Incorporació de funcionalitats a la xarxa social kPAX : *autenticació amb UOC, Facebook, Twitter, Google+*

Carlos Da Costa
Màster en Programari Lliure - PFM
Administració Web i Comerç electronic

Consultor : *Daniel Riera Terrén*
Tutor : *Francisco Javier Noguera Otero*
4 de Desembre 2016

Resum de Projecte

L'objectiu general d'aquest projecte és la incorporació de funcionalitats a la xarxa social kPAX. I més concretament, la possibilitat de fer l'autenticació dels usuaris mitjançant les seves credencials de la UOC i de les xarxes socials Facebook, Twitter, google+.

kPAX[1] és una xarxa social basada amb ELGG[2], per aprenentatge mitjançant jocs educatius, creada per la UOC l'any 2011.

Té una arquitectura nucli basada en serveis que permet desenvolupar jocs i aplicacions externes que interactuïn amb el nucli. kPAX2, actualment està migrat a ELGG 2.x en la part de *FrontEnd* la qual suporta la xarxa social i es comunica amb el nucli a través de crides a serveis web.

En l'actualitat encara no s'han desenvolupat totes les funcionalitats necessàries, i en aquest projecte s'ha fet el desenvolupament d'una d'elles, la possibilitat d'autenticació en plataformes d'un usuari de la UOC, o de Facebook, Twitter, Google+. És a dir, fer un connector que permeti el registre i la registre i validació dels usuaris per la UOC, Facebook[3], Twitter[4], i Google+[5] a través d'OAuth[6] en un entorn NODE.JS[7] i un FrontEnd ELGG. Concretament, volem afegir dues noves funcionalitats:

- Desenvolupar un nou connector per permetre l'autenticació d'usuaris fent servir la xarxa de la UOC, la de Facebook, Twitter, i Google+.
- Utilitzar eines d'autenticació com OAuth, passaport, amb els servidors externs a través d'un nou servidor NodeJs.

Project Overview

The overall objective of this project is the incorporation of social network features kPAX. And more specifically, the possibility of user authentication using their credentials at the UOC and the social networks Facebook, Twitter, google +.

kPAX[1] is a social network based with ELGG[2] for learning through educational games created by the UOC in 2011.
It has a core architecture based on services and external applications to develop games that interact with the kernel. kPAX2 currently is migrated ELGG 2.2.2 on the frontend that supports the network and communicates with the core through Web services calls.

Now I have not developed all the functionality needed, and this project has been the development of one of them, the possibility of a user authentication platforms UOC or Facebook , Twitter, google +. That is, make a plugin that allows the registration and validation of users validated by the UOC Facebook[3] , Twitter[4] and Google+[5] through OAuth2[6] in Node.js[7] environment and ELGG frontend module with KAPX.
Specifically, we added two new features:

- Develop a new plugin to allow user authentication using the network of UOC, that of Facebook, Twitter, and Google+.
- Use tools such as OAuth authentication, passport with external servers through new server NodeJs.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

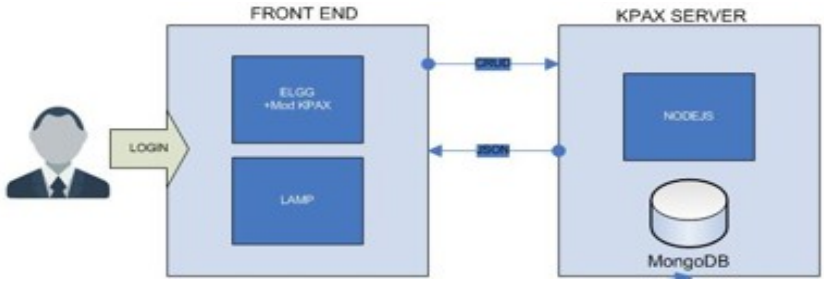
kPAX és una xarxa social basada amb ELGG, per aprenentatge mitjançant jocs educatius, creada per la UOC.
Actualment només existeix l'autenticació local al servidor ELGG + kPAX.
kPAX està format per una part servidora y altre client.
El servidor estava fet en JEE però s'ha canviat a Nodejs i les crides s'han adaptat a REST.
Com a novetat, a part de tenir la possibilitat d'entrar amb l'autenticació local, ho podrem fer amb l'usuari i contrasenya de la UOC, de Facebook, de Twitter o amb google+.

Per a poder realizar aquest projecte, és necessari modificar el codi en la que surt el formulari d'accés, i fer tot el codi a la part servidor que faci la comunicació entre el formulari PHP, el propi servidor, i els servidors de les xarxes socials amb OAuth2.

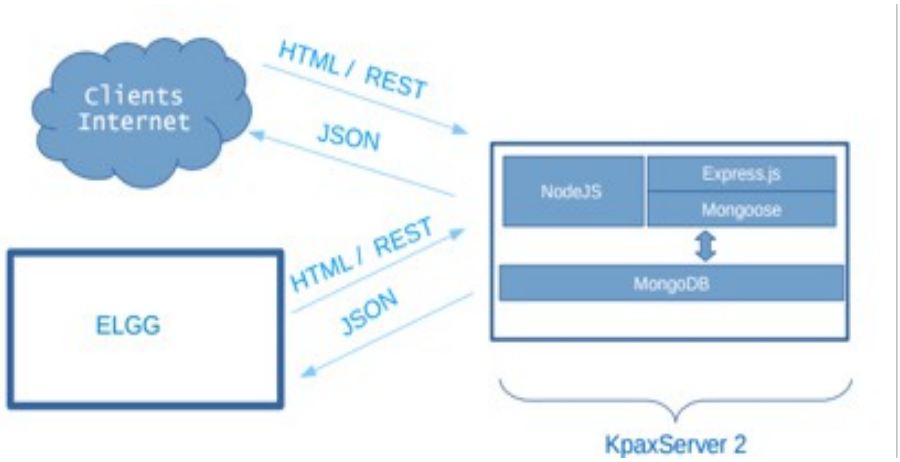
1.2 Objectius del Treball

L'objectiu específic del projecte és crear un connector que permetrà delegar l'autenticació d'usuaris de la plataforma kPAX a través de la UOC, i les xarxes socials com Facebook,Twitter,i Google+ sense haver de donar-se d'alta a kPAX2

L'arquitectura actual és la següent, sense comptar amb la part dels servidors externs (servidors de Twitter, Google...).



Il·lustració 1: Figura1



1.3 Enfocament i mètode seguit

Sabent que el producte kPAX ja existeix i està en funcionament, l'estratègia a seguir és la d'adaptar el producte existent.

Primer de tot s'ha d'actualitzar algunes parts, com per exemple, el ELGG a la nova versió, i instal·lar el servidor kPAXServer.

A partir d'aquí, s'han de fer proves de funcionament, com connectar a la Base de Dades, proves d'autenticació, exemples de OAuth, etc etc.

1.4 Planificació del Treball

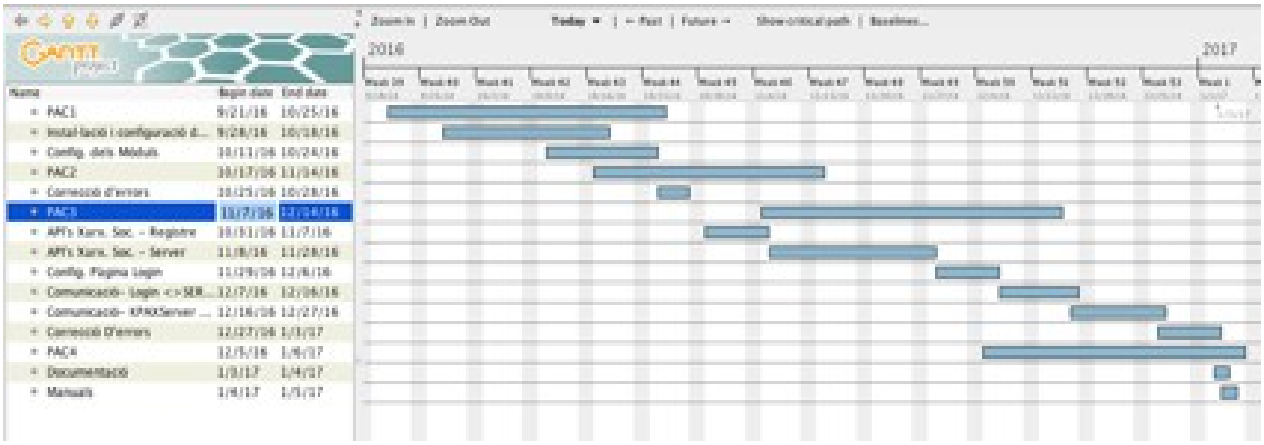
Planificació a seguir :

- Com es realitzarà ?
    - Instal·lació i configuració del servidor
    - Configuració dels mòduls
    - Avaluació, i correcció d'errors
    - Desenvolupament de la part servidor que fa les comunicacions
    - Configuració i presentació pàgina de Accés
    - Configuració de les API's dels servidors

- Desenvolupament de tota la comunicació entre els diferents mòduls
- Avaluació, i correcció d'errors

- Quines eines ?
  - Express /Node.Js
  - Vim per editar el codi
  - Robomongo per gestionar la Base de dades
- Estàndards
  - PDF, DOCX, diagràmes de disseny (UML)...
- Metodologia
  - Trello : eina de seguiment
  - Github local/web : Gestió de codi i versions

Planificació temporal del projecte (Figura3)



Il·lustració 3: Figura3 (Diagrama de Gannt)

1.5 Aspectes rellevants :

Pàgina login

La pàgina de *login* mostra diferents botons que permeten a l'usuari decidir si vol accedir de forma local o mitjançant una de les xarxes socials integrades a kPAX. El llenguatge de programació es PHP. S'utilitzarà CURL i s'enviarà amb HTML/REST.

API's xarxes socials

Pel que fa a les API's, farem les gestions oportunes en cada pàgina (UOC, Facebook, Twitter, google+) per generar i obtenir les claus, ID's i vincles pel tema d'accés.

Autentificacions OAuth

En aquest apartat, crearem la pàgina per fer totes les autentificacions i un altre per guardar les dades dels connectors. S'utilitzarà OAuth per l'autentificació, CURL per contestar les peticions amb JSON i el servidor respondrà amb HTML/REST.

Connexió Base de dades

En aquest apartat, crearem els esquemes per la base de dades dels usuaris, ja que es guardaran les dades dels usuaris. Es guarden per poder consultar si l'usuari està donat d'alta o no.

2. Estudi de Viabilitat

2.1- Consideracions Prèvies

L'estudi de viabilitat consisteix a saber si un projecte és raonable i funciona bé, és a dir la condició d'un camí que està fet però s'ha de saber si es pot transitar o no.

Un treball pot estar ben fet, però potser no és viable. Per això, quan fem l'estudi de viabilitat ens centrarem en mostrar els recursos disponibles, l'accessibilitat dels recursos que s'utilitzaran, i el que es vol realitzar com a projecte. Sempre tenint molt clar el que volem mostrar.

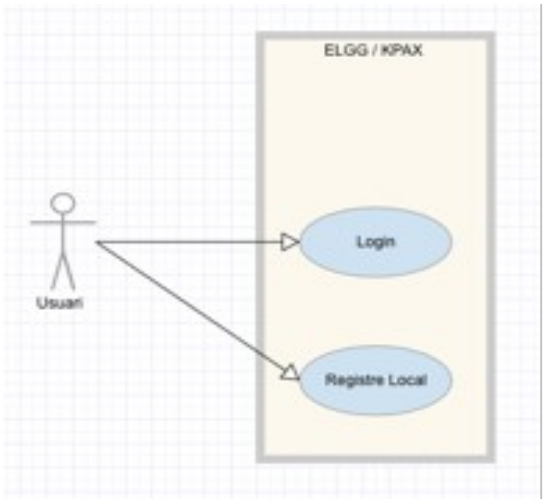
Ja que és un projecte de modificació, ja sabem que és raonable, i funciona bé. El que farem és actualitzar i millorar el sistema d'accés, és a dir solament farem millores.

2.2- Necessitats i Requisits

En aquest apartat, i tenint en compte que ja sabem el que farem, és captar les necessitats i els Requisits del sistema.
Requisits: fa falta que el sistema actual de kPAX, es modifiqui i s'actualitzi l'apartat d'autentificació amb el nou servidor (Nodejs + Mongodb) amb la UOC i les xarxes socials.

2.3- Situació Actual

En aquest apartat, hem d'identificar el sistema que es vol fer, i els usuaris implicats en el sistema.
En la figura posterior, es pot veure la situació actual.
Un usuari pot entrar si ja està registrat o bé, registrar-se localment.



Il·lustració 4: Figura 4

- El Sistema web està implementat amb kPAX2
- Es un model Client-Servidor basat amb ELGG
- El servidor en el que està implementat ELGG es LAMP[10]
- El servidor que farà les conexions cap a fora es Node.Js i MongoDB.
- La capa de presentació està desenvolupada amb PHP[11], HTML[12], Javascript[13] i CSS[14].
- La capa de servidor està desenvolupada amb Express, passport i OAuth.

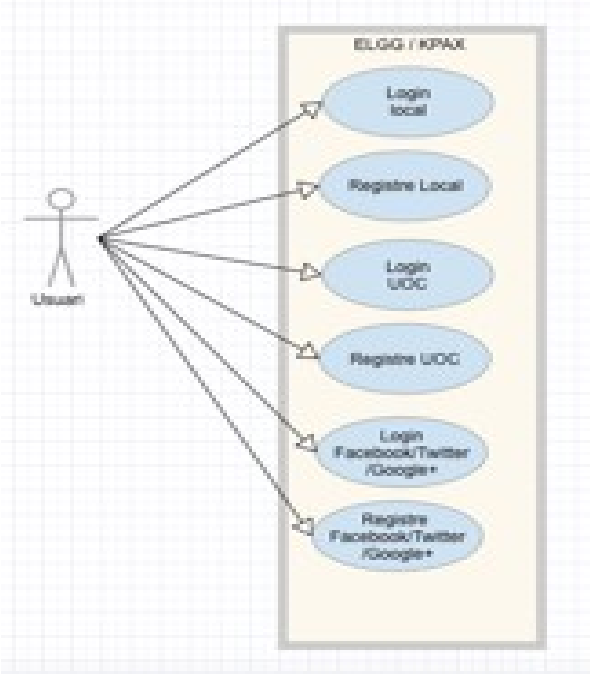
2.4- Requisits del Sistema

En aquest apartat, es descriuen els requisits del sistema a partir de la situació actual. Els requisits serien els de la *Figura 5*, és a dir, poder fer *login* o registrar-nos.

Volem que, en l'apartat de client, hi hagui un vincle o botó que ens faciliti l'accés a les xarxes socials. I per la part de servidor, volem que aquest accés es faci amb NodeJS i autenticació OAuth.

Volem :

- Un connector en PHP que ens permeti l'autenticació a les xarxes socials.
- Utilitzar Node.Js amb Passport i OAuth per la comunicació entre el nostre servidor i els servidors de les xarxes socials.
- Desenvolupar una interfície d'usuari en HTML i CSS a la qual afegirem una serie de botons en la mateixa pantalla d'autenticació.



Il·lustració 5: Figura5

2.5- Estudi de les Alternatives

Per poder realitzar l'estudi de viabilitat, fa falta que estudiem les alternatives que hi ha al sistema que volem realitzar. Per aquest motiu i perquè tinguem clar que la nostra solució és la millor, haurem de fer una comparativa amb els següents aspectes:

1-Microsoft Windows + aplicació propietària: en aquest cas, el sistema operatiu del servidor web serà Microsoft Windows 2016, i la gestió de continguts es farà mitjançant l'adquisició d'un programa específic, com Sharepoint i IIS per posar l'accés Web.

El programa de gestió de continguts triat compleix els requisits funcionals i tècnics definits en aquest sentit. Pel que fa als requisits legals i econòmics de la solució proposada, el sistema operatiu no compleix el que s'ha expressat, i el programa de gestió de continguts només ho fa en part. S'hauria de refer tot el sistema de jocs tot de nou, i no és el que volem.

2-Ubuntu/Linux + aplicació propietària: en aquest cas, el sistema operatiu del servidor web seria Ubuntu/Linux (en qualsevol de les seves distribucions).

Com a servidor de *FrontEnd*, LAMP (Linux, Apache, MySql, PHP) i com a *BackEnd*, Nodejs + MongoDB.

Els costos d'adquisició imputats a cada una de les solucions són:
Microsoft Windows + Sharepoint = = 500 € + 120 (10€x12) € = 620 €.
Ubuntu/Linux + aplicació lliure = = 0 € + 0 € = 0 €.

Veien que solament amb el sistema Operatiu ja hi ha diferencia en el que fa el Cost, ja podem decantar-nos per la segona opció. A part, el sistema actual ja és amb Linux.

2.6- Selecció i Possible Solució

Com ja hem vist en l'apartat anterior, la solució més viable és la que actualment està implantada, i per tant ens adaptarem al sistema implantat.
Muntarem el servidor de NodeJS i executarem l'aplicació per escoltar les peticions que ens arribin d'ELGG. Aquesta aplicació farà la comunicació amb els servidors que no formen part d'ELGG (UOC, Facebook, Twitter, Google).
Una vegada tingui aquesta part solucionada i funcionant, montarem un connector al servidor ELGG amb la interfície per la part del client, i la comunicació amb el servidor NodeJS.

3. Anàlisi

Després de seleccionar la millor solució, hem de fer l'anàlisi del sistema; Per tant s'ha de fer la descripció, i determinar la comunicació entre sistemes existents i els seus requisits exactes.

3.1- Anàlisi de Viabilitat

Requisits :

- Canviar el sistema d'accés actual (actualment solament en local)
- Continuar amb el mateix format (CSS)
- Adaptar el sistema de *login* perquè la validació la faci NodeJs en lloc de PHP.
- Validar el sistema perquè utilitzem OAuth i Passport
- Utilitzar les dades obtingudes de les xarxes socials i de la UOC.

3.2- Cost

3.2-1. Cost Actual

Actualment el cost del sistema és 0. Ja que el sistema Operatiu és Linux (cost 0), el client LAMP (cost 0), i el servidor també (cost 0).

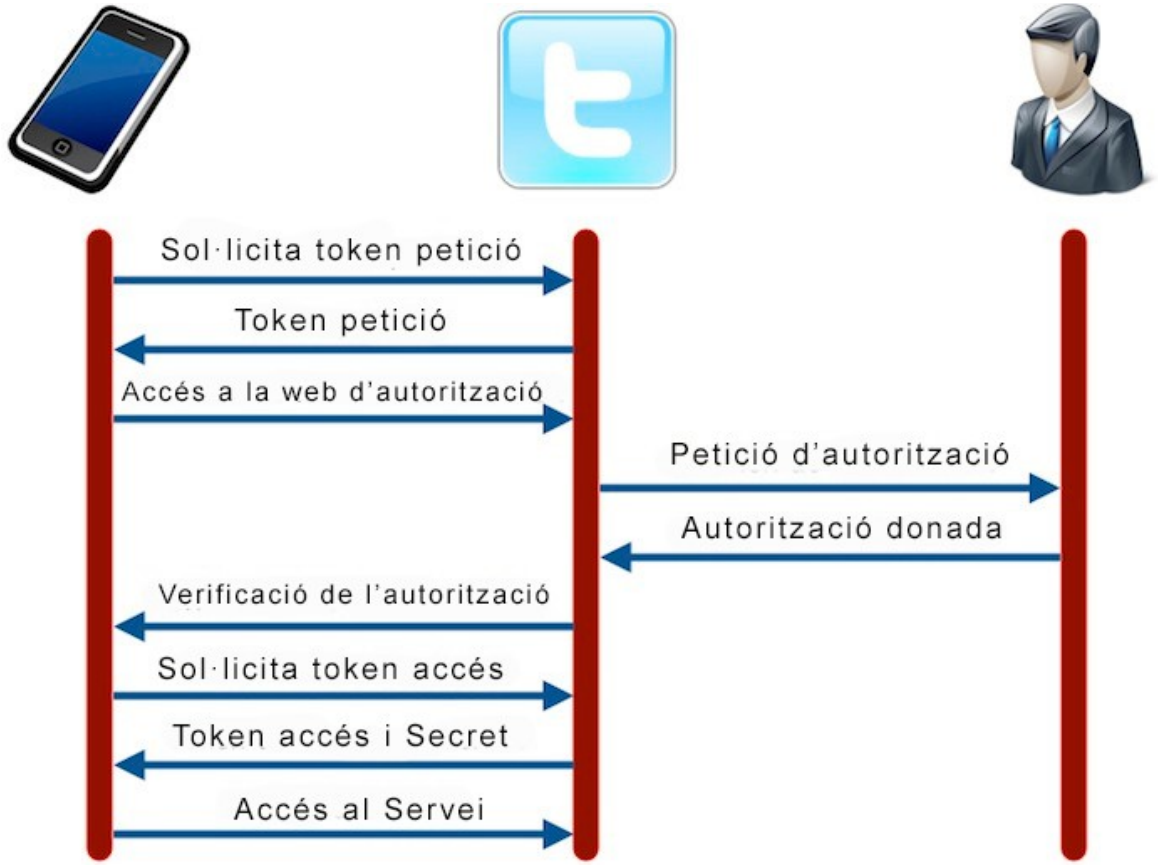
3.2-2. Cost Total

L'únic cost que es podria imputar es el que es passa al fer el desenvolupament, però com és un PFM, no s'imputarà cap cost. El sistema Operatiu es Linux, per lo que el cost es 0. El sistema de contigut es ELGG adaptar amb el mòdul kPAX, i tots dos son a cost 0.

3.2- Oauth

És un protocol que permet fluxos simples d'autorització per a llocs web o aplicacions informàtiques.
OAuth permet a un servidor (exemple: Twitter) compartir la seva informació amb el client (exemple : navegador web) sense compartir tota la seva identitat.

OAuth és un mètode per interactuar amb dades protegides i publicar-les.
OAuth proporciona als usuaris un accés a les seves dades al mateix temps que protegeix les credencials del compte.
Exemple d'ús :



Fem una petició de token al servidor de Twitter, i aquest ens retorna un token, el que ens permetrà verificar que el token eés correcte. Ens demanarà la clau d'accés o token i el secret, i ens donarà accés al servei perquè a posteriori l'usuari pugui accedir amb les seves dades al portal.

3.3 – Passport

Passport és un programa intermitg per NodeJs. Està disenyat per un propòsit únic: per autenticar sol·licituds.
Quan s'escriuen mòduls, Passport delega totes les funcionalitats a l'aplicació.
Tradicionalment, els usuaris es connecten proporcionant un nom d'usuari i una contrasenya.
Amb l'aparició de les xarxes socials, l'inici de sessió únic mitjançant un proveïdor Oauth com Facebook o Twitter s'ha convertit en un mètode d'autenticació molt popular.

Exemple de l'útilització de passport :

```
app.post('/login', passport.authenticate('local', { successRedirect: '/',
                                                    failureRedirect:  '/login'  }));
```

3.3 – NodeJS

NodeJS es un entorn d'execució per JavaScript. Node.js utilitza un model d'operacions Entrada/Sortida sense bloqueig i orientat a esdeveniments, que el fa lleuger i eficient. Creat en un entorn d'execució de JavaScript orientat a esdeveniments asíncrons, Node està dissenyat per construir aplicacions en xarxa escalables.

Les operacions de xarxes basades en fils són relativament ineficients i són molt difícils d'utilitzar. A més, els usuaris de Node estan lliures de preocupacions sobre el bloqueig del procés, ja que no existeix. Gairebé cap funció en Node realitza E/S (Entrada/Sortida) directament, així que el procés mai es bloqueja. A causa que no hi ha bloqueig és molt raonable desenvolupar sistemes escalables en Node.

3.4 – MongoDB

Es una base de dades àgil que permet als esquemes canviar ràpidament quan les aplicacions evolucionen, proporcionant sempre la funcionalitat que els desenvolupadors esperen de les bases de dades tradicionals, tals com índex secundaris, un llenguatge complet de cerques i consistència estricta.

4.Disseny del Sistema

4.1 – Arquitectura

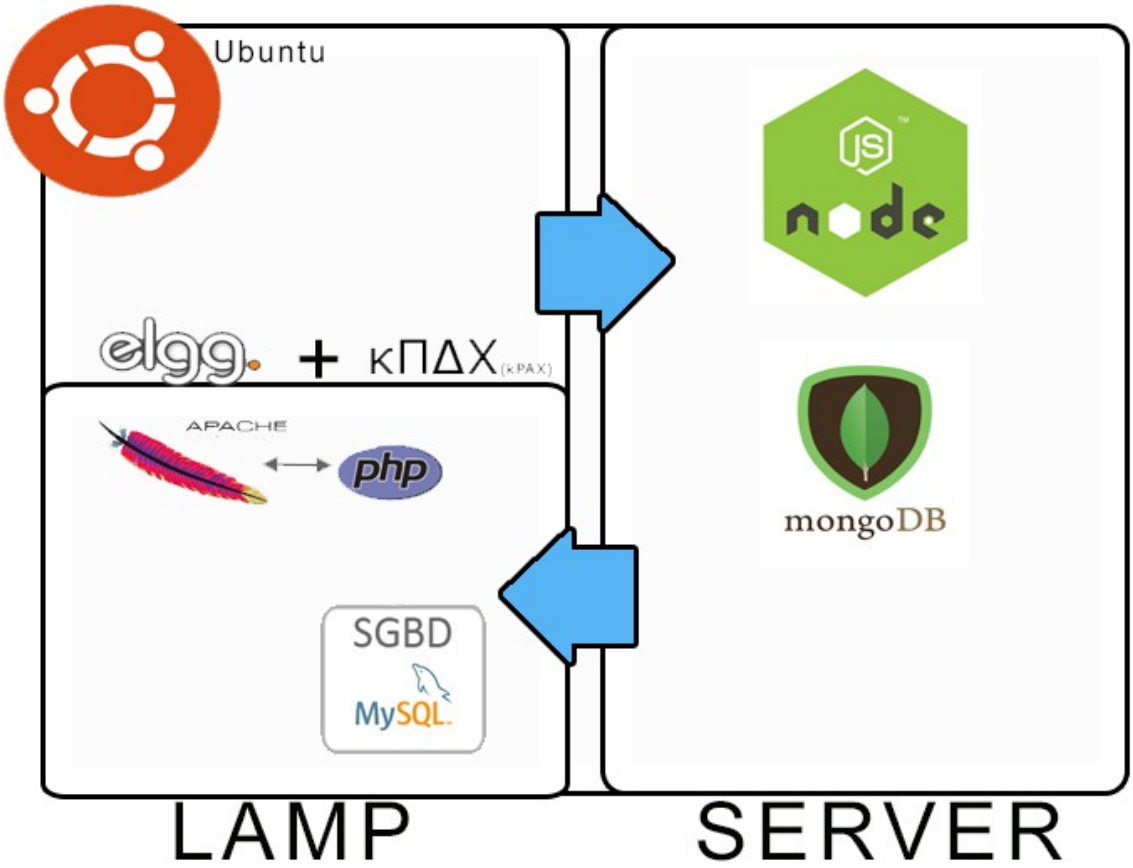
Identificarem els components del sistema amb una serie de diagrames per poder descriure el disseny.

L'arquitectura que volem implementar és, per una banda el Servidor que suportarà tots els serveis, en el nostre cas es Linux (Ubuntu).

Després 'instal·lar el servidor, hem d'instal·lar els serveis LAMP. En el script que ens han passat per instal·lar ELGG + Kpax ja ens instal·la els serveis mencionats.

Per acabar l'arquitectura, hem d'instal·lar el servidor NodeJs i la Base de dades MongoDB.

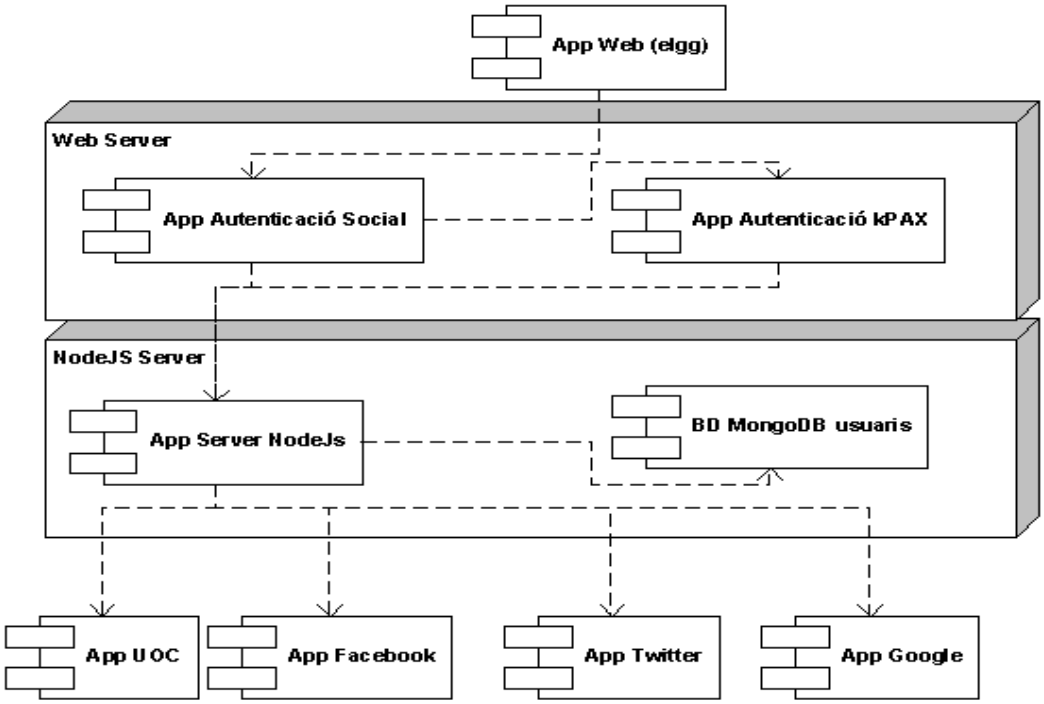
L'arquitectura bàsica del nostre projecte es pot visualitzar en la figura.



En el nostre cas, tindrem 2 servidors en un mateix sistema Operatiu: un servidor LAMP amb el sistema de gestió de continguts ELGG i kPAX, que tindran Mysql com a Base de dades local, i l'altra servidor, NodeJS amb la base de dades MongoDB, per gestionar els accessos, autoritzacions i permisos. Aquest segon guardarà les dades en la Base de dades dels usuaris de les xarxes socials.

4.2 – Revisió de casos d'ús

Utilitzaré diagrames de components i targetes CRC per poder mostrar el nivell d'arquitectura.



Il·lustració 6: Diagrama de components

Desenvoluparé 2 components :

1. App Autenticació Social. Es la part de login en la qual posaré els botons de les xarxes socials i és fa la connexió amb el servidor.

App Autenticació Social	
-Permet l'intercanvi de les dades dels usuaris amb el servidor -Envia les dades dels usuaris kPAX	-Usuaris kPAX -Usuaris xarxes socials

Taula 1: Autenticació 1

2. App Server Node.JS. Es la part de servidor que rep les peticions de login i fa les consultes als servidors de les xarxes socials. Fa també la comunicació amb la Base de Dades per verificar els usuaris i per registrar-ne.

App Server NodeJs	
-Permet la comunicació amb les App de les	-Usuaris kPAX

xarxes socials i obtenir dades dels usuaris. -Permet interactuar amb la base de dades dels usuaris. -Reb les dades d'autenticació de les xarxes socials i les tracta.	-Usuaris xarxes socials -Base de Dades kPAX d'usuaris i de xarxes socials

Taula 2: Autenticació 2

5.Planificació del Sistema i Desenvolupament

5.1- Establiment dels requisits del Sistema

Amb aquesta fase, completarem els requisits vistos en apartats anteriors amb la informació dels usuaris. Es dividiran en subsistemes segons tinguem definit el sistema principal.

Sistema operatiu : ubuntu

kernel : 4.8.0-22

Pel primer servidor ens fan falta els següents complements :

-Apache2

-PHP5

-CURL per PHP

-XML per PHP

Pel segon servidor ens fan falta els següents complements :

-NodeJS (v. 4.2.6)

-MongoDB (v. 2.6.11)

-Express (v. 4.13.4)

-Oauth (v. 1.0.4)

-Passport (v.0.3.2)

-Passport-local (v. 1.0.0)

-Passport per Facebook ,Twitter, google+, passport per CAS

-npm (gestor de paquest per NodeJS)

-body-parser, serve-favicon, cookie-parser : Son middleware que s'han d'instal·lar de nou ja que ja no formen part de Express. Surten com a “deprecated” i no es pot fer servir en versions d'express superiors a la 4.0.

Un altre dels requisits es modificar el comportament d'altres components, ja que en noves versions, canvia el mode de funcionament.

5.2- Nivells d’Arquitectura i Estàndards

Definim les normes i estàndards de disseny. Així identifiquem els components per poder dissenyar el sistema amb més detall.

Hi han 2 tipus d'arquitectura :

- Conceptual : Relacions entre els grans blocs del sistema sense entrar al detall.
- Lògica : detall i definició de les interfícies entre els components.

- Per generar la documentació s'utilitza el format ODT (Open Document Format)
- Per fer els diagrames, utilitzo el format UML.

5.3- Casos d'ús

Aquests son els casos d'ús principals del projecte.

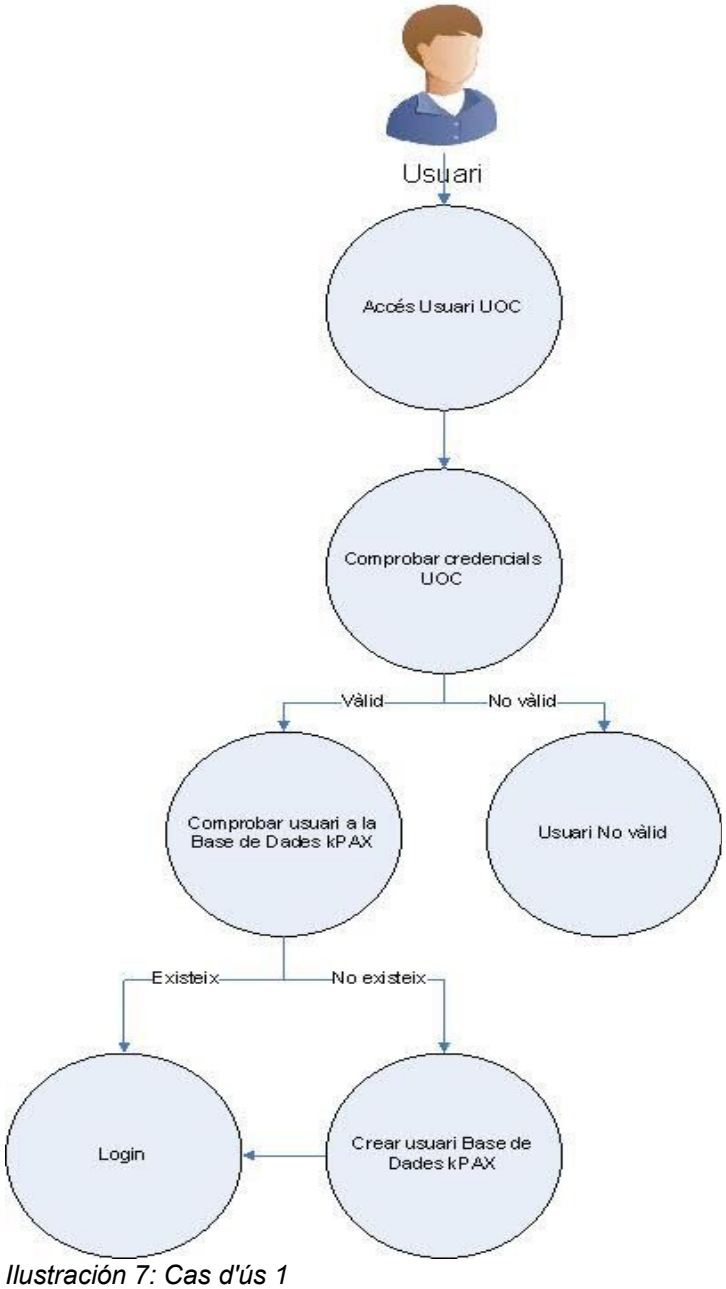


Ilustración 7: Cas d'ús 1

L'usuari demana accés amb el botó de la UOC. Es comproven les credencials de l'usuari al servidor de la UOC. Si és vàlid, és comprova a la Base de dades si l'usuari existeix llavors s'autentifica automàticament. En cas contrari, és dona un formulari perquè es registri i pugui autentificar-se.

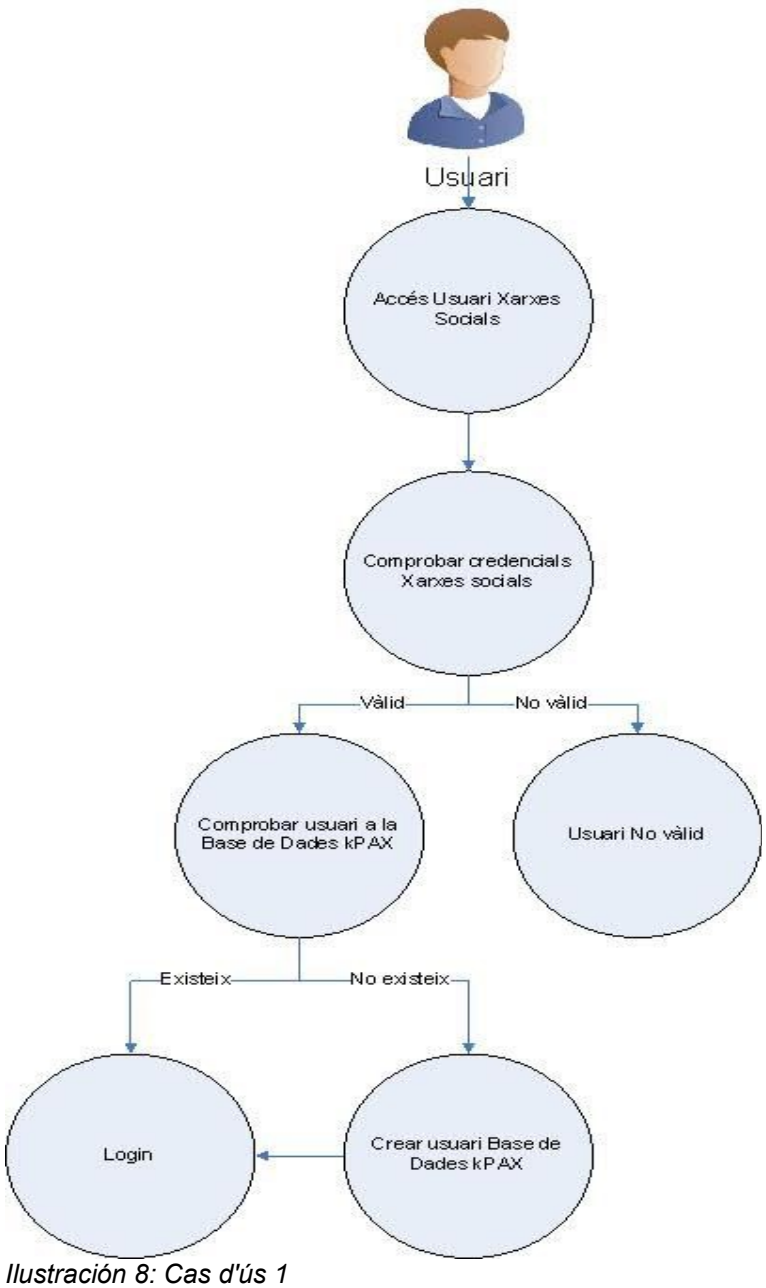


Ilustración 8: Cas d'ús 1

Es el mateix sistema que el de la UOC, però per cada proveïdor de xarxes socials.

En aquest cas, faig un únic diagrama ja que son els mateixos per cada xarxa social.

Taula de condicions Cas d'ús 1 :

Cas d'ús	Comprovació credencials UOC
Descripció	L'usuari s'identifica amb les

	credencials de la UOC
Actors	Usuari
PreCondicions	- Inici de l'accés amb les credencials de la UOC -Ha de tenir un compte de la UOC
PostCondicions	Validació per servidor de la UOC i accés a les dades

Taula 3: Cas d'ús 1

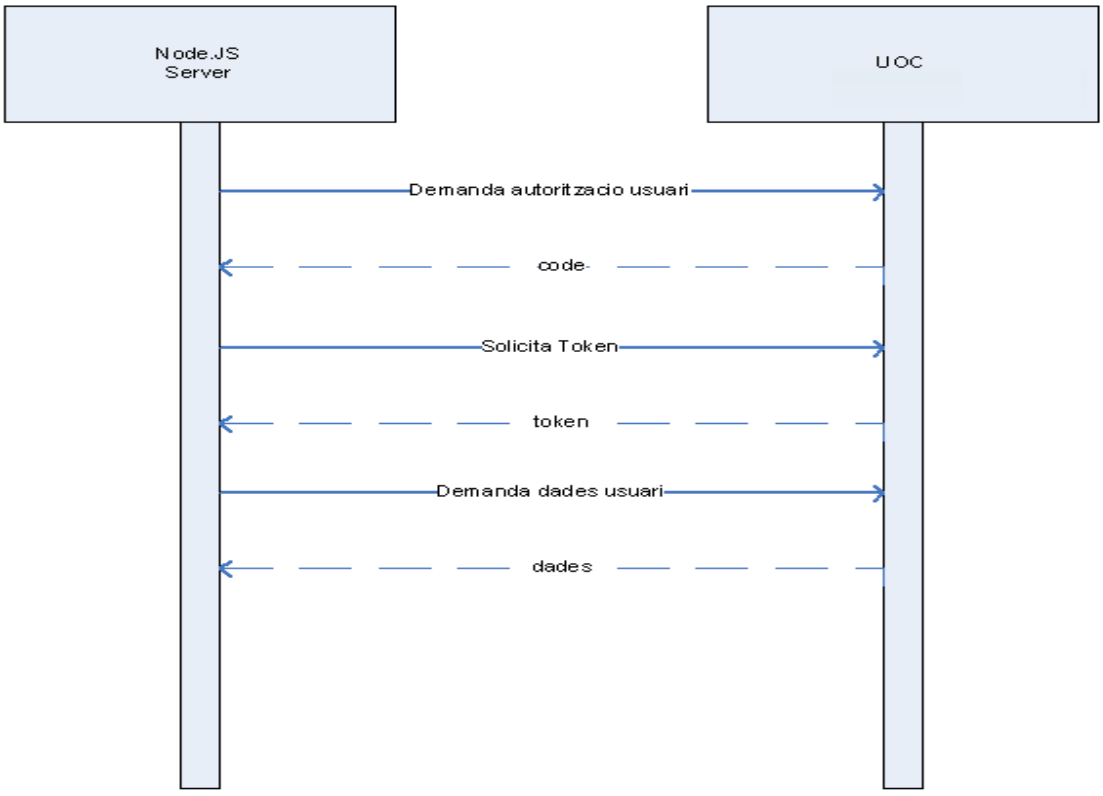
Taula de condicions Cas d'ús 2 :

En aquest cas, faig una única taula ja que son les mateixes per cada xarxa social.

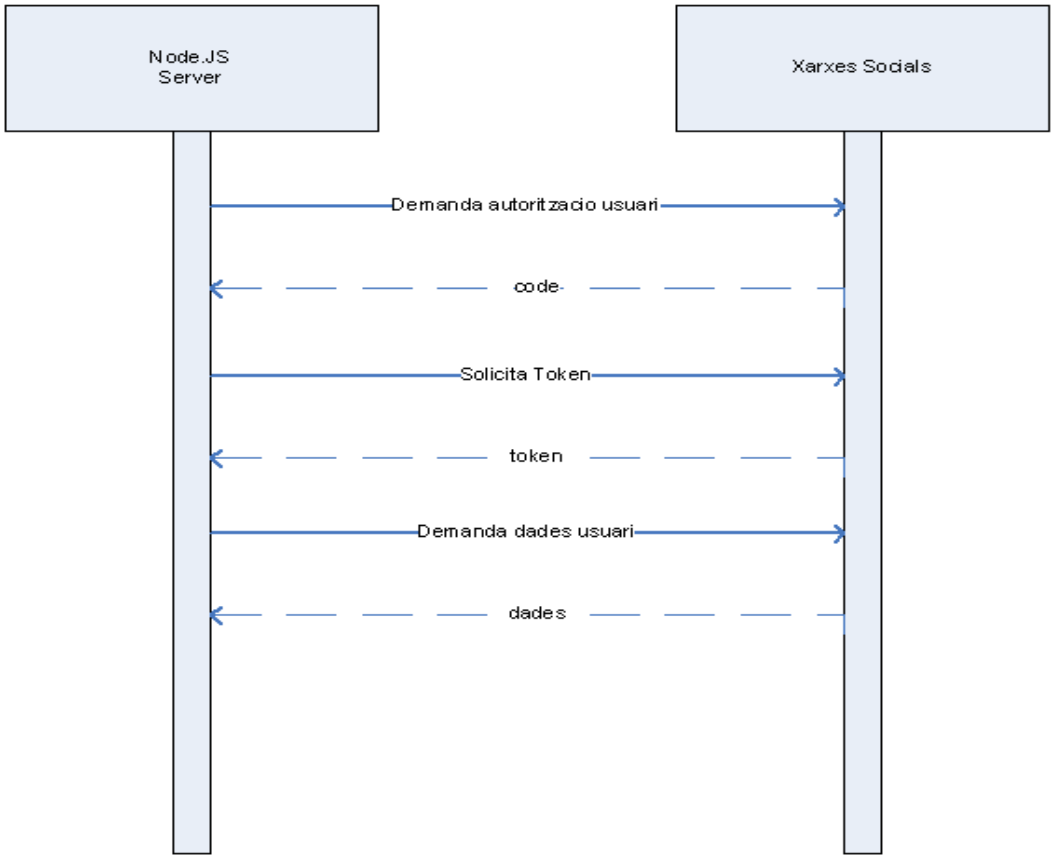
Cas d'ús	Comprovació credencials per cada Xarxa Social
Descripció	L'usuari s'identifica amb les credencials de Facebook, o de Twitter o de Google+
Actors	Usuari
PreCondicions	- Inici de l'accés amb les credencials de les xarxes socials -Ha de tenir un compte en alguna de les xarxes socials esmentades.
PostCondicions	Validació per servidor de cada xarxa social i accés a les dades dels usuaris

Taula 4: Cas d'ús 2

I



Il·lustració 9: Diagràma de Seqüència 1



Il·lustració 10: Diagràma de Seqüència 2

5.4- Llicències més adequades

Components	**Versió**	**Llicència**
ELGG	v. 2.2.3	GPL
MongoDB	v. 2.6.11	GNU AGPL
Node.JS	v. 4.2.6	MIT
Servidor Linux	Ubuntu 16.10	GPL
Apache	v. 2.4.18	Llicència Apache 2.0
PHP	v. 7.0.8-3ubuntu3	Llicència PHP

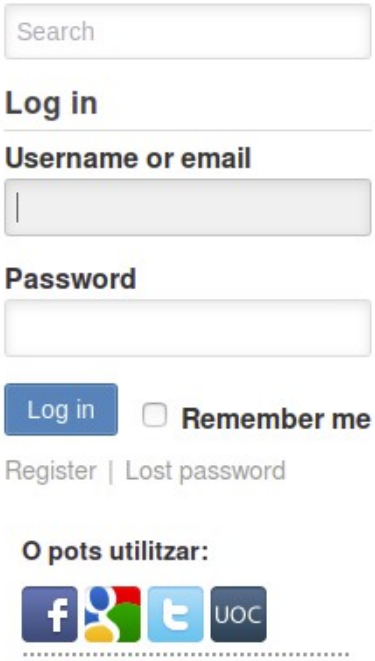
Taula 5: llicències

Les llicències que hem escollit són bàsicament llicències *Open Source*.

Són les adequades perquè són eines molt utilitzades arreu del món i i a més gratuites.

5.5- Definició d'Interfícies d'usuari

Afegim els botons adequats per cada servei.



Per cada proveïdor afegim una icona i en el codi, hem d'afegir, les claus privades, els tokens i les rutes "callback" pel retorn i comunicació de les dades dels servidors.

5.6- Requisits d'implantació

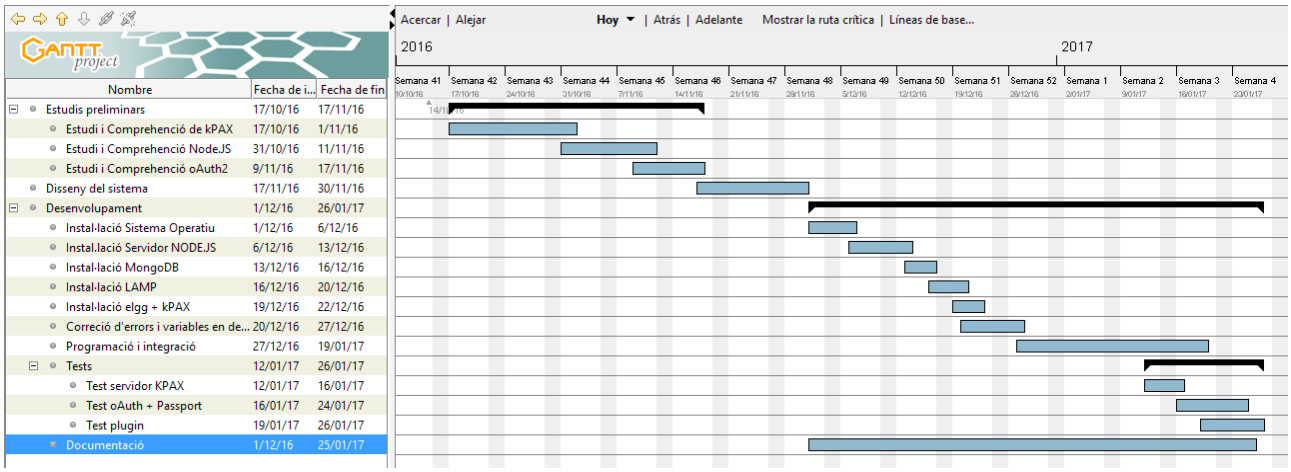
- Desenvolupament d'una aplicació de servidor (Node.JS) per fer la comunicació entre el plugin (login) i els servidors externs.
- Desenvolupament de un plugin per permetre l'autenticació (en PHP).
- Creació de la Base de Dades a MongoDB i l'estructura de la taula d'usuaris.

5.7- Planificació

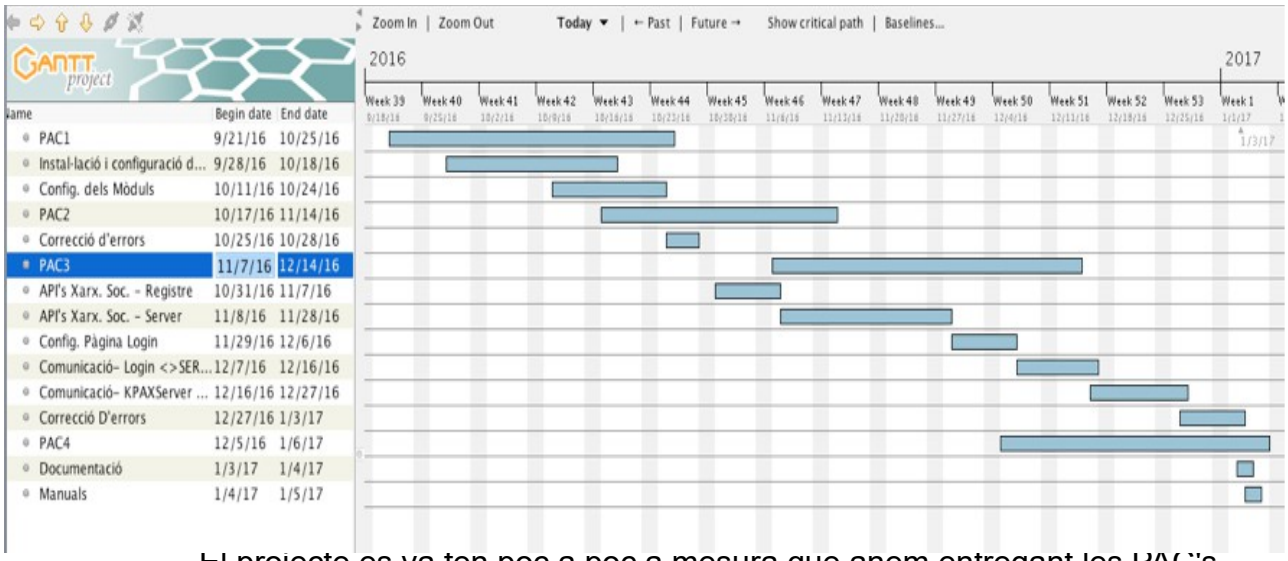
Per mostrar la planificació aproximada del desenvolupament del projecte, utilitzaré un diagrama de GANTT.

5.7-1. Calendari de desenvolupament

En aquest desenvolupament, el nostre calendari es veu limitat per l'acabament del PFM, per això el calendari es basa en l'entrega de les PAC'S.



5.7-2. Diagrama de Gantt



El projecte es va fer poc a poc a mesura que anem entregant les PAC's.

5.7-3. Documentació tècnica

Pel que fa a la documentació tècnica, s'entregarà un petit manual de com funciona l'aplicació. A part, en el codi, es posen comentaris de com funciona cada part del codi.

6. Desenvolupament i Implantació

El desenvolupament es divideix en dues parts:

- Desenvolupament de l'aplicació del servidor Node.JS
- Desenvolupament de l'aplicació (Connector) de kPAX

Estructura bàsica connector kPAX :

manifest.xml : conté les metadades del plugin en que s'indiquen les dependències,versió, autor, descripció del plugin...

start.PHP : es la pàgina principal. Conté el codi que s'executarà al carregar el connector.

README : Informació del connector.

/graphics/ : on es guarden les imatges

/views/ : vistes de l'usuari

Estructura bàsica del servidor Node.JS :

App.js : pàgina principal que enllaça amb les altres i en al que es defineixen les variables, port obert...

Package.json : arxiu en el que es guarden les dependències de les aplicacions que s'utilitzen

/routes : routes que enllaça la pàgina principal al rebre peticions

/config : es guarden les configuracions, com les claus OAuth, i els passaports.

/views : son les vistes que es fan servir per ensenyar al client, en el nostre cas només les he fet servir per fer proves.

Pel que fa a la implantació, Simplement lliurarem el codi i els responsables de l'aplicació kPAX ho revisaran, faran un "merge¨ amb el repositori Github i ho posaran en funcionament si s'escau.

.

6.1- Formació

La formació dels usuaris és únicament informar de les noves funcionalitats. Com que no afecta directament als usuaris, no implica fer formació ni a usuaris ni a desenvolupadors.

Pel que fa als desenvolupadors, hi ha aquesta documentació que explica com funciona el projecte, i al codi hi haurà descripcions específiques a cada apartat.

6.2- Implantació del sistema i Proves

Per tal d'implantar el sistema en un entorn de producció caldrà simplement instal·lar el connector en el servidor de producció i configurar correctament els inputs necessaris dins el formulari de *settings*.

Les proves que cal realitzar són les mateixes que es van dissenyar en l'entorn de desenvolupament.

6.3- Manteniment i Suport

Una vegada implantat i posat en l'entorn de producció no es requereix un manteniment especial. Només en el cas que hi hagués alguna modificació en el connector, modificació en la part servidor, s'hauria de revisar que tot continués funcionant correctament i no s'hagués quedat en desús alguna llibreria o similar.

6.4- Nous Desenvolupaments

Pel que fa als nous desenvolupaments, s'ha documentat cada funció i procediment pel tal que puguin adaptar el codi o replicar-lo en un altre entorn.

7. Conclusions

Amb aquest projecte s'aconsegueix, passar una part fonamental, que és l'autenticació d'usuari en diferents plataformes de xarxes socials, amb el nou sistema ELGG amb NodeJS i MongoDB.

Es deixarà la plataforma preparada i documentada per futurs projectes.

Es posen en pràctica molts sistemes que desconeixia com la programació amb NodeJS[7], la base de dades MongoDB, la configuració i detecció d'errors que van apareixent a mesura que avances amb el projecte; La metodologia de projecte amb GitHub, l'aplicació kPAX.

He après vàries coses, de les quals podria destacar el servidor Nodejs[7], la base de dades Mongodb, i les comunicacions amb Api's de les xarxes socials.

Mantenir la planificació ha estat molt difícil, ja que en alguns moments he hagut de dedicar més hores a la meva feina que les previstes i no he pogut disposar de més temps per dissenyar i desenvolupar aquest projecte.

La metodologia és adecuada, ja que no solament fa que planifiquis millor, sinó que fa que no perdis el fil. Si no tingués aquesta metodologia, potser havia hagut de refer i modificar alguna part del projecte. Com per exemple, fer el disseny d'una base de dades, i a meitat del projecte, donar-se compte que falta algun camp.

Pel que fa a les parts assolides, puc dir que he aconseguit fer gairebé totes les parts del projecte. Les comunicacions amb els proveïdors ha sigut correcta, menys una la qual no funciona. La que no funciona és la part de la UOC, he fet les proves amb les dades que m'han donat i no ha estat possible connectar-me. El servidor "oslo.uoc.edu" no contesta, i impedeix que pugui contrastar les dades de l'usuari i que es pugui validar. El que sí que he configurat per un possible canvi de servidors, es l'autenticació amb CAS[15] ja que el meu tutor m'ha comentat que els desenvolupadors de la UOC estaven implementant aquest servei.

He tingut molts problemes amb la ELGG, ja que no reconeixia les funcions pròpies del seu sistema. És a dir, si jo posava la funció "ELGG_echo(...)", em feia l'error típic en aquest cas que és la pàgina en blanc. I d'altres problemes com funcions en estat "deprecated".

Ara que he acabat, canviaria alguna funció perquè sigui més automàtica, ja que per la falta de temps no he pogut implementar.

He pogut veure que el que he implementat funciona i es possible connectar-se des de la web a les xarxes socials i això m'ha donat bastant satisfacció. Després de no poder avançar pels errors, he pogut avançar molt ràpidament per resoldre'ls.

D'OAuth he après que és possible simplificar el procés d'autenticació entre servidors i que és possible amb poques línies de codi fer una feina que sembla molt complicada. Hi ha exemples d'OAuth, i això facilita que sigui molt simple de fer servir.

A partir d'aquí, i després del que he dit sobre el codi de CAS de la UOC, solament canviant la direcció del servidor i algun paràmetre que hi ha, ja és podria fer servir.

Es poden afegir els proveïdors que es vulguin, saben en tot moment quina PASSPORT és l'adequat, ja que no tots els servidors reenvien tota la informació. Per exemple, pel que fa al servidor Twitter, no envia el correu de l'usuari com fan els altres, i s'ha de verificar a la base de dades en lloc del correu, el nom d'usuari.

Puc dir que, per acabar, és un projecte molt enriquidor i aporta molts aspectes que desconeixia.

8. Glossari

***1 - kPAX** *Plataforma d'aprenentatge en xarxa.*
LINK : http://www.innovaUOC.org/showcase/uploads/media/in_pid1111_art_cat.pdf

***2 - ELGG** *Plataforma de xarxes socials de codi obert.*
LINK : https://ELGG.org/

***3 - Facebook** *Xarxa social.*
LINK : https://www.facebook.com

***4 - Twitter** *Servei de microblogging.*
LINK : https://Twitter.com

***5 - Google+** *Xarxa social.*
LINK : https://plus.google.com/

***6 - OAuth2** Estàndard *obert per permetre autoritzacions segures.*
LINK : https://OAuth.net/2/

***7 - NODE.JS** *entorn multiplataforma de codi obert en la capa de servidor basat en llenguatge de programació ECMAScript.*
LINK : https://nodejs.org/es/

***8 – HTML/REST** *La Transferencia de Estado Representacional (en* inglés *Representational State Transfer) o REST es un estilo de arquitectura software para sistemas* hipermedia *distribuidos como la* World Wide Web*.*
LINK: **https://es.wikipedia.org/wiki/Transferencia_de_Estado_Representacional**

***9 – JSON** *(Javascript Object Notation - Notación de Objetos de Javascript) es un formato ligero de intercambio de datos.*
LINK *: http://www.json.org/json-es.HTML*

***10 – LAMP** **L**inux (Sistema Operatiu), **A**pache (Servidor Web), **M**ysql (Gestor de base de dades), **PHP** (Llenguatge de programació)
LINK : https://sourceforge.net/projects/lampas/

***11 – PHP** Llenguatge de programació disenyat pel contingut dinàmic de la web
LINK : http://php.net/manual/es/intro-whatis.php

***12 – HTML** *HyperText Markup Language* llenguatge web
LINK : http://www.w3schools.com/html/

***13 – Javascript** llenguatge de programació orientat a objectes
LINK : https://www.javascript.com/

***14 – CSS** Full d'estils en cascada
LINK : http://www.w3schools.com/css/

***15 – CAS servei d'autenticació centralitzada**
LINK: **https://en.wikipedia.org/wiki/Central_Authentication_Service**