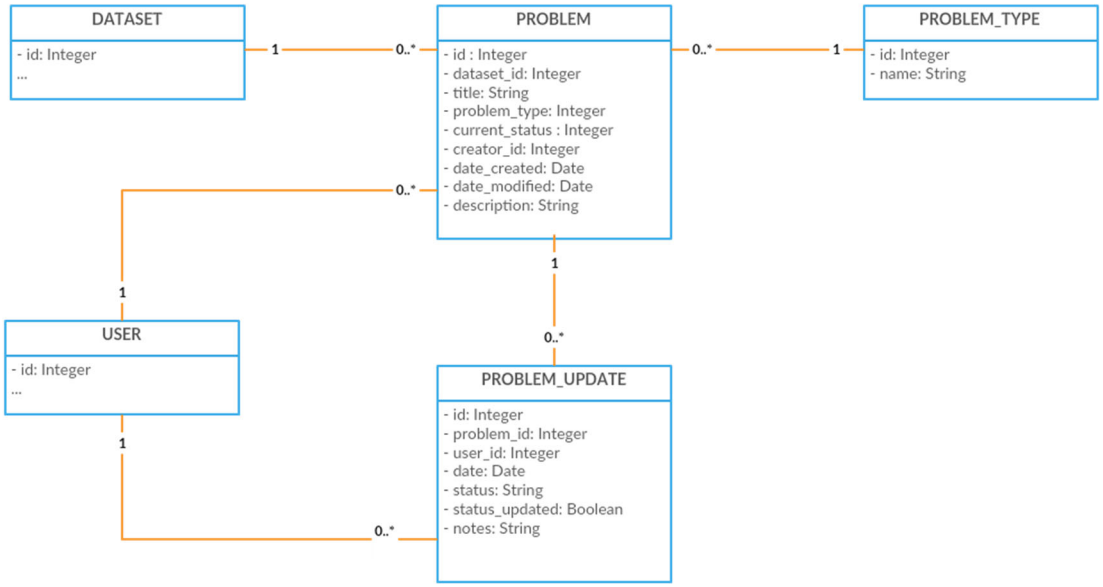


Figura 6: Diagrama UML del subsistema de notificació de problemes

2.5. Arquitectura

L'arquitectura de la infraestructura sobre la que treballarem es basa en un servidor Nginx que fa de frontal web mitjançant un proxy al servei d'aplicacions i serveix els recursos estàtics. Apache2 funciona en aquest cas com a servidor d'aplicacions amb mod_wsgi.

Per a desenvolupar les funcionalitats indicades, es desenvoluparà una extensió per a CKAN, que ens permet afegir noves funcionalitats dins de CKAN sense necessitat de modificar el core, fent que el manteniment i actualització del software base no resulti compromès.

Crearem un sol projecte d'extensió, que anomenarem *ckanext-qadashboard*, i dins d'aquest projecte crearem dos plugins diferents:

- *QA dashboard*
- Notify problems

Dins d'aquests plugins, s'utilitzarà l'arquitectura MVC per tal de separar la capa del model, de la lògica del control i la seva presentació.

2.5.1. Capa de controlador

Es crearà un controlador per a cada plugin i un fitxer Python per a cadascun d'ells:

- NotifyProblemsController (notify_controller.py)
- QADashboardController (qadashboard_controller.py)

Tots dos han d'estendre la classe *BaseController* del paquet *ckan.plugins.toolkit*, que proporciona classes d'utilitat per a la creació d'extensions per a CKAN.

2.5.2. Capa de model

Les dades del projecte s'emmagatzemaran a les tables necessàries a la BBDD PostgreSQL, dins del mateix esquema de CKAN. Per tal de facilitar la instal·lació de l'extensió resultant, s'afegiran els scripts de creació de taules i inserció de les dades necessàries dins d'una classe de comandes que anomenarem *QADashboardCommands*, que ha d'estendre la classe *ckan.plugins.toolkit.CkanCommand*.

Per abstraure la interacció amb la base de dades dels controladors, s'implementaran les classes Python necessàries per a gestionar el model amb la llibreria *sqlalchemy*.

2.5.3. Capa de presentació

La capa de presentació s'implementarà amb HTML5 i Jinja2 per a la generació del codi font HTML.

Per a canviar la presentació dels elements, s'utilitzarà CSS3 i Bootstrap 3, que s'inclou al core de CKAN des de la versió 2.8, que és la que s'utilitzarà per a aquest proejcte.

També s'inclouran algunes millores en la usabilitat de les funcionalitats amb JavaScript. Per als gràfics s'utilitzarà la llibreria Chart.js, disponible amb llicència MIT i que ja proporciona diversos tipos de gràfics que necessitarem per implementar alguns dels blocs del quadre de comandaments.

3.
4. Desenvolupament

4.1. Instal·lació del software base

Per tal de desenvolupar el nostre projecte, necessitem instal·lar el software base que serà CKAN, a més a més d'algunes extensions bàsiques que ens proveiran de la informació bàsica per al nostre quadre de comandaments.

En primer lloc, s'ha creat una màquina virtual amb Virtualbox amb una imatge d'Ubuntu server 16.04 de 64 bits, que ens permetrà fer una instal·lació des de paquet, tal i com s'explica a la documentació de CKAN[1].

Seguint la documentació, abans d'instal·lar el paquet de CKAN, hem de descarregar i configurar els seus requeriments, que són:

- Nginx
- Apache2 amb mod_wsgi
- Solr
- PostgreSQL
- Redis

A continuació, descarreguem i instal·lem el paquet .deb de CKAN:

```
wget http://packaging.ckan.org/python-ckan_2.8-xenial_amd64.deb
sudo dpkg -i python-ckan_2.8-xenial_amd64.deb
```

Una vegada instal·lats, configurats i iniciats tots els serveis necessaris, hem de canviar la configuració bàsica de CKAN editant el fitxer */etc/ckan/default/production.ini* per a indicar l'URL de connexió a la BBDD correcta, l'URL del servidor Solr i l'URL d'accés a CKAN, que en el nostre cas hem establit com a http://ckan, ja que estem accedint al servidor en l'entorn local i no s'accedeix des de internet, i executar la comanda que genera les taules necessàries a la BBDD:

```
sudo ckan db init
```

CKAN s'instal·la a un entorn virtual propi, comú a les aplicacions Python[2], per a que les versions requerides de les llibreries que utilitza no interfereixin amb la resta d'aplicacions Python del mateix sistema, per tant, per a executar les comandes d'interacció amb CKAN, per exemple, la comanda *paster*, cal que primer activem el seu entorn virtual (virtualenv):

```
. /usr/lib/ckan/default/bin/activate
cd /usr/lib/ckan/default/src/ckan
```

Amb la comanda *paster create-test-data* carreguem algunes dades de proves que ens poden servir per al desenvolupament:

```
paster create-test-data -c /etc/ckan/default/production.ini
```

[1]https://docs.ckan.org/en/latest/maintaining/installing/install-from-package.html
[2]https://docs.python.org/3/tutorial/venv.html

En aquest punt, creem l'usuari d'accés amb rol administrador amb la comanda *paster sysadmin* com s'indica a la guia de primers passos[3]:

```
paster sysadmin add admin email=ammendoza@uoc.edu name=admin -c
/etc/ckan/default/production.ini
```

Comprovem que podem accedir a l'URL de CKAN des de el nostre SO principal i que ens podem identificar.

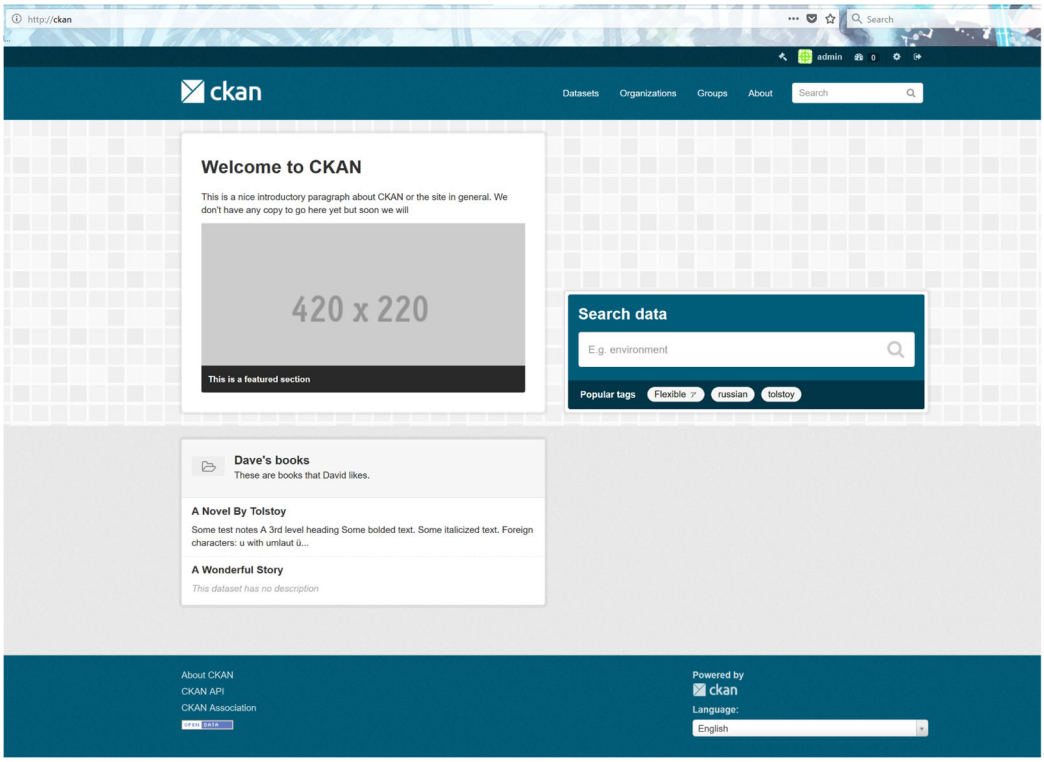


Figura 7: Pàgina principal de CKAN a la nostra instal·lació

4.1.1. Habilitar el mode de desenvolupament

Fins a aquest punt s'ha instal·lat CKAN per a que funcioni amb Nginx com a frontal i Apache2 amb mod_wsgi com a servidor d'aplicacions, una configuració recomanable per a l'entorn de producció, però es pot iniciar la instal·lació de CKAN amb *paster* per tal d'accedir al catàleg en mode de desenvolupament. La documentació de CKAN per a la instal·lació des de codi font[4] inclou els passos a realitzar per a habilitar aquest mode, però hem de tenir en compte que ja hem instal·lat CKAN des de paquet i per tant, molts dels passos no seran necessaris.

En primer lloc, copiem l'arxiu */etc/ckan/default/production.ini* a */etc/ckan/default/development.ini* i canviem la següent línia per habilitar el mode debug:

```
debug = true
```

[3]https://docs.ckan.org/en/latest/maintaining/getting-started.html
[4]https://docs.ckan.org/en/2.8/maintaining/installing/install-from-source.html

Com que hem instal·lat CKAN des de paquet, també hem d'instal·lar algunes llibreries de desenvolupament per a Python necessàries per a habilitar les ferramentes de debug:

```
sudo apt-get install python-dev libpq-dev
```

A aquest mode de desenvolupament no s'accedeix mitjançant Nginx, si no que s'ha d'iniciar un procés *paster* que serveix al port 5000. Aquest servei el podem iniciar com a tasca de segon pla amb la següent comanda:

```
nohup paster serve /etc/ckan/default/development.ini &
```

Si accedim ara des de el navegador a l'URL http://ckan:5000 veiem que apareix informació de debug al peu de cada pàgina que no es mostrava amb la configuració de producció.

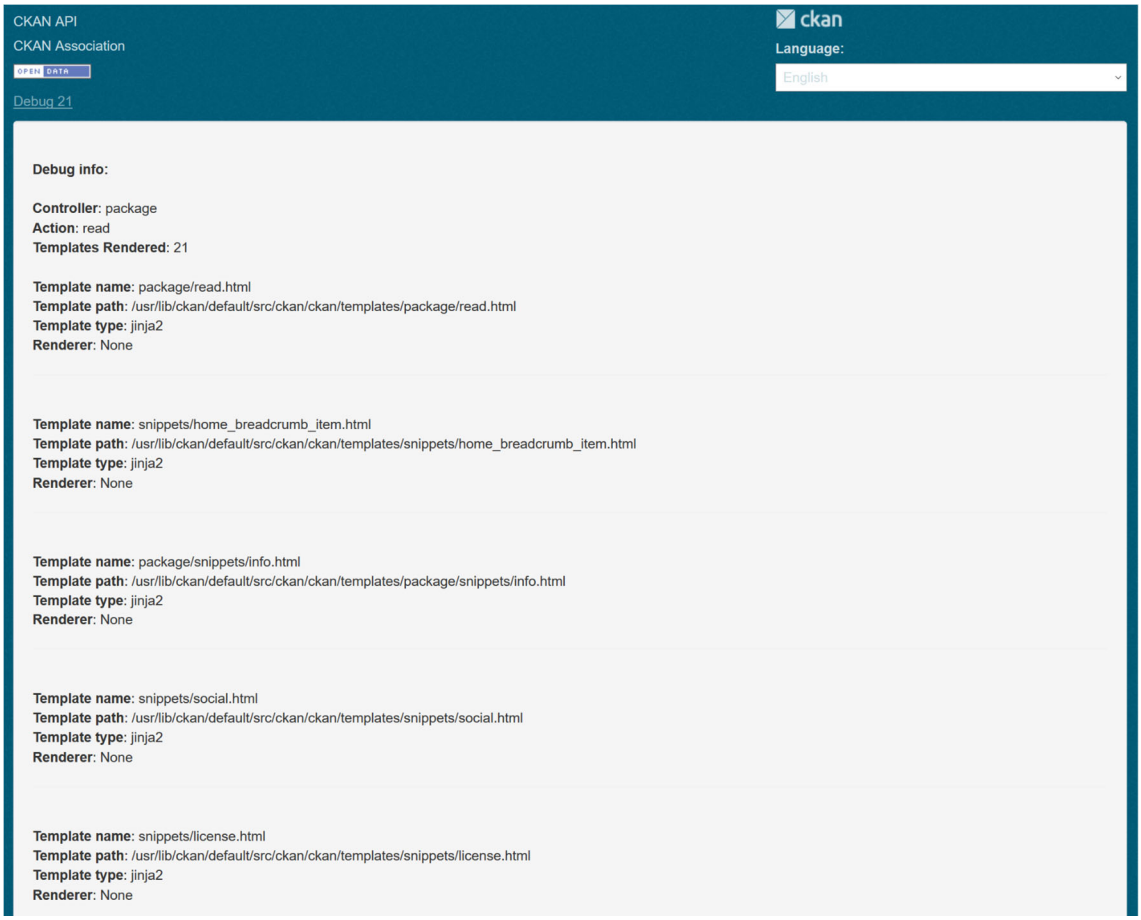


Figura 8: Informació de debug de CKAN

En aquest punt podríem parar el servei d'Apache2 i el de Nginx, ja que no ens resulten necessaris mentre estem servint les peticions amb *paster serve*.

4.1.2. Instal·lació d'extensions

A sobre de CKAN s'han instal·lat les següents extensions, disponibles a Github amb llicències copyleft:

- ckanext-archiver: https://github.com/ckan/ckanext-archiver (llicència MIT)
- ckanext-report: https://github.com/datagovuk/ckanext-report (llicència GNU Affero)
- ckanext-qa: https://github.com/ckan/ckanext-qa (llicència MIT)

Dins del readme.MD d'aquests projectes trobem les seves instruccions d'instal·lació pas a pas i com executar els seus tests.

S'ha de tenir en compte que aquestes extensions s'establiran com a requisits per a la instal·lació de la extensió que desenvoluparem, i per tant, les haurem d'incloure a les instruccions d'instal·lació que afegirem al fitxer README.md del nostre projecte, seguint les pautes habituals d'un projecte a Github.

4.2. Creació del projecte

Per a crear l'estructura del projecte d'extensió, activem l'entorn virtual de CKAN i executem la següent comanda paster[5]:

```
paster --plugin=ckan create -t ckanext ckanext-qadashboard
```

El wizard de creació del projecte ens demana la següent informació per tal de completar la informació bàsica del projecte:

- Descripció
- Etiquetes
- Nom de l'autor
- E-mail de l'autor
- Compte de Github de l'autor

Una vegada hem proporcionat la informació, es genera l'estructura bàsica del projecte, dins de la qual hem de tenir especialment en compte els següents arxius i carpetes:

- ckanext
  - qadashboard
    - fanstatic: conté els arxius CSS i JavaScript

[5]https://docs.ckan.org/en/latest/extensions/tutorial.html#creating-a-new-extension

        - i18n: conté els arxius .po i .mo per a les claus localitzades per idioma.
        - plugin.py: classe controladora del plugin inicial, QadashboardPlugin, es genera junt amb el projecte
        - public: conté els arxius que hagin d'incloure's a una ruta publica del portal, per exemple, les imatges.
        - templates: conté les plantilles HTML codificades amb Jinja2
        - tests: conté les classes amb la definició dels tests per a l'extensió
    - ckanext_qadashboard.egg-info: conté les metadades del projecte, les que hem emplenat en executar la comanda paster.
    - dev-requirements.txt
    - requirements.txt: defineix els requisits de l'extensió per instal·lar-los mitjancant pip
    - setup.py: conté la configuració bàsica de l'extensió

El projecte generat s'ha pujat a un repositori Github, on es pujaran els canvis al llarg del projecte per tal de mantenir un control de versions: https://github.com/ammendoza/ckanext-qadashboard

Com que en el moment de crear el projecte s'ha afegit un sol plugin, generem un altre controlador Dins d'aquest projecte ja podem crear les classes per a cadascun dels plugins que desenvoluparem dins de aquesta extensió, per a fer-ho, creem un nou fitxer python dins de ckanext-qadashboard/ckanext/qadashboard amb la classe per al subsistema de notificació de problemes, que també ha d'estendre de la classe ckan.plugins.SingletonPlugin:

```
import ckan.plugins as plugins
import ckan.plugins.toolkit as toolkit

class NotifyProblemsPlugin(plugins.SingletonPlugin):
    plugins.implements(plugins.IConfigurer)

    # IConfigurer
    def update_config(self, config_):
        toolkit.add_template_directory(config_, 'templates')
        toolkit.add_public_directory(config_, 'public')
        toolkit.add_resource('fanstatic', 'qadashboard')
```

Per tal de que la extensió conegui l'existència d'aquest nou plugin, hem d'afegir la classe dins del fitxer *setup.py* a la propietat *entry_points*:

```
entry_points='''
        [ckan.plugins]
          qadashboard=ckanext.qadashboard.plugin:QadashboardPlugin
```

```
        notifyproblems=ckanext.qadashboard.plugin_notify:NotifyProblems
Plugin
```

4.3. Extractes del codi

Dins de les classes principals dels plugins d'aquesta extensió es defineixen les noves rutes que habiliten aquests plugins i que s'afegeixen a les rutes definides al core de CKAN i la resta de plugins instal·lats. Es tracta d'un mapeig entre les URLs a les que accedeixen els usuaris i els mètodes concrets dels controladors que s'han d'ocupar de servir les vistes demanades.

```
def before_map(self, map):

    map.connect(
        'dataset_problems',
        '/dataset/problems/{package_id}',

controller='ckanext.qadashboard.problem_controller:ProblemController',
            action='view_dataset'
    )

    [...]

    return map
```

Dins del mètode del controlador, es realitzen les comprovacions de seguretat i tota la lògica necessària que s'ha de passar a la vista. Finalment, s'indica que es renderitzi la pàgina i s'indica quina és la plantilla Jinja2 que s'ha d'utilitzar per a la vista.

```
def view_dataset(self, package_id):

    self.__check_permissions__(package_id)

    problems = _model.Problem.by_package(package_id=c.pkg.id)

    return toolkit.render('package/problems.html', extra_vars={
           'problems': problems
    })
```

La capa de model d'aquesta extensió s'implementa amb SQLAlchemy, que proporciona un a capa d'abstracció sobre la base de dades, ja que es tracta d'una llibreria d'ORM. Tota la implementació del model es troba al fitxer model.py, un exemple de definició d'entitat amb SQLAlchemy és la que es fa per a la taula de problemes:

```
problem_table = Table('problem', meta.metadata,
     Column('id', types.UnicodeText, primary_key=True,
default=_types.make_uuid),
     Column('package_id', types.UnicodeText,
ForeignKey('package.id')),
     Column('title', types.UnicodeText),
     Column('problem_type', types.UnicodeText,
ForeignKey('problem_type.id')),
     Column('current_status', types.UnicodeText, default=Status.OPEN),
```

```
     Column('creator_id', types.UnicodeText, ForeignKey('user.id')),
     Column('date_created', types.DateTime,
default=datetime.datetime.now),
     Column('date_modified', types.DateTime,
default=datetime.datetime.now),
     Column('description', types.UnicodeText))
```

Com es pot observar, es defineixen totes les propietats de les taules com les claus primàries i tipus de columnes i les relacions amb altres entitats mitjançant claus foranies.

Una vegada feta aquesta definició d'entitats, només cal definir un mètode d'inicialització que cridi al mètode create_all i inserir les dades inicials:

```
def init_tables():
    meta.metadata.create_all(model.meta.engine)
    model.Session.add(ProblemType(name='Broken link'))
    model.Session.add(ProblemType(name='Wrong data'))
    model.Session.add(ProblemType(name='Insufficient information'))
    model.Session.add(ProblemType(name='Other'))
    model.Session.commit()
```

Per a cridar a aquest mètode, s'ha creat la classe *QADashboardCommands* que implementa la classe *ckan.plugins.toolkit.CkanCommand* i que permet definir mètodes per a la seva execució des de comandes shell mitjançant la comanda *paster* de CKAN. Es defineix en primer lloc un mètode *command,* que s'encarrega de rebre els paràmetres de la comanda i cridar al mètode de la mateixa classe adient en cada cas, i en segon lloc, tants mètodes com comandes es volen implementar. En el nostre cas, només implementem una comanda, *initdb*, que crida al mètode d'inicialització de les taules que hem definit a la classe del model.

```
from ckanext.qadashboard import model as plugin_model
import ckan.plugins as plugins
import logging

log = logging.getLogger(__name__)

class QADashboardCommands (plugins.toolkit.CkanCommand):

    '''
    Creates needed plugin tables
    Use:
        paster --plugin=ckanext-qadashboard qadashboard initdb -c
/etc/ckan/default/production.ini
    '''

    summary = __doc__.split('\n')[0]
    usage = __doc__
    min_args = 0
    max_args = 2


    def command(self):
        if not self.args or self.args[0] in ['--help', '-h', 'help']:
            print self.usage
            sys.exit(1)
```

```
        cmd = self.args[0]
        self._load_config()

        if cmd == 'initdb':
            self.initdb()
        else:
            self.log.error('Command %s not recognized' % (cmd,))


    def initdb(self):

        plugin_model.init_tables()
        print('All tables created.')
```

4.4. Seguretat

A tots els mètodes dels controladors s'inclouen comprovacions de permisos per tal de que un usuari que no tingui els permisos adients pugui accedir o actualitzar dades que queden fora del seu abast.

Per exemple, en el cas dels mètodes del controlador del subsistema de notificació d'errors, es comprova que l'usuari tant que l'usuari té permisos per a a veure el dataset com que sigui un usuari identificat, ja que els usuaris que no estan identificat també poden veure els datasets que estan publicats:

```
def __check_permissions__(self, package_id=None):

    if package_id:
        context = {'model': model, 'session': model.Session,
                'user': c.user, 'for_view': True,
                'auth_user_obj': c.userobj}
        data_dict = {'id': package_id}
        try:
           c.pkg_dict = logic.get_action('package_show')(context,
data_dict)
           c.pkg = context['package']
           dataset_type = c.pkg_dict['type'] or 'dataset'
        except logic.NotFound:
            abort(404, _('Dataset not found'))
        except logic.NotAuthorized:
            abort(403, _('Unauthorized to read dataset %s') % id)

     if not c.userobj:
        abort(403, _('Unauthorized to read problems'))
```

Per altra banda, la utilització de llibreries comuns en entorns Python, com SQLAlchemy, també proporciona una capa de seguretat addicional, ja que inclouen mecanismes de seguretat per evitar, per exemple, la injecció SQL.

4.5. Tests

S'han realitzat diferents proves funcionals per a comprovar que el comportament del desenvolupament és l'esperat amb diferents jocs de dades i perfils amb la definició dels casos de prova que es detallen a continuació.

CP01.01	Accés al quadre de comandaments com a usuari identificat
Resultat esperat	Es mostra el quadre de comandaments sense errors.
Resultat obtingut	OK, es mostra el quadre de comandaments sense errors.

CP01.02	Accés al quadre de comandaments com a visitant
Resultat esperat	Es mostra un missatge d'error a l'usuari indicant que no té permisos per veure la pàgina.
Resultat obtingut	OK, es redirigeix a l'usuari a la pàgina d'identificació i es mostra un missatge d'error a l'usuari indicant que no té permisos per veure la pàgina.

CP01.03	La gràfica de nivells de qualitat mostra valors de 0 a 5
Resultat esperat	La gràfica de nivells de qualitat només inclou el rang de valors de 0 a 5 i no té decimals.
Resultat obtingut	OK, la gràfica no mostra altres valors com None o 1.5

CP01.04	Els datasets amb baixa qualitat tenen valors de 0 a 3
Resultat esperat	Dins de la llista de datasets amb un baix nivell de qualitat, només s'han d'incloure els que tinguin nivells entre 0 i 3.
Resultat obtingut	OK, no es mostren datasets amb nivells 4 o 5.

CP01.05	Els datasets amb baixa qualitat estan ordenats per nivell ascendent
Resultat esperat	Dins de la llista de datasets amb un baix nivell de qualitat, aquests s'ordenen per nivell ascendent.
Resultat obtingut	OK, l'ordre és sempre ascendent.

CP01.06	El llistat de problemes reportats té limit
Resultat esperat	El llistat d'últims problemes reportats als meus datasets ha de mostrar com a màxim 5 problemes.
Resultat obtingut	OK, es mostren com a màxim 5

CP01.07	La gràfica de vistes mostra un valor per a cada dia
Resultat esperat	La gràfica de vistes ha de mostrar un valor per a cada dia de l'última setmana, encara que un d'ells no tingui cap vista

Resultat obtingut	OK, es mostren tots els dies i alguns amb valors 0

CP01.08	**La mitjana de qualitat correspon a la mitjana dels valors que apareixen a la gràfica de nivells de qualitat**
Resultat esperat	La mitjana de qualitat que es mostra a la taula de mitjanes correspon a la mitjana dels nivells de qualitat que apareixen a la gràfica.
Resultat obtingut	OK, els valors sí que corresponen

CP01.09	**No es mostren les mitjanes individuals als administradors**
Resultat esperat	La taula de resum de mitjanes no es mostra si l'usuari es sysadmin o pot administrar tots els datasets.
Resultat obtingut	OK, no es mostra si l'usuari pot editar tots els datasets del site

CP02.01	**Els nous problemes es creen amb estat «Obert»**
Resultat esperat	Quan es crea un nou problema, aquest apareix sempre amb estat «Obert»
Resultat obtingut	OK, quan es creen sempre tenen estat «Obert»

CP02.02	**Els visitants no poden veure el formulari de nou problema**
Resultat esperat	Si un usuari que no està identificat tracta d'accedir a la pàgina per a afegir un nou problema, es mostra un missatge indicant que no té permisos.
Resultat obtingut	OK, es mostra un error 403 (forbidden)

CP02.03	**No es pot afegir un problema sense informació**
Resultat esperat	Un usuari identificat no pot afegir un nou problema deixant en blanc el camp de títol o descripció.
Resultat obtingut	OK, es mostra un missatge d'error si el camp títol o descripció no s'ha emplenat.

CP02.04	**El problema canvia d'estat**
Resultat esperat	Quan un editor canvia l'estat d'un problema, aquest es mostra sempre amb el nou estat.
Resultat obtingut	OK, quan es canvia es veu el seu nou estat a la pàgina de detall del problema i a tots els llistats.

CP02.05	Llistat d'actualitzacions del problema
Resultat esperat	Quan un problema té actualitzacions, es poden veure totes a la pàgina de detall de problema.
Resultat obtingut	OK, de cada actualització es veu: canvi d'estat, usuari i descripció.

CP02.06	Els visitants no poden veure la pàgina de detall d'un problema
Resultat esperat	Si un usuari que no està identificat tracta d'accedir a la pàgina de detall de problema, es mostra un missatge indicant que no té permisos.
Resultat obtingut	OK, es mostra un error 403 (forbidden)

CP02.07	Els visitants no poden veure la pàgina de llistat de problemes d'un dataset
Resultat esperat	Si un usuari que no està identificat tracta d'accedir a la pàgina de problemes d'un dataset, es mostra un missatge indicant que no té permisos.
Resultat obtingut	OK, es mostra un error 403 (forbidden)

CP02.08	Els visitants no poden veure la pàgina de llistat de problemes dels meus datasets
Resultat esperat	Si un usuari que no està identificat tracta d'accedir a la pàgina de problemes d'un dataset, es mostra un missatge indicant que no té permisos.
Resultat obtingut	OK, es redirigeix a la pàgina d'identificació i es mostra un missatge d'error.

Per a comprovar que s'inclouen cassos de prova per a cadascuna de les funcionalitats que s'han desenvolupat (fins ara, les que s'inclouen al MVP), es completa la següent matriu de correspondència.

	H01.00	H01.01	H01.02	H01.03	H01.04	H01.05	H01.06	H01.07	H02.00	H02.02	H02.04	H02.05	H02.06
CP01.01	x												
CP01.02	x												
CP01.03		x											
CP01.04			x										

CP01.05			x										
CP01.06				x									
CP01.07					x								
CP01.08						x							
CP01.09						x	x	x					
CP02.01									x				
CP02.02									x				
CP02.03									x				
CP02.04										x			
CP02.05											x		
CP02.06											x		
CP02.07												x	
CP02.08													x

4.6. Versions de l'aplicació

S'han generat les següents versions de llançament de l'extensió, etiquetades al seu repositori de Github:
https://github.com/ammendoza/ckanext-qadashboard/releases

- 1.0: inclou la primera versió dels requisits funcionals que s'inclouen al MVP.
- 1.1: afegeix la funcionalitat de filtrat per tipus de problema als llistats de problemes.

4.7. Documentació del projecte

La documentació de l'extensió desenvolupada, que inclou els seus requisits, instruccions d'instal·lació i el manual d'usuari, es troba al fitxer readme.rst de l'arrel de projecte i que es pot veure a la pàgina principal del repositori a Github: https://github.com/ammendoza/ckanext-qadashboard

4.8. Bugs

No s'han detectat bugs en aquest punt del desenvolupament.

4.9. Possibles millores

A més a més de la implementació de la resta de funcionalitats descrites al disseny de l'aplicació que no corresponen al MVP, es podria millorar la validació

dels paràmetres GET i POST per assegurar-se de que no es puguin explotar vulnerabilitats com, per exemple, afegir codi JavaScript dins dels camps de text.

També caldria ampliar el joc de proves per tal de donar una major cobertura al comportament esperat i incloure tests unitaris que es puguin executar de manera automàtica en un sistema d'integració continua.

5. Conclusions

En general, considero que s'han complert els objectius en quant a que s'ha aconseguit implementar el producte viable mínim a més a més d'alguna funcionalitat addicional complint la planificació proposta inicialment, encara que m'hauria agradat poder implementar algunes funcionalitats més que s'havien plantejat al disseny de l'extensió.

Quant a les lliçons apreses i que poden servir per a futurs projectes, he trobat que una de les principals dificultats d'afrontar un projecte com aquest que es basa en un software ja existent és que requereix conèixer les tecnologies que utilitza, que sovent son moltes i aquestes no sempre estan documentades adientment. Cal acostumar-se a llegir el codi per tal d'entendre com s'han d'utilitzar o quin es el seu comportament.

Per altra banda, i es un punt comú a qualsevol projecte web, també s'han de tenir coneixements de les aplicacions de tercers necessàries al seu funcionament com els servidors web, servidors d'aplicacions, servidors de BBDD i de indexació. S'han de saber també treballar amb diferents sistemes operatius sense utilitzar l'entorn gràfic, ja que als servidors reals es tracten d'aprofitar al màxim els recursos del sistema.

El preparar l'entorn de desenvolupament, instal·lar i configurar tot el software necessari és un punt prou important d'un projecte d'aquest tipus; cal incloure a la planificació de qualsevol projecte de desenvolupament el temps necessari per a fer-ho correctament i poder comprovar que tot funciona com s'espera per tal de fiançar la base sobre la que desenvoluparem i no trobar problemes inesperats durant les fases posteriors.

6. Glossari

- CKAN: software per a la creació de portals de dades obertes de codi obert i basat en Python.
- *Dataset*: conjunt de dades o col·lecció de dades. Es pot assimilar a una taula de base de dades.
- Recurs: fitxer en un format concret que es publica dins d'un dataset. Un mateix *dataset* el poden comprendre molts recursos amb formats diferents o agrupacions de les dades (per anys, per organismes...)

7. Bibliografia

1. Berners-Lee, T. *Linked Data – Design Issues* [en línia]. 27 juliol 2006. <https://www.w3.org/DesignIssues/LinkedData.html> [Data de consulta: 30 setembre 2018]