Transports Metropolitans de Barcelona

Universitat Oberta de Catalunya

Manteniment Embarcat WEB

Domínguez López, Emiliano
Màster Universitari Enginyeria Informàtica
Desenvolupament d'aplicacions WEB

Nom Consultor/a: Lorente Puchades, Ignasi
Nom Professor/a responsable de l'assignatura: Córcoles Briongos, César

Data Lliurament: 12 Juny 2017

FITXA DEL TREBALL FINAL

Títol del treball:	*Manteniment Embarcat WEB*
Nom de l'autor:	*Emiliano Domínguez López*
Nom del consultor/a:	*Ignasi Lorente Puchades*
Nom del PRA:	*César Córcoles Briongos*
Data de lliurament (mm/aaaa):	*06/2017*
Titulació o programa:	*Màster Universitari Enginyeria Informàtica*
Àrea del Treball Final:	*Desenvolupament d'aplicacions WEB*
Idioma del treball:	*Català*
Paraules clau	*Manteniment, Embarcat, WEB*

Resum del Treball

Aquest treball final de màster pretén assolir una nova eina WEB que permeti realitzar tasques de manteniment OnLine en els equips embarcats dels autobusos de TMB, mitjançant eines actuals com poden ser portàtils, tablets, smartphones, etc. i aprofitant les capacitats de comunicacions que tenim actualment.

El treball final que es desenvoluparà es pot dividir en 3 blocs:

1. Estudiar les diferents opcions disponibles al mercat que es poden implantar com a servidor web per a implementar la solució que proporcioni les millors prestacions en un equip amb uns recursos limitats.
2. Desenvolupar i instal·lar l'estructura optima que sorgeixi de l'estudi previ. Aquesta permetrà connectar qualsevol dispositiu amb connexió (WIFI/3G/4G) i navegador web a la informació de la que disposin els equips que hi han embarcats.
3. Desenvolupar la pròpia WEB que mostrarà tota la informació disponible de les dades dels equips embarcats, i que haurà de ser responsive per a adaptar-se a les diferents grandàries de pantalles.

El resultat esperat és un producte que pugui ser implantat en tota la flota d'autobusos de TMB (aproximadament 1100 autobusos), que sigui útil per a realitzar el manteniment de forma remota (en la mesura que sigui possible) i que permeti a l'equip assolir la resta de funcionalitats de les que és responsable actualment.

És un treball molt ambiciós que permetrà assentar unes bases per a futures implementacions que es basaran, en part, en el resultat d'aquest treball.

Abstract

"Maintenance of "On Board Equipment" TMB Buses using a Web Application"

With this work I want to develop a new Web tool which allows us to make maintenance tasks about on board equipment in the bus fleet of TMB. This tool has to take advantage of the new available communication systems in the buses like WiFI/3G/4G. This tool must be able to execute in a wide range of devices like Smartphones, hand-held devices, laptops, and so on. In addition, the tool must allow concurrent access and be easily scalable.

This work is divided into three blocks:

1. It's necessary to do a study to know what solutions exist and their performance. For example, I will study NGINX, Ruby On Rails and Django. Then I'll choose the best option to develop the final System.
2. With the best option chosen, I'll install it in a virtual machine with a Linux System and this machine will have limited resources like the device on the bus. This development has to interact with other systems to collect all the information to show in this Web Application.
3. The last step will be to develop a Web Page and install in the Web Server chosen in the step 1 and this web has to be responsive hence it will be able to show in different screen sizes correctly.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

La tecnologia que hi ha embarcada actualment en la flota d'autobusos de TMB està basada en una arquitectura de sistemes distribuïts amb una xarxa de comunicacions que es basen en protocols TCP/IP i models de dades XML. En concret aquesta arquitectura està composta per 5 equips que tenen les següents funcionalitats:

- **Sistema Gestor d'Energia (des d'ara SGE)**: És l'equip que s'encarrega de fer la gestió energètica de la resta dels equips embarcats; els alimenta en una determinada seqüencia d'inici, els apaga en funció de l'estat del vehicle, i els controla per si algun dels equips no està comunicant-se amb la resta, per exemple s'hi ha algun equip que no ha iniciat correctament o s'ha quedat bloquejat.

  Una altra funcionalitat que té aquest equip, encara que no està funcionant al 100% de la flota, és la de comunicar-se amb els equips interns de l'autobús (com les centraletes electròniques) per bus CAN (Controller Area Network) que és l'estàndard de comunicacions en els vehicles.

- **Sistema d'Informació a l'usuari (des d'ara SIU):** Aquest equip s'encarrega de proporcionar informació a l'usuari. Aquesta informació pot ser auditiva (indicant mitjançat síntesis de veu la següent parada, o el recorregut del bus per a persones invidents), per mitjans visuals (mostrant informació d'alteracions i properes parades amb els panells led tricolor interiors, destinacions en frontals exteriors, vídeos informatius en pantalles TFT, etc.).

- **Localitzador (des d'ara LOC):** Proporciona la localització en temps real del vehicle i s'encarrega d'informar als dispositius de comunicació la seva posició actual per a que el centre de control pugui realitzar les regulacions oportunes. Aquesta geolocalització ajuda al SIU per a sintetitzar la propera parada.

- **Radio Tetra:** Actualment és l'encarregada de realitzar les comunicacions de la posició del bus al centre de control i a més, de permetre les comunicacions de veu entre el centre de control i el conductor de l'autobús.

  Aquesta comunicació de dades es veurà ampliada amb les noves comunicacions 4G que se li han incorporat a la xarxa embarcada.

- **Cpu Principal (CPU_COM):** Aquest equip és el que fa d'enllaç amb la resta d'equips mitjançant una xarxa Ethernet i gestiona totes les informacions que li poden aportar la resta d'equips. S'encarrega també de les comunicacions WiFI i 4G amb els serveis centrals, proporciona connexió WiFI al passatge mitjançant el servei de Barcelona WiFI i interactua amb informació dels serveis centrals i el conductor mitjançat el TFT que executa el Sistema d'Informació al Conductor (des d'ara SIC).

  Dins d'aquest equip s'executen diferents aplicatius que controlen el correcte funcionament de tot el conjunt i a més pot executar l'actual aplicació de manteniment embarcat que és vol substituir.

En la figura 1 es pot veure com és l'estructura embarcada definida fins ara:

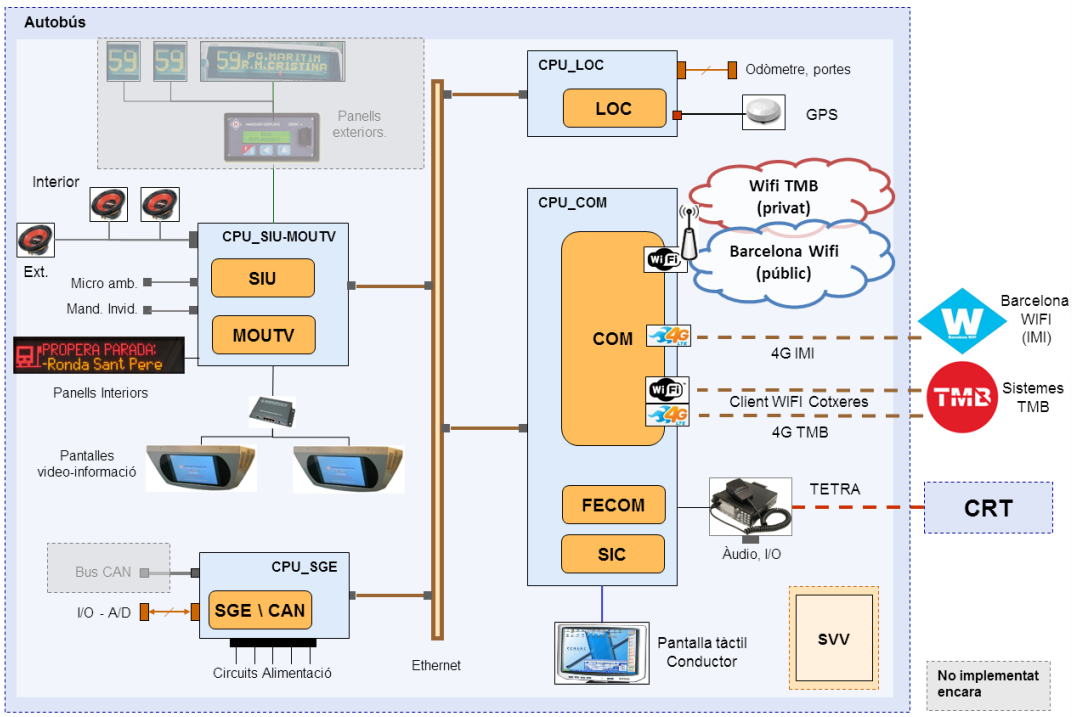


Figura 1 Estructura equips embarcats

L'aplicació de manteniment embarcat que tenim actualment s'ha d'executar manualment des de la pantalla TFT del conductor, i encara que la Cpu Principal ens permet accedir al que s'està visualitzant en la pantalla TFT mitjançant connexions VNC (Virtual Network Computing), si de forma remota accedim a visualitzar l'aplicatiu de manteniment embarcat, el conductor es queda sense la seva informació de regulació del servei.

L'aplicació de manteniment embarcat que està funcionant actualment en el sistema es mostra en la figura 2. I en la figura 3 es mostra l'arbre d'opcions de la informació relacionada amb la Cpu Principal:

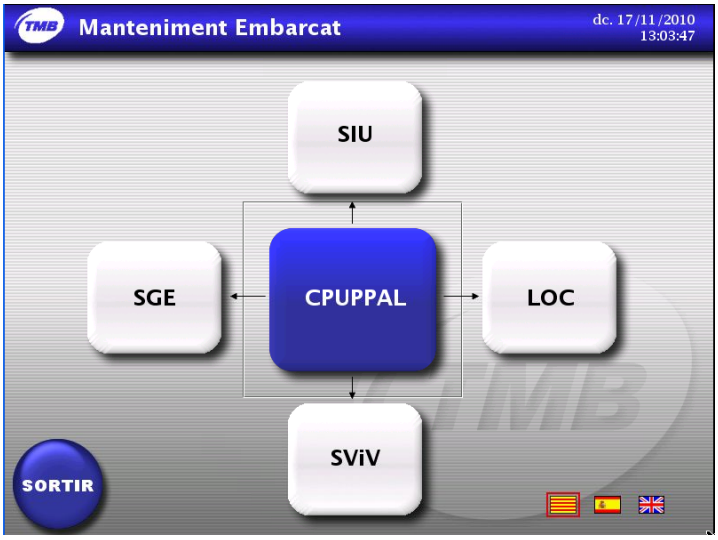


Figura 2 Aplicació de Manteniment embarcat

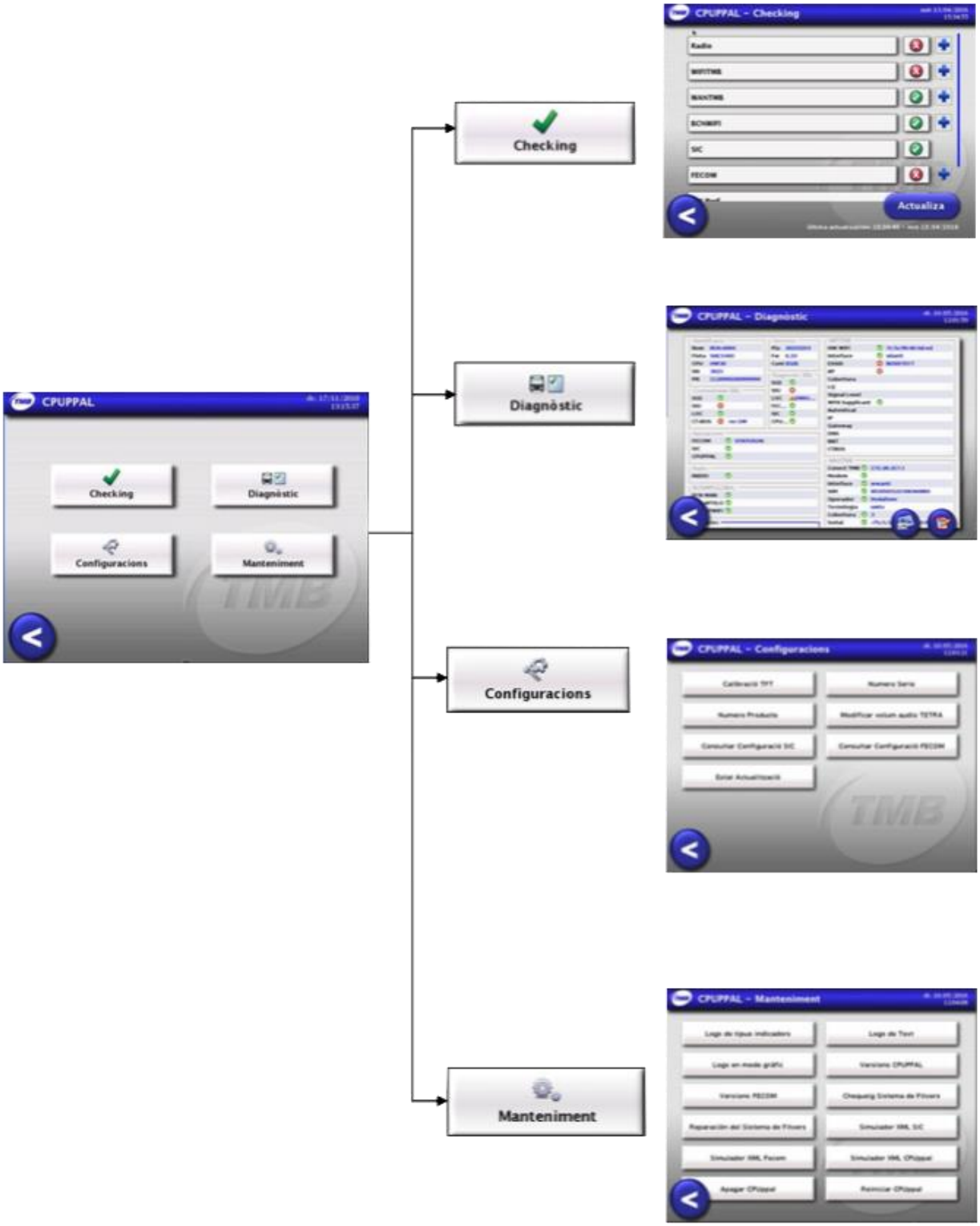


Figura 3 Arbre de funcionalitats

Per aprofitar les comunicacions 4G de les que disposem actualment i poder visualitzar el manteniment embarcat d'un autobús que estigui en línia de forma remota, la solució parcial que hem implementat ha estat mitjançant un Script en Bash, connectar-se per SSH (Secure SHell) a l'equip, redireccionar el display en les opcions de VNC i llavors executar el manteniment embarcat en el PC de forma remota.

Aquesta solució pal·liativa té molts inconvenients:

- Nomes permet l'accés d'una persona o PC, es a dir, no permet concurrència.
- El procés és molt feixuc, encara que s'ha facilitat amb el Script que s'ha implementat.
- No es pot accedir des d'un dispositiu qualsevol i que no sigui de la xarxa de TMB.

L'objectiu que es persegueix amb la realització d'aquest projecte és un sistema que permeti realitzar tasques de manteniment Online de forma preventiva i/o correctiva, perquè implementar una solució que permeti accés a un manteniment correctiu Online pot facilitar a molts departaments de l'empresa la visualització en temps real de casuístiques que indiquin funcionaments anòmals dels equips i/o de l'autobús.

Per exemple, podem tenir un avis del centre de control que un autobús no està realitzant la síntesis de les parades de la línia on està circulant, llavors personal de manteniment podria accedir de forma online al manteniment embarcat, visualitzar l'estat dels equips, les versions de les seves aplicacions, els fitxers de configuració, etc. i intentar de forma remota solucionar el problema, tot això sense retirar el vehicle del servei, i el que és més important sense interferir en el funcionament del conductor, de l'autobús i sense afectar en res a l'usuari.

Actualment, si ocorre l'exemple proposat l'autobús estaria circulant tot el torn sense informar de les parades als usuaris i quan arribes al seu Centre Operatiu seria quan el personal de manteniment realitzaria les tasques oportunes per a solucionar el problema.

Amb aquesta eina que es vol implementar es pot aconseguir un millor servei de cara al client i un manteniment més potent i efectiu.

1.2 Objectius del Treball

Els objectius que es volen aconseguir amb la consecució d'aquest projecte són:

1. Estudiar la viabilitat d'aquest tipus d'aplicació en l'entorn on s'haurà d'executar a nivell de rendiment, usabilitat, seguretat, connectivitat, etc.
2. Desenvolupar i migrar l'actual aplicació de manteniment embarcat cap a una aplicació tipus Web.
3. Permetre consultar i/o configurar opcions en background sense interferir en la pantalla del conductor.
4. Permetre l'accés concurrent des de diferents dispositius amb connectivitat WiFI/3G/4G i amb navegador Web.
5. La web que es desenvolupi ha de ser "responsive" per a que es pugui visualitzar correctament des de qualsevol dispositiu i mida de pantalla.
6. El desenvolupament que es realitzi ha de ser fàcilment escalable per la inclusió de nous equips en la xarxa embarcada.

1.3 Enfocament i mètode seguit

Hi han dos possibilitats per a enfocar aquest projecte:

1. Estudiar l'actual aplicació de manteniment embarcat i tractar d'adaptar-la per a les noves funcionalitats que s'han plantejat en els objectius.
2. Realitzar una nova aplicació per a donar-li un aspecte renovat (l'actual té aproximadament 8 anys).

Encara que la segona opció és molt ambiciosa i possiblement molt extensa per a aconseguir-la en l'espai de temps del que es disposa, és més interessant perquè amb aquest desenvolupament es veurà quina es la viabilitat tècnica de desenvolupar aquest tipus d'aplicació i es podrà aprofitar aquest coneixement i

ampliar-lo a més sistemes. Es a dir, a més d'aconseguir un producte nou que ens permeti ampliar les funcionalitats que tenim ara mateix, ens proporcionarà un "know how" que es podrà incloure en projectes futurs.

1.4 Planificació del Treball

S'han identificat 3 fases per al desenvolupament d'aquest projecte, que són:

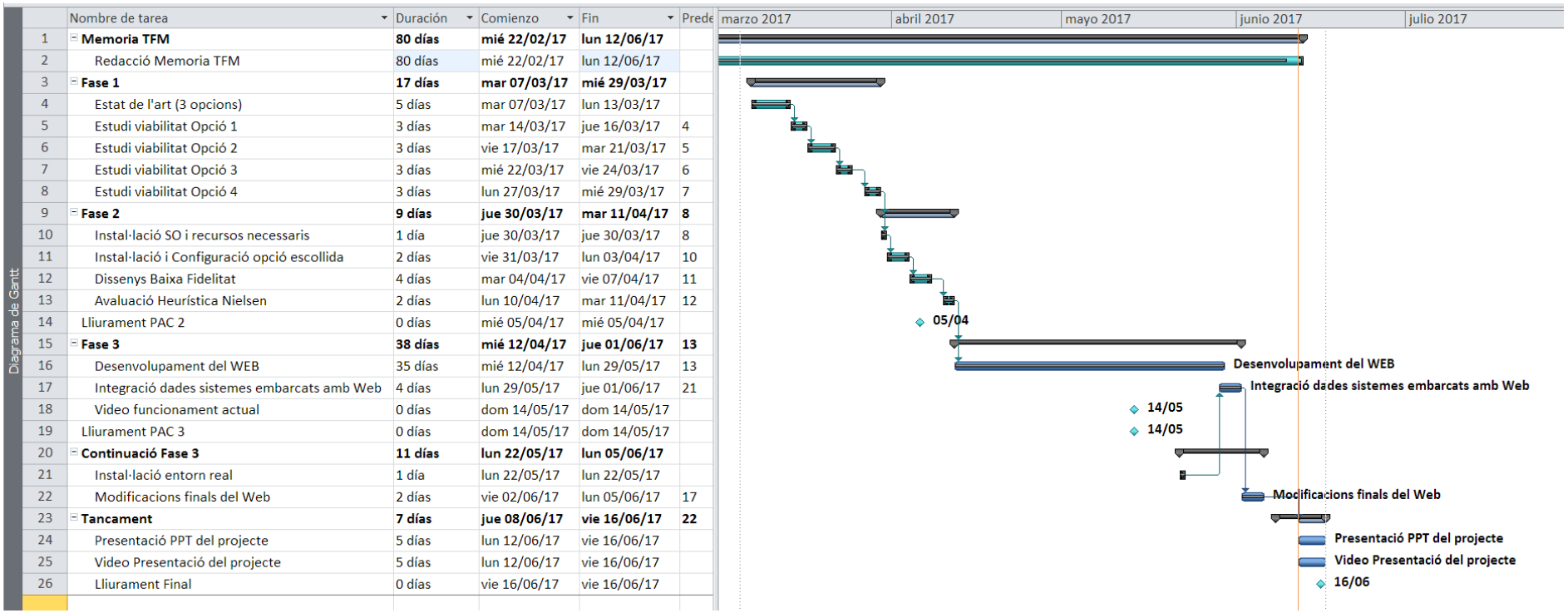
1. Estudiar les diferents opcions de servidors web que existeixen actualment, i la seva viabilitat tècnica per a fer-la servir en aquest projecte.
    a. Dins d'aquesta fase s'ha de realitzar una tria dels referents actuals en el que a servidors web es refereix per a estudiar el funcionament, els recursos, plataformes, etc. on es poden implantar.
    b. Realitzar una prova de concepte senzilla amb cada opció, per tal de tenir dades quantitatives i qualitatives de quina pot ser la millor opció a implementar.
2. Instal·lar en un entorn de desenvolupament la millor opció escollida de les estudiades en la fase anterior.
    a. Un cop s'hagin observat els comportaments, els rendiments de processador, RAM, accessos E/S, etc. es realitzarà la instal·lació completa del servidor web que millors prestacions ens hagi resultat en una maquina virtual amb el sistema operatiu on s'instal·larà finalment la solució, i els recursos de maquinari que té l'equip físic.
3. Implementar la Web amb totes les característiques i objectius definits en els apartats anteriors, es a dir que sigui usable, responsive, disseny centrat en l'usuari, etc. Dins d'aquesta fase s'haurà de:
    a. Fer dissenys de baixa fidelitat i l'avaluació heurística amb els 10 principis de Nielsen.
    b. Implementar la web pròpiament dita amb els feedbacks que tinguem de l'avaluació heurística.
    c. Dotar de dinamisme a l'aplicació web interactuant amb les dades que proporcionaran la resta d'equips.

d. Estudiar les opcions d'accessos concurrents quan es tracti de configuracions, es a dir, estudiar com es poden resoldre els casos en que 2 o més accessos estiguin intentant modificar una variable a l'hora.

Pel que fa als recursos necessaris per a poder realitzar aquest projecte, tenim de 3 tipus:

- **Maquinari**:
  - ✓ PC amb accés a Internet.
  - ✓ Tablets i Smartphones amb diferents mides de pantalla per a provar com es comporta la web "responsive".
- **Programari**:
  - ✓ Aplicació per a crear i manipular maquines virtuals (Virtual BOX, VMWare,...)
  - ✓ Sistema Operatiu basat en Linux (Ubuntu 14.04.05 LTS).
  - ✓ Editors i IDEs per al desenvolupament del codi:
    - Sublime Text
    - ...
- **Informació digital, llibres, articles, etc**.

Amb la distinció de les 3 fases i els objectius a assolir, la planificació que es proposa portar a terme és la mostrada en la figura 4:



	Nombre de tarea	Duración	Comienzo	Fin	Prede
1	**Memoria TFM**	**80 días**	**mié 22/02/17**	**lun 12/06/17**	
2	Redacció Memoria TFM	80 días	mié 22/02/17	lun 12/06/17	
3	**Fase 1**	**17 días**	**mar 07/03/17**	**mié 29/03/17**	
4	Estat de l'art (3 opcions)	5 días	mar 07/03/17	lun 13/03/17	
5	Estudi viabilitat Opció 1	3 días	mar 14/03/17	jue 16/03/17	4
6	Estudi viabilitat Opció 2	3 días	vie 17/03/17	mar 21/03/17	5
7	Estudi viabilitat Opció 3	3 días	mié 22/03/17	vie 24/03/17	6
8	Estudi viabilitat Opció 4	3 días	lun 27/03/17	mié 29/03/17	7
9	**Fase 2**	**9 días**	**jue 30/03/17**	**mar 11/04/17**	**8**
10	Instal·lació SO i recursos necessaris	1 día	jue 30/03/17	jue 30/03/17	8
11	Instal·lació i Configuració opció escollida	2 días	vie 31/03/17	lun 03/04/17	10
12	Dissenys Baixa Fidelitat	4 días	mar 04/04/17	vie 07/04/17	11
13	Avaluació Heurística Nielsen	2 días	lun 10/04/17	mar 11/04/17	12
14	Lliurament PAC 2	0 días	mié 05/04/17	mié 05/04/17	
15	**Fase 3**	**38 días**	**mié 12/04/17**	**jue 01/06/17**	**13**
16	Desenvolupament del WEB	35 días	mié 12/04/17	lun 29/05/17	13
17	Integració dades sistemes embarcats amb Web	4 días	lun 29/05/17	jue 01/06/17	21
18	Video funcionament actual	0 días	dom 14/05/17	dom 14/05/17	
19	Lliurament PAC 3	0 días	dom 14/05/17	dom 14/05/17	
20	**Continuació Fase 3**	**11 días**	**lun 22/05/17**	**lun 05/06/17**	
21	Instal·lació entorn real	1 día	lun 22/05/17	lun 22/05/17	
22	Modificacions finals del Web	2 días	vie 02/06/17	lun 05/06/17	17
23	**Tancament**	**7 días**	**jue 08/06/17**	**vie 16/06/17**	**22**
24	Presentació PPT del projecte	5 días	lun 12/06/17	vie 16/06/17	
25	Video Presentació del projecte	5 días	lun 12/06/17	vie 16/06/17	
26	Lliurament Final	0 días	vie 16/06/17	vie 16/06/17	

Figura 4 Diagrama de Gantt

1.5 Breu sumari de productes obtinguts

Els productes que s'obtindran d'aquest treball seran:

- El desenvolupament de la pagina web,
- Els fitxers de configuració necessaris per al servidor web, connector entre servidor web i servidor d'aplicacions i del propi servidor d'aplicacions.

1.6 Breu descripció dels altres capítols de la memòria

Els següents capítols d'aquesta memòria s'estructuren de la següent manera:

El **capítol 2** farà una descripció de quin és l'estat actual de l'aplicació que es pretén canviar, i quin és l'escenari que es vol aconseguir a futur.

El **capítol 3** estudia l'estat de l'art de totes les parts involucrades en aquest projecte. A més permetrà veure les opcions més indicades per al desenvolupament que es vol realitzar.

Un cop s'han vist i estudiat les diferents opcions per a implementar la solució, el **capítol 4** posa a prova diferents escenaris per a veure els diferents recursos que són necessaris per a utilitzar un escenari o un altre. D'aquests escenaris ha de sortir un escenari òptim per a continuar amb el desenvolupament de l'aplicació web.

El **capítol 5** mostra els dissenys de baixa fidelitat que, un cop avaluats amb els principis heurístics de Nielsen al **capítol 6**, seran els que s'implementaran com a aplicació web.

El **capítol 7** especificarà el desenvolupament web, així com les estratègies adoptades, les funcions que es poden implementar, la comunicació amb les dades de la resta d'equips, etc.

El **capítol 8** identificarà els bugs que hagin pogut sortir dels test amb usuaris o de l'estudi en profunditat de l'aplicació web. Es farà una estimació econòmica del procés de desenvolupament portat a terme en el **capítol 9**, i finalment en el **capítol 10** es mostraran les conclusions a les que s'hagin arribat amb aquest

projecte, es a dir, quins són els punts forts i febles d'aquest projecte, quines han estat les lliçons apreses, que ha quedat fora del projecte, quines seran les següents passes, etc.

2. Descripció dels escenaris

2.1 Escenari actual

Tal i com s'ha explicat en el punt 1.1 Context i justificació del treball en l'escenari actual es disposa d'un equip que executa un sistema operatiu Linux Debian 8, i sobre aquest sistema operatiu s'executa l'aplicació denominada SIC (Sistema d'informació al conductor); mitjançant aquesta aplicació es pot executar l'aplicació de "Manteniment embarcat".

L'estructura software de l'equip actual seria la que es mostra en la figura 5:

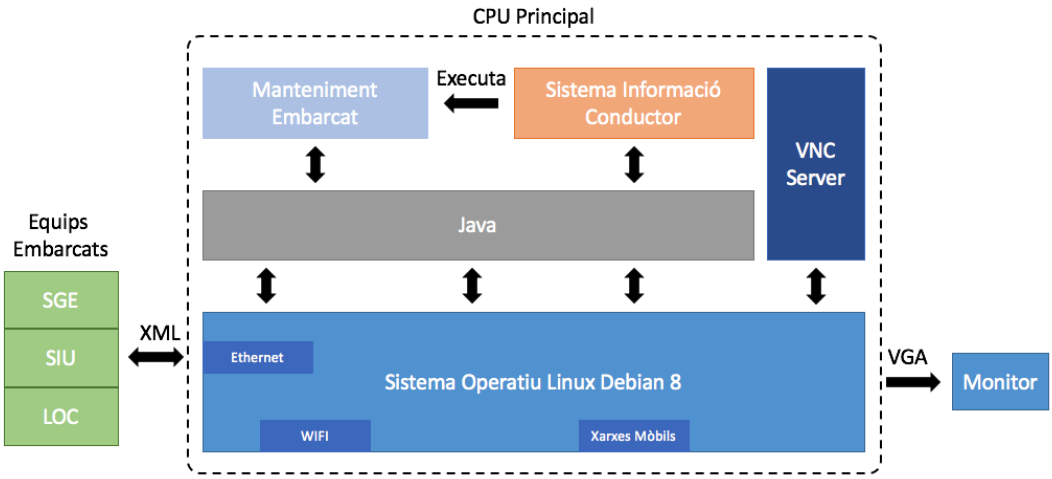


Figura 5 Equip CPU actual

L'equip disposa de les següents característiques hardware:

- Processador de doble nucli x64
- Memòria RAM 4GBytes
- Interfícies de comunicació: Ethernet, WiFI, 3G/4G, Bluetooth.
- Altres dispositius: GPS, acceleròmetre,...

2.2 Escenari futur

Per aconseguir els objectius d'aquest projecte, el plantejament que es vol desenvolupar és el que es mostra en la figura 6:

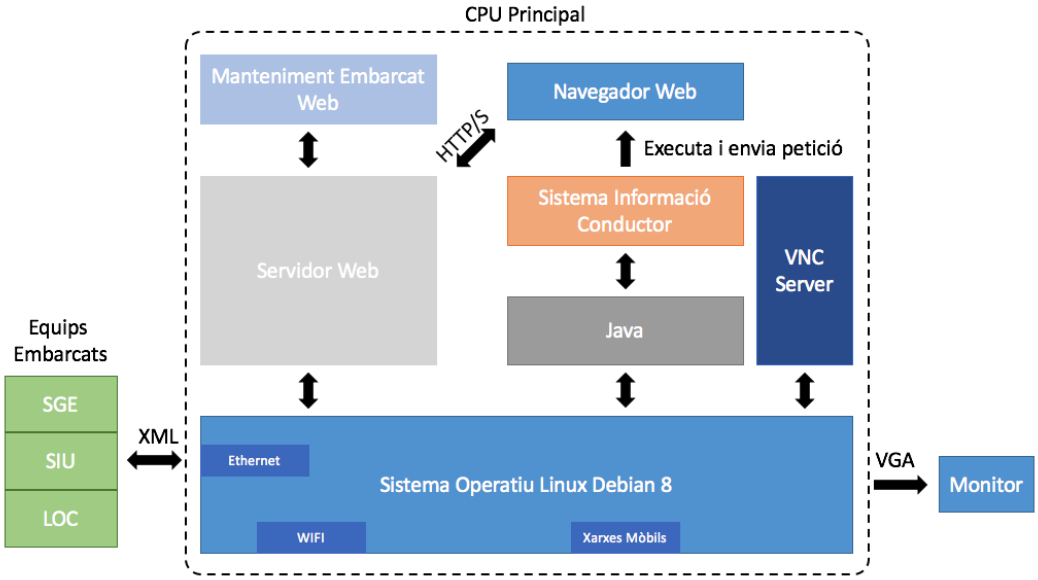


Figura 6 Equip CPU futur

L'estructura necessària serà un servidor web d'aplicacions que ens permeti accedir a una adreça web mitjançant un dispositiu amb navegador web, i a traves d'aquesta pagina web visualitzar la informació proporcionada per la resta dels equips embarcats.

Aquest disseny el podríem descompondre en 3 capes ben diferenciades:

1. **Capa de presentació:** Es referiria a la part que l'usuari podrà veure i manipular, es a dir, en aquest cas seria la pròpia web.

2. **Capa de negoci:** Aquesta capa s'encarregarà de rebre les peticions dels usuaris i de retornar els resultats a la capa de presentació.

Aquesta capa es té que comunicar per una banda amb la capa de dades per obtenir i/o calcular els resultats que ha demanat l'usuari, i per altra banda, amb la capa de presentació perquè ha de proporcionar les dades que es visualitzaran en la capa de presentació.

3. **Capa de dades:** Aquesta capa s'encarrega d'emmagatzemar les dades i posar-les a disposició de la capa de negoci.

Aquest model és molt semblant al patró d'arquitectura de programació MODEL-VISTA-CONTROLADOR, on tindríem que el model seria l'equivalent a la capa de dades, la vista l'equivalent a la capa de presentació i el controlador l'equivalent a la capa de negoci.

Per assolir aquesta solució per capes s'ha escollit l'esquema de la figura 7:

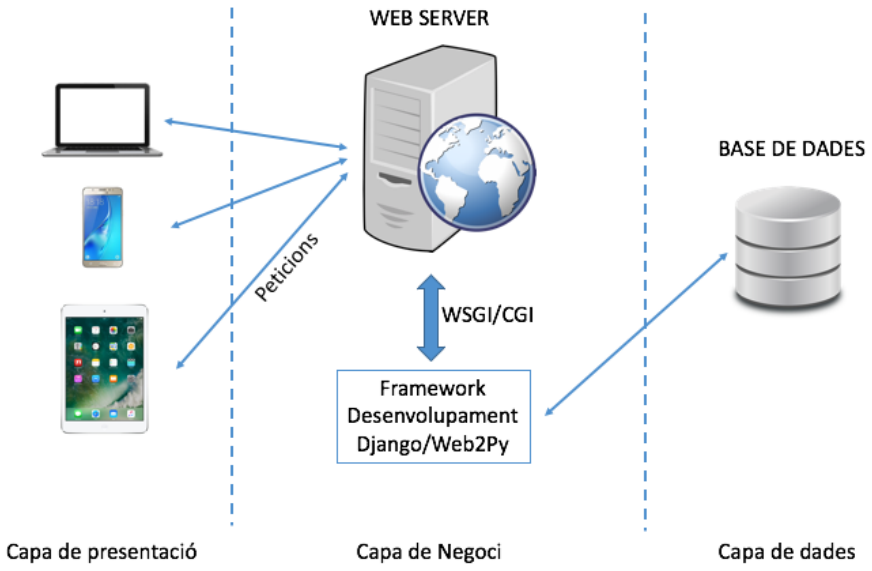


Figura 7 Estructura de la solució (3 capes)

3. Estat de l'art

Tenint en compte els elements que hem d'escollir per abordar la solució plantejada en aquest projecte, s'han cercat les opcions de les que disposem actualment per a assolir els objectius, tant en servidor web d'aplicacions com en frameworks de desenvolupament web.

3.1 Servidors Web

A Internet es poden trobar multitud d'opcions de servidors web. A continuació es descriuen els 3 més rellevants actualment: NGINX, Apache2 i Lighttpd.

3.1.1 NGINX

És un servidor web HTTP lliure i de codi obert que té un alt rendiment, estabilitat i baix consum de recursos. És utilitzat per llocs com Netflix, Airbnb, Heroku, GitHub i WordPress, entre d'altres.

Aquest servidor executa un procés "master" i diversos processos "worker". La funcionalitat principal del procés "master" és la de llegir i avaluar la configuració i mantenir els processos "worker" actius. Els processos "worker" s'encarreguen de fer el processament de les peticions.

Nginx implementa un model basat en esdeveniments i mecanismes dependents del Sistema Operatiu per a distribuir de forma eficient les peticions entre els processos "worker". El numero de "workers" és pot configurar i el seu valor òptim depèn de factors relacionats amb el hardware del dispositiu on s'executa. S'aconsella configurar el nombre de processos "worker" amb el numero de nuclis que tingui el processador on s'executi.

Aquest servidor també permet equilibrar la carrega HTTP per a distribuir eficientment el tràfic cap a diverses aplicacions web, i d'aquesta manera permet millorar el rendiment, l'escalabilitat i la fiabilitat de les aplicacions.

Normalment aquest servidor es configura com a Proxy invers o "reverse Proxy" per a redirigir les peticions que rep cap a un servidor WSGI (Web Server Gateway Interface) que produeix contingut dinàmic mitjançant l'execució de

codi Python. Un cop el servidor WSGI té la resposta (normalment HTML, JSON o XML), el Proxy invers respon al client amb aquest resultat.

La figura 8 mostra de forma esquemàtica el funcionament de NGINX:

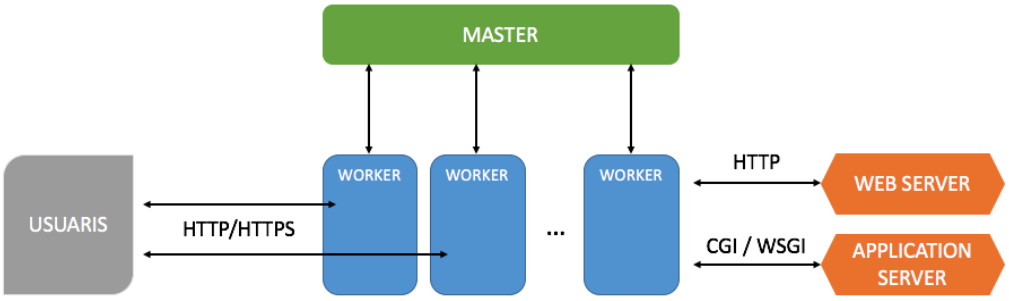


Figura 8 Estructura de NGINX

3.1.2 Apache2

Ha estat el servidor web més utilitzat des del seu llançament en 1995. És de codi obert i gratuït, i proporciona versions per a tots els sistemes operatius més importants.

Entre les seves característiques més destacades tenim que és altament configurable, permet bases de dades amb autenticació i la majoria de vulnerabilitats de seguretat descobertes i resoltes nomes les poden fer servir usuaris locals i no pas de forma remota.

Aquest servidor web disposa de diversos mòduls que l'aporten moltes funcionalitats, sobretot mòduls externs per a disposar de continguts dinàmics com en el cas de Nginx.

3.1.3 Lighttpd

És una altra opció de web server ràpid, segur, i fiable. Consumeix menys recursos de memòria i de processador que d'altres opcions per això està optimitzat per a entorns on la velocitat és molt important.

De la mateixa manera que ocorria amb les dues opcions anteriors, pot fer servir extensions per a proporcionar continguts dinàmics.

3.1.4 Comparativa

Com els servidors web definits en els punts anteriors són els més reconeguts a Internet i els que millors resultats aporten, s'han trobat recursos on s'ofereixen les comparatives entre els 3 servidors webs del llistat anterior, com les que es mostren en les figures 9 i 10:

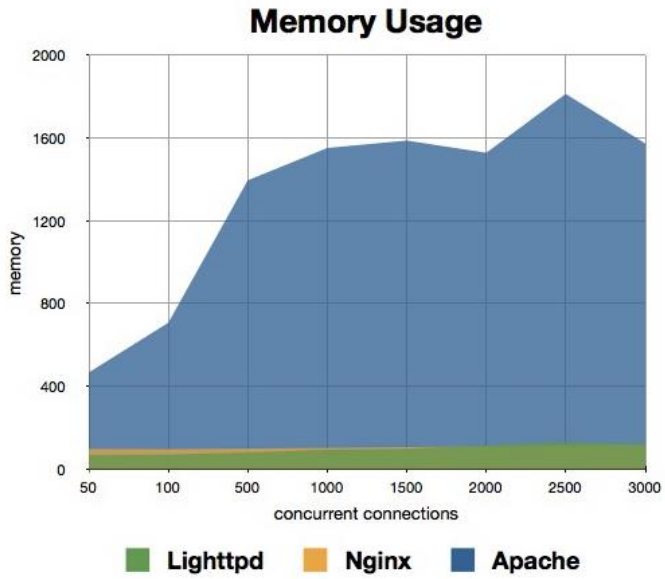


Figura 9 (font: https://help.dreamhost.com/hc/en-us/articles/215945987-Web-server-performance-comparison)

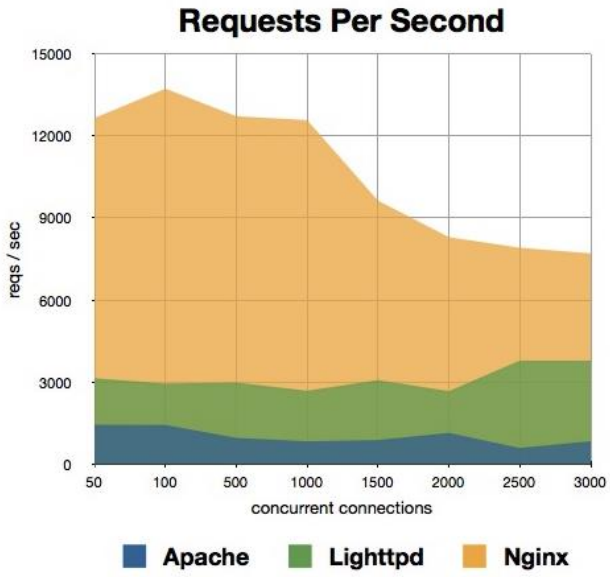


Figura 10 (font: https://help.dreamhost.com/hc/en-us/articles/215945987-Web-server-performance-comparison)

3.2 Servidors d'aplicacions WEB

Com a principals servidors d'aplicacions web tenim 3, tots ells basats en el llenguatge de programació Python ja que és un llenguatge interpretat i té una sintaxi que el fa molt intuïtiu.

3.2.1 Django

Té com a objectiu principal facilitar la creació de llocs web complexos, re-utilitzar, el desenvolupament ràpid i el principi de no repetir (DRY en angles, Don't Repeat Yourself). Es basa en el patró de programació Model-Vista-Controlador, és de codi obert i té un ampli suport i seguiment en Internet.

3.2.2 Web2Py

De la mateixa manera que Django, Web2Py és de codi obert i pretén donar suport al desenvolupament àgil de software i a la creació de webs escalables i segures. De la mateixa manera que Django, es basa en el patró Model-Vista-Controlador i disposa d'una interfície "administrativa" que permet configurar, i/o modificar codi, funcions,... directament des del navegador web on es visualitza la web. Encara que té un ampli suport a Internet, es troba molta més informació i recursos per Django.

3.2.3 Flask

S'anomena microframework perquè no necessita biblioteques o eines concretes. Existeixen múltiples extensions que es poden utilitzar amb flask per a donar-li les funcionalitats necessàries.

3.3 Continguts dinàmics

Els servidors web no executen directament les aplicacions web desenvolupades sinó que reben peticions des d'un navegador web i retornen codi HTML. Aquest codi HTML és estàtic i com que els servidors web que s'han definit no tenen la possibilitat de generar codi per a visualitzar continguts dinàmics, es fan necessàries unes interfícies que implementin un protocol per a respondre a les necessitats de comunicació entre les aplicacions web i els servidors web per aconseguir dinamisme en els resultats.

Els protocols que es poden utilitzar per als servidors web que s'han definit anteriorment són extensions de **CGI (Common Gateway Interface)** i **WSGI (Web Server Gateway Interface)**.

3.3.1 CGI. Common Gateway Interface

CGI és un estàndard que permet la comunicació entre aplicacions i servidors web. Un programa CGI s'executa en temps real i pot anar generant informació dinàmica a partir de certs paràmetres que li proporcioni el servidor web, i al final retornar els resultats per a que el propi servidor els pugui mostrar.

Dintre d'aquest estàndard trobem extensions com FastCGI.

3.3.2 WSGI. Web Server Gateway Interface

És similar al CGI, es a dir, és una interfície que permet comunicar els servidors web amb les aplicacions web. Està basat en CGI però amb modificacions relacionades amb el llenguatge Python.

Aquesta interfície té 2 costats, per una banda el gateway o servidor (on podrien estar Nginx o Apache) i per altra banda el costat de l'aplicació on estarien les aplicacions desenvolupades en Django, Flask o Web2Py, entre d'altres.

Quan el servidor web rep una petició, el costat servidor executa l'aplicació i li proporciona la informació necessària (que pot haver estat introduïda per l'usuari en la petició), l'aplicació processa aquesta petició i retorna els resultats al servidor.

Dintre d'aquesta interfície trobem, entre d'altres, uWSGI i Gunicorn (Green Unicorn).

4. Estudi de la viabilitat tècnica de la solució

Fins aquest punt s'ha especificat el tipus de hardware on s'ha de desenvolupar aquesta solució i també s'han definit les principals opcions disponibles per a assolir aquest projecte. Per tant, resta escollir la solució més adequada i que s'ajusti el millor possible tant a les funcionalitats com als recursos hardware.

Per poder realitzar aquesta elecció recrearem el hardware del que disposem mitjançant una maquina virtual i amb un sistema operatiu Ubuntu 14.04.05 LTS amb Debian 8 com el de l'equip embarcat.

S'instal·laran 4 escenaris:

- Apache-WSGI-Web2Py
- Nginx-uWSGI-Web2Py
- Nginx-uWSGI-Django
- Nginx-Gunicorn-Django

Disposem del software Apache Benchmark instal·lat a la maquina Host que ens permet realitzar peticions a un servidor web de forma concurrent i especificar diferents paràmetres per posar a prova un servidor web. Es realitzaran peticions concurrents per a veure com afecta en els recursos de la maquina virtual i quins són els límits que es poden aconseguir. En concret, es faran proves amb 5, 10, 20 i 50 peticions concurrents per segon.

En tots els escenaris s'ha tingut en compte la configuració del servidor web així com el servidor de continguts dinàmics. Com que el numero de "workers" recomanats en Nginx és el nombre de processadors del sistema (en el nostre cas 2) s'han configurat 2 workers, i pel que fa als processos WSGI, el recomanable és que siguin el doble del nombre de processadors, en el nostre cas s'han configurat amb 4 processos WSGI.

Amb els resultats obtinguts d'aquesta eina i els consums dels recursos del propi sistema operatiu estarem en disposició d'escollir la solució més optima.

4.1 Resultats obtinguts

Els resultats de les proves d'estres realitzades amb els diferents entorns de proves s'han dibuixat gràficament, mostrant el consum de CPU-RAM i s'han adjuntat a l'annex d'aquesta memòria.

Les figures 11 i 12 mostren un resum dels resultats que s'han obtingut en la realització de totes les proves mostrades en les gràfiques anteriors:

	%CPU repòs	%CPU amb 5 peticions concurrents	%CPU amb 10 peticions concurrents	%CPU amb 20 peticions concurrents	%CPU amb 50 peticions concurrents
Apache + WSGI + Web2Py	2'01%	8'49%	14'01%	25'11%	41'22%
NGINX + uWSGI + Web2Py	2'71%	6'66%	11'088%	18'45%	34'53%
NGINX + uWSGI + Django	2'13%	2'77%	3'166%	4'24%	6'76%
NGINX + Gunicorn + Django	2'03%	2'8%	3'2%	4'08%	6'90%

Figura 11 Taula resum consum CPU

	Temps mitjà resposta 5 peticions concurrents	Temps mitjà resposta 10 peticions concurrents	Temps mitjà resposta 20 peticions concurrents	Temps mitjà resposta 50 peticions concurrents
Apache + WSGI + Web2Py	141'435 mS	290'022 mS	608'332 mS	1553'060 mS
NGINX + uWSGI + Web2Py	63'838 mS	108'026 mS	209'339 mS	564'734 mS
NGINX + uWSGI + Django	6'521 mS	11'034 mS	21'679 mS	54'123 mS
NGINX + Gunicorn + Django	6'20 mS	11'68 mS	21'453 mS	53'569 mS

Figura 12 Taula resum temps resposta mitjà

Tal i com es pot observar en les taules, hi ha una diferencia important entre el rendiment d'una aplicació Django i una aplicació Web2Py, en realitat això es degut a que Web2Py per defecte té una pagina web amb imatges, fitxers d'estils (CSS), etc. i en general és més pesada que la pagina web que té per defecte Django. Per a les proves de rendiment referides al temps de resposta per peticions, la grandària en bytes s'ha igualat en totes 2 webs de test.

L'elecció entre el Web Server Nginx i Apache és clara, ja que tots els millors resultats obtinguts han estat per part de Nginx. Pel que fa al framework de desenvolupament, escollirem Django perquè té més recorregut i té més suport en webs d'Internet, llibres, etc. I entre Gunicorn i uWSGI, tots 2 tenen un comportament similar, però Gunicorn és més senzill a l'hora d'instal·lar, amb la qual cosa l'escenari escollit serà:

- NGINX
- Gunicorn
- Django

5. Dissenys de baixa fidelitat

Un cop tenim la solució d'arquitectura escollida, resta el disseny del web que es desenvoluparà.

Per a començar a dissenyar-lo s'han realitzat un seguit de Wireframes que ajuden a fer-se una primera idea de com pot ser l'aplicació web. Aquests dissenys serveixen per a 2 objectius principals:

1. Els usuaris finals poden tenir una primera impressió de com seran les pantalles que hi hauran en l'aplicació, a més, encara que sigui un disseny de baixa fidelitat, els usuaris o l'expert en usabilitat poden observar errors o funcionaments que no siguin apropiats i es poden fer canvis abans de implementar codi.

2. El propi desenvolupador amb aquests wireframes disposa d'unes plantilles a partir de les quals pot començar a implementar codi. A més, com que aquestes pantalles han estat validades per l'expert en usabilitat i/o pels usuaris, ja no té que realitzar diferents dissenys i proves.

S'ha pres la decisió de nomes realitzar els models per a mòbil, en un principi, per poder assolir el projecte quasi sencer, encara que hi haurà certs aspectes de disseny que ja permetran que les pantalles s'adaptin a navegadors més grans, gracies a la utilització de Bootstrap.

Els dissenys que s'han realitzat són molt similars a les pantalles de l'aplicació actual, es va decidir així per tal de no canviar el "model mental" al personal de manteniment que actualment fa servir aquesta aplicació.

5.1 Wireframes

A continuació es fa un recull de les pantalles més característiques que s'han dibuixat, comparant-les amb les actuals.

5.1.1 Login

Aquesta pantalla permet identificar-se com a usuari de l'aplicació de manteniment embarcat. Actualment no hi ha una distinció entre usuaris, però es vol portar a terme aquest control mitjançant LDAP per poder restringir certes manipulacions en funció del perfil i poder saber qui ha realitzat segons quina acció.

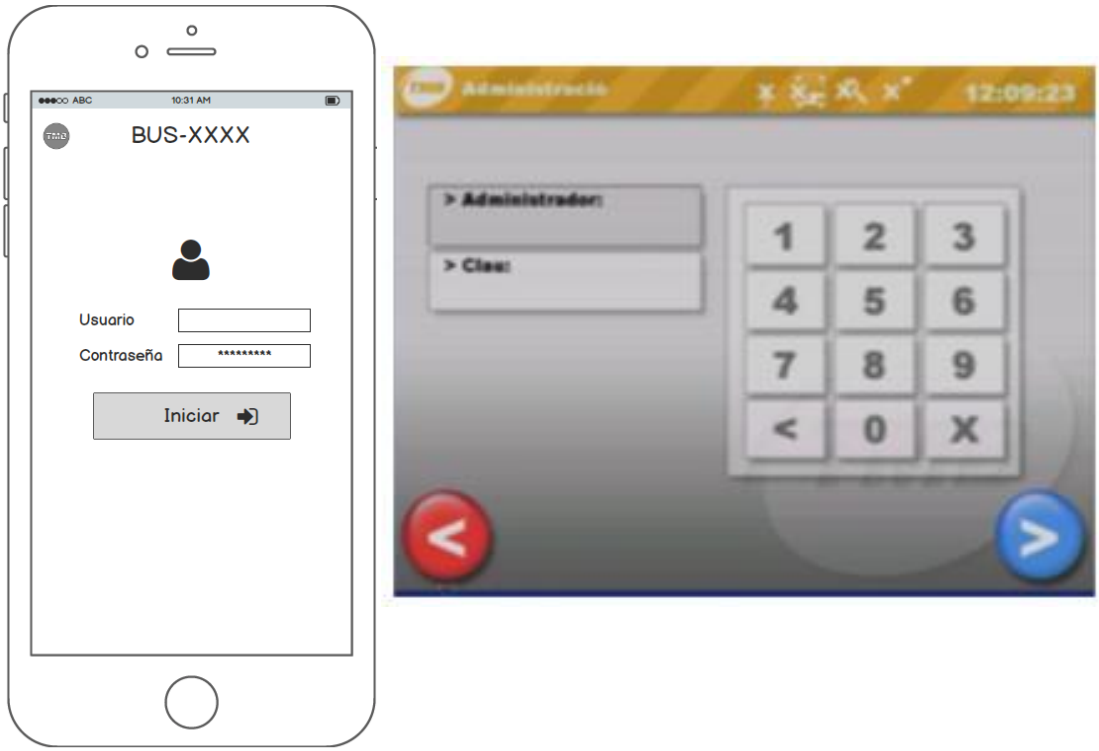


Figura 13 Pantalla de Login

5.1.2 Barra de navegació superior

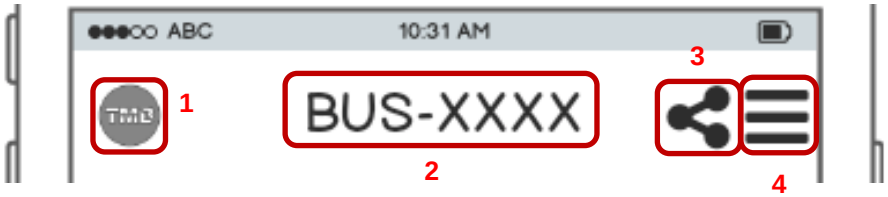


Figura 14 Barra de navegació superior

La barra de navegació superior disposa (d'esquerra a dreta) de:

1. Logotip de l'empresa. Permet tornar a "home" o inici en qualsevol moment.
2. El títol (BUS-XXXX) mostra a l'usuari en quin autobús està realitzant les tasques de manteniment.
3. Botó per a "compartir" que permetrà emmagatzemar en el dispositiu o enviar per correu electrònic informació rellevant relacionada amb els equips de l'autobús. En prémer aquesta opció, apareixerà el menú amb les opcions d'emmagatzemar o enviar per e-mail:

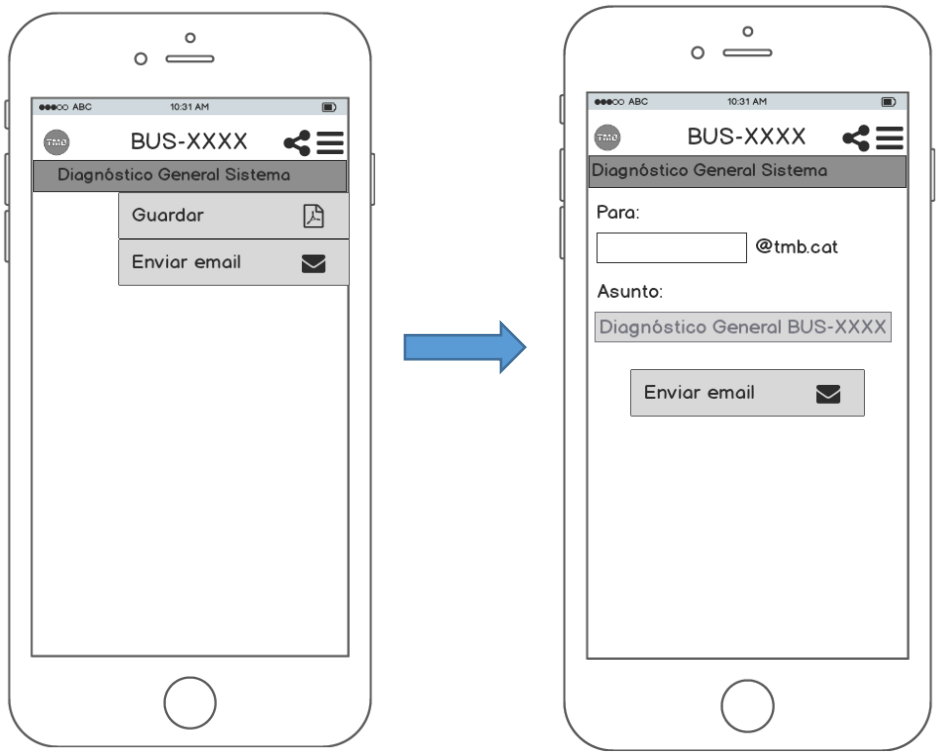


Figura 15 Opcions de compartir resultats (mail o emmagatzemar en dispositiu)

4. La darrera opció permet mostrar/ocultar el menú inicial amb totes les opcions disponibles:

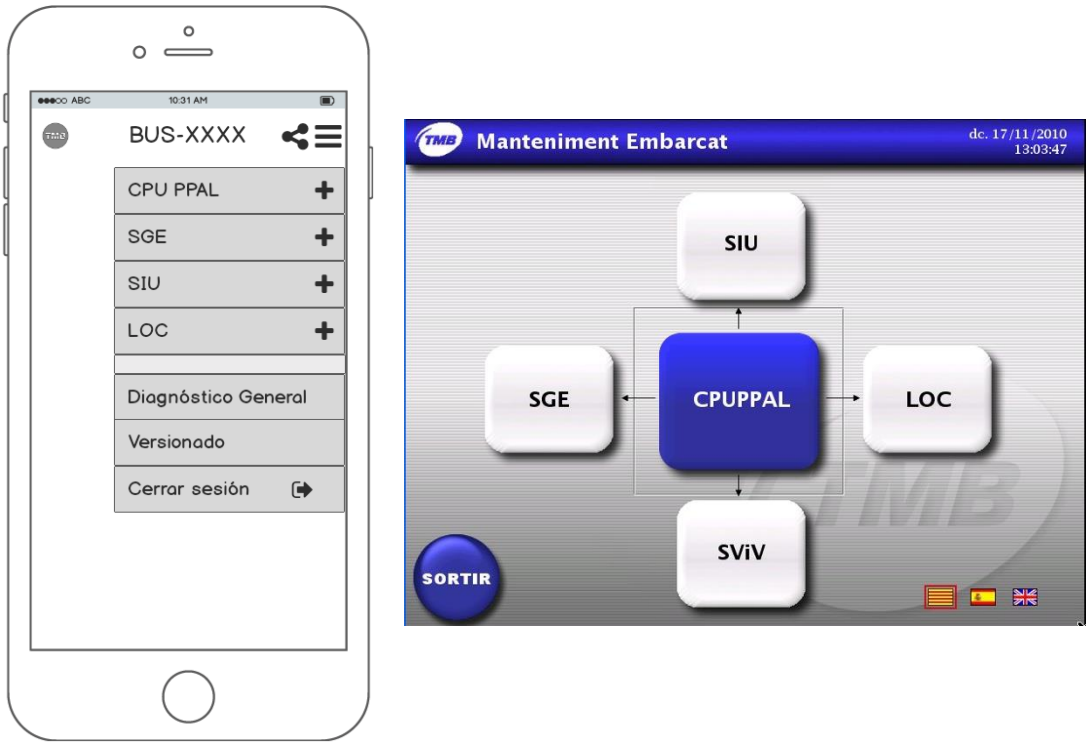


Figura 16 Menú principal

En aquesta pantalla es permet la navegació entre els diferents equips.

En la nova pantalla que s'ha dissenyat s'han introduït noves funcionalitats que són molt útils quan s'està realitzant el manteniment, ja que permeten directament accedir-hi a opcions que actualment tenen una "profunditat de menús" molt gran.

5.1.3 Submenús dels equips

Quan es selecciona un equip en concret, aquest disposa de diferents opcions que permeten visualitzar diferent informació, això es mostra mitjançant l'ampliació dels menús disponibles mitjançant uns menús anomenats "dropdown".

Aquestes opcions no són fixes, es a dir, tant el nombre d'equips com les opcions de cadascun dels equips poden variar mitjançant uns fitxers de configuració (s'explicaran en més profunditat en l'apartat de implementació) on es defineixen els equips, les opcions, les comandes a executar, etc.

Per aquest motiu, en la següent figura nomes es mostra la captura de l'opció CPUPPAL per poder comparar-la amb l'actual:

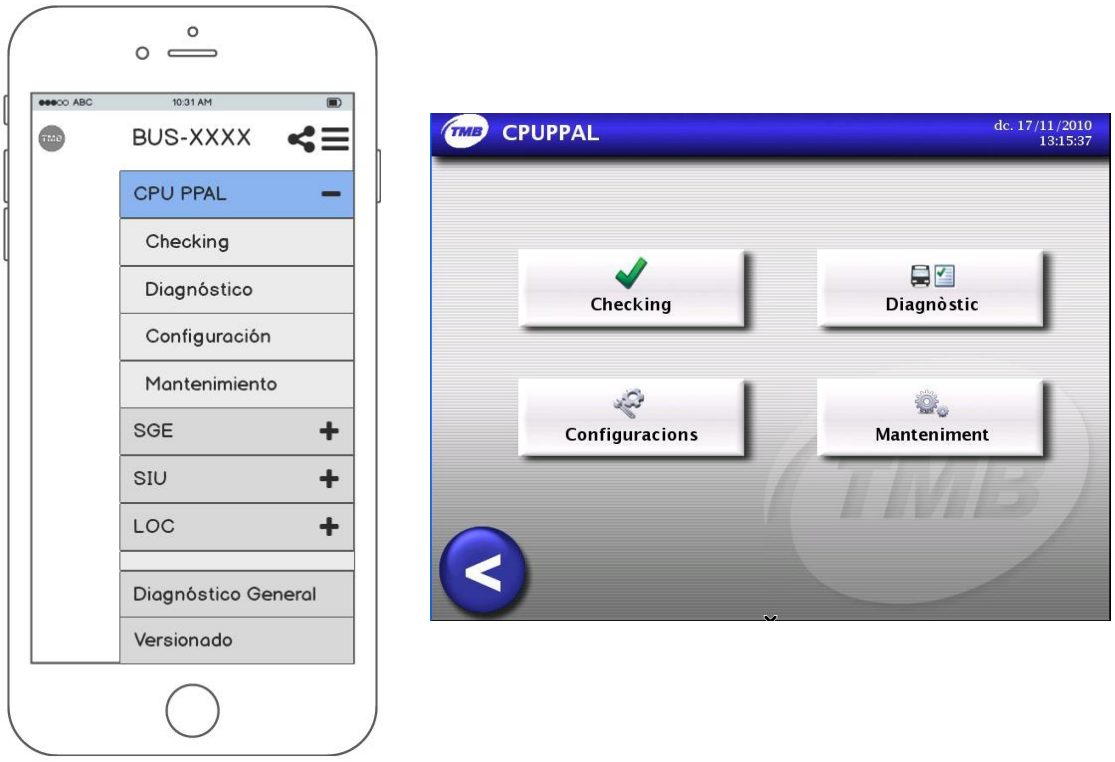


Figura 17 Opcions per equip (CPUPPAL)

5.1.4 Pantalla Checking

Aquesta pantalla mostra d'una ullada els processos més rellevants de l'equip, si estan funcionant correctament, i en cas d'error, permet aprofundir en mostrar informació més detallada:

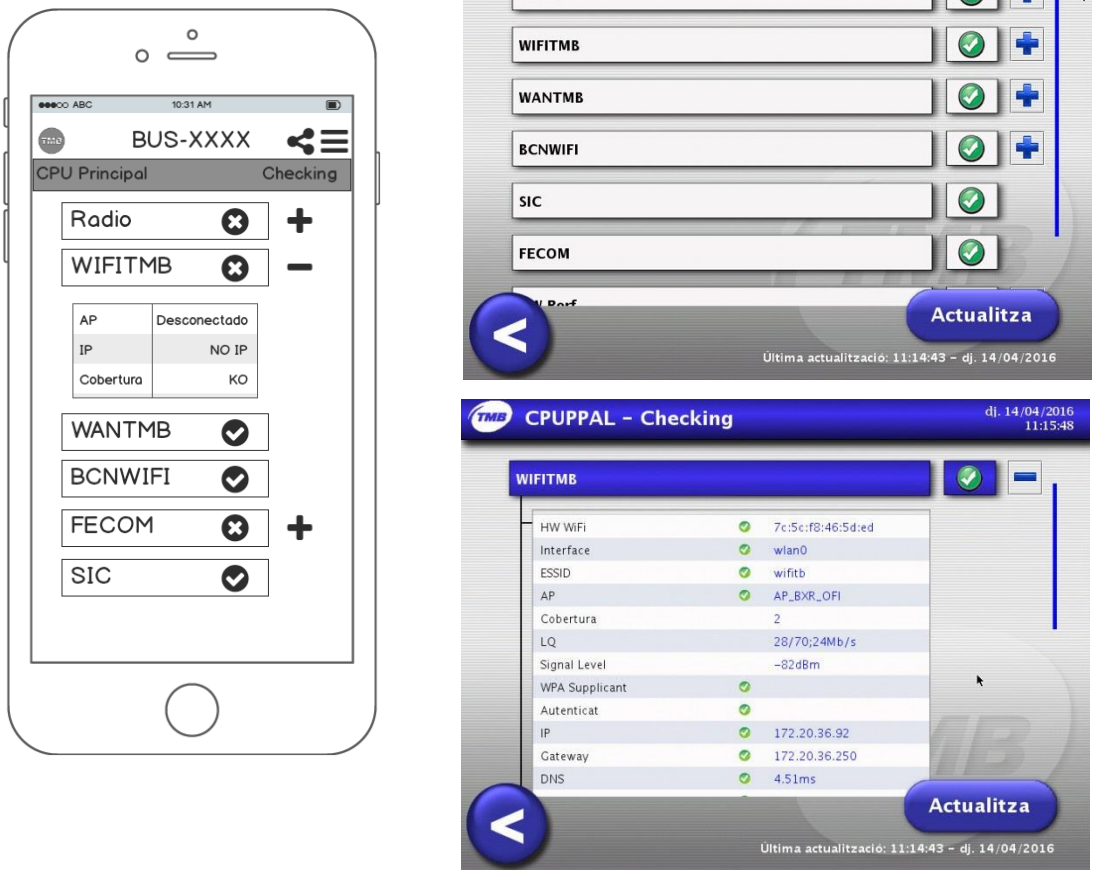
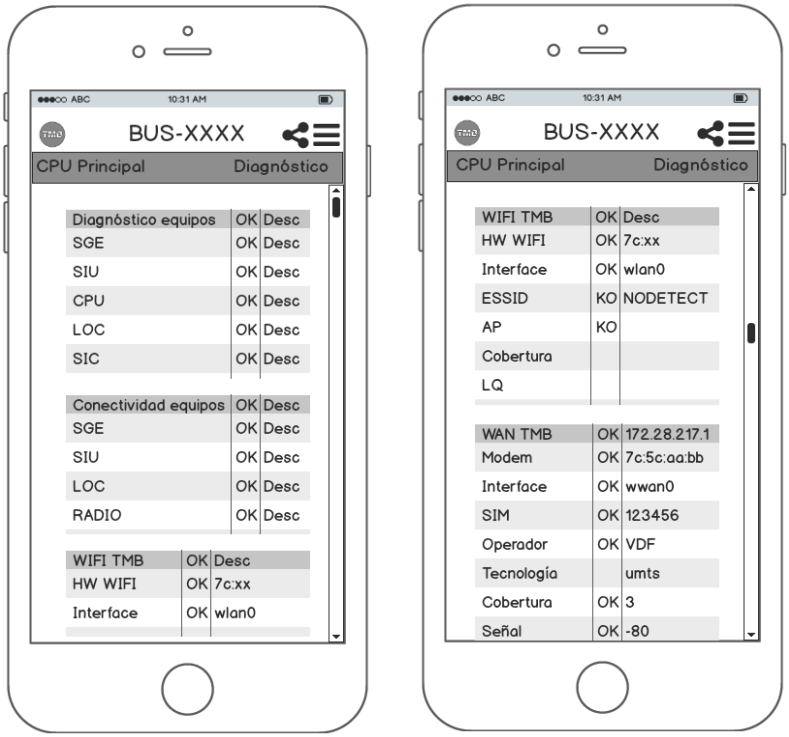


Figura 18 Opció de checking

5.1.5 Pantalla Diagnòstic

Aquesta pantalla permet veure informació de tot l'equip, i si s'està visualitzant el "diagnòstic" de la CPU PPAL, visualitza informació del sistema sencer, es a dir, mostra l'estat general de la resta d'equips, i la informació més important de tot el sistema.

Actualment l'opció "Diagnòstic" de la CPU PPAL es comporta igual que l'opció que hem inclòs en el menú desplegable denominat "Diagnòstic General". S'ha dissenyat així perquè a futur el Diagnòstic General del menú inicial podrà oferir més informació que el de CPU PPAL.



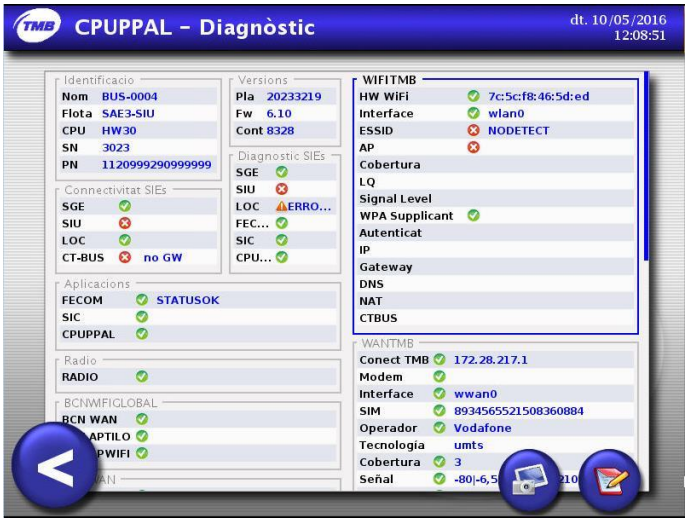


Figura 19 Diagnòstic CPUPPAL

5.1.6 Pantalla Configuració

Aquesta pantalla permet accedir a opcions de configuració dels equips, com són, per exemple, el calibratge del tàctil del TFT, canviar volums del so, configurar números de sèrie, etc.

Figura 20 Menú configuracions

5.1.7 Pantalla de manteniment

Aquesta pantalla permet accedir a accions que pot realitzar el personal de manteniment com per exemple visualitzar logs, reiniciar o apagar l'equip de forma controlada, veure i/o reparar l'estat del sistema de fitxers, etc.

En el wireframe no s'han dibuixat totes les opcions que hi ha en l'actual aplicació perquè potser es realitzen canvis o agrupacions d'opcions.

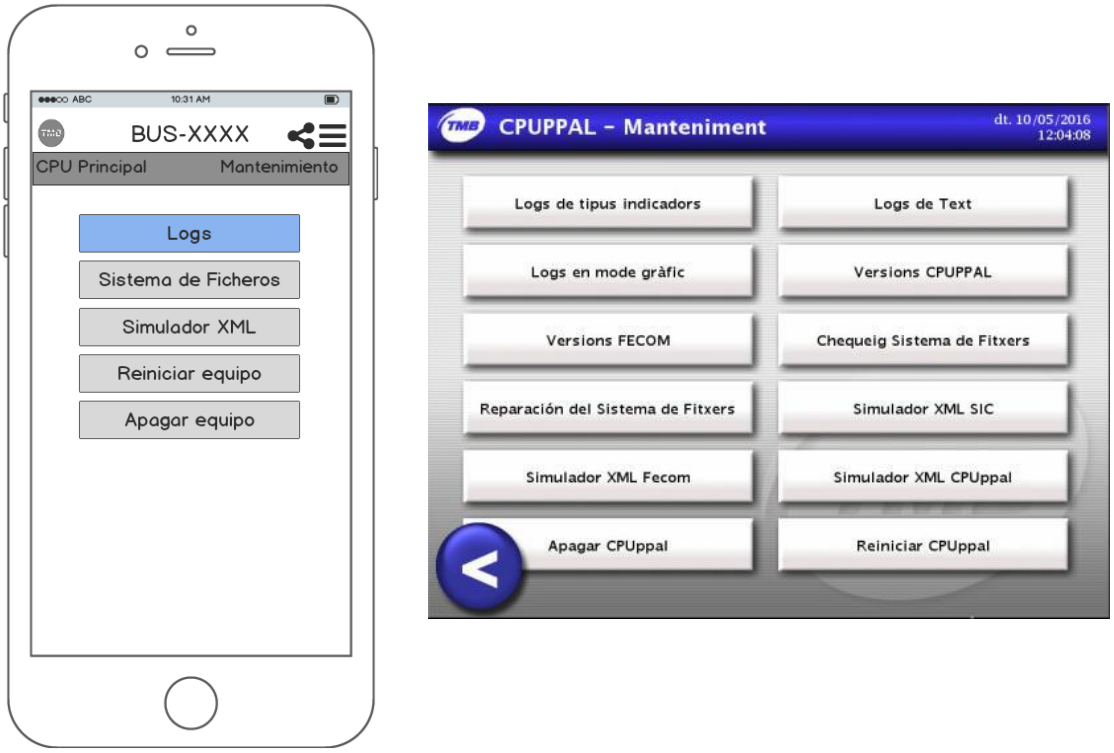


Figura 21 Menú manteniment

5.1.8 Opcions generals del menú principal

Fora de les opcions dels diferents equips trobem 3 opcions que s'han inclòs en el menú principal, que són:

- Diagnóstico General: Inicialment aquesta opció mostrarà la mateixa informació que si seleccionem CPUPPAL > Diagnóstico (Punt 5.1.5) però en el futur, aquesta opció podrà donar més informació en un format similar al que s'ha mostrat en el punt 5.1.5.

- Versionado: Aquesta opció ens permetrà veure les versions dels diferents equips i de les diferents aplicacions que s'estan executant. Actualment no hi ha una pantalla que mostri tota aquesta informació, s'ha d'accedir a cada equip i a les diferents opcions de cada equip per a poder visualitzar les diferents versions:

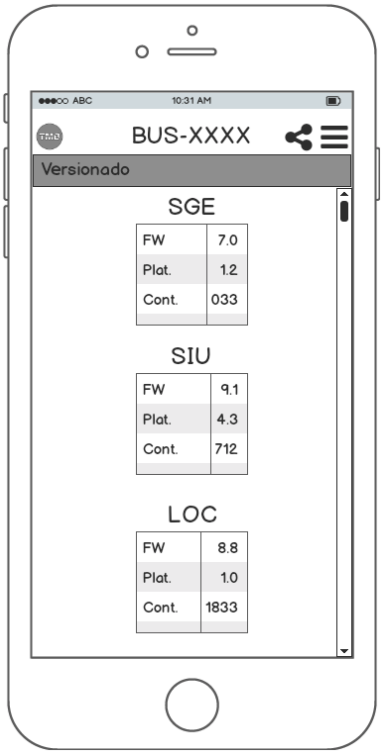


Figura 22 Opció nova "Versionado"

- Cerrar sesión: Aquesta opció permetrà sortir de l'aplicació, i on pren molta importància és quan aquesta aplicació s'executi en la pantalla del conductor. En ser una aplicació web s'haurà d'executar un navegador, i quan es seleccioni l'opció "logout" s'haurà de tancar el navegador per a que el conductor pugui tenir visible la seva aplicació de regulació.

5.1.9 Altres pantalles

Fins aquí s'han mostrat les pantalles principals de navegació de l'aplicació, però n'hi han d'altres com les configuracions de canvi del numero de sèrie, la visualització de logs, etc. que també s'han dibuixat per a tenir un esbós de com s'hauran de visualitzar. Aquestes pantalles s'han inclòs en l'annex d'aquesta memòria.

6. Avaluació Heurística segons els 10 heurístics de Nielsen

Per tal d'avaluar la usabilitat dels dissenys de baix nivell i de les pantalles de la web, s'ha realitzat l'avaluació heurística amb els 10 principis heurístics de Nielsen.

A continuació s'analitzen els resultats d'aquesta avaluació heurística i quins són els principis que s'acompleixin i quins no.

6.1 Visibilitat de l'estat del sistema.

L'usuari en tot moment està informat de les operacions que està realitzant i de la ubicació dins de la pagina web perquè s'ha dibuixat una barra per sota de la barra de navegació que ho indica.

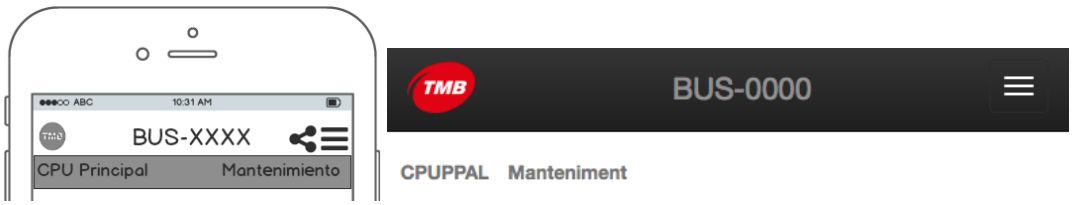


Figura 23 Estat del sistema

I quan es realitza una operació que triga en respondre, es mostra un "loader" per a que l'usuari sàpiga que la web està realitzant una operació i no s'ha quedat bloquejada:

Figura 24 Loader

6.2 Adequació entre el sistema i el món real.

El llenguatge que s'ha utilitzat en el web és el mateix que ja coneix l'usuari degut a que s'han utilitzat els mateixos textos que es fan servir en l'actual aplicació.

La informació s' ha ordenat d'una forma molt similar a com es mostra en l'aplicació actual.

Figura 25 Adequació entre sistema i món real

6.3 Llibertat i control per part de l'usuari.

L'usuari pot sortir davant d'alguna opció que sigui errònia i pot desfer o repetir accions, per exemple, com a la pantalla del conductor no es veurà la barra de navegació, s'ha inclòs un botó de tornar per a que l'usuari pugui desfer l'opció que ha escollit. O per exemple, si s'ha equivocat en introduir un numero en una configuració, també pot retornar. A més, sempre tindrà l'opció de prémer el logotip per tal de tornar a l'inici de l'aplicació:

Figura 26 Botó Tornar

6.4 Consistència i estàndards.

Com el web ha de ser “responsive”, s’ha seguit la convenció del botó “menú” que apareix en la majoria de webs quan la pantalla assoleix una certa mida d’amplada. Conceptualment, els usuaris ja comprenen que aquest botó desplegarà un menú.

De la mateixa manera, el logotipus de l’empresa normalment redirigeix a l’inici del web.

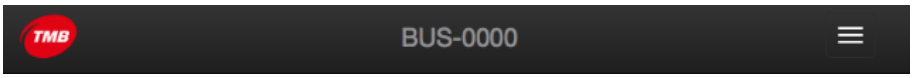


Figura 27 Estàndards

6.5 Prevenció d’errors.

En el primer disseny del web no s’ha implementat cap mecanisme per a la prevenció d’errors per part de l’usuari.

En posteriors fases de disseny, fora de l’abast d’aquest projecte, es pot implementar per exemple un nombre màxim de caràcters per a la configuració del numero de sèrie, que actualment no pot superar els 4 caràcters, però això es faria aprofitant els fitxers de configuració, on s’inclouria un paràmetre nou que indiques el nombre màxim de caràcters que pot ocupar aquest camp. D’aquesta manera no hi haurà paràmetres que poden ser modificables en el futur implementats en el codi, sinó en els fitxers de configuració:

Figura 28 Prevenció d'errors

6.6 Reconeixement abans que record.

L'avantatge de partir d'una aplicació ja existent és que s'ha dissenyat el web de manera molt semblant a l'aplicació original, en conseqüència, els usuaris tenen assolides moltes de les funcionalitats en el seu "model mental", és per això que els elements del web són visibles, comprensibles i molt similars al de l'aplicació actual.

6.7 Flexibilitat i eficiència en l'ús.

En el disseny del web s'han inclòs funcionalitats que permeten a l'usuari guanyar en productivitat, per exemple amb les funcionalitats de "Diagnostic General" i "Versions" que són 2 de les opcions més utilitzades en el manteniment diari.

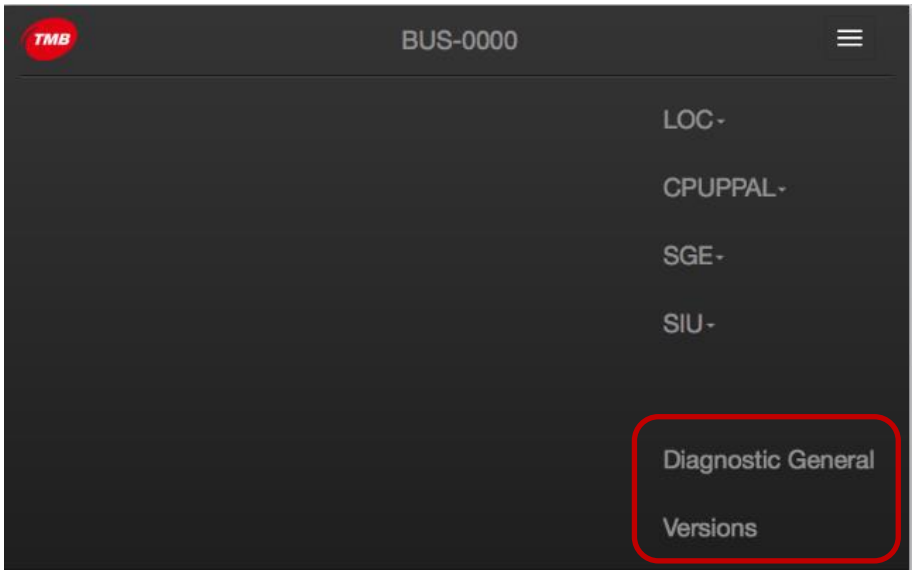


Figura 29 Funcionalitats per guanyar en productivitat

6.8 Disseny estètic i minimalista.

El disseny que s'ha pensat tant amb els wireframes com en la maquetació del web ha estat senzill i no hi ha cap element que distregui a l'usuari, es a dir, no s'han inclòs efectes, molts colors, etc.

6.9 Ajuda als usuaris a reconèixer i diagnosticar els errors i recuperar-se'n.

Degut a que aquesta web no té gaires opcions per a que l'usuari cometi errors, es a dir, està més basada a donar informació que no pas a realitzar moltes configuracions, no s'han dissenyat pantalles d'error per a informar a l'usuari que ha comés alguna errada.

Si s'han dissenyat pantalles per a informar a l'usuari si s'ha produït alguna errada en la crida d'alguna comanda en el sistema, que pot portar a no retornar dades, per exemple davant una caiguda de les comunicacions entre els equips:

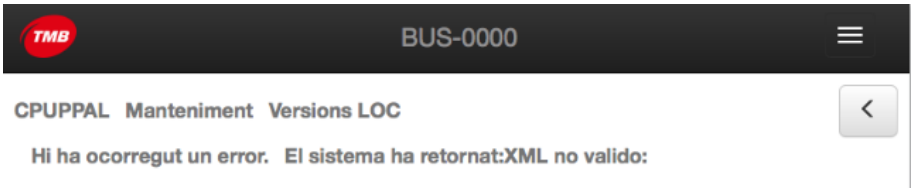


Figura 30 Error en el retorn de l'execució d'una comanda

6.10 Ajuda i documentació.

En aquesta primera fase d'implementació, no hi ha mecanismes d'ajuda, com poden ser "tooltips", opcions d'ajuda, etc.

Com ja hem indicat en anteriors apartats, l'avantatge de "replicar" una aplicació, com és aquest cas, és que l'usuari ja coneix el funcionament de l'aplicació que s'està replicant i com que el disseny realitzat té un aspecte molt similar, això fa que l'usuari ràpidament assoleixi el seu funcionament.

7. Disseny de l'aplicació WEB

L'entorn on s'instal·larà aquesta aplicació i l'arquitectura d'equips que s'ha especificat en punts anteriors poden esdevenir en canvis de diferents aspectes, com poden ser instal·lacions de nous equips, noves opcions que permetin obtenir altres dades o d'altres tipus que encara no es tenen.

És per això que l'aplicació web que es desenvolupi s'ha de poder adaptar a aquests canvis sense la necessitat de realitzar nous desenvolupaments "ad-hoc", es a dir, l'aplicació ha de permetre una certa llibertat de configuració per tal d'abordar noves funcionalitats i nous equips mitjançant fitxers de configuració, que un cop interpretats facin que el web pugui ser escalable en funcionalitats, equips, etc.

Per altra banda, les comunicacions que es realitzen entre els diferents equips són missatges SOAP-XML i aquests missatges s'han d'interpretar per tal de mostrar la informació demanada.

En darrer terme, per aconseguir les respostes XML dels diferents equips s'aprofiten scripts realitzats en BASH que demanen informació a tots els equips de la xarxa embarcada. Per aquest motiu, aquest desenvolupament ha de ser capaç de fer crides al sistema operatiu i capturar les dades que rebi, per després mostrar-les en la web.

7.1 Estructura de l'aplicació WEB

Per tal de mostrar com s'ha desenvolupat l'aplicació WEB i de quines parts està formada, la figura 31 mostra l'estructura de l'aplicació, per a continuació definir part per part de que s'encarrega cada element de l'aplicació.

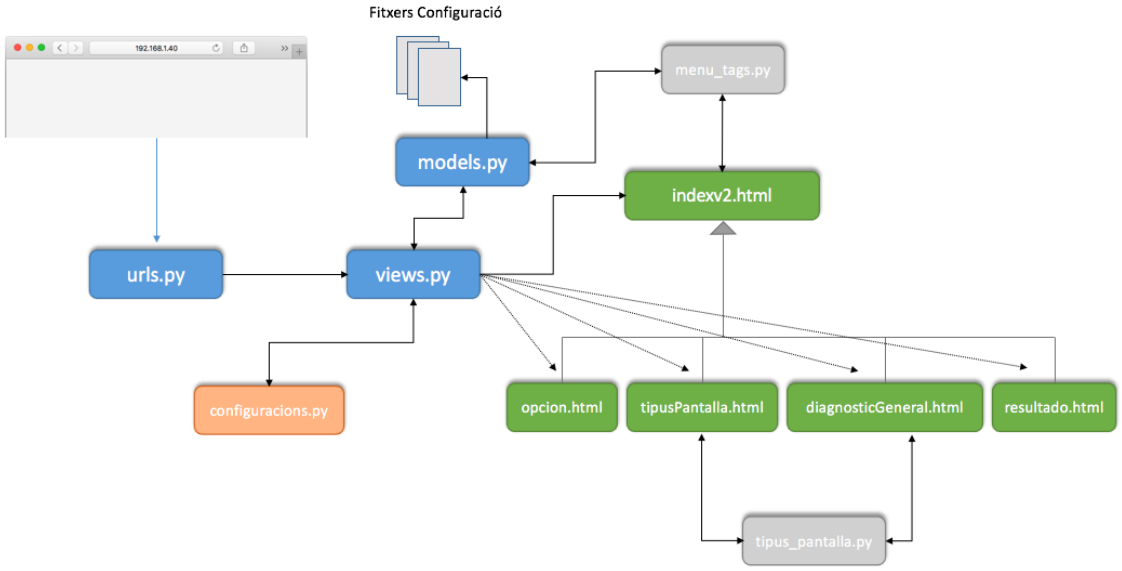


Figura 31 Estructura aplicació WEB

Cadascun dels elements mostrats en la figura 31 tenen una funció en concret, i les fletxes indiquen de quina manera interactuen els diferents elements entre si.

1. El Navegador WEB. S'encarrega de fer les peticions, rebre respostes, i interpretar el codi html per a mostrar-ho a l'usuari.

2. Fitxers propis d'un projecte Django

    a. `urls.py`. Defineix les URLs que seran admeses en l'aplicació i cap a quin mètode del fitxer "`views.py`" han de dirigir-se quan són cridades. Aquestes URLs es poden definir mitjançant expressions regulars.

    b. `views.py`. És el fitxer de vistes on estan els mètodes que s'executen quan s'adreça una URL en concret. Aquests mètodes poden rebre informació de l'usuari, processar-la amb l'ajuda d'altres mètodes, i cridar a un fitxer html per a que mostri aquesta nova informació.

    c. `models.py`. Aquest fitxer conté una sèrie de mètodes que s'encarreguen de la recol·lecció de totes les dades que puguin fer

falta en les vistes. Per exemple, s'encarreguen de la lectura dels fitxers de configuració.

3. Fitxers de configuració. Són fitxers de text pla on s'indiquen totes les funcionalitats que es podran visualitzar i l'usuari podrà escollir.

4. Templates HTML. Tenen el codi necessari per a mostrar o demanar informació a l'usuari. Aquests fitxers HTML poden contenir estructures de dades recaptades a traves de mètodes Python per a mostrar-les aprofitant elements HTML.

    a. `indexv2.html`. Aquest és el fitxer "inicial" o "home", es a dir, quan s'executa l'aplicació i s'accedeix a la URL inicial, és la pagina web que es visualitza. La resta de fitxers html de la llista estenen d'aquest fitxer, això significa que és similar a una herència en classes UML. Quan es visualitzin els següents fitxers, primer es mostrarà el contingut d'aquest fitxer i a continuació el contingut dels altres fitxers. Això és molt útil perquè, per exemple en el nostre cas, la barra de navegació que hi ha implementada en aquest fitxer es mostrarà en totes les pantalles, i amb aquesta "herència" nomes implementem una vegada el codi de la barra de navegació.

Figura 32 indexv2.html

b. `opcion.html`. Aquest fitxer s'encarrega de 2 funcions. La primera és mostrar l'equip i l'opció escollida (en la figura 36 LOC Manteniment) i la segona, mostrar una llista de botons amb els menús disponibles de cada equip i de cada opció que hi ha definides en els fitxers de configuració (en la figura 33 són totes les opcions disponibles per al menú de manteniment de l'equip LOC).

Figura 33 opcion.html

c. `tipusPantalla.html`. Aquest fitxer mostra els diferents tipus de pantalla que hi han disponibles i que es defineixen en els fitxers de configuració (es veuran la definició dels tipus de pantalla en els fitxers de configuració). Aquestes pantalles poden mostrar informació de diferent tipus, com poden ser dades dels equips o del propi autobús, modificacions de paràmetres de configuració, etc.

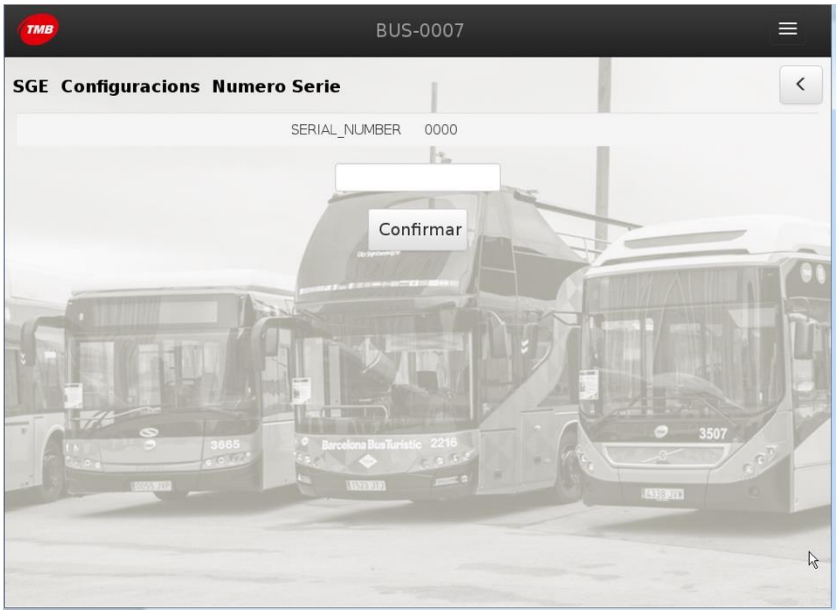


Figura 34 tipusPantalla.html (modificacio1)

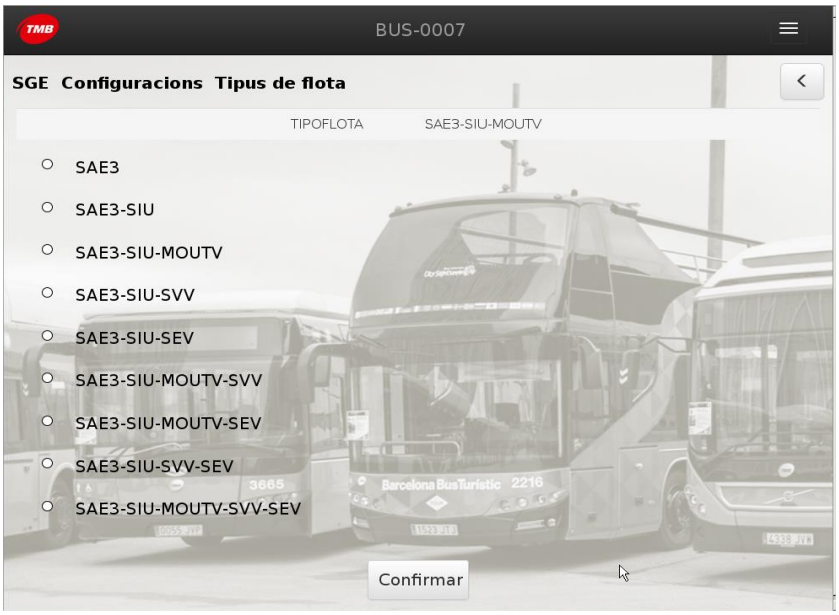


Figura 35 tipusPantalla.html (modificacio2)

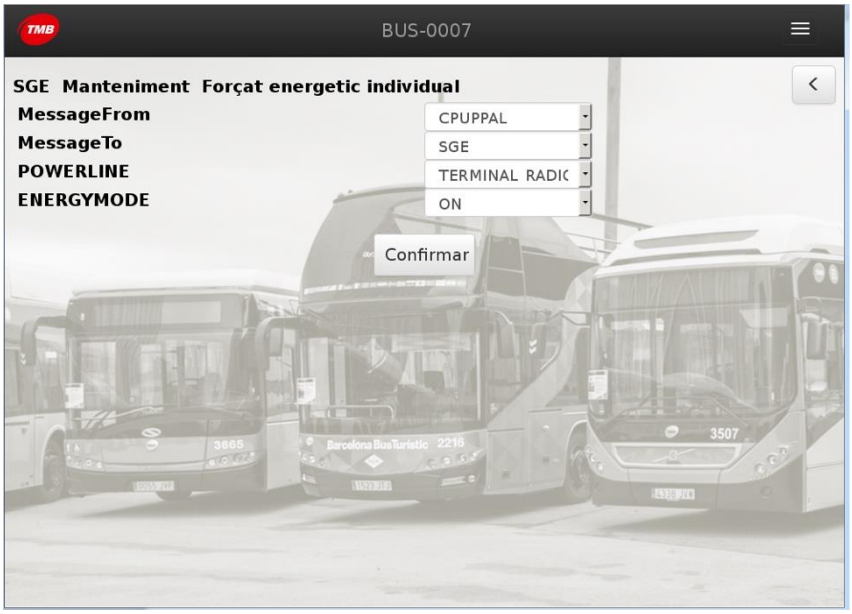


Figura 36 tipusPantalla.html (comanda4)

d. `diagnosticGeneral.html`. Aquest fitxer s'encarrega de mostrar tota la informació general de la que es disposa en el sistema embarcat. Són una sèrie de taules que mostren diferents estats (OK/KO/WARNING) dels diferents elements del sistema.

e. `resultado.html`. Aquest fitxer mostra possibles missatges d'error. És un fitxer genèric al que se li passa un diccionari indicant si hi ha ocorregut un error, i el text informatiu de l'error.

Figura 37 resultado.html

5. Resta de fitxers de l'aplicació.

    a. `configuracions.py`. Aquest fitxer conté els mètodes necessaris per a muntar missatges XML a partir de valors que hagin pogut estar introduïts per l'usuari i després s'encarrega d'enviar-los a l'equip que correspongui.

    b. Fitxers de tags personalitzats. Aquests fitxers defineixen un tipus de mètode que pot ser cridat directament des del codi HTML i retornar valors per tal de mostrar-los en HTML. Són molt útils perquè proporcionen molta potencia per a realitzar cerques, agrupar dades, etc.

        i. `menu_tags.py`. És l'encarregat de proporcionar-li al fitxer inici totes les opcions i equips disponibles a traves dels fitxers de configuració. A més, proporciona la "calca" del bus que és el títol de la barra de navegació.

        ii. `tipus_pantalla.py`. Aquest té la lògica de tots els tipus de pantalla definits en el punt 7.2.4.1 dels fitxers de configuració. S'encarrega de la cerca d'informació, d'executar les comandes Linux que siguin necessàries, capturar els resultats, aplicar filtres XPATH per a extreure el valor de variables dels fitxers XML, etc.

7.2 Funcionament de l'aplicació

En els següents apartats s'indica com seria el funcionament de tota l'aplicació web, des que es demana per la seva adreça URL fins que es mostren resultats pel navegador.

Per a definir aquest funcionament i quins elements o processos intervenen, aprofitarem l'esquema de la figura 31 i pas a pas es mostrarà com interactuen les diferents "peces" que formen l'aplicació.

7.2.1 Crida des del navegador a la URL inici

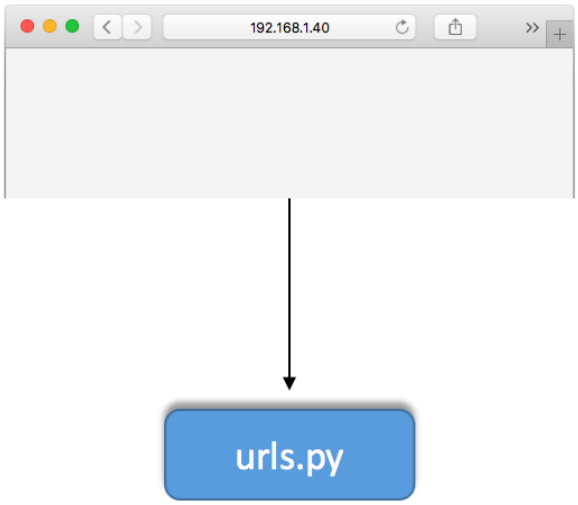


Figura 38 Crida des del navegador a la pagina d'inici

En introduir des d'un navegador l'adreça web d'inici (si executem mitjançant Python aquest projecte l'adreça seria /myApp/), el fitxer `urls.py` s'encarrega de cridar el mètode del fitxer `views.py` definit.

L'aspecte del fitxer urls.py es mostra a continuació (nomes és un fragment):

```
urlpatterns = [
    url(r'^$', views.index, name='index'),
    url(r'^options/(?P<equipo>\D+)/(?P<opcion>\D+)/', views.options, name='options'),
```

Figura 39 urls.py

Cada línia defineix una possible adreça url permesa mitjançant expressions regulars.

En la figura 40 es defineix cada camp:

Figura 40 Línia urls.py

1. Patró d'expressió regular. Quan la correspondència és exacta, es cridarà al mètode del fitxer `views.py` indicat en el segon paràmetre.

2. Indica el mètode del fitxer `views.py` que s'haurà d'executar en coincidir l'adreça URL amb el patró (en aquest cas el mètode índex del fitxer `views.py`).

3. Defineix el nom amb el que es podrà readreçar des d'altre fitxer HTML a aquest mètode del fitxer `views.py`.

En els patrons d'expressions regulars es poden incloure paràmetres que després es passaran al mètode corresponent, per exemple en la segona línia mostrada del fitxer `urls.py` tenim que:

```
    url(r'^options/(?P<equipo>\D+)/(?P<opcion>\D+)/', views.options, name='options'),
```

Figura 41 Línia urls.py amb paràmetres

Per exemple, en introduir la url "`myApp/options/SGE/manteniment`" es cridarà al mètode "`options`" de `views.py` amb els paràmetres "`equipo=SGE`" i "`opcion=manteniment`".

7.2.2 Crida a mètodes de views.py

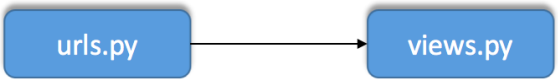


Figura 42 Enllaç urls.py amb views.py

Quan es produeix la coincidència amb un dels patrons de l'expressió regular del fitxer `urls.py`, es crida al mètode corresponent de `views.py`.

L'aspecte d'aquest fitxer es mostra en la següent captura:

```
def index(request):
    return render(request, 'myApp/indexv2.html')

def options(request, equipo, opcion):...

def tipusPantalla(request, equipo, opcion, numMenu):...

def modificaciones(request, equipo, opcion, missatgeMenu):...
```

Figura 43 views.py

Com es pot observar, per a les urls que hem mostrat en la captura del fitxer `urls.py` hi han els corresponents mètodes que seran cridats, amb els paràmetres d'entrada que corresponguin.

Els mètodes de `views.py` principalment s'encarreguen d'aplicar certes regles de negoci a dades i després renderitzar els fitxers html en funció del mètode que hagi estat cridat.

En les captures del punt 7.1 ja hem vist les diferents pantalles html que podrà veure l'usuari en funció del mètode de `views.py` que s'hagi executat. Però aquest fitxer també ha de carregar dades per a després mostrar-les en les pantalles html i/o rebre dades introduïdes per l'usuari per tal de realitzar configuracions en els equips embarcats.

7.2.3 Càrrega de dades

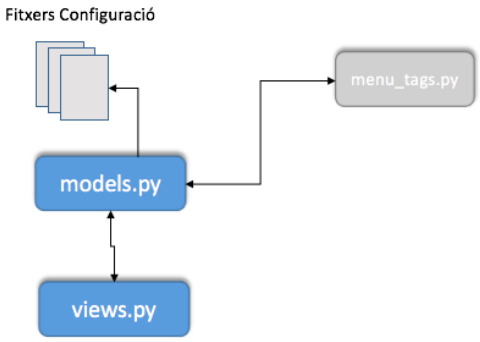


Figura 44 Càrrega de dades

La carrega de dades pot ser cridada per 2 llocs diferents, per un costat per el fitxer `views.py` que aprofitarà les dades per a muntar el fitxer html que es mostrarà a l'usuari, o per altra banda, per un fitxer de "tags personalitzats" que també s'encarrega de proporcionar dades al fitxer html per a que el contingut d'aquest html no sigui estàtic sinó dinàmic.

Sigui per la banda que sigui, el fitxer `models.py` s'encarrega de carregar i crear l'estructura de dades que sigui necessària per a muntar inicialment els menús, opcions, equips, etc. a partir dels fitxers de configuració. Aquests fitxers de configuració es llegeixen sencers una sola vegada amb la primera crida al fitxer `models.py` ja que es crea una classe Singleton de la qual nomes existirà

una instancia i la resta de vegades que s'accedeixi a la classe seran operacions de lectura. Com que nomes es llegeixen un cop els fitxers de configuració a l'inici de l'execució de l'aplicació, si es porta a terme una modificació en aquests fitxers s'ha de reiniciar l'aplicació per a que es puguin llegir les noves opcions configurades.

7.2.4 Lectura dels fitxers de configuració

Per a que l'aplicació sigui escalable i permeti introduir noves opcions, equips, etc. es necessiten uns fitxers de configuració i aquests fitxers han d'incloure el necessari per a reflectir ampliacions i/o modificacions.

Com a punt de partida d'aquests fitxers de configuració s'han aprofitat els que hi ha en l'aplicació actual i s'han modificat en la mesura que ha estat necessari per a acoblar-los en el nou disseny de l'aplicació web.

Aquests fitxers de configuració tenen les següents característiques:

- El nom del fitxer es el nom de l'equip amb l'extensió "`.properties`". Aquesta extensió de fitxer es degut a que l'aplicació actual instal·lada en l'equip es va desenvolupar en JAVA.

  El fitxer de configuració de cada equip pot disposar de diferents seccions. Aquestes seccions són les que es mostraran en el menú de la web i es defineixen entre claudàtors:

  i. Checking → `[checking]`
  ii. Diagnostic → `[diagnostic]`
  iii. Configuracions → `[configuracions]`
  iv. Manteniment → `[manteniment]`

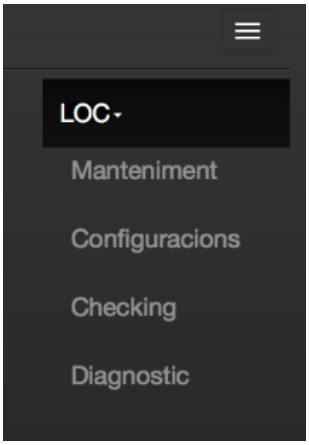


Figura 45 Seccions de l'equip LOC

Per afegir-ne una altra secció seria suficient amb la inclusió al final del fitxer d'una nova secció escrita entre claudàtors:

v. Nova Secció → `[nova seccio]`

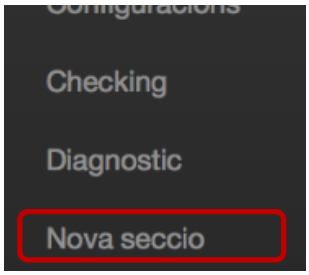


Figura 46 Inclusió nova secció

- Cada secció contindrà els diferents menús que es mostraran en la web, i cada menú té uns paràmetres de configuració fixes i uns altres que depenen del tipus de funcionalitat que s'estigui configurant.

  A continuació es defineixen els paràmetres que són fixes en cada configuració de menús i els valors que poden tenir, i a l'annex s'inclouen altres camps que poden ser utilitzats per aportar-li més valor i funcionalitat a l'opció.

7.2.4.1 Tipus de pantalla

El tipus de pantalla ens permet saber de quina manera es mostrarà la informació en el web i quins tipus de resultats s'obtindran en seleccionar l'opció.

Els diferents tipus de pantalla que hi han definits actualment són:

- Comanda1: Executa una acció (comanda Linux) i no fa servir el paràmetre retornat, es a dir, si s'ha pogut executar correctament o no.
- Comanda2: Executa una acció (comanda Linux) però en un altre equip, amb la qual cosa necessita accedir-hi a l'equip per SSH i executar la comanda.
- Comanda3: Executa una comanda per a que un equip li retorni un XML amb informació relativa al seu estat.
- Comanda4: Executa una comanda que es composa de diferents valors configurables per l'usuari, com són desplegables, sliders, etc.
- Modificacio1: Executa una comanda per a modificar un camp d'un equip que ha estat introduït per l'usuari en mode text.
- Modificacio2: Executa una comanda per a modificar un camp d'un equip però en format llista d'opcions per a seleccionar-ne una d'elles.
- Consulta2: Executa una consulta XML a un equip i de la resposta que rep selecciona i mostra els valors que s'hagin de mostrar, (especificats en els fitxers de configuració).
- SimuladorXML: Aquesta opció permet mostrar una sèrie d'opcions que envien un fitxer XML precarregat a un equip en concret.

Aquesta llista de tipus de pantalles es podria ampliar amb altres funcionalitats que no estiguin contemplades en les pantalles definides anteriorment, tenint en compte que s'hauria de definir una pantalla per a mostrar la informació que retornarà la nova opció (codi HTML) i la lògica necessària per a interpretar la informació que s'ha de demanar i la que es rep (codi Python).

7.2.4.2 Text de l'opció

És el text que es mostrarà a l'usuari i li permetrà entendre que és el que podrà fer amb aquesta opció. Hi han definits 3 idiomes, català, castellà i anglès, però en aquesta implementació nomes s'ha tingut en compte el missatge en català.

7.2.4.3 Estat de l'opció

Aquest estat nomes indica si l'opció està disponible per a que la seleccioni l'usuari o no.

S'ha especificat aquest camp de configuració per al desenvolupament i és molt útil per a depurar opcions noves, es a dir, es pot definir un nou menú i mentre està en període de proves, en producció aquest camp es desactiva i no pot ser seleccionat pels usuaris.

7.2.4.4 Icona

S'ha inclòs per a indicar la ubicació d'una icona i poder-la mostrar en les opcions que pot escollir l'usuari. Aquest camp es pot deixar buit si no hi ha icona que mostrar.

7.2.4.5 Comanda

Aquest camp definirà la comanda Linux que s'ha d'executar per tal d'obtenir les dades que espera l'usuari. Si la comanda necessita de paràmetres extra per a executar-se, hi han altres camps per a definir-los.

7.2.4.6 Exemple de Menú

En la figura 47 es mostra un exemple de menú de configuració.

```
############
## Menu 5
############
menu5.tipusPantalla=Comanda2
menu5.missatgeMenu.ca=LOGS funcionamiento
menu5.missatgeMenu.es=LOGS funcionamiento
menu5.missatgeMenu.en=LOGS funcionamiento
menu5.estat=activat
menu5.icon=
menu5.comanda=/usr/local/tmb/scripts/sshlogin.exp loc cat funcionamientoLOC*.log
```

Figura 47 Menú exemple

Com es pot veure, cada línia del menú comença pel nom del menú, que no es mostrarà a la web però serveix per agrupar les opcions d'un mateix menú, i els

paràmetres estan separats per un punt. El valor del paràmetre s'inclou després del signe "=".

7.2.5 Estructura de dades per emmagatzemar els fitxers de configuració

Per a mantenir un alt grau d'abstracció i escalabilitat s'ha decidit no tenir classes a mida per emmagatzemar la informació dels fitxers de configuració, que ens faria tenir una forta restricció a l'hora d'emmagatzemar les dades, sinó s'ha decidit emprar un diccionari que és una estructura de dades amb una clau única i el seu valor (que pot ser alhora un altre diccionari). Això que ens permetria crear configuracions de quasi qualsevol tipus.

El diccionari amb les dades dels fitxers de configuració tindria la següent estructura:

```
{
  'LOC':
  {
    'manteniment':
    {
      'menu1':
      {
        'tipuspantalla': 'Consulta2',
        'missatgemenu':
        {
          'ca': 'Localitzacio GPS',
          'es': 'Localizacion GPS',
          'en': 'GPS Localization'
        },
        'estat': 'activat',
        'icona': ''
    },
    'diagnostic':
    {
      ...
    }
  }
  'SIU':
  {
    'manteniment':
      …
```

Jeràrquicament, el diccionari té com a claus els equips disponibles al sistema i com a valors de cada equip hi han diccionaris amb les diferents opcions descrites en els fitxers de configuració.

Amb aquest disseny els fitxers de configuració poden créixer de la manera que vulguem i indiferentment del tipus de dades que han d'emmagatzemar,

quantitat de dades, etc. Per aquesta flexibilitat s'ha optat per un diccionari en comptes de definir classes a mida.

La figura 48 mostra de manera gràfica com s'interpretaria el diccionari d'equips i opcions com si fossin carpetes:

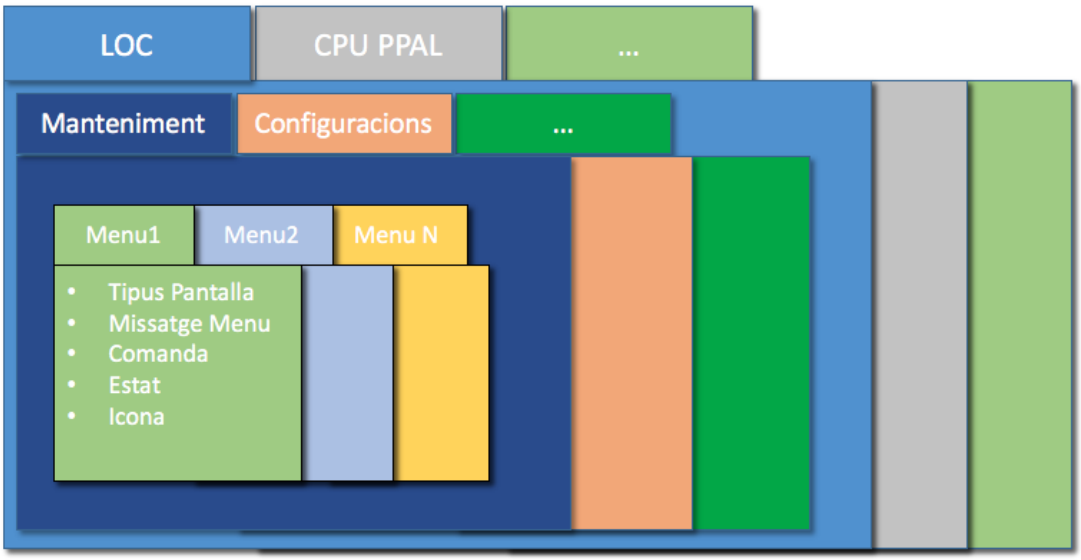


Figura 48 Diccionari de configuracions

Per exemple, per accedir-hi al tipus de pantalla del menú numero 1 del menú de manteniment de l'equip Localitzador, es faria de la següent manera:

```
Diccionari['LOC']['manteniment']['menu1']['tipuspantalla']
```

7.2.6 Inclusió de dades dinàmiques en els fitxers HTML

Tal i com hem vist en l'estructura de l'aplicació WEB, hi han una sèrie de plantilles HTML però aquests fitxers són estàtics, es a dir, inicialment nomes mostren una informació (botons, etiquetes, camps de text, ...) però nosaltres necessitem cert grau de llibertat per a mostrar un tipus d'informació o un altre en funció dels fitxers de configuració i dels resultats que s'obtinguin en interrogar als diferents equips del sistema. Per aquest motiu, en Django es poden crear uns tipus de mètodes que es poden cridar des del fitxer html per a

que executin certs processos, i les dades que es retornin, mostrar-les en el WEB.

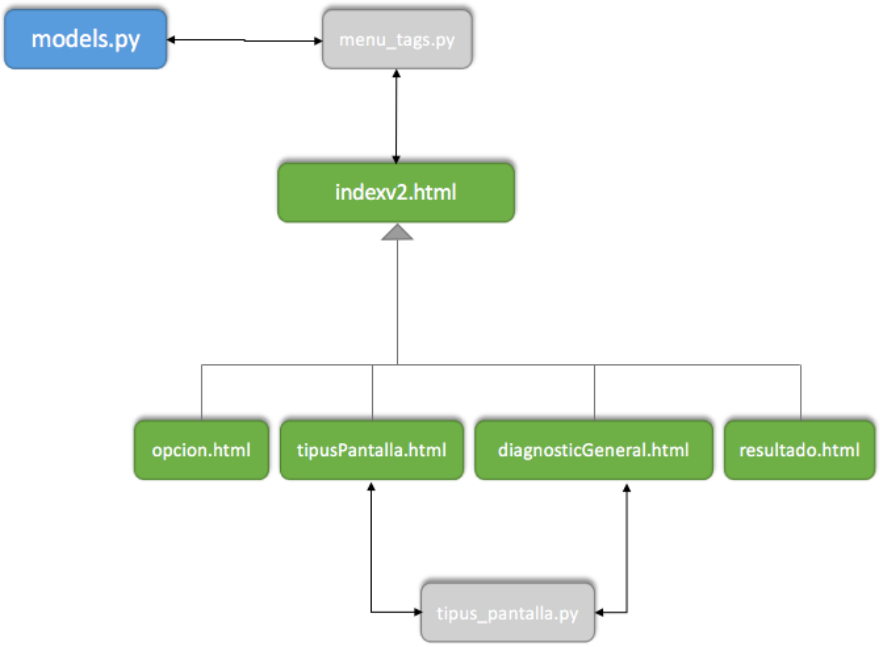


Figura 49 Fitxers tags personalitzats

En l'estructura de l'aplicació tenim per una banda que el fitxer "indexv2.html" pot interactuar amb "`menú_tags.py`" i el fitxer "`tipus_pantalla.py`" pot interactuar amb "`tipusPantalla.html`" i "`diagnosticGeneral.html`".

7.2.6.1 Interacció menú_tags.py i indexv2.html

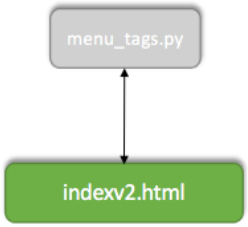


Figura 50 Enllaç entre menú_tags i indexv2.html

Tal i com s'ha indicat en punts anteriors, la pantalla d'inici indexv2.html, de la que estenen la resta de fitxers HTML, nomes s'encarrega de mostrar la barra de navegació amb els menús disponibles estrets dels fitxers de configuració.

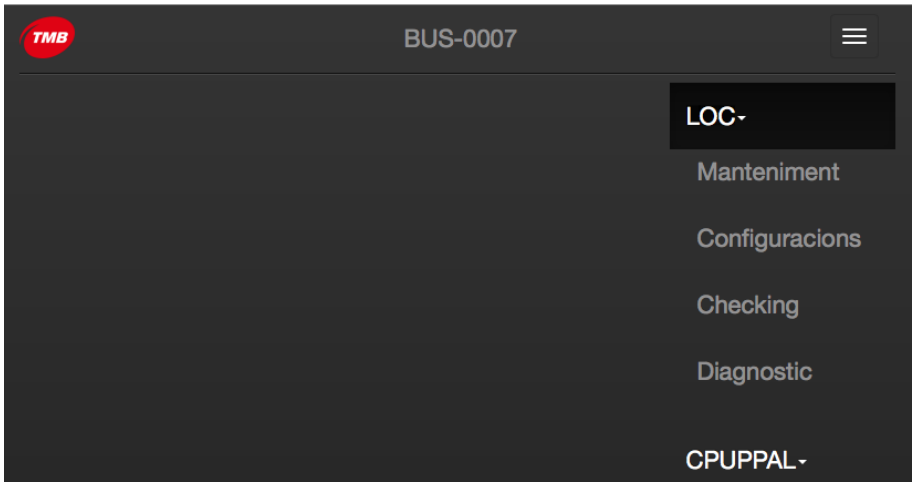


Figura 51 Barra de navegació (comprimida)

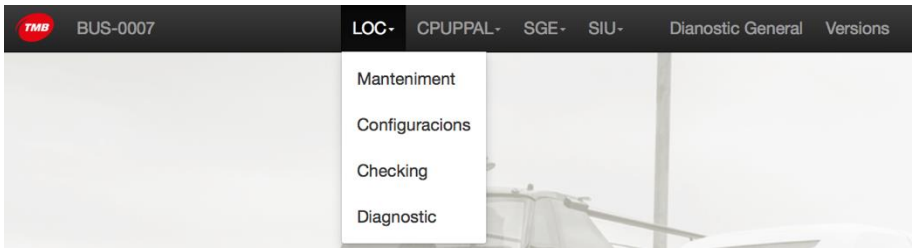


Figura 52 Barra de navegació

Django permet utilitzar una sèrie de tags en els fitxers html per a dotar de certa lògica al fitxer HTML, i "dibuixar" en funció de variables, bucles, etc. objectes, textos, ... A l'annex es mostraran alguns dels tags que s'han utilitzat en aquest projecte per tal de veure la potencialitat d'aquests tags.

Com ja s'ha indicat, amb Django es poden definir fitxers de "tags personalitzats" que ens facilitin una tasca en concret i des del fitxer html cridar a funcions d'aquests fitxers i mostrar les dades que retornin en el format que es decideixi.

Per exemple, en les captures de la figura 51 i 52 que mostren els menús d'equips i opcions, quan s'ha de dibuixar aquesta barra de navegació es crida a

una funció “personalitzada” que retorna del diccionari que s’ha muntat amb les opcions dels fitxers de configuració, els equips i els menús disponibles. Mitjançant un bucle es recorre tot el diccionari amb les opcions disponibles i es dibuixen en la barra de navegació. Per aquest motiu, entre d’altres, l’estructura de diccionari emprada en aquest projecte ens funciona molt bé perquè permet iterar molt fàcilment, capturar les claus i els valors d’una forma molt senzilla i fer escalable l’aplicació en funcions, equips, menús, etc. perquè no hi ha cap limitació a l’hora d’iterar sobre les opcions del diccionari.

7.2.6.2 Interacció tipus_pantalla.py amb tipusPantalla.html i diagnosticGeneral.html

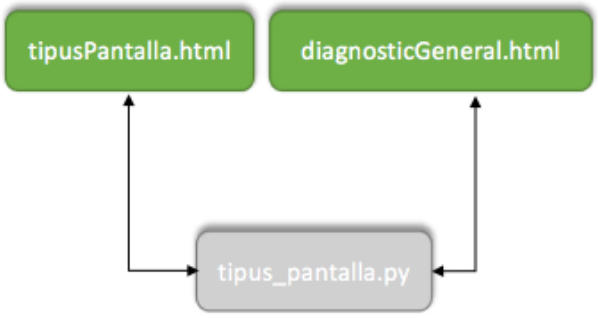


Figura 53 tipus_pantalla.py i fitxers html

Un cop es demana mostrar diferents tipus de pantalla, dades dels equips, configuracions, etc. fa falta que tinguem certa informació per a tomar decisions a l’hora de muntar la pantalla html.

Pel que fa a la lògica per a mostrar els diferents tipus de pantalla tenim un altre fitxer de tags personalitzats que ens ajuden a realitzar aquestes accions. Dins d’aquestes funcions “personalitzades” tenim com es dibuixa cada pantalla, l’execució de comandes en Linux, la interpretació de les respostes XML-SOAP, etc. Podríem dir que la lògica per a mostrar tots els valors que poden proporcionar els equips, així com les modificacions que podem realitzar com a usuaris, es realitza amb l’ajuda dels mètodes inclosos en aquest fitxer de tags personalitzats.

7.2.7 Lectura dels resultats XML

Totes les comunicacions que podem fer entre els diferents equips del sistema i l'aplicació web que s'ha desenvolupat es realitzen mitjançant missatgeria XML; l'aplicació envia un missatge de consulta i l'equip consultat respon amb un missatge XML.

Els missatges de consultes son fitxers XML que ja estan disponibles en l'equip, llavors l'equip nomes a de cridar al procés que s'encarregui d'enviar un cert fitxer xml a un equip (els executables i la ubicació dels fitxers a enviar estan definits en els fitxers de configuració), executar-lo i esperar la resposta.

Un cop es rep la resposta, s'ha de processar per tal d'extreure les dades que s'han demanat (variables definides en els fitxers de configuració).

A continuació es mostra un exemple de fitxer resposta XML d'un equip:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
        <soap:Header>
                <h:SIEheader xmlns:h="urn:TMB:SIE:Header">
                        <h:MessageFrom>CPUPPAL</h:MessageFrom>
                        <h:MessageTo>CPUPPAL</h:MessageTo>
                        <h:MessageType>COMMANDRESPONSE</h:MessageType>
                        <h:MessageName>Diagnostic de la CPUPPAL</h:MessageName>
                        <h:DataPath>s:CPUPPAL/s:Diagnostic</h:DataPath>
                </h:SIEheader>
        </soap:Header>
        <soap:Body>
                <s:CPUPPAL xmlns:s="urn:TMB:SIE:System:CPUPPAL" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
                        <s:Diagnostic s:name="Aplicacions" s:status="OK">
                                <s:DiagnosticKey s:name="FECOM" s:status="OK">STATUSOK</s:DiagnosticKey>
                                <s:DiagnosticKey s:name="SIC" s:status="OK"/>
                                <s:DiagnosticKey s:name="CPUPPAL" s:status="OK"/>
                        </s:Diagnostic>
                </s:CPUPPAL>
        </soap:Body>
</soap:Envelope>
```

Figura 54 Resposta XML

Per ser més eficients s'han aprofitat els avantatges de XPATH que permet accedir a certs camps del fitxer XML sense haver de recórrer totes les etiquetes, comprovant si és la que s'està cercant.

Amb la llibreria de Python que ens permet accedir a la informació d'un fitxer XML, abans d'indicar la ruta on es troba el valor que estem cercant, hem de definir un espai de noms (namespace), ja que en el fitxer XML està definit.

En el cas d'exemple, el nom d'espai seria el següent:

```
namespace = {'h': 'urn:TMB:SIE:Header', 's': 'urn:TMB:SIE:System:CPUPPAL', 'soap': 'http://www.w3.org/2003/05/soap-envelope'}
```

Figura 55 Nom d'espai

Un cop tenim definit l'espai de noms, podem fer una consulta directament a un cert camp indicant la ubicació dins del fitxer XML; per exemple, per accedir al valor de l'atribut FECOM, faríem:

```
'*/s:CPUPPAL/s:Diagnostic/*[@s:name="FECOM"]'
```

Quan en alguna funció definida en els fitxers de configuració s'hagi de cercar el resultat de varies variables (estaran separades per '|'), farem el mateix procés però iterarem totes les variables que ens indiqui el fitxer de configuracions.

8. Bugs identificats

Com a Bugs identificats tenim:

1. S'ha inclòs un "loader" per tal de mostrar a l'usuari quan una acció triga en executar-se que sàpiga que l'aplicació no s'ha quedat bloquejada.

   Si retornem mitjançant el botó del navegador, el loader es torna a mostrar, en canvi si es retorna amb el botó de retorn que s'ha inclòs en la pantalla de 800x600 si que s'atura i no es mostra.

   L'error està identificat i es necessita que en retornar amb el botó del navegador que es faci un refresc de la pagina completament per tal que el "loader" s'aturi.

2. Relacionat amb el "loader", no s'ha inclòs aquest element en totes les accions que puguin trigar un temps en executar-se. S'hauria d'incloure en totes aquelles que puguin deixar la pantalla congelada.

3. La imatge de background queda distorsionada si es selecciona el diagnòstic general.

4. Falten funcionalitats per implementar i s'haurien de protegir per a que l'usuari en aquestes fases inicials, si selecciona aquestes opcions no implementades se li mostri un missatge d'opció en construcció.

5. Falta per implementar l'actualització de dades cada cert temps, ja que es va provar a realitzar un refresc però els recursos de CPU que necessitava l'aplicació s'incrementaven prop del 50-60%.

6. S'hauria d'implementar un mecanisme que permeti carregar dades en background, es a dir, que l'aplicació actualitzi la seva visualització i quan es tinguin les dades, llavors s'actualitzin en el web.

7. Faltaria implementar la creació del missatge de configuració XML quan es mostren diferents desplegables.

8. Quan es mostren més d'un slider per a configurar valors, l'actualització de l'etiqueta amb el seu valor nomes fa cas d'un slider. S'ha de contemplar que

els scripts que llegeixen aquests valors puguin distingir entre els diferents sliders que pugui haver-hi en pantalla.

9. Han faltat incloure tooltips quan es mostri l'aplicació web en pantalla d'escriptori o pantalla gran.

9. Valoració econòmica

Per mostrar la valoració econòmica, primer de tot s'han d'identificar els perfils que haurien intervingut en el desenvolupament d'aquest projecte:

1. **Enginyer de sistemes**. S'hauria encarregat de l'estudi de l'estat de l'art per a la solució demanada, dels estudis dels diferents escenaris, i les diferents instal·lacions que haguessin estat necessàries, a més de realitzar les proves de rendiment, i ens hagués proporcionat un informe dels pros/contres de les diferents opcions disponibles.

2. **Arquitecte Software**. Perfil dedicat a desenvolupar la millor estratègia de desenvolupament amb les opcions escollides per l'enginyer de sistemes. L'arquitecte hauria proporcionat un disseny a gran nivell de com s'estructuraria la solució per a que el programador/desenvolupador pogués portar a terme el dissenyat per l'arquitecte.

3. **Desenvolupador/programador**. Amb sòlids coneixements de HTML/CSS/Javascript i Python per a desenvolupar tota l'aplicació WEB amb el disseny proporcionat per l'arquitecte software.

4. **Expert en usabilitat**. Necessari per al disseny dels wireframes, de l'avaluació heurística i per a la realització dels tests d'usuari.

Des de que es va començar amb el projecte amb data 22 de febrer fins al 12 de Juny data de lliurament, hi han hagut un total de:

- 75 dies feiners
- 36 dies festius

Per al còmput d'hores s'ha estimat una mitjana de 5 hores al dia, perquè els dies feiners no es dedicaven tantes hores com els dies festius.

Perfil	Dies	Hores	Preu/hora	Subtotal
Enginyer de Sistemes	10	50	80€	4.000€
Arquitecte Software	5	25	100€	2.500€
Desenvolupador	40	200	60€	12.000€
Expert en usabilitat	9	45	80€	3.600€
Subtotal	74	370		
TOTAL				**22.100€**

10. Conclusions

L'assoliment d'aquest projecte ha estat una tasca molt dura, amb molt feina a fer, molts recursos per investigar, però en definitiva ha estat un treball molt enriquidor i satisfactori.

En primer lloc aquest treball m'ha permès endinsar-me en el desenvolupament d'aplicacions WEB, en el llenguatge Python i en tot l'ecosistema que hi ha al voltant dels servidors web d'aplicacions.

De les lliçons apreses, em quedo amb que de vegades el primer camí qué penses pot ser la millor opció per afrontar la solució a un problema, al final pot no ser-ho. Aquesta reflexió és deguda a que parlant de la implementació d'aquest projecte amb el meu cap i un company de feina vaig entendre que una altra solució molt més portable podria haver estat separar l'aplicació WEB per una banda, desenvolupada en HTML5, CSS3 i Javascript. Aquesta WEB s'hagués comunicat amb un middleware mitjançant, per exemple, missatges JSON, i en darrer terme una capa de comunicació entre el middleware i l'equip. D'aquesta manera tindríem que la capa de presentació, que seria la nostra WEB es podria portar a qualsevol infraestructura perquè no estaria vinculada a res, nomes a una capa de comunicació JSON, i això permetria portar-la, per exemple als servidors centrals i des d'aquests servidors comunicar-se amb el middleware, estigui on estigui.

Aquest projecte tenia uns objectius molt ambiciosos per al temps del que es disposava i no s'ha pogut arribar a tots els objectius que s'havien definit com a desitjables per aquest projecte. Per exemple, no s'han realitzat els tests d'usuaris perquè el desenvolupament ha arribat a dates de quasi el lliurament final. S'havien plantejat com una tasca a realitzar i planificada al Diagrama de Gantt, però no ha estat possible realitzar-la a temps.

Segons el meu parer, vaig dedicar-li massa temps a la instal·lació dels escenaris que serien motiu de prova, però va ser degut al munt de problemes que vaig tenir per a que funcionessin degudament. Si aquesta tasca no s'hagués allargat tant de temps possiblement els tests d'usuaris m'hagués donat temps de fer-los.

S'ha intentat seguir la planificació amb molta exigència, però en combinar altres factors, com per exemple el treball, els imprevistos, etc. amb aquest treball final de màster, fa que no sempre es pugui avançar al ritme desitjat.

En els moments que no s'ha pogut seguir la planificació, s'han aprofitat els caps de setmana per avançar amb molta més força i temps.

A més, durant el desenvolupament de la WEB sortien imprevistos en la implementació que evidentment no estaven contemplats a la planificació.

Les línies que s'haurien de seguir a futur en següents fases d'aquest projecte serien:

1. Integrar la WEB implementada en Django a l'equip físic però executant-se amb el servidor web d'aplicacions (NGINX) i el connector escollit (Gunicorn), per tal de veure quin és el consum de recursos CPU i RAM que necessiten.

2. Finalitzar les funcionalitats que no s'han implementat així com els diferents tipus de pantalla que no s'han desenvolupat per qüestió de temps, com per exemple el login d'accés.

3. Realitzar els tests d'usuaris, amb els Pre-test, escenaris, tasques, post-test, etc.

4. Millorar l'aspecte visual de la WEB i el "responsive", aprofitant els inputs que ens puguin aportar els tests d'usuaris.

5. Decidir de quina forma s'accedeix des de l'exterior a la WEB, perquè per qüestions de seguretat, no es pot obrir lliurament un port de comunicacions a l'equip.

Per finalitzar, hem trobo molt satisfet de la feina feta ja que s'ha treballat molt dur, he après moltes coses tant en relació al desenvolupament WEB com del framework Django. I nomes em queda la insatisfacció de no haver assolit tots els objectius a temps.

11. Glossari

- **3G/4G**. Tercera i quarta generació de transmissió de dades en mobilitat mitjançant UMTS(Universal Mobile Telecommunications System) i LTE (Long Term Evolution).
- **Background**. Execució de processos en segon pla, es a dir, sense interferir en el procés principal. En una aplicació amb interfície gràfica és molt important tenir en compte realitzar les operacions que triguin massa en respondre en segon pla, per tal que la interfície gràfica no es quedi "bloquejada o congelada".
- **Bash**. (Bourne again Shell). La seva funció consisteix en interpretar ordres i comandes a traves de la consola del sistema. Es basa en la Shell de UNIX i és compatible amb POSIX.
- **Dropdown**. És una opció que permet desplegar altres funcions com per exemple un menú dropdown, que en seleccionar-lo es desplega en alguna direcció per a mostrar les opcions que conté.
- **Ethernet**. És un estàndard per a xarxes d'àrea local i es troba en la capa de nivell d'enllaç del model OSI.
- **Framework**. Es tracta d'un esquema per a desenvolupar codi i implementar una aplicació.
- **Gateway**. És un dispositiu que possibilita compartir recursos entre diferents dispositius.
- **Host**. Un equip HOST (amfitrió) és un equip que pot acollir un altre sistema GUEST (convidat), i pot estar connectat a una xarxa per aprofitar l'accés a recursos, etc.
- **Http**.(HyperText Transfer Protocol). Aquest protocol de transmissió permet realitzar comunicacions a traves de la WWW. Aquest protocol defineix la sintaxis i la semàntica que faran servir els elements software de la WEB.
- **IDE**. (Integrated Development Environment). És un aplicatiu per a desenvolupament que facilita el treball al desenvolupador. Segons el tipus de IDE pot proporcionar-li al desenvolupador eines com creació d'interfícies gràfiques, depuradors de codi, auto-completat d'instruccions, etc.
- **Java**. És un llenguatge de programació executat en una maquina virtual que permet abstraure's del tipus de maquina i/o processador i adaptar-se a diferents entorns. El codi és similar a C++, però un cop compilat es generen bytecodes que nomes interpreta la maquina virtual de JAVA.

- **Know-how**. És un terme que prové de l'anglès i fa referencia al coneixement que s'ha adquirit o es podrà adquirir.

- **LDAP**. (Lightweight Directory Access Protocol). És un protocol a nivell d'aplicació que permet l'accés a un directori distribuït per a cercar diferent informació en un entorn de xarxa. Mitjançant aquest protocol es permet fer grups, proporcionar a aquests grups privilegis d'accés, etc.

- **Linux**. Sistema operatiu lliure i de codi obert que es basa en UNIX, i que el va començar a implementar Linus Torvalds. Existeixen moltes distribucions basades en aquest kernel.

- **Mvc**. (Model-View-Controller). Es un patró de desenvolupament software que tracta de separar les diferents parts de negoci d'una aplicació en diferents papers com són el model per a la gestió de dades, les vistes per a "dibuixar" la informació a l'usuari i els controladors que fan de pont entre el model i les vistes. És un patró de desenvolupament molt emprat en l'actualitat per permetre aquesta separació. En realitzar aquesta separació correctament, es poden intercanviar "peces" sense massa implicació.

- **Proxy**. És un servidor que fa d'intermediari entre les peticions que fa un client a un servidor.

- **Ram**. (Random Access Memory). Memòria d'accés aleatori i volàtil; aquest tipus de memòria té una velocitat d'accés molt més alta que la velocitat d'accés a un disc dur, per aquest motiu s'intenta que les aplicacions que han d'executar-se ràpidament o han de tenir una resposta prou rapida s'executin en memòria RAM.

- **Responsive**. El disseny responsive permet que l'aplicació (sigui web o nativa) s'adapti a les diferents mides de pantalla dels actuals dispositius. Per això aquest concepte de responsive també aplica en el mon del desenvolupament per a dispositius mòbils perquè aquests disposen de diferents mides de pantalla i els elements s'han de mostrar d'una forma coherent sigui quina sigui la mida de la pantalla.

- **Script**. És un arxiu que conté un seguit d'ordres que s'executaran seqüencialment. En Linux pot ser un script en bash i en Windows pot ser un arxiu per lots .bat

- **Singleton**. És un patró de desenvolupament que nomes permet tenir una instancia d'una classe. D'aquesta manera es té controlada la quantitat d'instàncies d'una mateixa classe.

- **Slider**. Es tracta d'un objecte visual (que està disponible en diferents entorns de programació) que es pot mostrar horitzontal o verticalment, i permet a l'usuari seleccionar un valor mitjançant el moviment del seu indicador.
- **Soap**. (Simple Object Access Protocol). És un protocol que defineix la manera en que 2 objectes es comunicaran mitjançant l'intercanvi de dades en format XML.
- **Spinner**. És un element que mostra a l'usuari que una tasca està trigant en executar-se, però que s'està portant a terme. D'aquesta manera indica a l'usuari que l'aplicació no s'ha quedat bloquejada.
- **SSH**. (Secure Shell). És un protocol que permet accés de forma remota mitjançant una consola i permet realitzar tasques d'administració, manteniment, etc.
- **Tcp/Ip**. Conjunt de protocols de xarxa en els que es basa Internet. Amb aquest conjunt de protocols es permet la transmissió/recepció d'informació entre diferents ordinadors o servidors.
- **Tooltips**. És un element visual que s'aplica en desenvolupaments d'interfícies gràfiques i proporcionen informació addicional a l'usuari sobre un element en concret. Aquest element s'activa en passar per sobre d'un l'element..
- **URL**. (Uniform Resource Locator). Fa referencia a la localització en la xarxa d'un recurs web en concret.
- **VNC**. (Virtual Network Computing). És un software lliure que permet el control d'un ordinador de forma remota. Aquest software no imposa restriccions en els sistemes operatius de l'ordinador servidor ni client.
- **WiFI**. Mecanisme de comunicació sense fils.
- **Wireframes**. És un esbós per ajudar a visualitzar com es pot implementar una certa pantalla web o aplicació nativa. Un wireframe pot ajudar tant al client a visualitzar una primera opció de l'aplicació que vol implementar, al dissenyador que aprofitarà els inputs del client per tal de modificar un disseny que no es massa detallat, als usuaris que es poden fer una idea de com serà el funcionament i al desenvolupador per tal de plasmar el disseny en la implementació del codi.
- **XML**. (eXtensible Markup Language). És un metallenguatge que permet emmagatzemar estructures de dades en format llegible.

12. Annexos

12.1 Consum de recursos dels diferents escenaris provats

12.1.1 Apache - WSGI - Web2Py

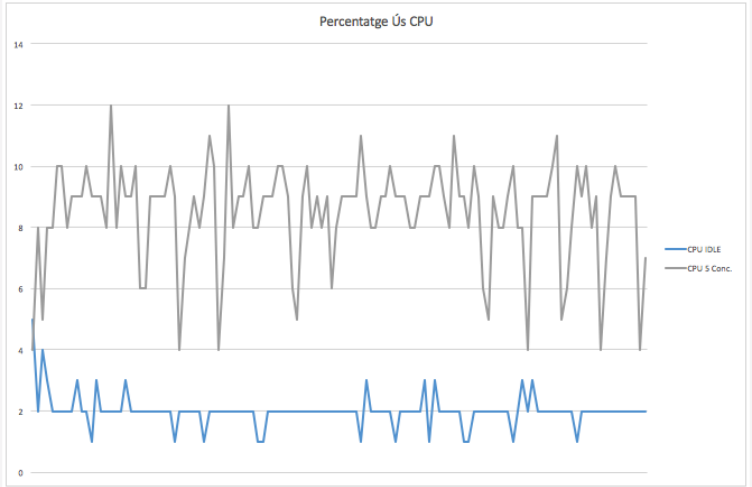


Figura 56 Percentatge CPU en repòs vs percentatge CPU amb 5 peticions concurrents

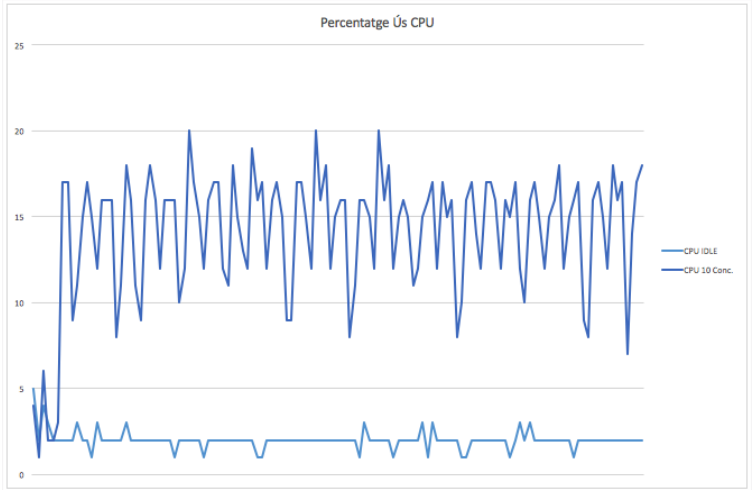


Figura 57 Percentatge CPU en repòs vs percentatge CPU amb 10 peticions concurrents

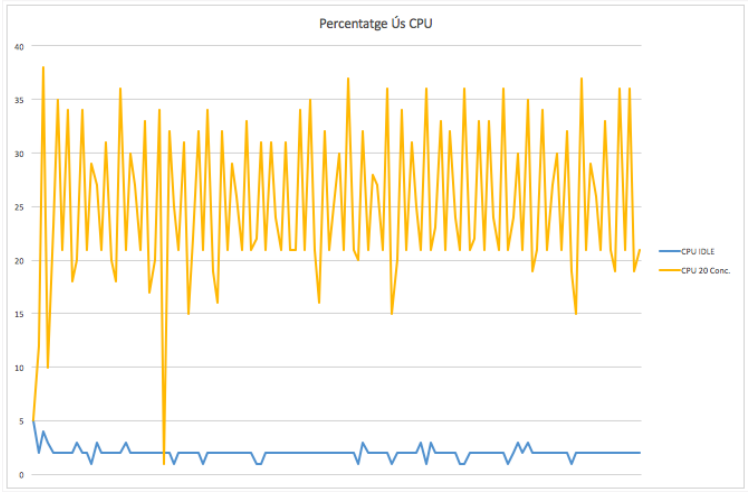


Figura 58 Percentatge CPU en repòs vs percentatge CPU amb 20 peticions concurrents

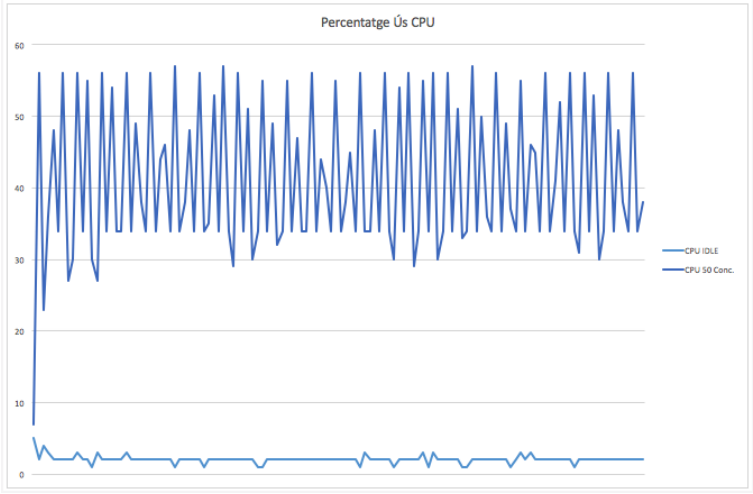


Figura 59 Percentatge CPU en repòs vs percentatge CPU amb 50 peticions concurrents

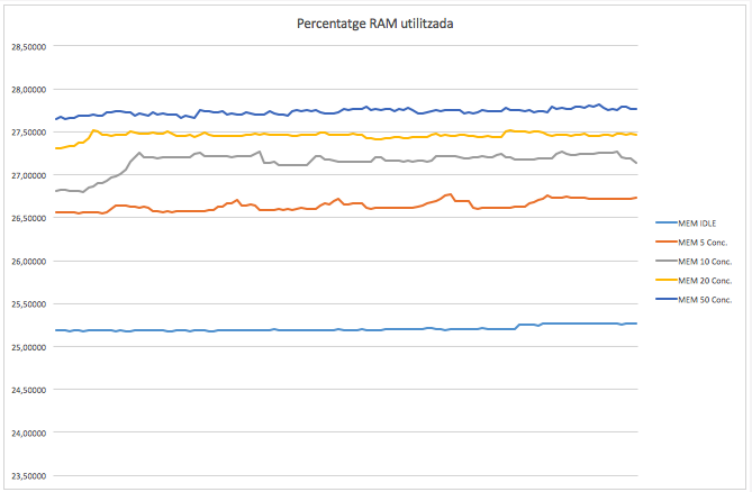


Figura 60 Percentatge de RAM utilitzada vs percentatge RAM amb peticions concurrents

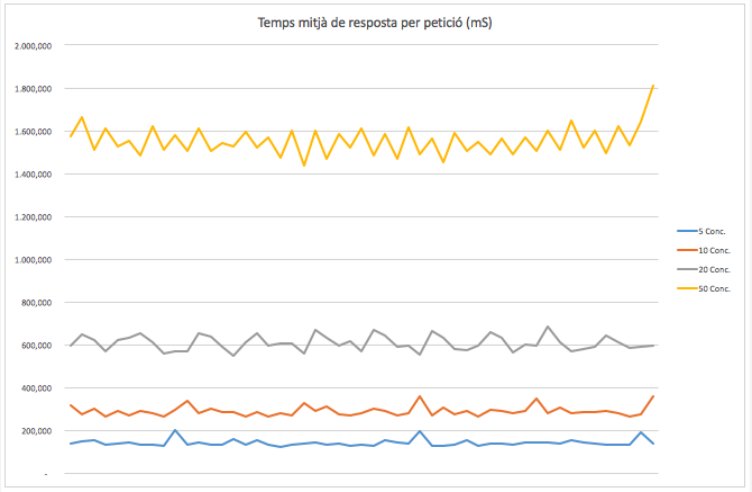


Figura 61 Temps mitjà de resposta per petició en mS

12.1.2 Nginx - uWSGI - Web2Py

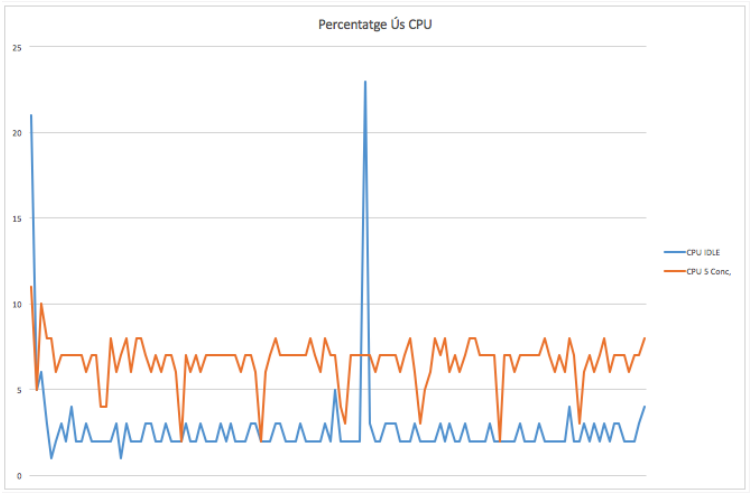


Figura 62 Percentatge CPU en repòs vs percentatge CPU amb 5 peticions concurrents

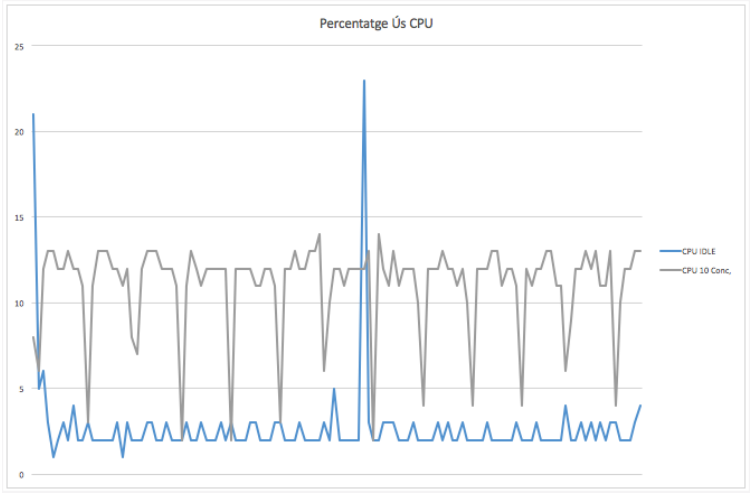


Figura 63 Percentatge CPU en repòs vs percentatge CPU amb 10 peticions concurrents

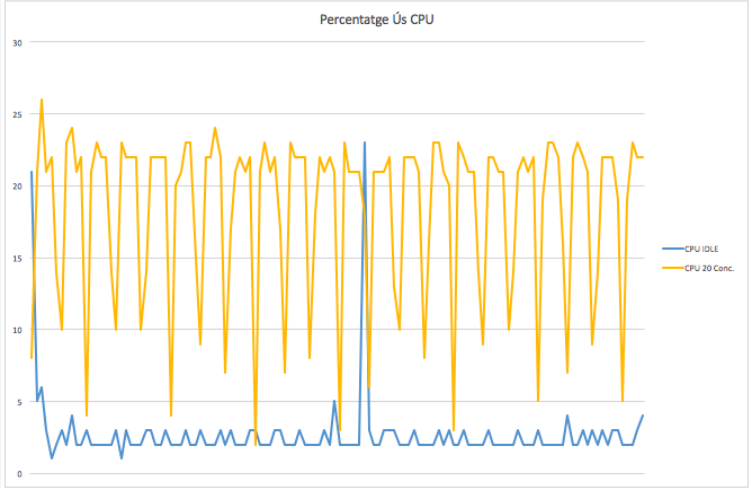


Figura 64 Percentatge CPU en repòs vs percentatge CPU amb 20 peticions concurrents

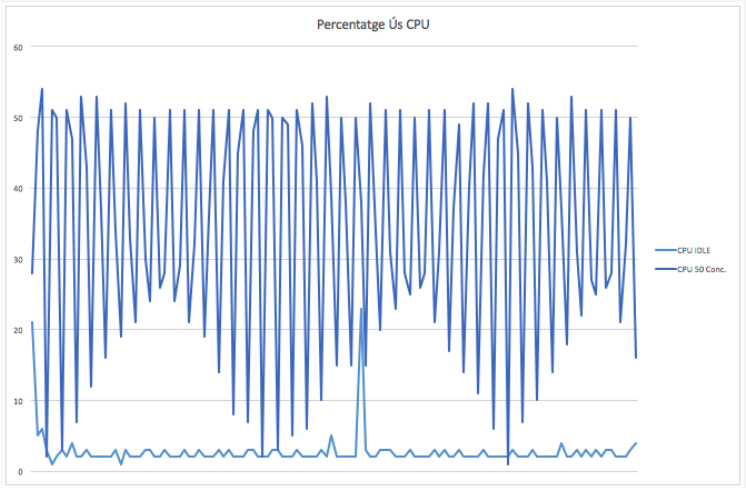


Figura 65 Percentatge CPU en repòs vs percentatge CPU amb 50 peticions concurrents

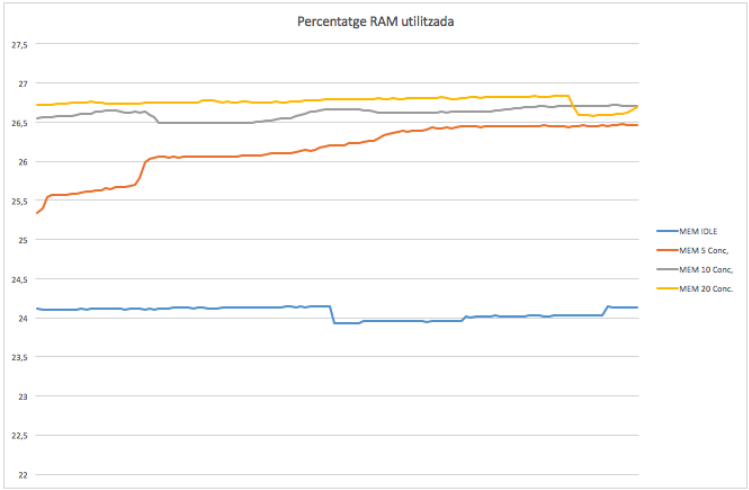


Figura 66 Percentatge de RAM en repòs vs percentatge RAM amb peticions concurrents

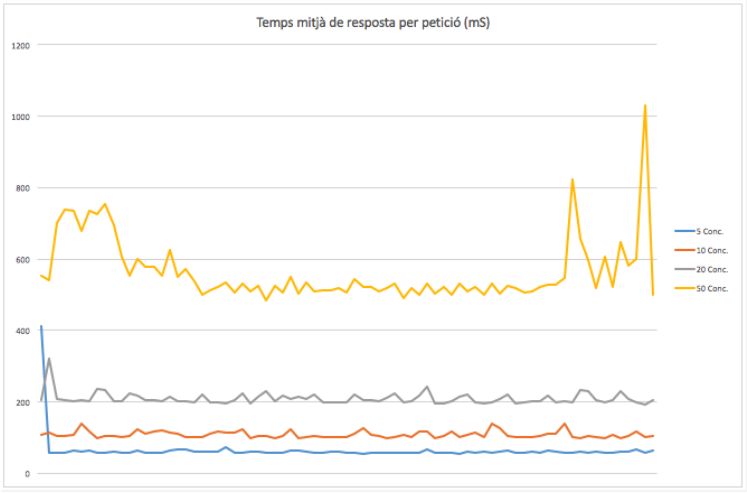


Figura 67 Temps mitjà de resposta per petició en mS

12.1.3 Nginx - uWSGI - django

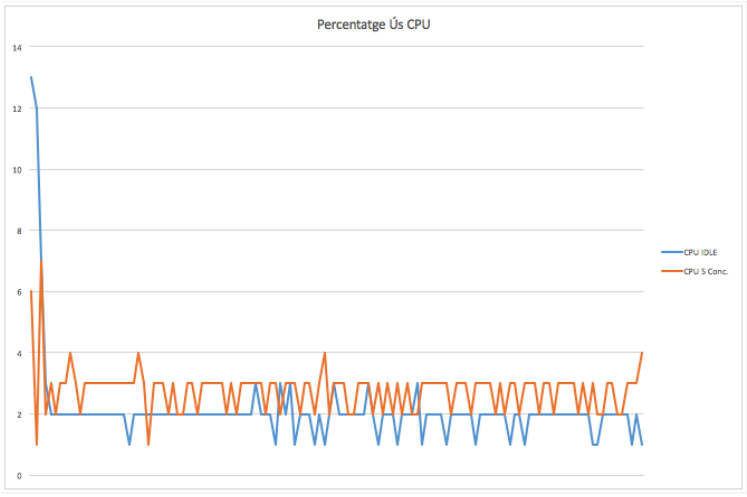


Figura 68 Percentatge CPU en repòs vs percentatge CPU amb 5 peticions concurrents

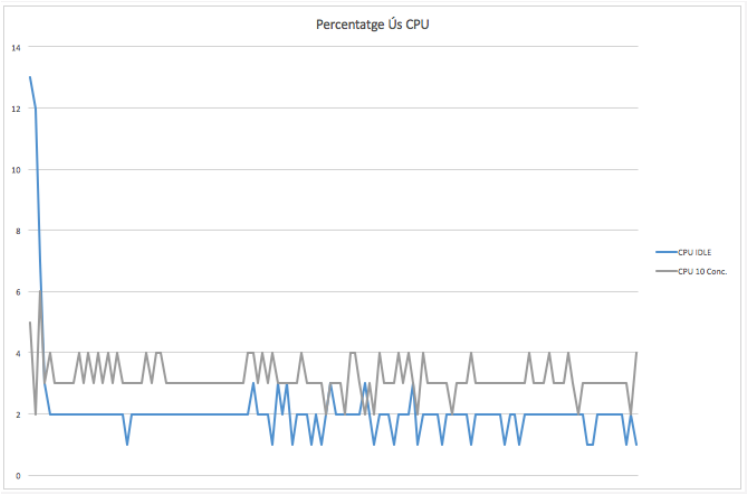


Figura 69 Percentatge CPU en repòs vs percentatge CPU amb 10 peticions concurrents

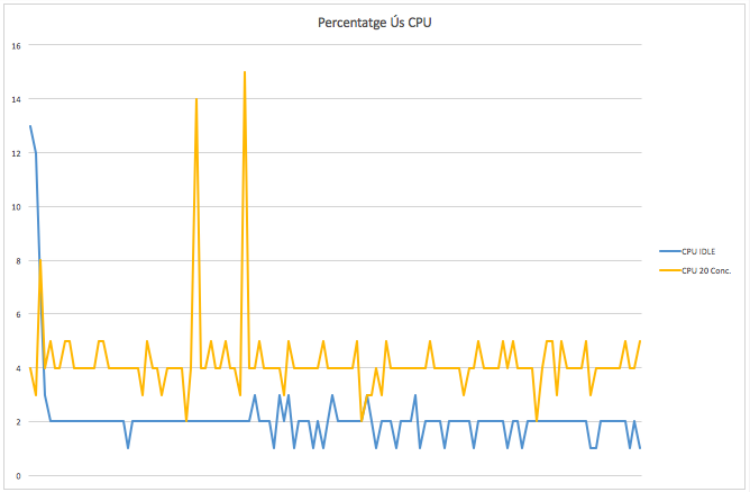


Figura 70 Percentatge CPU en repòs vs percentatge CPU amb 20 peticions concurrents

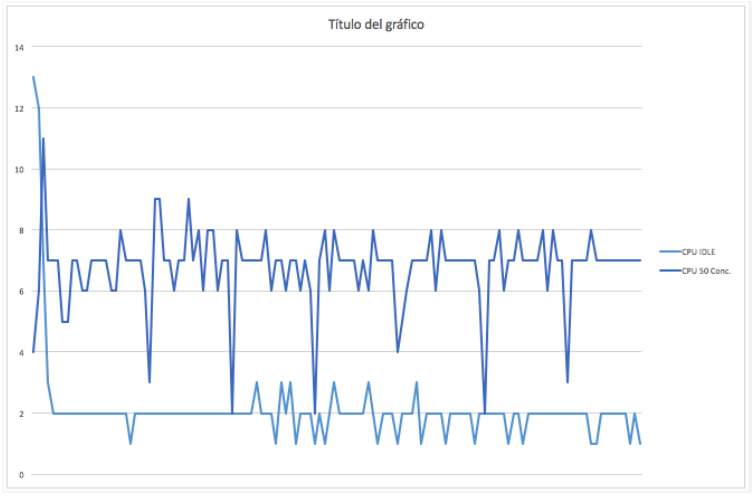


Figura 71 Percentatge CPU en repòs vs percentatge CPU amb 50 peticions concurrents

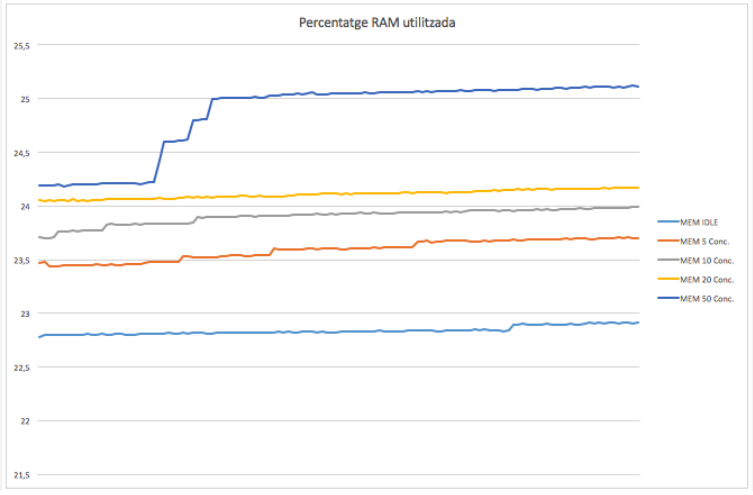


Figura 72 Percentatge de RAM utilitzada vs percentatge RAM amb peticions concurrents

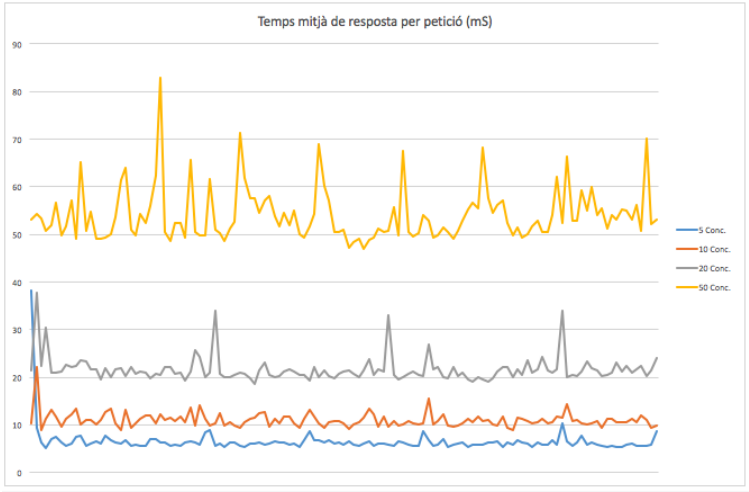


Figura 73 Temps mitjà de resposta per petició en mS

12.1.4 Nginx - Gunicorn - django

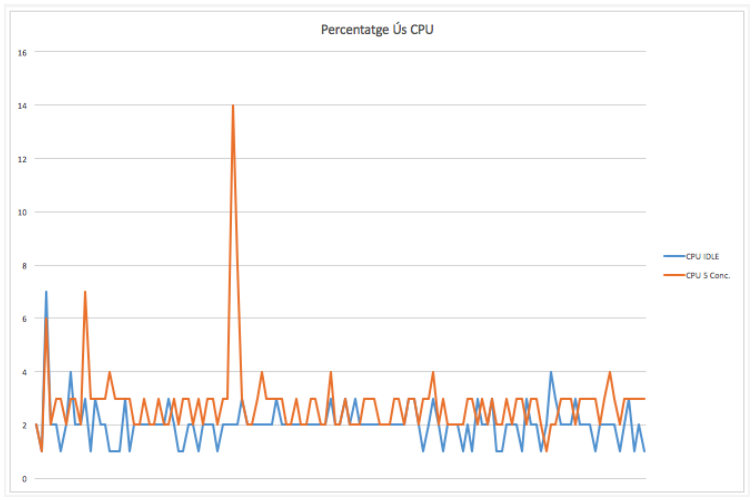


Figura 74 Percentatge CPU en repòs vs percentatge CPU amb 5 peticions concurrents

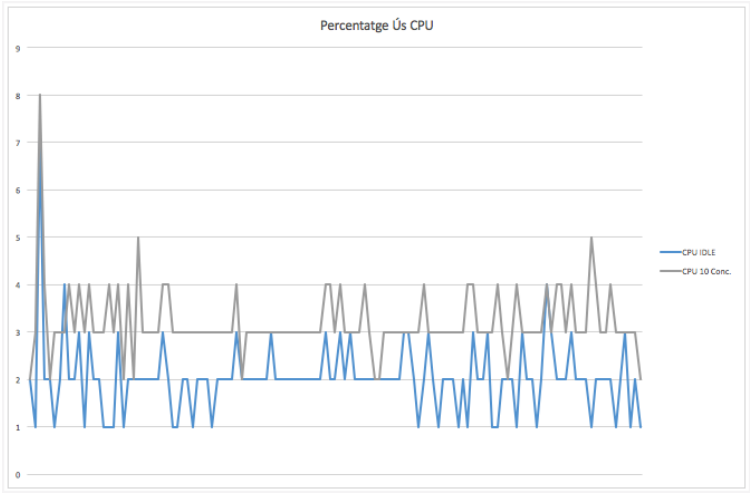


Figura 75 Percentatge CPU en repòs vs percentatge CPU amb 10 peticions concurrents

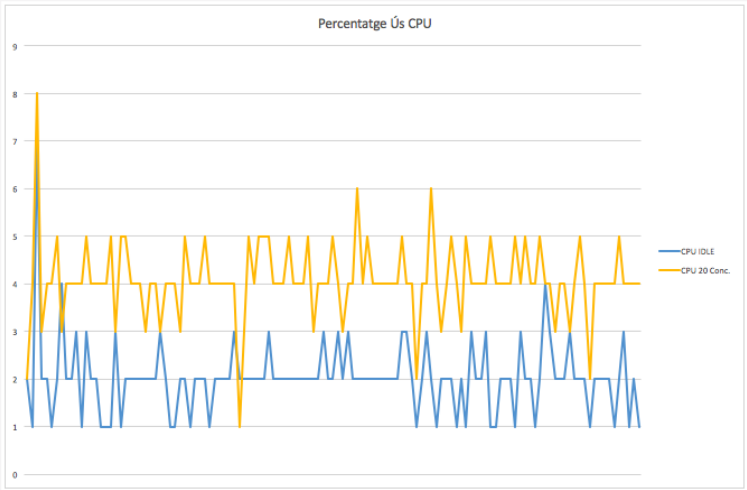


Figura 76 Percentatge CPU en repòs vs percentatge CPU amb 20 peticions concurrents

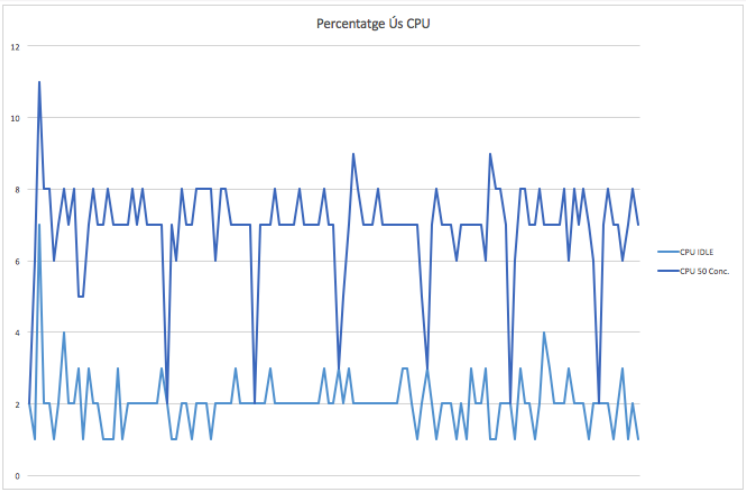


Figura 77 Percentatge CPU en repòs vs percentatge CPU amb 50 peticions concurrents

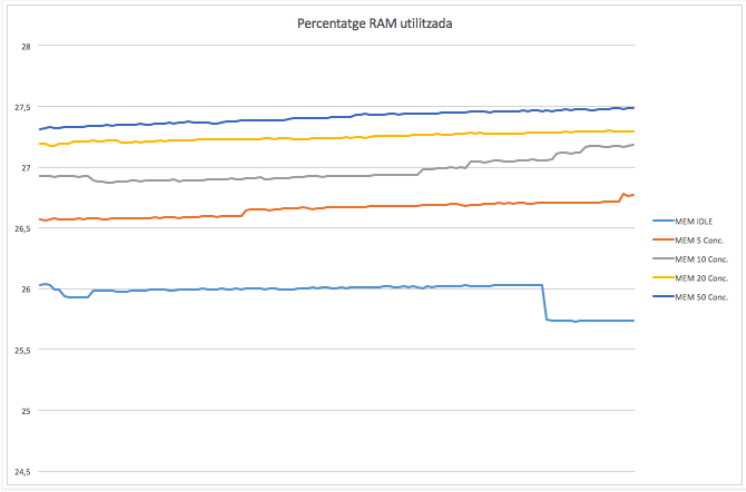


Figura 78 Percentatge de RAM utilitzada vs percentatge RAM amb peticions concurrents

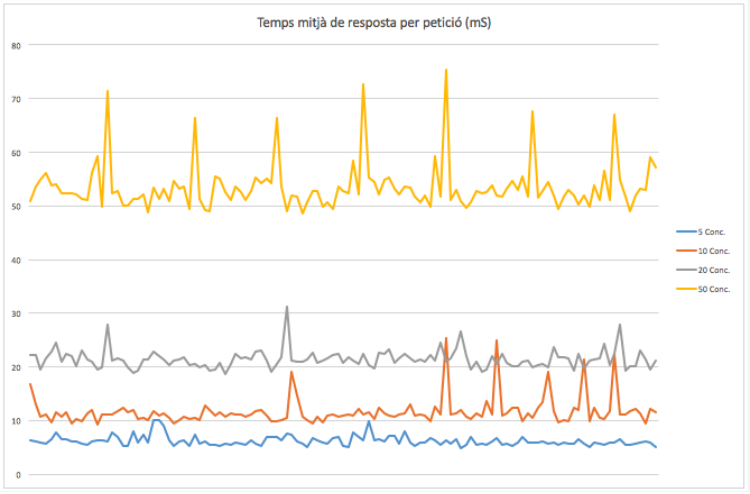


Figura 79 Temps mitjà de resposta per petició en mS

12.2 Wireframes dissenyats per a l'aplicació

Figura 80 Opció de Calibració TFT Seleccionada

Figura 81 Modificar Número Sèrie. Modificar número Producte

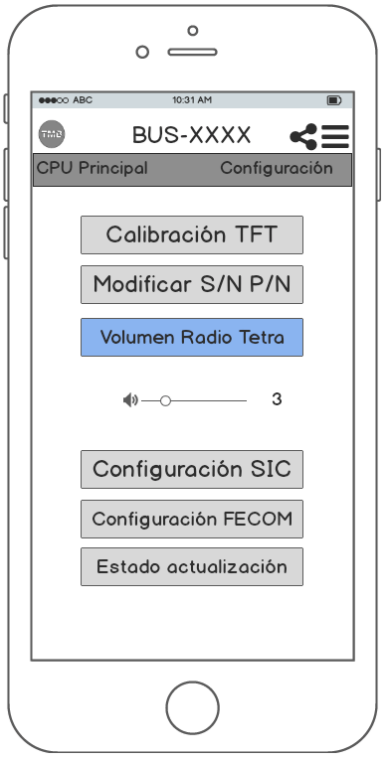


Figura 82 Configurar volum Radio Tetra (Pot ser modificada encara)

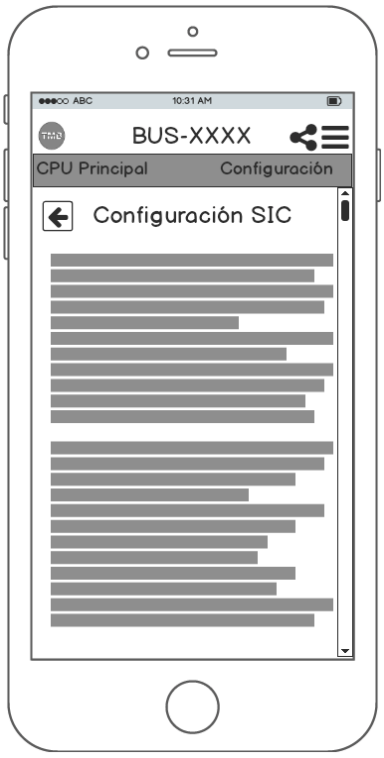


Figura 83 XML mostrant configuració SIC

Figura 84 Visualització actualitzacions

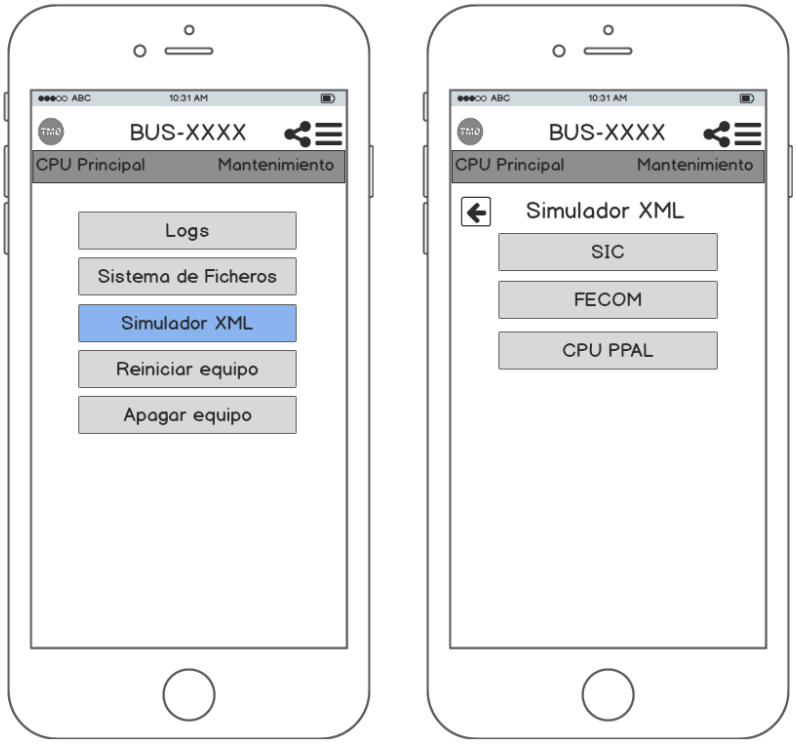


Figura 85 Opcions Simulador XML

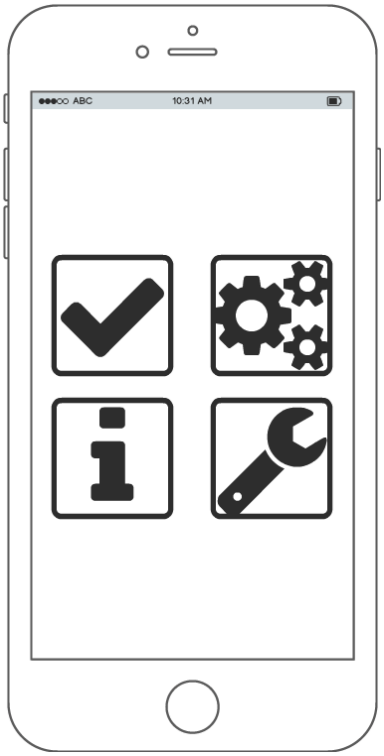


Figura 86 Prototip per a no mostrar text

12.3 Resta de paràmetres de configuració per als fitxers de configuració

A l'apartat 7.2.4 s'han definit els paràmetres que s'han d'incloure de forma obligatòria per a formar la configuració d'un menú per a l'aplicació.

A continuació es llisten altres opcions que permeten configurar menús més elaborats o que aportin més funcionalitats a les pantalles que s'han dissenyat.

- `fitxerXMLConsulta`. Quan un menú necessita enviar un fitxer de consulta per a rebre una resposta es defineix la ubicació del fitxer que servirà per a fer la consulta.

```
menu5.fitxerXMLConsulta=ConfiguracionSistema.xml
```

Figura 87 fitxerXMLConsulta

- `keyAConsultar`. En rebre una resposta XML potser nomes volem visualitzar unes keys en concret. En aquesta línia de configuració es descriuen les variables que es volen extreure separades per '|'.

```
menu1.keyAConsultar=MOTOR|CONTACTO|PUERTA_OPEN
```

Figura 88 keyAConsultar

- `consultaEnBucle`. Hi ha certes opcions que es vol que contínuament s'estiguin actualitzant, perquè són valors que poden canviar en un cert instant de temps. Aquest camp indica si s'hauran de refrescar els valors contínuament. Pot valer "`true`" o "`false`", encara que no s'ha implementat la funcionalitat de refrescar en aquest projecte.

```
menu1.consultaEnBucle=true
```

Figura 89 Consulta en bucle

- `temps`. Aquesta configuració està relacionada amb l'anterior, i indica cada quant temps es realitza l'actualització. És un valor expressat en mil·lisegons.

```
menu1.temps=2000
```

Figura 90 temps d'actualització

- `fitxerXMLModificacio`. Quan es realitza una operació de modificació, s'ha d'enviar un missatge XML amb les noves configuracions realitzades. Aquest camp de configuració indica on es troba la base del missatge XML que s'haurà de formar amb les modificacions que hagi indicat l'usuari per a després enviar-lo.

```
menu2.fitxerXMLModificacio=/sge/ModificarConfiguracion.xml
```

Figura 91 fitxer XML modificació

- `keyAModificar`. Aquesta línia de configuració està directament relacionada amb l'anterior, ja que a més de formar el fitxer XML amb la modificació que hagi realitzat l'usuari, s'ha d'incloure la modificació de la key en concret. Aquesta línia de configuració indica quina key s'ha de modificar.

```
menu2.keyAModificar=CALCA
```

Figura 92 key a modificar

- `numDesplegables`. Hi ha un tipus de pantalla que permet mostrar uns desplegables amb valors, entre els que l'usuari pot seleccionar un d'ells. Aquesta línia de configuració indica quants desplegables apareixeran en la pantalla. A més, les configuracions que es definiran a continuació es tenen que repetir per a cadascun dels desplegables que s'hagin indicat.

```
menu7.numDesplegables=3
```

Figura 93 numero de desplegables

- `desplegable`. Per a cadascun dels desplegables que s'hagin definit en el numero de desplegables de la configuració anterior, s'han de definir les següents configuracions:
    - `nombre`. És el nom que apareixerà al costat del desplegable i serveix per a informar a l'usuari que és el que podrà escollir amb aquest desplegable.

```
menu7.desplegable1.nombre=MessageFrom
```

Figura 94 nom del desplegable

- `valor`. Com que en el desplegable apareixeran una quantitat de valors per a escollir, en aquest camp es defineixen els valors de la llista separats pel caràcter '|'.

```
menu7.desplegable1.valor=CPUPPAL|CTBUS
```

Figura 95 valors del desplegable

Si el valor es defineix com a "spinner" llavors no es mostrarà un desplegable sinó que es mostrarà un "slider" que permeti escollir un valor a l'usuari:

```
menu1.desplegable4.valor=spinner
```

Figura 96 valor spinner

Figura 97 slider per seleccionar un valor

- `rang`. Si el que s'ha de mostrar és un slider, es necessiten certs valors de configuració com són el valor màxim, el mínim, el valor actual i la distancia entre un valor i el següent. Aquests valors es defineixen separats entre ',' i l'ordre és:
  - valor actual
  - valor mínim
  - valor màxim
  - separació entre valors

```
menu1.desplegable4.rang=10,0,60,1
```

Figura 98 valors de configuració del slider

- `recurs`. Aquesta línia de configuració està relacionada únicament amb el diagnòstic general del sistema. Cada recurs indicarà una comanda a executar i un temps d'actualització.

- `comanda`. Defineix el bash a executar amb els paràmetres necessaris.

```
recurs1.comanda=diagnostic.sh -id -abv
```

Figura 99 Comanda d'un recurs

- `temps`. Defineix el temps en mil·lisegons per actualitzar els valors.

```
recurs1.temps=10000
```

Figura 100 Temps d'actualització

12.4 Tags Django utilitzats

Com s'ha indicat a la descripció de l'estructura de l'aplicació web, Django ens permet incloure en els fitxers de codi html uns "tags" que incrementen la potencialitat del web que es visualitzarà incloent-hi certa lògica que disminueix la quantitat de codi i incrementa l'entesa del mateix.

A continuació s'han inclòs les que més s'han utilitzat en aquest projecte.

12.4.1 `{{ x }}`

Mostra en el codi html el valor de la variable x.

Quan es crida a un template html des del fitxer `views.py` se li poden passar variables, i aquest interpretar-les i mostrar-les.

Per exemple, si en la variable "`equipo`" li hem passat el valor "SGE", la següent línia de codi HTML:

```
<label>{{ equipo }}</label>
```

Figura 101 {{ variable }}

Mostraria en un navegador una etiqueta que mostraria SGE. I el codi HTML equivalent seria:

```
<label>SGE</label>
```

Figura 102 Codi HTML equivalent

12.4.2 `{% if elif else %} {% endif%}`

És l'estructura condicional, equivalent a altres llenguatges de programació. En funció de si la condició éscerta o falsa, es realitzaran unes accions o unes altres. En el exemple següent serà mostrar o no mostrar la variable "equipo" sempre hi quan no contingui el valor "None":

```
{% if equipo != "None" %}
    <label>{{ equipo }}</label>
{% endif %}
```

Figura 103 Estructura IF

12.4.3 `{% for ... in ... %} {% endfor %}`

Permet la iteració d'una variable ja sigui un diccionari o una llista.

Es tracta d'un bucle "`for`" similar al que es troben en altres llenguatges. A continuació es mostra un exemple utilitzat en aquest projecte:

```
{% for opcion in resultado %}
    <div class="radio" style="padding-left: 50px;">
        <input type="radio" name="modificacio1_value" value="{{ opcion }}"/>
        <label style="font-size: 20px; padding-bottom: 20px;">{{ opcion }}</label>
    </div>
{% endfor %}
```

Figura 104 Bucle For

En aquest exemple es faran tantes iteracions com objectes hi hagi en la variable "`resultado`".

A cada iteració es crearà un bloc `<div class="radio"...>` amb el contingut que hi ha entre els tags `<div>`.

Si suposem que hi ha 3 valors en la variable `resultado`, el codi html resultant que podríem visualitzar en un navegador web seria el següent:

```
<div class="radio" style="padding-left: 50px;">
    <input type="radio" name="modificacio1_value" value="{{ opcion }}"/>
    <label style="font-size: 20px; padding-bottom: 20px;">{{ opcion }}</label>
</div>
<div class="radio" style="padding-left: 50px;">
    <input type="radio" name="modificacio1_value" value="{{ opcion }}"/>
    <label style="font-size: 20px; padding-bottom: 20px;">{{ opcion }}</label>
</div>
<div class="radio" style="padding-left: 50px;">
    <input type="radio" name="modificacio1_value" value="{{ opcion }}"/>
    <label style="font-size: 20px; padding-bottom: 20px;">{{ opcion }}</label>
</div>
```

Figura 105 Codi equivalent en HTML

12.4.4 `{% url 'method' parameters %}`

Apunta a la direcció que executarà el mètode "`method`" i li passarà els paràmetres que s'introdueixin a continuació.

Aquest tag té molt de potencial perquè no hem de fixar la url ja que aquí definim el mètode del fitxer `views.py` tal i com ho hem definit al fitxers `urls.py`. Així, encara que l'aplicació canviï d'ubicació, l'adreça url apuntarà correctament a on té que anar. En el següent exemple, l'adreça url serà anar al mètode "`options`" del fitxer `views.py` amb els paràmetres `equipo` i `opcion`.

```
href="{% url 'options' equipo=equipo opcion=opcion %}"
```

Figura 106 {% url %}

12.4.5 `{% load static %} {% static file %}`

Carrega fitxers estàtics com arxius d'estils, imatges o scripts javascript i els va a buscar a una carpeta anomenada static de l'aplicació on es demana. De la mateixa manera que ocorria amb el tag anterior, amb aquest tag no ens hem de preocupar quan canviem la ubicació de l'aplicació. Sempre anirà a l'adreça relativa on es trobin els fitxers estàtics.

12.4.6 `{% extends 'file.html' %}`

En un fitxer html es pot incloure el codi d'un altre fitxer html i d'aquesta manera no es repetirà codi que ens pot fer falta en diferents arxius.

En el nostre cas necessitem que la barra de navegació estigui disponible en tots els fitxers html, amb la qual cosa tots els fitxer html estenen del fitxer on hi és aquesta barra de navegació.

12.4.7 `{% load fitxer de tags personalitzats %}`

Tal i com s'ha definit en la memòria, en aquest projecte s'han definit fitxers amb tags personalitzats que amplien les funcionalitats de les que ja proporciona Django.

Per executar aquestes funcions, primer s'ha de carregar el fitxer que conté les funcionalitats personalitzades per després cridar-les dintre del codi html.

A continuació es mostra un exemple:

```
{% load tipus_pantalla %}

{% consulta2 menu as resultado %}
```

Figura 107 Càrrega i execució de codi personalitzat

Primer es càrrega el fitxer que conté les funcionalitats que s'han definit com a personalitzades, en el cas d'exemple el fitxer "`tipus_pantalla`". A continuació, es crida al mètode "`consulta2`" passant-li com a paràmetre la variable `menu`, i el resultat que ens retorni aquesta funció s'emmagatzemarà en la variable "`resultado`".

La capçalera del mètode "consulta2" es mostra a continuació:

```
def consulta2(menu):
```

Figura 108 Capçalera mètode personalitzat

12.5 Execució de l'aplicació en Django

Per a executar la solució tal i com està ara s'ha de tenir instal·lat Django 1.8 (versions posteriors han de funcionar igual però no s'ha comprovat) i en una maquina amb Sistema Operatiu Linux (no s'ha comprovat el funcionament en una maquina Windows).

Un cop es té Django instal·lat s'ha de navegar fins a la carpeta `django-project/tfm` i executar la comanda:

```
python manage.py runserver 0:8000
```

El 0 es pot substituir per 127.0.0.1 que és l'adreça de localhost i el 8000 que és el port es pot substituir pel numero de port que desitgem. Un cop executada la comanda apareixerà en la línia de comandes:

```
Django version 1.8, using settings 'tfm.settings'
Starting development server at http://0:8000/
Quit the server with CONTROL-C.
```

Per accedir-hi a l'aplicació WEB s'ha d'introduir en el navegador el següent:

```
http://localhost:8000/myApp/
```

13. Bibliografia

Recursos WEB

https://gorails.com/setup/ubuntu/14.04 [8/3/2017]
https://help.dreamhost.com/hc/en-us/articles/215945987-Web-server-performance-comparison [13/3/2017]
https://www.digitalocean.com/community/tutorials/how-to-serve-django-applications-with-uwsgi-and-nginx-on-ubuntu-14-04 [15/3/2017]
https://www.digitalocean.com/community/tutorials/how-to-set-up-django-with-postgres-nginx-and-gunicorn-on-ubuntu-14-04 [15/3/2017]
https://www.fdi.ucm.es/profesor/jpavon/web/32-CGI.pdf [24/3/2017]
http://www.web2py.com/ [24/3/2017]
http://nginx.org/en/docs/beginners_guide.html [24/3/2017]
http://nginx.org/en/docs/ngx_core_module.html#worker_processes [24/3/2017]
http://lethain.com/overview-of-single-vs-multi-server-architecture/ [24/3/2017]
http://gunicorn.org/ [24/3/2017]
https://uwsgi-docs.readthedocs.io/en/latest/ [24/3/2017]
https://httpd.apache.org/docs/2.4/programs/ab.html [26/3/2017]
http://www.html5xcss3.com/p/responsive-menus-tutorials.html
https://docs.python.org/2/library/configparser.html [10/4/2017]
https://docs.python.org/2/library/xml.etree.elementtree.html [1/5/2017]
https://es.wikipedia.org/wiki/Wikipedia:Portada [Durant tot el projecte]
https://stackoverflow.com [Durant tot el desenvolupament]
https://www.youtube.com/watch?v=NjE_Or4VIlU#t=515 [27/3/2017]
https://www.youtube.com/watch?v=xWInGelLGN8#t=256 [27/3/2017]
https://www.html5rocks.com/en/mobile/responsivedesign/ [27/3/2017]
https://www.slideshare.net/yiibu/adaptation-why-responsive-design-actually-begins-on-the-server [28/3/2017]
https://www.slideshare.net/yiibu/pragmatic-responsive-design [30/3/2017]
http://www.hongkiat.com/blog/responsive-web-tutorials/ [30/3/2017]
https://www.w3schools.com [Durant tot el desenvolupament]
https://docs.djangoproject.com/en/1.11/ [Durant tot el desenvolupament]
http://getbootstrap.com/components/ [Durant tot el desenvolupament]
https://docs.djangoproject.com/en/1.10/howto/custom-template-tags/ [Durant tot el desenvolupament]
https://docs.djangoproject.com/en/1.10/ref/templates/builtins/ [Durant tot el desenvolupament]
https://docs.python.org/2/library/subprocess.html [22/5/2017]
http://effbot.org/zone/element-namespaces.htm [1/6/2017]

Llibres

Títol: RESPONSIVE WEB DESIGN
Autor: Ethan Marcotte

Títol: MOBILE FIRST
Autor: Luke Wroblewski