Universitat Oberta de Catalunya

Memòria projecte final de carrera

Programació d´aplicacions per a mòbils usant HTML5

B4zaar

– Implementació mitjançant pila complerta JavaScript –

Estudiant:	Albert Doménech Olivera
Estudis:	Enginyeria informàtica
Semestre:	Segon
Tutor:	Carles Sànchez Rosa

Barcelona, a 3 de Juny, 2015

1 Resum

La intenció última d'aquest projecte es presentar una serie de tecnologies molt recents en el camp amb l'objectiu de produir una aplicació distribuïda, de la que la part visible està implementada en forma d'aplicació SPA Javascript.

Per crear aquesta aplicació client he utilitzat un marc de treball «Ionic» que estic convençut que agafarà rellevància en un futur immediat. El gran encert del seu disseny en la meva opinió, és la naturalitat amb la qual s'integra amb un marc de treball més conegut com a AngularJS així com la seva integració amb phoneGap.

«Ionic» ens afegeix una capa d'abstracció més on ens permet incorporar els pegats de «PhoneGap»/«Cordova» per accedir a les funcionalitats natives dels dispositius d'una manera senzilla i ordenada que pretén facilitar el seu manteniment

Per altra banda, la incorporació d'una sèrie de tecnologies, en alguns casos disruptives, que ens permeten mantenir l'operació, gestió i monitoratge del sistema durant tot el seu cicle de vida, donen un valor afegit a l'experiència.

Utilitzant la tecnologia de contenidors es desplega la part servidor del sistema en múltiples entorns aconseguint una unificació al mateix temps que capacitats d'alta disponibilitat i capacitat d'escalar horitzontalment sense que es converteixi en una experiència traumàtica.

Resumint podríem dir que el projecte reposa sobre tres grans pilars tecnològics:

1. La pila MEAN com a tecnologia de desenvolupament web
2. El marc de treball «Ionic» com enllaç a les funcionalitats natives dels dispositius
3. La tecnologia de contenidors per desplegar i operar el servei.

Totes les tecnologies i components utilitzats, considero que són bastant recents i és possible que tinguin alguns aspectes encara per polir, però la meva impressió és que conformen una pila molt sòlida i coherent que suposa un salt qualitatiu important.

Sumari

Llista d'Imatges

Llista d'abreviacions i símbols

API	Application Programming Interface
APK	Application Package File
ARM	Advanced RISC Machine
C2DM	Android Cloud to Device Messaging Framework
CDN	Content delivery network
GCM	Google Cloud Messaging for Android
HTML5	HyperText Markup Language version 5
JSON	JavaScript Object Notation
LXC	Linux Containers
MEAN	Mongo, Express, Angular, Node Stack
MVC	Model View Controller
NPM	Node Package Manager
NVM	Node Version Manager
REST	Representational State Transfer
SPA	Single Page Aplication
SVG	Scalable Vector Graphics
UML	Unified Modeling Language
VPS	Virtual Private Server

Llista de projectes i serveis utilitzats

MongoDB	https://www.mongodb.org/	Es tracta d'una base de dades orientada a document en algun punt intermedi entre les NoSQL, clau valor, i les bases de dades relacionals, essent el seu principal punt fort la seva flexibilitat i capacitat d'adaptació.
NodeJS	https://nodejs.org/	Ens proporcionarà el mecanisme per executar JavaScript en el servidor, representa el cor de la pila, en forma de marc de treball construït al voltant del motor JavaScript V8 de Google encarregat de compilar el codi font per generar codi multi-plataforma
Loopback	http://loopback.io/	Marc de treball construït sobre NodeJS i basat en Express especialitzat en el desenvolupament de APIs
AngularJS	https://angularjs.org/	És un marc de treball declaratiu Javascript que segueix el patró d'arquitectura MVC, especialitzat en desenvolupament frontal, el model correspon a informació en format JSON.
Ionic	http://ionicframework.com/	Marc de treball i desenvolupament especialitzat en aplicacions mòbils híbrides basat en Angular que proporciona una infraestructura sòlida amb els pegats PhoneGap / Cordova bàsics integrats.
Vagrat	https://www.vagrantup.com/	Eina de creació, gestió i operació d'entorns virtuals, permetent utilitzar diferents tecnologies de virtualització «Virtualbox» o «VMWare». Ens permet proveir d'entorns virtuals flexiblement.
Docker	https://www.docker.com/	Capa d'abstracció sobre LXC que incorpora accés a tot un catàleg d'imatges de contenidors que ens facilitaran el desplegament i distribució de qualsevol servei
CoreOs	https://coreos.com/	Sistema operatiu lleuger de codi obert basat en el nucli de Linux, derivat de ChromeOS preparat per proveir infraestructura en alta disponibilitat sota demandat. La tecnologia subjacent es la de contenidors, concretament opera molt bé amb Docker.
DigitalOcean	https://www.digitalocean.com	Proveïdor de serveis al núvol, proporciona recursos en forma de VPS (que anomenen Droplets) que s'adapten a qualsevol necessitat. El servei incorpora un servei bàsic de copia de seguretat i monitoratge.
Cloudinary	http://cloudinary.com/	Servei dorsal per la gestió d'imatges, permeten la modificació de les mateixes que són entregades mitjançant l'ús d'un CDN
StrongOps	https://strongops.strongloop.com/	És un servei de monitoratge de rendiment per aplicacions NodeJS, s'integra amb Loopback i ens permetrà consultar les mètriques relacionades amb la nostra API
Genymotion	https://www.genymotion.com/	Continuació del projecte AndroVM proveeix màquines virtuals d'Android per l'arquitectura x86 millorant el rendiment de l'emulador oficial.

2 Prefaci

2.1 Origen del projecte

L'origen d'aquest projecte es dóna fa aproximadament un any, en el context d'una conversa entre amics. Compartint la tarda amb un company lleidatà, tal dia com avui, aquest em va preguntar per la possibilitat de desenvolupar una aplicació per dispositiu mòbil, amb la intenció de fomentar el consum entre particulars, molt similar amb cert producte comercial que en l'actualitat esta tenint bastant èxit.

L'excitació inicial per intentar afrontar aquest repte, va ser un exercici força interessant, però com passa en la majoria d'aquests casos, a poc a poc va anar quedant en res. El company se'n va despreocupar un cop donada la idea i va traslladar el seu focus d'interès sobre altres projectes, també s'ha de dir que ja en aquell moment semblava una idea poc recomanable intentar reproduir un producte que ja existia i ja estava agafant certa embranzida comercial.

De totes maneres aquesta va ser l'excusa perfecta per fer una mica d'investigació, i adonar-me en molt breu temps les tecnologies lliures relacionades, havien sofert un salt evolutiu molt important. De manera que en el moment que vaig veure la possibilitat de realitzar un projecte en aquesta àrea, vaig recuperar la idea inicial del company com un exercici que permetés agafar experiència amb aquestes noves tecnologies i explorar la franja existent entre les àrees tradicionals d'enginyeria del programari i sistemes.

2.2 Motivació

Tota la meva experiència professional sempre ha girat entorn de l'àrea de sistemes, essent aquesta la meva parcel·la d'especialització professional. Durant l'última dècada he pogut observar com aspectes troncals concrets al voltant d'aquest àmbit han sofert una important convulsió. Si fa deu anys les meves tasques principals giraven entorn a garantir el manteniment físic de servidors molt específics on residien els serveis i en proporcionar alta disponibilitat dels mateixos, en funció de la seva criticitat, habitualment mitjançant la duplicació d'elements físics d'infraestructura.

Actualment, a partir principalment de la irrupció de les tecnologies de visualització, ara ja fa algun temps, s'han simplificat enormement la complexitat inherent als

sistemes dedicats, en abstreure i unificar la part física dels sistemes i per tant simplificar tota la gestió del maquinari associat.

Tot i que a priori la virtualització suposa un canvi de paradigma molt important que facilita el desplegament i aprovisionament de nous sistemes d'una manera impensable fins fa pocs anys, col·lateralment també ha provocat una redefinició de les figures dels administradors i arquitectes de sistemes, que en l'actualitat no reposen tant clarament sobre aspectes de manteniment de maquinari com anteriorment, ja que es sol utilitzar maquinari d'ús habitual, minimitzant l'impacte que puguin oferir les particularitats dels diferents sistemes físics i oferint l'alta disponibilitat com un servei elemental proveït per la mateixa tecnologia de virtualització.

D'aquesta manera perfils com el meu ens hem hagut d'actualitzar per dues vies principalment, per una banda incorporant constantment nous coneixements en el nostre bagatge referent a noves plataformes d'infraestructura o de desplegament al núvol. I per l'altra, entrant parcialment en l'àrea del desenvolupament, perquè amb l'aparició de noves tecnologies i metodologies de programari àgils, així com amb una, cada vegada major conscienciació general sobre la necessitat de tenir en compte els sistemes pensant en tot el seu cicle de vida, en aquest context apareix la figura que recentment es coneix com «devops» Un perfil responsable del sistema tant durant la seva fase de desenvolupament com d'operació, difuminant la línia entre les dues disciplines tradicionals.

La motivació per realitzar aquest projecte radica en el fet que els desenvolupaments d'aplicacions mòbils són en general, d'unes dimensions raonables com per ser assumides per un bon desenvolupador o un equip reduït. De fet l'aparició de les diferents plataformes de publicació i distribució de aplicacions per dispositius mòbils, així com l'existència d'eines de codi lliure amb un nivell de qualitat prou madur, considero que han democratitzat en certa manera el mercat de producció i distribució de programari.

Addicionalment un enfocament ampli del projecte, pensant en tot el seu cicle de vida em permetrà definir aspectes relatius al desplegament i operativa del servei que m'ajuden a integrar noves tecnologies molt actuals i explorar aquesta gran franja difusa entre el desenvolupament i les operacions.

3 Requeriments previs

L'objectiu és la realització d'una aplicació que dirigida exclusivament a la compra / venda entre particulars. El que es pretén és proveir una aplicació que promocionar un consum de proximitat utilitzant les possibilitats de posicionament dels dispositius.

Idealment un usuari utilitzarà la càmera del seu dispositiu per publicitar els productes donant-los d'alta en el catàleg de productes de l'aplicació, on incorporarà la seva descripció, un missatge de contacte i dades de la seva localització.

Addicionalment ha de permetre certa interacció entre les persones interessades en un objecte determinat i aquella que l'ha publicitat a través del catàleg de l'aplicació, per tal de tancar les condicions finals de la transacció.

Un requeriment desitjable seria certa integració amb xarxes socials de manera que donés cert valor afegit als perfils d'usuari així com al sistema d'avaluacions que permeti minimitzar els possibles vicis ocults d'aquest tipus de transaccions.

L'aplicació s'hauria de poder executar en el nombre més gran de dispositius mòbils possible, que haurien de disposar d'una càmera i un mecanisme de posicionament que permeti realitzar les fotografies de catàleg d'un producte i posicionar-lo geogràficament.

L'aplicació mòbil descarregarà les dades del catàleg necessàries per operar basant-se en els paràmetres de proximitat geogràfica entre els productes i els usuaris que l'estan consultant això implica que el sistema haurà d'estar preparat per emmagatzemar informació de geo-localització.

4 Llistat d'objectius del projecte

- **Disseny d'una aplicació per mòbils basada en HTML5 que resolgui un problema concret.**
  - Selecció d'una pila actual de desenvolupament que potenciï la simplicitat en l'elaboració i manteniment d'aquest tipus d'aplicacions.
  - Utilització d'eines de prototipatge pel disseny de les vistes.
  - Utilitzar una eina que ens permeti adaptar el model de referència RM-ODP de disseny d'aplicacions distribuïdes de manera formal.
- **Desenvolupament correcte i complet de la aplicació dissenyada.**

  - Desplegament de la pila de desenvolupament necessari per realitzar els productes necessaris per posar en marxa els sistemes necessaris.
  - Emmagatzemar tot el codi i altres enginys relacionats de l'aplicació en un sistema de control de versions.
  - Realitzar els components de l'aplicació utilitzant un desenvolupament orientat a proves.
  - Utilitzar serveis web lleugers (Restfull/JSON) per la publicació dels serveis dorsals.
  - Utilitzar disseny adaptatiu en la part frontal per facilitar la integració en múltiples dispositius hardware amb característiques de visualització diferents.
  - Garantir un grau de seguretat acceptable en les comunicacions entre la part dorsal i frontal del sistema.
- **Integració de l'aplicació en múltiples plataformes.**
  - Integrar les funcionalitats natives de dispositiu necessaris en l'aplicació.
  - Compilació de l'aplicació sobre múltiples plataformes.
  - Empaquetament de l'aplicació per ser distribuïda sobre múltiple plataformes.
  - Distribució de l'aplicació a través de múltiples plataformes (en el seu defecte documentació del procés).
- **Posada en marxa de l'aplicació.**
  - Disseny, implementació i desplegament de la infraestructura necessària per executar l'aplicació.
  - Desplegament dels serveis del dorsal del sistema.
  - Instal·lació de l'aplicació en múltiples dispositius.
  - Comprovació la correcta execució de totes les funcionalitats del sistema.
- **Documentar i valorar l'experiència.**
  - Recopilar tota la documentació relacionada amb les diferents fases del projecte (requeriments, disseny, tècnica i d'usuari).
  - Analitzar l'aplicació resultant i la documentació generada.
  - Redactar les conclusions del projecte.
  - Desenvolupar els materials multimèdia per presentar el projecte.
- **Assegurança de qualitat (QA).**

- Raonar les decisions preses en l'àmbit de les tecnologies emprades per desenvolupar el projecte.
- Donar prioritat a la selecció d'aquelles eines i tecnologies que millorin el desenvolupament d'aquest tipus de projecte a efectes d'agilitat, productivitat, reutilització i claredat en el codi.
- Descripció i compliment de les bones pràctiques recomanades per aquelles tecnologies i estàndards que escollim per desenvolupar el nostre producte.

5 Abast del projecte

Si ens fixem en l'extensió que ha d'abastar el projecte, el que es pretén es que cobreixi tot el cicle de vida de l'aplicació, començant per aquest mateix procés de presa i anàlisi de requeriments fins acabar en la fase de manteniment de l'aplicació un cop ja desplegat el servei i distribuïdes interfícies sobre diferents plataformes. A partir d'aquí propers desenvolupaments correctius o evolutius haurien de tractar-se en projectes independents.

Pel que fa a la part dorsal del sistema desenvolupat aquest hauria d'estar dimensionat per servir en l'ordre dels milers d'usuaris amb un rendiment suficient perquè l'experiència d'usuari de les funcionalitats descrites, siguin fluides i no bloquegin la càrrega de l'aplicació frontal que s'ha de realitzar en un temps raonable sense donar la percepció de lentitud.

S'ha de tenir en compte en aquest sentit, que els dispositius que s'utilitzaran per fer les proves, tot i no ser d'última generació disposen ja de diversos nuclis de processament i la memòria suficient per no representar un coll d'ampolla del rendiment, aquest és un requeriment indispensable perquè l'experiència d'usuari sigui acceptable en una aplicació híbrida en l'execució de la lògica de presentació en el client.

6 Planificació

6.1 Fases del projecte

El projecte es divideix en quatre grans fases que corresponen a les diferents PAC de les que esta conformada l'assignatura, el final de cada fase correspon amb una fita que

certifica el compliment dels terminis establerts. Cal tenir en compte que el projecte queda vinculat a les diferents dates d'entrega pròpies de l'assignatura.
La meva planificació inicial, no havia tingut en compte aquest factor i s'havia establert en funció d'una estimació basada en l'esforç estimat de cada tasca tenint en compte que la data de finalització del projecte quedés aproximadament fixada durant la finalització del semestre. El resultat va ser un projecte que s'ha demostrat massa ambiciós, que durant la fase de desenvolupament es va haver d'anar adaptant gradualment a la realitat d'una complexitat elevada, pel gran nombre de components que hi intervenen, així com els factors més humans de disponibilitat de recursos.

Lamentablement el temps d'inici del projecte, l'estudi de les noves tecnologies m'han anat fent ajustant la planificació final amb un enfocament molt més modest del que pensava assolir, però donant màxima prioritat a les dates d'entrega fixades.

He de reconèixer que al començar el projecte era poc conscient de les problemàtiques (riscos) que m'aniria trobant, així com del meu index real de productivitat amb aquestes tecnologies. He intentat anar ajustant l'abast però mantenint una estructura de components suficient per a aproximar-me el màxim possible al projecte inicial, d'on s'han anat eliminant algunes funcionalitats que no les considerava troncals.

Tot i els imprevistos amb els quals he anat xocant durant el projecte, en aquest punt previ a la finalització del mateix, la sensació és que ha estat un bon exercici amb què he après unes quantes lliçons al mateix temps que he profunditzat en un seguit de tecnologies molt noves que estic segur utilitzaré en futurs projectes.

Així, seguint l'esquema d'entregues de l'assignatura podem fraccionar el projecte en les següents fases:

6.1.1 Fase 1: Pla de treball (PAC1)

Fase corresponent al període compres entre el 25 de febrer del 2015 a l'11 de març del 2015, en aquesta fase s'ha realitzat les següents activitats:

- Anàlisi i proves amb programari de desenvolupament i infraestructura.
- Estudi d'oportunitat.
- Desenvolupar el document de pla de treball

6.1.2 Fase 2: Anàlisis i disseny (PAC2)

Aquesta fase correspon al període comprés entre el 12 de març del 2015 i el 8 d'abril del 2015, en aquesta fase s'afronten les següents tasques:

- Seleccionar les tecnologies per desenvolupar el producte
- Especificar l'aplicació.
- Dissenyar la interfície d'usuari.
- Desenvolupar el document de requeriments.
- Desenvolupar el document de decisions tecnològiques.
- Desenvolupar el document de disseny inicial.
- Desenvolupar el document de desplegament d'eines

6.1.3 Fase 3: Implementació (PAC3)

Correspon al període comprés entre el 9 d'abril del 2015 i el 6 de maig del 2015 on s'ha afrontat la implementació amb les següents tasques:

- Desenvolupament del servei dorsal.
- Desenvolupament de l'aplicació.
- Creació dels enginys de desplegament.
- Proves i correcció d'errors.

6.1.4 Fase 4: Memòria i presentació virtual

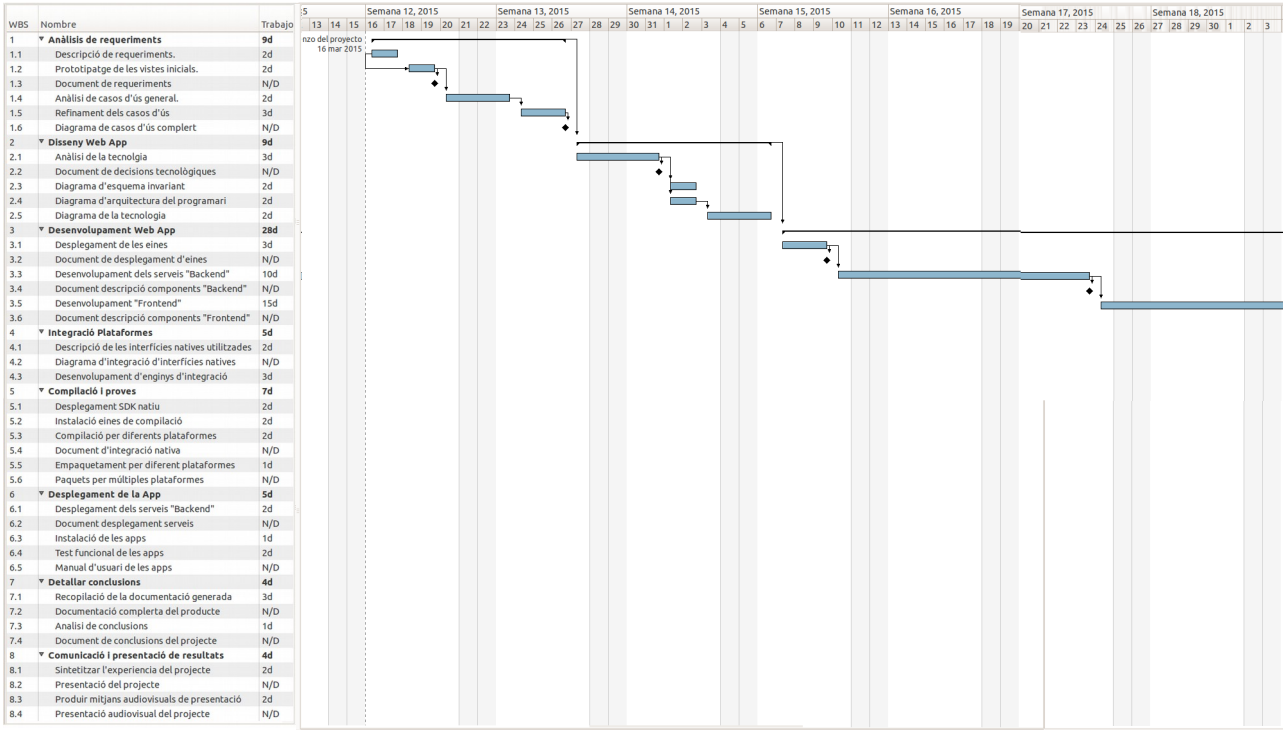
Aquesta fase correspon al període comprés entre el 7 de maig del 2015 i el 3 de juny del 2015, en aquesta s'afronten les següents tasques de conclusió, presentació i finalització del projecte:

- Confeccionar la memòria del projecte.
- Confeccionar una presentació amb els punts més destacables.
- Realitzar la presentació virtual.

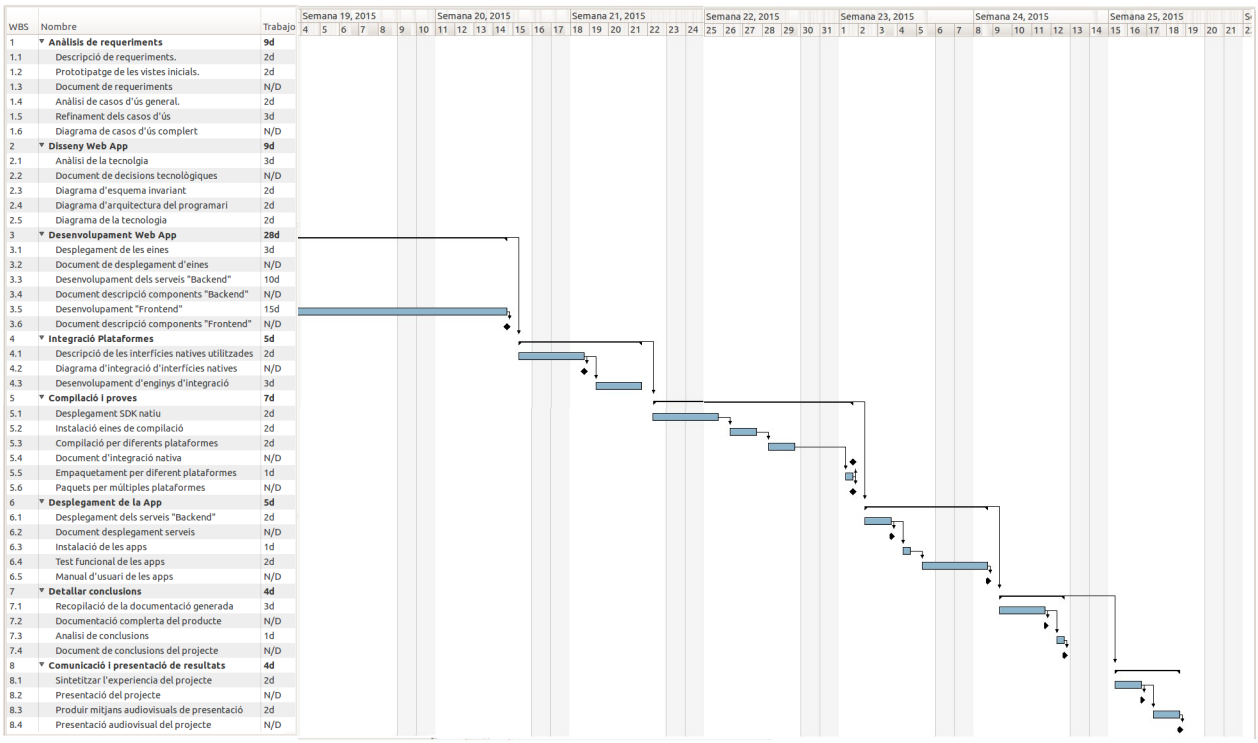
6.2 Diagrama de Gantt Inicial

Aquest diagrama correspon a la meva planificació inicial, on no es van tenir en compte les dates d'entrega de l'assignatura i es va fer l'exercici de planificar lliurement les fases en funció de les necessitats específiques i esforç. Tenint en compte únicament unes dates aproximades d'inici i finalització del semestre. Adjunto aquest diagrama perquè crec que afegeix cert valor observar l'evolució del projecte que en general ha

hagut de reduir els seus objectius i abast per ajustar-se correctament als requeriments demanats:



WBS	Nombre	Trabajo
1	**Anàlisis de requeriments**	**9d**
1.1	Descripció de requeriments.	2d
1.2	Prototipatge de les vistes inicials.	2d
1.3	Document de requeriments	N/D
1.4	Anàlisi de casos d'ús general.	2d
1.5	Refinament dels casos d'ús	3d
1.6	Diagrama de casos d'ús complert	N/D
2	**Disseny Web App**	**9d**
2.1	Anàlisi de la tecnolgia	3d
2.2	Document de decisions tecnològiques	N/D
2.3	Diagrama d'esquema invariant	2d
2.4	Diagrama d'arquitectura del programari	2d
2.5	Diagrama de la tecnologia	2d
3	**Desenvolupament Web App**	**28d**
3.1	Desplegament de les eines	3d
3.2	Document de desplegament d'eines	N/D
3.3	Desenvolupament dels serveis "Backend"	10d
3.4	Document descripció components "Backend"	N/D
3.5	Desenvolupament "Frontend"	15d
3.6	Document descripció components "Frontend"	N/D
4	**Integració Plataformes**	**5d**
4.1	Descripció de les interfícies natives utilitzades	2d
4.2	Diagrama d'integració d'interfícies natives	N/D
4.3	Desenvolupament d'enginys d'integració	3d
5	**Compilació i proves**	**7d**
5.1	Desplegament SDK natiu	2d
5.2	Instalació eines de compilació	2d
5.3	Compilació per diferents plataformes	2d
5.4	Document d'integració nativa	N/D
5.5	Empaquetament per diferent plataformes	1d
5.6	Paquets per múltiples plataformes	N/D
6	**Desplegament de la App**	**5d**
6.1	Desplegament dels serveis "Backend"	2d
6.2	Document desplegament serveis	N/D
6.3	Instalació de les apps	1d
6.4	Test funcional de les apps	2d
6.5	Manual d'usuari de les apps	N/D
7	**Detallar conclusions**	**4d**
7.1	Recopilació de la documentació generada	3d
7.2	Documentació complerta del producte	N/D
7.3	Analisi de conclusions	1d
7.4	Document de conclusions del projecte	N/D
8	**Comunicació i presentació de resultats**	**4d**
8.1	Sintetitzar l'experiencia del projecte	2d
8.2	Presentació del projecte	N/D
8.3	Produir mitjans audiovisuals de presentació	2d
8.4	Presentació audiovisual del projecte	N/D



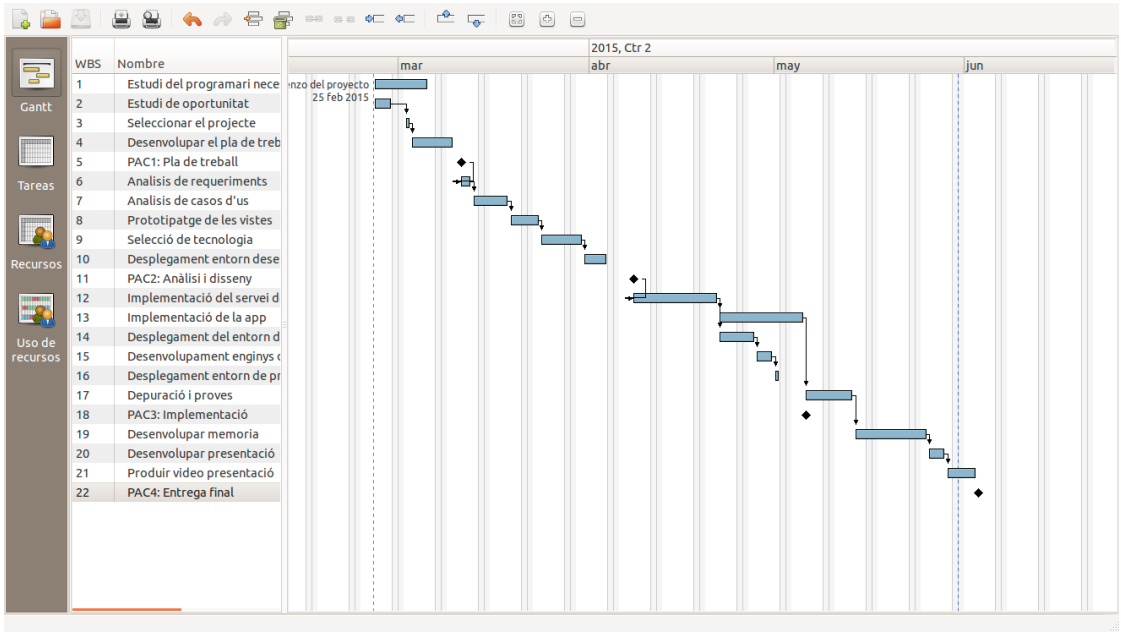
WBS	Nombre	Trabajo
1	**Anàlisis de requeriments**	**9d**
1.1	Descripció de requeriments.	2d
1.2	Prototipatge de les vistes inicials.	2d
1.3	Document de requeriments	N/D
1.4	Anàlisi de casos d'ús general.	2d
1.5	Refinament dels casos d'ús	3d
1.6	Diagrama de casos d'ús complert	N/D
2	**Disseny Web App**	**9d**
2.1	Anàlisi de la tecnolgia	3d
2.2	Document de decisions tecnològiques	N/D
2.3	Diagrama d'esquema invariant	2d
2.4	Diagrama d'arquitectura del programari	2d
2.5	Diagrama de la tecnologia	2d
3	**Desenvolupament Web App**	**28d**
3.1	Desplegament de les eines	3d
3.2	Document de desplegament d'eines	N/D
3.3	Desenvolupament dels serveis "Backend"	10d
3.4	Document descripció components "Backend"	N/D
3.5	Desenvolupament "Frontend"	15d
3.6	Document descripció components "Frontend"	N/D
4	**Integració Plataformes**	**5d**
4.1	Descripció de les interfícies natives utilitzades	2d
4.2	Diagrama d'integració d'interfícies natives	N/D
4.3	Desenvolupament d'enginys d'integració	3d
5	**Compilació i proves**	**7d**
5.1	Desplegament SDK natiu	2d
5.2	Instalació eines de compilació	2d
5.3	Compilació per diferents plataformes	2d
5.4	Document d'integració nativa	N/D
5.5	Empaquetament per diferent plataformes	1d
5.6	Paquets per múltiples plataformes	N/D
6	**Desplegament de la App**	**5d**
6.1	Desplegament dels serveis "Backend"	2d
6.2	Document desplegament serveis	N/D
6.3	Instalació de les apps	1d
6.4	Test funcional de les apps	2d
6.5	Manual d'usuari de les apps	N/D
7	**Detallar conclusions**	**4d**
7.1	Recopilació de la documentació generada	3d
7.2	Documentació complerta del producte	N/D
7.3	Analisi de conclusions	1d
7.4	Document de conclusions del projecte	N/D
8	**Comunicació i presentació de resultats**	**4d**
8.1	Sintetitzar l'experiencia del projecte	2d
8.2	Presentació del projecte	N/D
8.3	Produir mitjans audiovisuals de presentació	2d
8.4	Presentació audiovisual del projecte	N/D

Data d'inici del projecte: 16/03/15 Data finalització del projecte: 18/06/15

Fites del projecte:

- Document de requeriments: 19/03/15
- Diagrama de casos d'ús complet: 26/03/15
- Document de decisions tecnològiques: 31/03/15
- Document de desplegament d'eines: 10/04/15
- Document descripció components dorsals: 23/04/15
- Document descripció components frontals: 14/05/15
- Diagrama d'integració d'interfícies natives: 19/05/15
- Document d'integració nativa: 31/05/15
- Paquets per múltiples plataformes: 31/05/15
- Document desplegament de serveis: 2/06/15
- Manual d'usuari de l'aplicació: 7/06/15
- Document de conclusions del projecte: 11/06/15
- Presentació del projecte: 16/06/15
- Presentació audiovisual del projecte: 18/06/15

6.3 Diagrama de Gantt final



Il·lustració 1: Planificació final

Data d'inici del projecte: 25/02/15 Data finalització del projecte: 03/06/15

Fites del projecte:

- PAC1 pla de treball: 11/03/15
- PAC2 Anàlisi i disseny: 08/04/15
- PAC3 Implementació: 06/05/15
- PAC4 Memòria i presentació 03/06/15

7 Entorn de desenvolupament

7.1 Descripció general

L'entorn pròpiament de desenvolupament es troba instal·lat localment en l'ordinador personal des d'on executem l'entorn integrat de desenvolupament, aquest executa el sistema operatiu de referència per aquest projecte Ubuntu 14.04 LTS. La instal·lació dels components es pot realitzar de múltiples maneres, en aquest cas s'ha intentat mantenir un compromís entre l'estabilitat del gestor de paquets, la necessitat de disposar de versions prou recents i la flexibilitat de fer proves amb múltiples versions dels productes.

Pel que fa a al maquinari utilitzat, personalment he fet anar un equip portàtil comú potenciat de memòria, les característiques concretes mes destacades serien:

- Processador: Intel(R) Core(TM) i5-2467M 1.6Ghz
- Memòria: 8Gb SODIMM @ 1333 MHz
- HD: SAMSUNG SSD PM830 2.5" 7mm 256GB SATAIII

A continuació es reprodueix un guió d'instal·lació dels components principals que es realitza de manera local, però posteriorment les comandes d'aquests guions, s'encapsularan en contenidors que ens permetrà realitzar una instal·lació coherent i aprovisionar un entorn de reproducció o provés així com l'entorn final. Aquests guions es presenten en el material annex.

7.2 Instal·lació de requeriments:

Inicialment necessitarem la instal·lació d'alguns requeriments bàsics per tal de compilar alguns elements de la nostra pila, això ho podem fer una crida al sistema gestor de paquets perquè els instal·li, prèviament resulta bona practica actualitzar el sistema al últim nivell de pegats:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install git build-essential openssl libssl-dev pkg-config
```

7.3 Instal·lació de la capa de dades:

Començarem per la instal·lació de la capa de dades, amb la base de dades MongoDB, aquesta ja existeix per defecte en el llistat de paquets disponibles per la distribució que hem seleccionat com a SO. El problema és que en el nostre cas voldrem disposar de funcionalitats bastant recents que surten de l'abast dels paquets disponibles per defecte.

De manera que el que farem és incloure en el nostre sistema de gestió de paquets les referències del repositori propi de MongoDB necessàries per poder utilitzar directament els paquets que allí es publiquen. Això ho realitzem en dos passos:

1. Afegir la clau de l'equip de MongoDB en el nostre llistat de claus confiables:

   ```
   sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv
   7F0CEB10
   ```

2. Afegir l'adreça del repositori en el nostre sistema de gestió de paquets:

```
echo 'deb http://downloads-distro.mongodb.org/repo/ubuntu-upstart dist 10gen'
| sudo tee /etc/apt/sources.list.d/mongodb.list
```

En aquest punt necessitem actualitzar el llistat de paquets disponibles per tenir en compte aquest nou origen que hem afegit, posteriorment podem instal·lar el paquet mitjançant la seva etiqueta especifica, aquesta solució és força interessant, ja que donat el cas, permetria gestionar les dues versions disponibles de manera simultània.

7.4 Instal·lació del servidor d'aplicacions:

Un cop ja tenim funcionant la base de dades, la part més important que ens restaria, seria instal·lar el servidor NodeJS com a cor de la nostra pila, per fer-ho utilitzarem l'eina NVM (Node Version Manager)1 que és un guió «bash» que ens facilita la instal·lació de múltiples versions de Node en el nostre sistema. Descarregarem el guió localment i l'executem amb la següent comanda:

```
curl https://raw.githubusercontent.com/creationix/nvm/v0.24.1/install.sh |
bash
```

Aquest ens crea un subdirectori en el nostre directori personal .nvm que emmagatzemarà els components del servidor, també ens crea algunes variables d'entorn que necessitem per tal d'operar amb l'eina per tant tancarem el terminal en ús i n'obrirem un de nou per fer aquests canvis efectius. Amb el nou terminal podem instal·lar una versió concreta que hem comprovat prèviament que no ens donarà problemes amb els altres components de la pila mitjançant una senzilla comanda:

```
nvm install v0.10.38
```

En aquest punt podem configurar l'entorn per utilitzar aquesta versió concreta de la rama estable del servidor node amb la comanda:

```
nvm use stable
```

Aquesta peça troncal de la nostra pila queda així coberta pel servidor JavaScript Node, que incorpora una eina pròpia de gestió de mòduls NPM que ens permet incorporar posteriorment funcionalitats en el nostre codi, empaquetades per la comunitat, fomentant així la reutilització de codi.

Un dels problemes que ens trobarem en un entorn de treball Ubuntu, que és el que utilitzarem com a sistema de referència, és que diversos mòduls que requerirem, s'han d'instal·lar per ser utilitzats de manera global en tot el sistema. La instal·lació d'aquest tipus de mòduls per defecte necessiten tenir permisos d'administrador del sistema, però inclús amb la utilització d'una eina del tipus **sudo**, aquesta configuració per defecte sol presentar problemes fàcilment i no queda mai clar on quedaran emmagatzemats els mòduls descarregats.

Després de detectar aquest problema he provat diverses solucions, essent la que s'exposa a continuació, la que s'adapta millor a la situació i la resol de la manera més elegant possible, ja que activa al gestor de paquets de node per instal·lar paquets globalment sense causar problemes amb les variables d'entorn.

```
echo prefix=`dirname $NVM_BIN` > ~/.npmrc
```

D'aquesta manera la instal·lació dels mòduls globals de node quedaran perfectament localitzats com a subdirectori en el nostre directori personal, concretament en aquell on està instal·lada la versió del servidor que NVM està utilitzant.

Cal tenir en compte aquesta situació particular, ja que instal·lacions de noves versions de node NVM i la seva posterior activació alterarien les variables d'entorn i ens obliga a eliminar l'entrada del fitxer `~/.npmrc` i repetir l'ultima comanda per reflectir el canvi.

Un cop desplegats els dos pilars de la nostra pila, el desplegament de la resta de productes relacionats se simplifica mitjançant l'ús de l'eina npm:

```
npm install -g bower grunt-cli gulp strongloop
```

7.5 Instal·lació del marc de treball per dispositius mòbils:

Tan sols ens quedaria instal·lar el marc de treball «Ionic» per tenir la nostra pila de desenvolupament completa, aquest punt en principi simple, es complica una mica per la necessitat de vincular el marc de treball amb «Cordova» i amb una plataforma de desenvolupament mòbil nativa. En el nostre cas per simplicitat de configuració

seleccionarem la plataforma de Google Android que haurem d'accedir a la seva web de descàrrega:

```
wget http://dl.google.com/android/android-sdk_r24.1.2-linux.tgz
```

Hem de parar compte en descarregar únicament el SDK, aquest funciona únicament en sistemes de 32bits i com que l'entorn de prova que estem utilitzant és de 64bits, haurem d'instal·lar les llibreries corresponents per poder executar-lo correctament. Ho podem fer amb la següent comanda:

```
sudo apt-get install lib32z1 lib32ncurses5 lib32bz2-1.0 lib32gcc1 g++-
multilib
```

Un cop complet aquest petit requeriment podem desplegar el SDK, personalment ho fet directament en el directori personal, per simplificar:

```
tar -zxvf android-sdk_r24.1.2-linux.tgz
```

Ja descomprimit haurem de preparar l'entorn per localitzar-lo, ho podem fer immediatament mitjançant l'exportació de variables d'entorn, però si volem fer el canvi persistent, s'ha de crear un fitxer amb les comandes perquè s'executin durant la càrrega del perfil: `/etc/profile.d/android.sh`

```
export ANDROID_HOME=~/android-sdk-linux
export PATH=$PATH:~/android-sdk-linux/tools:~/android-sdk-linux/platform-
tools
```

Aquesta instal·lació encara no està completa, hem d'executar la seva consola un primer cop per acabar de descarregar tots els paquets que necessitem, per fer-ho podem cridar la següent comanda:

```
android sdk
```

Aquesta primera execució ja ens seleccionarà els mínims components requerits per completar les eines necessàries per poder compilar i emular nativament, l'únic que observarem, és que per defecte l'eina tendeix a seleccionar les plataformes i les imatges més recents, per exemple, en el meu cas selecciona per defecte android 5.1.1, aquesta versió és massa recent i podria presentar algun problema, de manera que seleccionarem addicionalment una versió inferior com la 4.4.2 per tenir una alternativa de garantia a efectes de comptabilitat amb la resta de components de la pila.

Un cop desplegat el SDK complert i preparades les variables d'entorn, ja ens trobem en disposició d'instal·lar «Cordova» e «Ionic» amb una senzilla comanda del gestor de paquets de node:

```
npm install -g cordova ionic
```

7.6 Instal·lació eines de desenvolupament.

Amb la pila completa, ja podem posar el focus en les eines que necessitem per dissenyar desenvolupar i provar l'aplicació, podem començar amb la instal·lació de l'entorn integrat de desenvolupament. Podem descarregar el producte WebStorm de JetBrains a l'adreça https://www.jetbrains.com/webstorm/download/ es distribueix en forma de paquet .tar comprimit. En tractar-se d'una aplicació Java requerirem prèviament una màquina d'execució en aquest llenguatge i l'equipament de desenvolupament que utilitzarem igualment durant l'empaquetament de l'aplicació. En el nostre entorn, la manera més senzilla d'instal·lar-ho és amb la següent comanda:

```
sudo apt-get install default-jre default-jdk
```

Finalment podem demanar una llicència gratuïta tant per professors com per alumnes amb l'únic requeriment d'enviar un correu electrònic, validat pel domini de la universitat a la següent adreça: https://www.jetbrains.com/estore/students/

És possible trobar un paquet per la nostra distribució de l'eina «Pencil» que s'ha utilitzat per la creació dels prototipus de les vistes que s'adjunten com annex, aquesta, és una eina curiosa realitzada per una companyia vietnamita, disponible a l'adreça del projecte: http://evoluspencil.googlecode.com/files/evoluspencil_2.0.5_all.deb cal fer notar però que tot i que l'eina compleix perfectament la funció, el projecte sembla una mica abandonat.

En el cas de «plantUML» tan sols hem baixar un paquet Java en format jar en la següent adreça: http://sourceforge.net/projects/plantuml/files/plantuml.jar/download. Posteriorment podrem utilitzar aquest paquet per generar els nostres diagrames amb una comanda similar a:

```
java -jar /home/albert/plantuml/plantuml.jar *.pml -tsvg
```

7.7 Instal·lació de les eines de virtualització

Si volem tenir en compte tot el cicle de vida del sistema d'informació, necessitem un entorn de preproducció o proves que s'assembli el màxim possible a l'entorn de producció, és per això que utilitzarem virtualització per poder aprovisionar aquest entorn, en el nostre propi equip on realitzem el desenvolupament.

En la selecció de components s'ha intentat ser simple per minimitzar la complexitat implícita en la utilització de tants components en la nostra solució, sortosament la plataforma Virtual Box mantinguda per Oracle ens proporciona una base sòlida.

Podem afegir la clau del repositori oficial per obtenir les versions més recents i actualitzar les fonts per tenir els paquets disponibles.

```
wget -q
http://download.virtualbox.org/virtualbox/debian/oracle_vbox.asc -O-
| sudo apt-key add -
sudo apt-get update && sudo apt-get install virtualbox-4.3
```

Addicionalment podem descarregar el paquet d'extensions oficial, això és degut al fet que Oracle no llicencia totes les funcionalitats sota llicenciament GPL, però no existeixen restriccions per a l'ús personal o d'avaluació. El podem descarregar de la següent adreça i la instal·lació es realitza des del mateix producte:

http://download.virtualbox.org/virtualbox/4.3.28/Oracle_VM_VirtualBox_Extension_Pack-4.3.28-100309.vbox-extpack

Amb la plataforma de visualització instal·lada podem desplegar els dos components amb els quals hi interactuarem. Per una banda «Vagrant» que ens donarà una capa d'abstracció al permetre desplegar entorns virtuals d'acord amb guions de text, la seva instal·lació és molt simple utilitzant el gestor de paquets del sistema:

```
sudo apt-get install vagrant
```

Per altra banda també requerim una eina d'emulació que sigui una mica més rapida que l'oficial del paquet de desenvolupament oficial. L'eina «Genymotion» té una versió destinada l'ús personal: https://www.genymotion.com/#!/download aquesta versió ja ens és útil per provar les funcionalitats natives que requerim en el nostre projecte. La instal·lació es limita a descomprimir el fitxer obtingut en un directori de preferència. En realitat l'emulador utilitza la plataforma «VirtualBox» incorporant un catàleg d'imatges Android x86 de diversos dispositius.

Un dels problemes més probables durant la seva utilització inicial, pot ser l'impossibilitat d'accedir a Internet des de l'emulador, per solucionar aquesta situació, hem d'afegir un segon dispositiu de xarxa en mode pont de xarxa.

8 Productes obtinguts

Llistat de productes obtinguts:

1. Document de pla de treball.
2. Document de requeriments.
3. Document de decisions tecnològiques.
4. Document de disseny inicial.

5. Document de desplegament d'eines.
6. Diagrames de disseny: Guions (plantUML) que representen la formalització del disseny que es basa en les interaccions de l'usuari i el model de dades principalment, així com una definició d'arquitectura a molt alt nivell.
7. Codi font dels productes: separat en dos components ben diferenciats.
   - **Servei dorsal** implementat (Loopback): Desplegar en proveïdor al núvol.
   - **Aplicació** implementada (Ionic): Client del servei per dispositius mòbils.
8. Enginys de desplegament:
   - **Guions de creació d'imatges** (Docker): Ens permeten generar imatges de contenidors amb la infraestructura necessària per desplegar el servei dorsal de manera coherent
   - **Guions d'operació del servei** (fleetctl): Ens permeten la desagregació de serveis facilitant-ne la seva gestió granular, facilitant posteriors operacions.
   - **Guions de creació d'entorn virtual** de proves (Vagrant): són una modificació de l'obtingut de https://github.com/coreos/coreos-vagrant perquè s'adapti a les nostres necessitats específiques.
9. **Instal·lador** de l'aplicació: Aquest es proveeix en forma de paquet d'instal·lació per la plataforma Android ARM degudament signat.
10. Memòria final del projecte:
11. Transparències de síntesis:
12. Presentació Virtual:

9 Anàlisi

En aquest apartat recollim els resultats de l'anàlisi inicial sobre les necessitats dels requeriments així com de la gestió completa del projecte.

9.1 Consideracions inicials

Els diagrames que es presenten per descriure els detalls de disseny s'han realitzat majoritàriament mitjançant l'eina «plantUML», després d'intentar diferents opcions, s'ha procedit de la següent manera, amb la consideració que resulta la més adequada a efectes de manteniment i reutilització:

1. Inicialment es crea un fitxer de text (amb l'extensió arbitraria .pml) per cada un dels diagrames a generar, marcant l'inici del document i el seu final amb les etiquetes @startuml i @enduml respectivament.
2. Es comprova la visualització dels diagrames mitjançant el pegat pel navegador Chrome anomenat «PlantUML Viewer» aquests ens permet navegar localment per interpretar directament el fitxer de text que hem generat, el diagrama de sortida, se'ns mostra directament a la finestra del navegador. Aquesta aproximació presenta l'avantatge que qualsevol modificació sobre el fitxer de text es reflexa automàticament en la representació gràfica.
3. Un cop tenim la descripció completa de tots els diagrames que volem mostrar en podem extraure una representació gràfica en un fitxer de format vectorial que serà introduïda finalment com imatge en el document.
4. Per fer-ho hem d'executar aquesta comanda en el directori on resideixen els fitxers de descripció de diagrames:

   ```
   java -jar /home/albert/plantuml/plantuml.jar -tsvg ./*.pml
   ```

   Cal fer notar el paràmetre que indica el format de sortida en «svg» així com la ruta del paquet de l'aplicació que volem executar que variarà en funció de cada instal·lació. Aquest tipus de fitxer de sortida ens proporciona el valor afegit d'adaptar-se fàcilment a diferents mides sense perdre qualitat en els detalls i per tant ens facilita la reutilització dels diagrames.
5. Finalment podem incloure directament els fitxers de text en el document final per mostrar com s'ha construït el diagrama així com la seva representació gràfica mitjançant el fitxer.

En la meva opinió, la potència d'utilització d'aquesta eina, radica en el fet que les descripcions dels diagrames estan realitzades en simples fitxers de text, habilitant la utilització de multitud d'eines existents per tractar aquest tipus de fitxers, per exemple facilitant la comparació, cerca de diferències o el versionat dels diagrames. És per aquest motiu que es reprodueixen les definicions dels diagrames, per fer evident la simplicitat d'ús d'aquesta eina així com la facilitat per integrar-la en qualsevol pila de desenvolupament incorporant els diagrames com a part del codi.

9.2 Requeriments

D'aquests se'n poden distingir i classificar principalment de dos tipus:

9.2.1 Requeriments no funcionals

Aquells que fan referencia a una característica requerida del sistema però que no representi una funcionalitat concreta, sinó que sigui una restricció concreta ja sigui en el procés de desenvolupament, el servei donat o altres aspectes que es contemplin al llarg del cicle de vida del producte.

ID Requisit	***Descripció del requisit***	***Prioritat 1:obligatori 2:valorable 3:opcional***
T0001	Entrega dels executables per executar l'aplicació en les principals plataformes mòbils.	1
T0002	Entrega dels enginys necessaris per desplegar el dorsal de l'aplicació en els diferents entorns necessaris	1
T0003	Utilització d'eines de codi obert que siguin de fàcil accés per qualsevol, proveint de suport per part de la comunitat que en faciliti l'aprenentatge.	1
T0004	Alta capacitat d'escalabilitat horitzontal del sistema en funció d'unes necessitats que podrien ser molt dinàmiques	1
T0005	El desenvolupament ha d'estar dirigit a proves intentant mantenir la cobertura més gran possible dels mateixos per tal de minimitzar l'impacte d'errors detectats en les darreres fases del desenvolupament.	2

9.2.2 Requeriments funcionals

Aquells referits a les característiques requerides del sistema que representi una capacitat d'acció del mateix, d'alguna manera representen les funcionalitats a desenvolupar en el producte.

ID Requisit	***Descripció del requisit***	***Prioritat 1:obligatori 2:valorable 3:opcional***
F0001	L'aplicació ha de ser capaç de publicar un producte associat a una posició geogràfica obtinguda del dispositiu.	1
F0002	La fotografia dels productes s'ha de poder fer amb la càmera del dispositiu.	1
F0003	L'aplicació ha de ser capaç de localitzar productes propers al dispositiu que executa la cerca, en un radi d'acció	1

ID Requisit	Descripció del requisit	Prioritat 1:obligatori 2:valorable 3:opcional
	prèviament determinat	
F0004	L'aplicació ha de permetre comunicar un possible comprador i venedor mitjançant una conversació en format de xat.	1
F0005	Durant qualsevol moment de la conversa el possible comprador ha de ser capaç d'anunciar que passa a recollir el producte en una localització pactada.	2
F0006	A partir del moment que un possible client decideix anar a cercar un producte podrà realitzar-hi una avaluació així com de la mateixa transacció.	2
F0007	L'aplicació ha d'incorporar algun mecanisme per comprovar que el possible comprador d'un producte s'ha presentat a la localització pactada.	2

9.3 Escenaris i components

L'escenari del projecte es l'aplicació distribuïda «B4zaar» que segueix un model client/servidor on la part que actua com servidor resideix en un proveïdor al núvol i la part que actua com a client en forma d'aplicació instal·lada en els dispositius mòbils dels usuaris. Els actors els diferents usuaris i rols definits, el servei dorsal i l'aplicació instal·lada en els dispositius dels usuaris.

9.3.1 Usuaris i rols

En el domini d'aquesta aplicació podem identificar fins a tres tipus d'usuaris diferents, aquells que no estan registrats són els més generals de tots, l'única iteració que possible per ells es la de registrar-se en l'aplicació els anomenarem anònims.

Una especialització d'aquests correspon a aquells usuaris que s'han registrat, tenen un perfil creat i per tant se'ls possibilita la iteració amb la majoria de les funcionalitats de l'aplicació, els denominarem registrats.

Finalment en l'extrem més especialitzat tenim l'usuari administrador que pot interactuar amb l'aplicació de la mateixa manera que un usuari registrat però addicionalment requereix poder interactuar per realitzar tasques de manteniment per resoldre problemes amb les dades introduïdes.

Finalment d'aquests usuaris registrats podem identificar dos tipus de rols que adopten en la seva iteració amb el sistema. Així un usuari registrat pot adoptar el rol de comprador quan cerca un producte en l'aplicació amb l'objectiu final de comprar-la, de la mateixa manera un usuari registrat pot adoptar el rol de venedor si publica un nou producte en l'aplicació amb l'objectiu final de poder-lo vendre.

9.4 Anàlisi de casos d'ús

El diagrama de casos d'ús general ens defineix les interaccions dels usuaris amb el sistema, aquest s'ha mantingut simple per poder ser fàcilment interpretat d'una ullada. Això ha tingut cert cost, ja que la complexitat ha quedat parcialment dissimulada en un cas d'ús potser una mica general «Realitzar transacció» que conté la lògica de negoci més important al contenir tota la comunicació entre els usuaris.

9.4.1 Diagrama Cas d'ús general

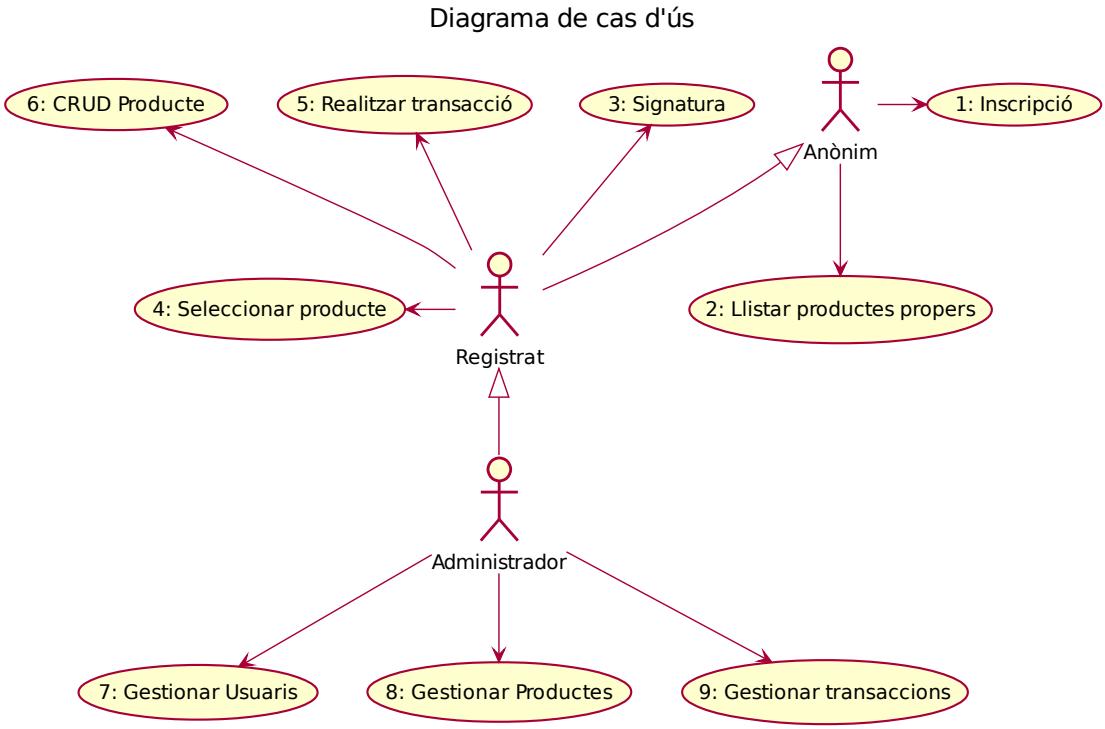
Com es pot observar, en el diagrama, un usuari no registrat, al que es denomina anònim, tan sols tindria la possibilitat d'inscriure's al sistema i visualitzar els productes del catàleg. Acció, aquesta última que es pot realitzar sense signar en el sistema.

Per contra un usuari registrat podria cridar pràcticament tots els casos d'ús addicionals, excloent aquells referits a tasques d'administració que de manera lògica queden vinculats a l'usuari administrador únicament.

Així per exemple, un usuari que ja està registrat pot cridar al cas d'ús referent a la signatura en el sistema, aquest forçosament hauria d'ésser el primer pas necessari per interactuar amb la resta de funcionalitats del sistema.

Un cop ha signat, l'usuari hauria de ser capaç de seleccionar un producte en el qual estigui interessat, essent aquesta selecció de producte el pas previ per iniciar qualsevol transacció.

Un cop seleccionat el producte podrà cridar «realitzar la transacció» aquest cas d'ús és el més carregat ja que implica tota la comunicació entre un possible comprador i un possible venedor i l'intercanvi d'informació de contacte previ a donar per finalitzada la transacció. Donada aquesta complexitat, s'ha optat per descriure'l posteriorment amb major detall mitjançant un «diagrama de seqüencia».



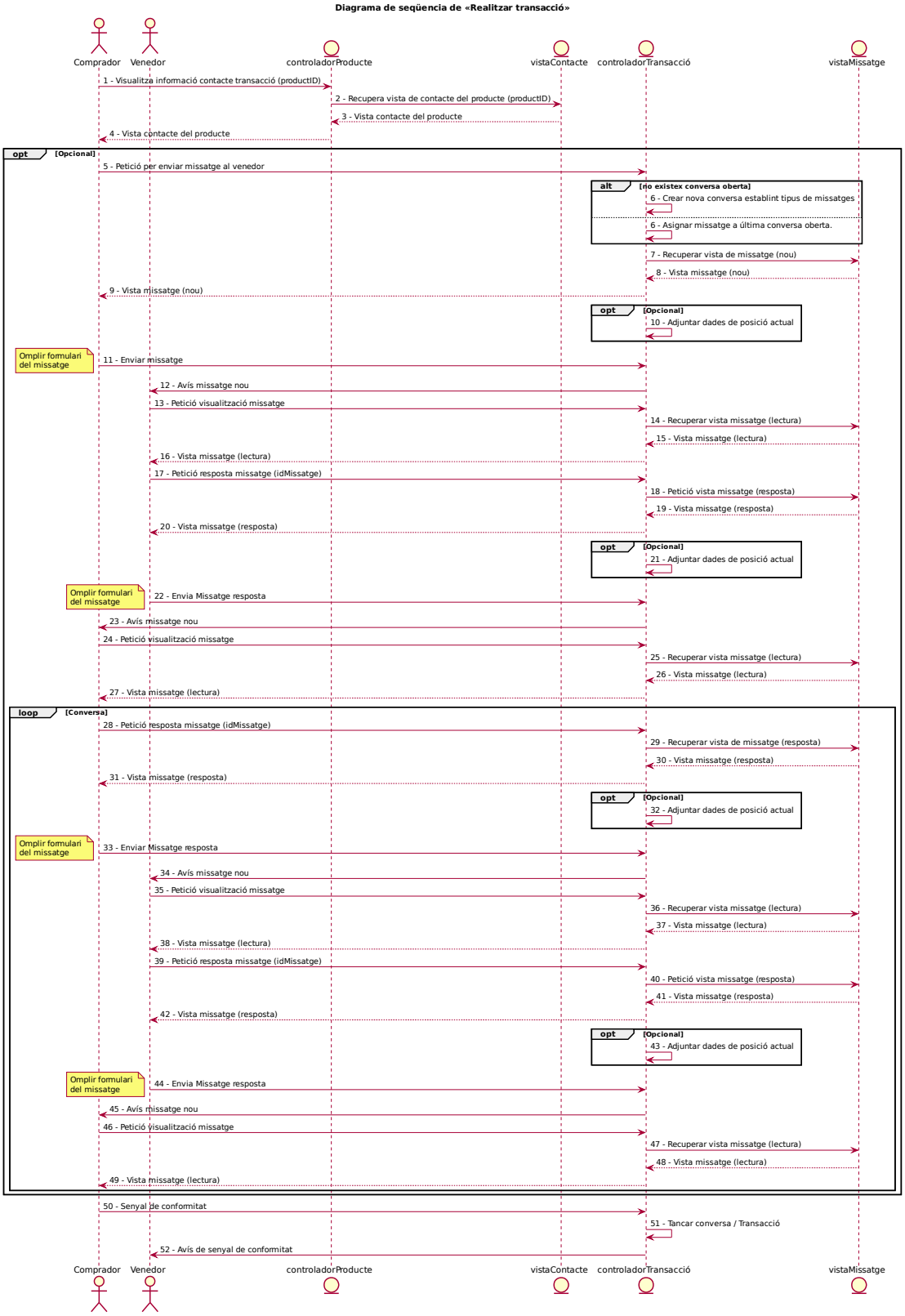
Il·lustració 2: Diagrama de cas d'ús general

9.4.2 Diagrama de seqüencia «Realitzar transacció»

Tal com ja hem comentat el cas d'ús «Realitzar transacció» requereix un nivell de detall major degut principalment a la complexitat que incorpora la funcionalitat de conversa entre els usuaris implicats en la transacció, que implica la comunicació entre dos actors iguals però que adopten rols diferents, de manera que afegim un diagrama addicional, en aquest cas de seqüència que faciliti la comprensió de la iteració dels diferents rols amb el sistema.

Per començar es defineixen els actors, que en aquest cas concret s'han definit tenint en compte els dos rols que pot adoptar un usuari registrat, així com les entitats del sistema que ens ajudaran a detallar la interacció que tenen aquests actors. Tal com es pot observar en la definició cada un d'aquests elements s'ha identificat de manera abreviada per tal de simplificar el màxim possible les posteriors regles de comunicació.

Inicialment un possible comprador requerirà al controlador de producte que mostri la informació de contacte que el venedor va dipositar en donar el producte d'alta en el sistema. El controlador de producte recuperarà la vista associada a la informació de contacte d'aquell producte concret i la mostrarà al comprador.



Il·lustració 3: Diagrama de seqüencia "Realitzar transacció"

En aquest punt el comprador, es pot donar per satisfet amb la informació obtinguda, podria enviar al venedor un senyal de conformitat amb la transacció i senzillament anar buscar el producte, el sistema per la seva banda modifica la transacció per considerar-la tancada i aquest fet hauria d'activar la possibilitat de valoració per part del comprador.

Opcionalment el comprador pot voler preguntar més detalls al venedor i en aquest sentit iniciar una conversació. Per fer-ho realitzarà una petició de nou missatge al controlador de transacció que li subministrarà la vista necessària perquè pugui redactar un missatge que automàticament queda vincular a un venedor un comprador i un producte concret.

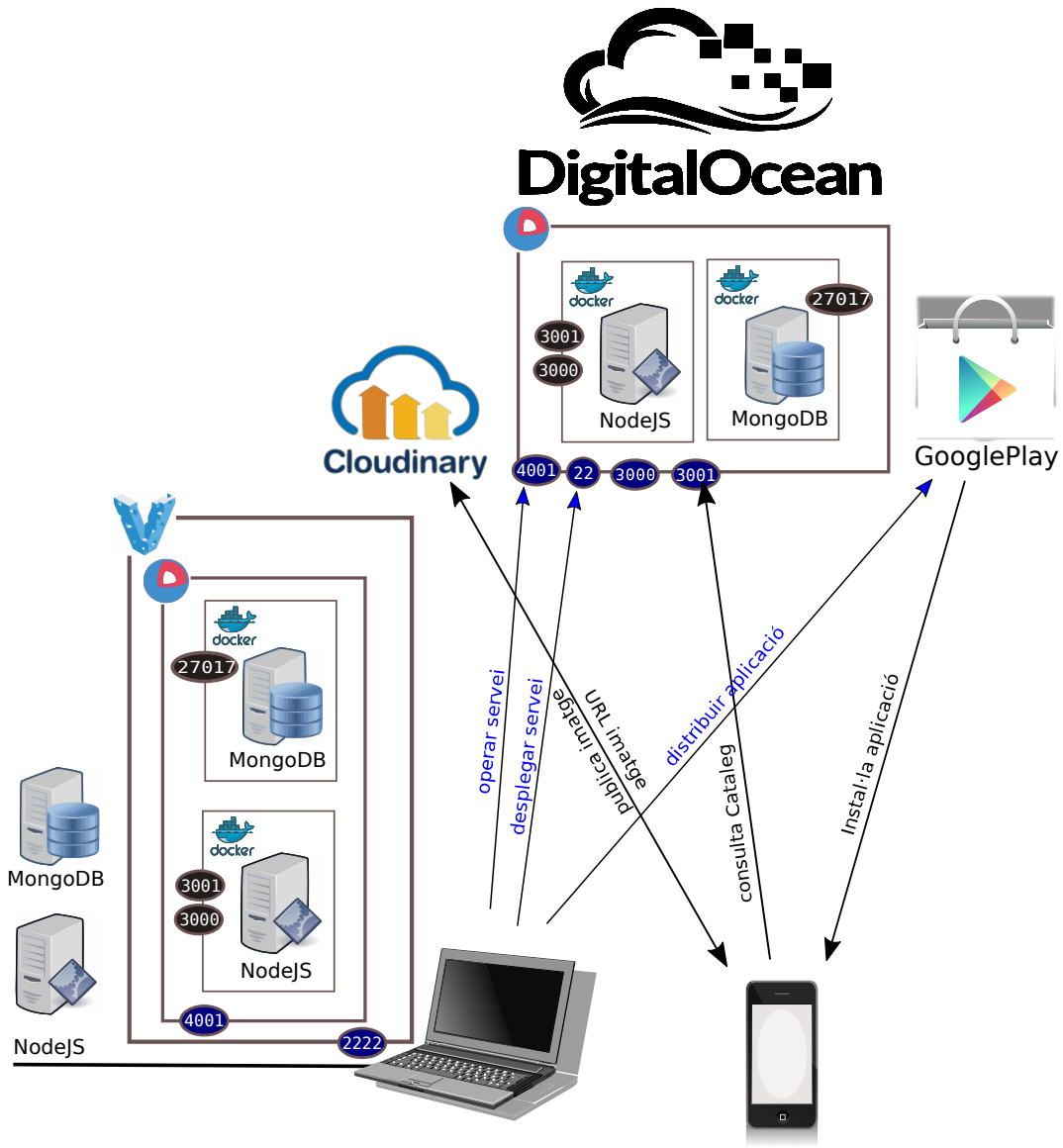
El controlador de transacció també s'encarrega de fer arribar al venedor la notificació d'entrada de missatge nou i igualment proporcionarà a aquest la vista adequada per poder respondre quan aquesta sigui la seva voluntat. El procés es pot repetir indefinidament fins que el comprador doni senyal de conformitat.

10 Disseny

10.1 Arquitectura física

L'arquitectura física d'aquest sistema té un vessant distribuït força important, principalment perquè el nostre servei dorsal es desplega en un servei al núvol amb capacitat per escalar el servei horitzontalment i geogràficament distribuït.

Addicionalment reposem sobre serveis externs que ens faciliten la gestió de tasques específiques. És el cas de «Cloudinary» que ens permet externalitzar la gestió d'imatges simplificant enormement la complexitat de l'aplicació.



Il·lustració 4: Arquitectura física

Aquesta aproximació avantatjosa en alguns aspectes, com per exemple que ens permet reduir l'arquitectura física gestionada pràcticament al mínim estant composta pel nostre equip de desenvolupament així com pels dispositius mòbils dels usuaris. Per altra banda també presenta problemes com per exemple la dependència implícita en la disponibilitat dels serveis. En aquest sentit s'hauria de plantejar en un projecte més complet el disseny de la redundància dels serveis externs crítics com els d'allotjament i el de gestió d'imatges per proveir major grau d'alta disponibilitat.

El diagrama anterior procura plasmar en major detall i de manera gràfica l'arquitectura física (o d'infraestructura) juntament amb alguns components lògics d'infraestructura així com les principals interaccions entre els mateixos.

D'esquerra a dreta i de baix a dalt podem identificar l'entorn de desenvolupament que s'executa localment, el de proves o pre-producció encapsulat en una màquina virtual servida per «Vagrant» i el de producció allotjat en un servei al núvol.

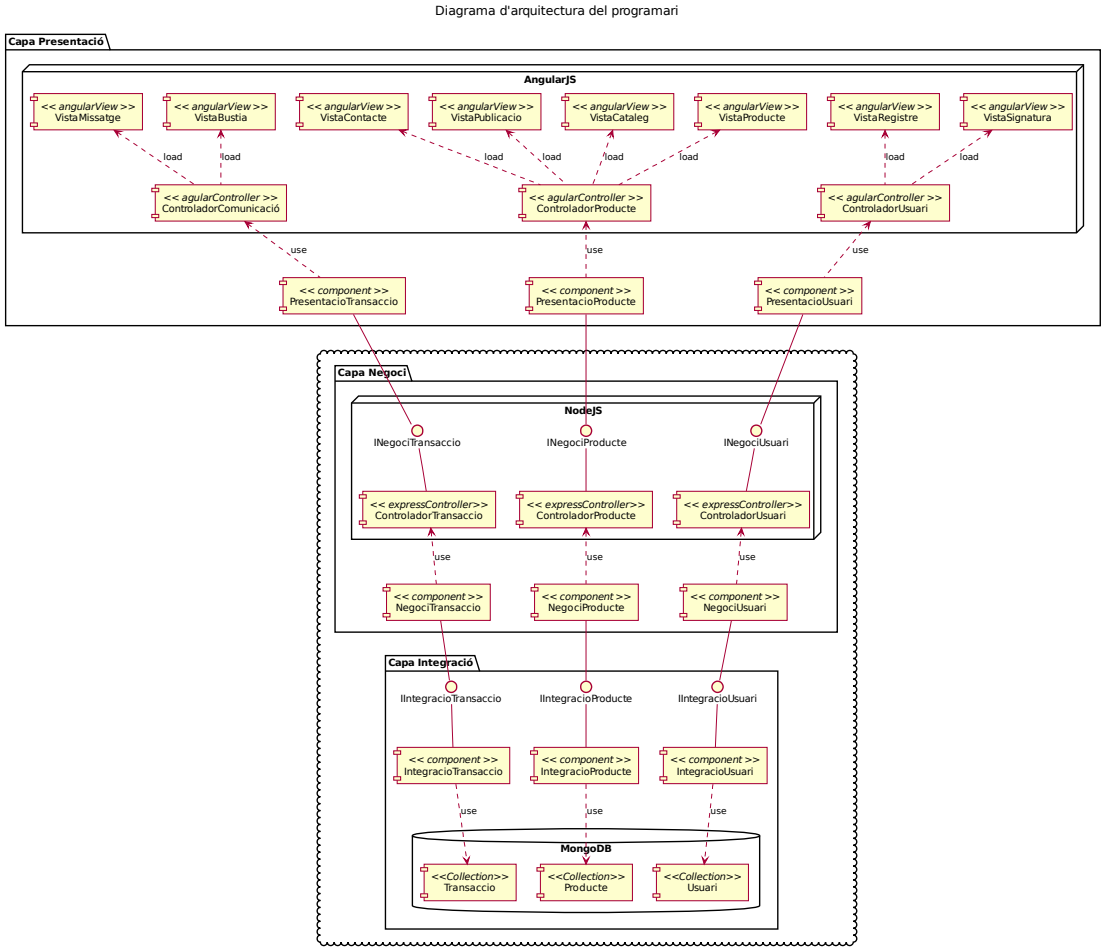
10.2 Arquitectura de programari

Per la seva banda l'arquitectura de programari es basa en una arquitectura distribuïda mixta d'un sistema client / servidor i d'una arquitectura de capes, on la capa de presentació residiria completament en el client i estaria implementada mitjançant el patró de disseny MVC, mentre que les capes de negoci i integració restarien en la part servidor.

En el nostre cas particular aquest servidor compost per un servei que podríem denominar de catàleg de productes, de manera que l'aplicació client ens permetrà tant publicar nous productes per vendre com cercar-ne per comprar-los en el sistema.

La comunicació entre la part client i la part servidor es realitzarà mitjançant serveis lleugers REST que publicaran les dades en format JSON, que al seu torn actuaran com el model del patró MVC en el client.

Per representar el punt de vista de la computació, és a dir el disseny intern de la funcionalitat del sistema i la seva arquitectura utilitzarem un diagrama de components dins d'una arquitectura de capes on cada capa en l'àmbit lògic es representa mitjançant un paquet UML



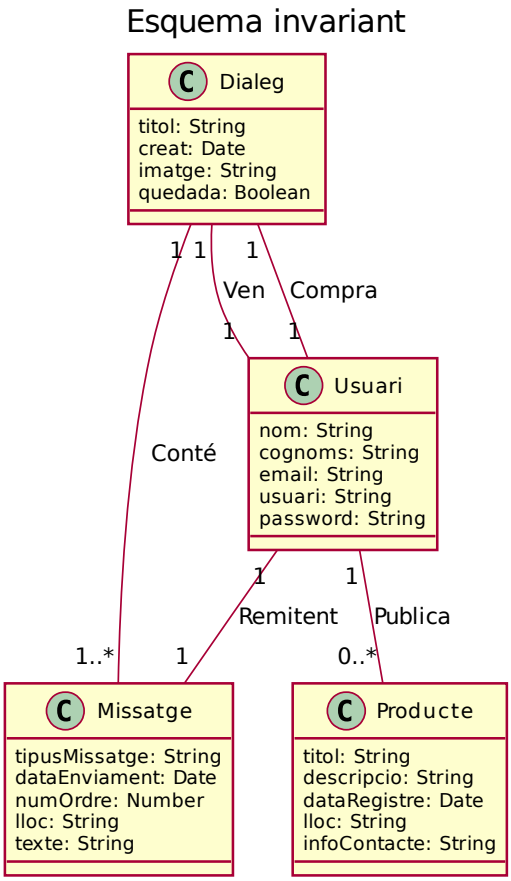
Il·lustració 5: Arquitectura de programari

10.3 Emmagatzemament

Podríem dir que a nivell d'arquitectura, aquest és un dels punts febles de la solució, el problema principal que se'ns presenta és que tot i que la tecnologia de contenidors ens ofereix una flexibilitat de desplegament impensable fa tan sols uns anys i que es tracta d'una de les tecnologies més disruptives de les que disposem actualment. Un dels punts que encara no té solucionat adequadament és el referent al de les persistència de les dades, ja que conceptualment els contenidors no funcionen com una màquina virtual, no emulem cap sistema, el que fem és aïllar els processos que ens interessen pels nostres components, de fet un contenidor es pot interpretar com l'execució encapsulada d'un procés determinat. Tot i que en el meu cas he fet una mica de trampa ,ja que aquest únic procés a executar, en el meu cas és «supervisor», cosa que em permet que aquest delegui l'execució als processos que determini, permeten l'elaboració de contenidors més complexos, al ser un paradigma focalitzat en

l'execució del procés (ideal pels microserveis) encara ha de madurar la persistència de les dades.

quan al model lògic de les dades, aquest ha sofert una forta evolució durant el desenvolupament del projecte, això és pel fet que inicialment pensava utilitzar la pila MEAN completa, això m'hauria permès utilitzar un component anomenat «mongoose» que possibilitava aplicar una major complexitat en el model de dades. Finalment he hagut de descartar l'idea, ja que m'havia quedat bloquejat en un moment donat i he pres la decisió d'utilitzar «Loopback» que em permetia acabar el servei dorsal amb menor esforç. Succeeix que els connectors del mateix s'associen directament a la base de dades, factor que ens limita més les possibilitats alhora de dissenyar el model de dades complexes.



Il·lustració 6: Model de dades

En l'esquema invariant podem observar el model de dades utilitzat finalment, podem comprovar que no s'assembla molt a un model relacional, ja que existeix informació redundada i relacions que originen bucle, això és degut a les característiques NoSQL i l'orientació a col·leccions de les dades pròpia del producte que utilitzem.

11 Implementació

11.1 Decisions d'implementació

Inicialment tenia molt clar que volia desenvolupar aquest projecte utilitzant JavaScript, les piles de desenvolupament complertes en aquest llenguatge ja són una realitat en els últims anys. La meva impressió és que les perspectives de futur d'aquestes tecnologies resulten molt prometedores, més si tenim en compte que darrere d'alguns components

d'aquests esta una companyia com Google que en principi sembla una bona garantia de continuïtat.

En les meves decisions tecnològiques vaig cometre un error, al seleccionar la pila MEAN habitual per desenvolupar el servi REST, el problema és que degut al meu nivell més aviat de principiant en el llenguatge vaig perdre molt de temps en l'adaptació i no em va acabar de funcionar mai correctament. En un moment determinat vaig haver de prendre la decisió de substituir el component «Express» de la pila MEAN per un altre component anomenat «Loopback» que és un marc de treball específic per aquest tipus de tasca.

Al final la valoració del canvi resulta positiva, ja que em va permetre desplegar el servei en un temps raonable però de totes maneres em va penalitzar i trobo que al final el resultat ha quedat una mica lluny del que pretenia aconseguir. Per altra banda em va sorprendre positivament al permetre'm incorporar un servei de monitoratge bàsic mitjançant la inclusió de l'agent de monitoratge de «StrongOps».

La utilització de contenidors per desplegar el servei és una de les pedres angulars del projecte. Personalment trobo que és una tecnologia també amb molt de futur i per la que han demostrat molt d'interès els principals actors del sector. És una tecnologia que encara té carències en alguns aspectes, com en el tema de la persistència de dades però que ens ofereix una flexibilitat adequada per desplegaments al núvol, permetent abaratir el cost d'operació dels projectes, ja que permet que els components aprofitin millor el maquinari, amb una incidència positiva en l'eficiència de la solució.

La decisió d'incorporar el marc de treball «Ionic» la considero igualment una decisió molt encertada, encara que sigui un projecte molt recent, la primera versió estable tot just s'acaba de publicar. Les facilitats que incorpora pel desenvolupament d'aplicacions mòbils són més que remarcables i considero que en els casos anteriors, és una eina de molt futur cridat a jugar un paper protagonista en el desenvolupament d'aplicacions híbrides. S'integra perfectament amb «AngularJS» permetent implementar l'aplicació en format SPA a la que podem afegir els pegats de «Cordova / PhoneGap» per executar les funcionalitats natives de la plataforma.

Aquesta aproximació em sembla força elegant i ens proporciona l'abstracció necessària com per poder centrar l'esforç en el desenvolupament de l'aplicació sense preocupar-nos sobre quina plataforma utilitzarem per desplegar-la. Finalment això és una característica de disseny que en absolut sembla casual i persegueix un salt evolutiu

en el desenvolupament d'aquest tipus d'aplicacions. Una de les raons de més pes per utilitzar tecnologia JavaScript prové del requeriment de multi-plataforma, hem de pensar que s'ha portat a pràcticament sobre qualsevol arquitectura que disposi d'implementació de navegador web.

11.2 Implementació del servei

El servei, tal s'ha implementat amb «loopback», el codi està disponible sota la carpeta b4zaar-strg de l'entrega. La part interessant, on la seva simplicitat resulta evident, es en la definició dels models, que es troba sota la carpeta common/models en ella, es podrà observar que existeix un fitxer de definició de cada model en json i una serie de ganxos associats a cada model en un fitxer Javascript amb el mateix nom.

El servei escolta sobre dos ports de servei, el port 3000 per connexions http i el port 3001 per https, les peticions sobre el primer port es redirigeixen automàticament sobre el segon, això ens afegeix xifrat en la comunicació. Els detalls de implementació es poden consultar sobre el fitxer server.js

Tot i la seva simplicitat, el servei fins i tot incorpora un agent de monitoratge que ens donarà unes mètriques bàsiques del seu funcionament. Això s'aconsegueix executant la següent comanda del propi marc de treball en el directori arrel del nostre projecte i contestar unes senzilles preguntes:

```
# activar stronOps
slc strongops
```

11.3 Implementació del client

El client, l'aplicació mòbil, ha estat desenvolupada en «Ionic» un marc de treball construït sobre «AngularJS» que es un altre marc de treball construït sobre «Jquery». Sembla com si en el desenvolupament frontal s'imposin aquestes arquitectures tipus «ceba» on múltiples capes ens donen cada cop una abstracció major envers un objectiu determinat.

En el cas de «Ionic» l'objectiu es molt clar en donar suport a la inclusió de components natius de dispositius mòbils sobre un desenvolupament Angular. Això ho aconsegueix ja sigui al incorporar integracions bàsiques com donant la possibilitat d'integrar altres pegats PhoneGap/Cordova via un component anomenat ngCordova.

Encara que ho desconec, dubto que existeixi una altra manera, utilitzant codi lliure que ens permeti incorporar aquests pegats d'una manera tan neta i ordenada com en aquest cas.

El desenvolupament el vaig iniciar mitjançant l'eina de comandes del propi marc de treball, aquesta ens permet desplegar plantilles totalment operatives que ens eviten la creació dels projectes de cero i promocionen les bones practiques en l'organització dels aspectes arbitraris del projecte.

```
ionic start B4zaar  sidemenu
```

Amb aquesta comanda es crea el projecte basat en la plantilla «sidemenu» completament operativa sobre la que hem vaig ficar a treballar.

Un dels avantatges d'utilitzar «loopback» en el apartat anterior, es que en proveeix d'un mètode per simplificar-nos el desenvolupament del client, en aquella part més delicada, l'acoblament amb el servidor. Això s'aconsegueix executant la següent comanda en el directori arrel del codi del nostre servei:

```
lb-ng server/server.js client/js/services/lb-services.js
```

Ens genera un fitxer de servei angular amb les operacions i els recursos necessaris per connectar mitjançant REST al servei. Després d'unes modificacions on s'han afegit els mètodes per autenticar els usuaris aquest fitxer queda en el client en la ruta:

```
B4zaar/www/js/services.js
```

De manera que essencialment tota la lògica de la aplicació desenvolupada la trobarem en el fitxer «controllers.js», aquest explota els recursos presentats com a serveis i defineix una serie de controladors associats amb les vistes que resideixen en:

```
B4zaar/www/templates
```

Especialment interessant és el controlador «ProductAddCtrl» on s'implementa la interacció amb les capacitats natives del dispositiu. En el moment de publicar un nou producte requerim d'una fotografia del mateix així com obtenir la seva localització.

L'apartat que requereix major atenció futura es la comunicació entre els usuaris, que ha estat implementada mitjançant un mecanisme de consulta al tenir problemes amb altres opcions molt és recomanades, aquesta funcionalitat no ha estat implementada de manera optima, on ha primat la necessitat de completar-la a temps.

11.4 Desplegament del servei

Per desplegar el servei dorsal de l'aplicació es requereix encapsular prèviament les funcionalitats en un contenidor transportable que ens faciliti el desacoblament amb el maquinari dedicat que presenta l'entorn de desenvolupament. La posada en producció

del servei es realitza en dos passos, desplegant primer en l'entorn de preproducció local en el nostre equip de desenvolupament i posteriorment en l'entorn de producció un cop hem comprovat el seu correcte funcionament, el codi que utilitzat en tots dos entorns varia mínimament, bàsicament només en la identificació de l'agent de monitoratge .

Per desplegar el servei dorsal disposem d'una col·lecció d'enginys que ens facilitaran enormement la tasca i ens donaran una visió molt clara de les possibilitats que disposem amb les tecnologies de contenidors. Aquests enginys resideixen en el directori «deploy» del projecte de servei dorsal.

11.4.1 Creació de les imatges dels contenidors

Un dels primers passos que hauríem d'afrontar és la de preparar les imatges que posteriorment utilitzarem en el nostre entorn. Aquestes estan realitzades a partir d'un guió «Docker» que trobarem en el subdirectori `deploy/docker` del codi del servei dorsal resident en el directori `b4zaar-strng`.

Les imatges són les mateixes independentment de l'entorn i per tant es poden generar en qualsevol dels entorns (la recomanació és fer-ho sempre que sigui possible en l'entorn de producció per motius de rendiment) però inicialment els he creat en l'entorn de desenvolupament és a dir en el meu equip físic.

Per crear les imatges he d'executar la següent seqüencia de comandes, suposant que les iniciem des del directori del servei dorsal:

```
cd deploy/docker/
docker build -t adomenech73/baseuoc baseuoc/
docker build -t adomenech73/mongodb mongodb/
docker build -t adomenech73/nodejs nodejs/
```

Amb les imatges generades que necessitarem per desplegar els nostres entorns, podem pujar el binari a un repositori públic específic que ens permetrà distribuir-les allí on necessitem mitjançant una CDN. La publicació de les imatges es realitza executant les següents comandes:

```
docker push adomenech73/baseuoc
docker push adomenech73/mongodb
docker push adomenech73/nodejs
```

Analitzem una mica més aquestes comandes. En les de creació utilitzem el paràmetre `-t` per identificar la imatge mitjançant una etiqueta o «tag». Aquesta

etiqueta es conforma per dues parts, una primera que identifica el meu directori personal en el servei públic d'imatges, que podem consultar en la següent adreça: https://registry.hub.docker.com/

La segona part de l'etiqueta identifica cada imatge en particular i posteriorment en un segon paràmetre cal indicar el directori on es troba el guió de creació d'aquesta imatge específica.

Resulta interessant fixar-se que en les comandes de publicació en el registre públic no apareix cap referencia al directori on es troba el guió, ja que identifiquem les imatges directament per la seva etiqueta. Això és degut al fet que les imatges, un cop creades s'emmagatzemen en un registre local que podem consultar amb la següent comanda:

```
docker images
```

Si ho fem, podrem comprovar per exemple que la mida de tots tres contenidors junts no sumen més de 1,5Gb, per entendre la transcendència hauríem de tenir en compte que un entorn similar utilitzant la virtualització tradicional fàcilment superaria els 20Gb

11.4.2 Desplegament en l'entorn de proves:

L'entorn de proves el desplegarem igualment en l'equip local però procurarem que sigui completament aïllat i reprodueixi el més fidedignament possible, el que serà l'entorn de producció final. Per crear-lo inicialment he descarregat una plantilla «Vagrant» en l'arrel del projecte que modificarem per adaptar-la al nostre cas particular, per fer-ho primer ens hem de situar en aquesta carpeta i després utilitzarem l'eina anomenada «git» amb la següent comanda:

```
git clone https://github.com/coreos/coreos-vagrant/
```

Això ens crearà un nou directori `coreos-vagrant` amb el contingut del repositori en la seva última versió. Dels fitxers que ens hi copia ens hem de fixar especialment en els següents fitxers sobre els quals realitzarem les modificacions:

- **Vagrantfile**: És el guió principal, al principi del mateix podem determinar alguns recursos assignats a la màquina virtual que desplegarem modificant el valor d'algunes variables. En el meu cas únicament he modificat els següents valors: $vm_memory = 2048 i $vm_cpus = 2.

Addicionalment necessitarem també utilitzar el servei de fitxers compartits entre la màquina virtual i l'equip físic per presentar el servei que volem desplegar l'interior de la màquina virtual, per aconseguir-ho, cap al final del guió he modificat la següent entrada per tenir en compte la localització dels fitxers del nostre projecte:

```
config.vm.synced_folder  "../b4zaar-strng",  "/home/core/share",
id: "core", :nfs => true, :mount_options =>
['nolock,vers=3,udp']
```

D'aquesta manera muntem un recurs a la carpeta «share» de l'usuari per defecte del node coreOS que despleguem.

- **user-data**: El node aprovisionara la informació d'usuari mitjançant «coreos-cloudinit» si localitza un fitxer user-data, això és important a tenir-ho en compte tant en l'entorn de proves com en l'entorn de producció. El fitxer l'he generat a partir de la plantilla que ens ofereix el projecte. L'única modificació destacable és la configuració del «token» d'identificació del «cluster».

CoreOs ens abstrau de la lògica necessària per gestionar sistemes en alta disponibilitat; si despleguem múltiples nodes, aquests s'associen al «cluster» automàticament i pràcticament no requereixen gestió, de manera que simplifiquen la solució, donant-li unes capacitats d'escalabilitat pràcticament il·limitades al mateix temps. Per fer-ho requereix que els nostres nodes s'identifiquin inicialment mitjançant un «token» únic que podem obtenir d'un servei públic.

Per obtenir-ne un de nou, tan sols hem d'adreçar el nostre navegador cap https://discovery.etcd.io/new i recollir el nou «token» de la resposta. Necessitem realitzar aquesta operació cada cop que modifiquem substancialment la configuració del nostre «cluster» o quan copiem el guió d'un ordinador a un altre. Utilitzem el resultat obtingut per modificar al principi de fitxer la cadena d'identificació del nostre «cluster», per exemple en el meu equip actualment estic utilitzant la següent entrada, que lògicament

hauria de ser modificada per qualsevol persona que volgués provar arrancar l'entorn en un altre equip físic:

```
discovery: https://discovery.etcd.io/2c94e720329568a3ee3a930f3aacd3eb
```

- **config.rb**: Aquest fitxer tan sols és realment necessari quan despleguem més d'un node, en ell podem modificar la configuració dels nodes individuals de «cluster» amb les mateixes variables que ho fèiem en el guió Vagrant.

En aquest punt ja podem aixecar l'entorn i connectar-hi amb les següents comandes executades en la carpeta `coreos-vagrant`:

```
vagrant up
vagrant ssh
```

Per apagar o eliminar l'entorn ho podem fer mitjançant les següents comandes executades en el mateix directori.

```
vagrant halt
vagrant destroy
```

Encara que hem de tenir en compte que si eliminem l'entorn haurem de generar un nou «token» d'identificació prèviament a tornar-lo a aixecar.

11.4.3 Desplegament en l'entorn productiu

La informació anterior ens serveix igualment per l'entorn de producció, on el procés és inclús molt més simple. Inicialment ens hem de registrar en el servei a través de l'adreça https://cloud.digitalocean.com i crear el que ells anomenen un Droplet. Aquest servei ens deixa desplegar una instal·lació de CoreOS al que poder associar unes claus per accessos segurs posteriors així com la informació de cloud-init que en l'entorn de proves es trobava en el fitxer `user-data`.

El problema principal que se'ns presenta és que en aquest cas en treballar amb una infraestructura remota no podem utilitzar els recursos compartits per aprovisionar el servei, això ho he resolt mitjançant «`ssh`» per una banda connecto al servidor i creo manualment la carpeta share, per després llençar una comanda en el portàtil per copiar-hi els fitxers del servei

```
# connectar al servidor de producció
ssh -i ~/.ssh/id_rsa core@46.101.158.187
mkdir share
# copiar-hi els fitxers del servei
scp -i ~/.ssh/id_rsa ./b4zaar-strng/*  core@46.101.158.187:~/share
```

```
scp -r -i ~/.ssh/id_rsa ./b4zaar-strng/common  core@46.101.158.187:~/share
scp -r -i ~/.ssh/id_rsa ./b4zaar-strng/deploy  core@46.101.158.187:~/share
```

Amb aquesta seqüencia, m'evito copiar-hi els mòduls de NodeJS que fàcilment poden tornar a ser descarregats durant l'arrencada dels serveis.

11.4.4 Arrancada i parada dels serveis

Aquest procés és exactament igual en tots dos entorns, ja que amb les accions anteriors aconseguim que quedin pràcticament idèntics, utilitzarem els guions. Inicialment hem de connectar a l'entorn, tal com hem comentat en tots dos casos i situar-nos sobre el directori `~/share` :

```
fleetctl submit deploy/fleetctl/mongodb.1.service
fleetctl submit deploy/fleetctl/nodejs.1.service
fleetctl start mongodb.1.service
fleetctl start nodejs.1.service
```

11.4.5 Operativa i gestió

Un cop desplegat els serveis podem continuar realitzant la gestió dels mateixos amb les eines que hem vist fins ara, podem llistar els serveis, visualitzar la informació de registre dels contenidors i inclús la informació de registre en cada servei individual. En el nostre cas concret això s'aconsegueix amb les següents comandes:

```
fleetctl list-units
fleetctl stop mongodb.1.service
fleetctl stop nodejs.1.service
fleetctl destroy mongodb.1.service
fleetctl destroy nodejs.1.service
journalctl -f
fleetctl journal mongodb.1.service
fleetctl journal nodejs.1.service
docker exec nodejs tail -n 100 -f /var/log/node.out.log
docker exec nodejs tail -n 100 -f /var/log/node.err.log
```

11.5 Instal·lació del producte

El producte en sí es l'aplicació pel dispositiu mòbil, aquesta es pot instal·lar a partir d'un paquet en format APK per la plataforma «Android». Encara que la meva idea inicial era proveir també el paquet d'instal·lació per la plataforma «Ios», tan sols seria necessari repetir el procés d'empaquetament en un entorn de desenvolupament sobre «MacOSX».

Un cop generat el paquet, com ha resultat d'una execució en l'entorn de desenvolupament integrat, en el cas de Android, addicionalment hem de signar aquest paquet per poder fer-lo instal·lable, el procés que he seguit per realitzar aquesta tasca es el següent:

```
# generem clau privada
keytool -genkey -v -keystore my-release-key.keystore -alias
alias_name -keyalg RSA -keysize 2048 -validity 10000

# signem el paquet
jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-
release-key.keystore platforms/android/build/outputs/apk/android-
armv7-release-unsigned.apk alias_name

# Alinear el APK
./zipalign -c -v 4 platforms/android/build/outputs/apk/android-armv7-
release-unsigned.apk
```

En aquest punt el APK ja es pot distribuir sense problema.

12 Conclusions

- Les tecnologies presentades es confirmen com productes de present amb una projecció en el futur important on el seu principal actiu és l'agilitat.
- La flexibilitat dels serveis en el nuvol, ens permet dimensionar fàcilment el servei, encara sota situacions a una demanda molt variable.
- La utilització d'una SPA com a base, encaixar fàcilment el desenvolupament web en el desenvolupament de aplicacions mòbils.
- La programació orientada a esdeveniments ens permet millorar la experiència d'usuari.
- «Ionic» és magnific especialment per integrar funcionalitats natives i permetre la inclusió de pegats PhoneGap/Cordova sense que sigui una experiència traumàtica.
- La utilització de serveis web externs en el nostres desenvolupaments ens ajuda a desagregar i externalitzar funcionalitats que d'altra banda ens podrien ser molt costoses.

12.1 Possibles millores

Totes les millores que se m'acudeixen són sobre l'aplicació client, ja que és l'aspecte més fluix del projecte i crec que no he estat capaç d'aconseguir tot el que pretenia inicialment.

Una de les millores més immediates enfocada a l'optimització del sistema seria la implementació d'un servei «push» per facilitar les comunicacions entre els usuaris de l'aplicació. No té gaire de sentit en una arquitectura principalment orientada als esdeveniments i l'asincronisme un sistema de comunicació totalment síncron basat en la comprovació periòdica del servei per detectar l'aparició de nous missatges.

Originalment em vaig plantejar la utilització dels serveis oferts per les plataformes de distribució, inicialment em vaig fixar amb C2DM pels sistemes Android que semblava que s'adaptava perfectament a les meves necessitats. Lamentablement aquest servei està discontinuat des del Juny del 2012 i el servei finalitza definitivament aquest Estiu, el seu substitut GCM senzillament té uns requeriments que el descarten directament en obligar-nos prèviament de registrar-nos com desenvolupador Android i signar cada petició de missatge que realitzem

Sembla així que l'alternativa més recomanable que ens quedaria seria la implementació d'un servei ad-hoc mitjançant la llibreria «socket.io», lamentablement el meu coneixement de la tecnologia és encara insuficient com per afrontar aquesta tasca amb garanties dins del marc d'aquest projecte.

Per altra banda una altra millora recomanable seria la inclusió d'algun sistema de valoració de les transaccions que donés certa garantia als usuaris i ajudés a generar una xarxa de confiança.

13 Bibliografia

- Leandro Navarro Moldes, Juan Manuel Marquès i Puig, *Arquitectura de sistemes distribuïts*, 3rd ed. Barcelona, UOC Setembre 2007
- Ethan Brown, *Web Development with Node & Express*, 1st ed. USA, O'Reilly Media Inc., July 2014
- Amit Gharat, Matthias Nehlsen, *AngularJS UI Development*, 1st ed. UK, Packt Publishing Ltd., October 2014
- Simon Holmes, *Mongoose for Application Development*, 1st ed. UK, Packt Publishing Ltd., October 2014

14 Altres fonts

- Loopback: http://docs.strongloop.com/display/public/LB/LoopBack
- Ionic: http://ionicframework.com/docs/
- Angular: https://docs.angularjs.org/api
- Docker: https://docs.docker.com/
- CoreOs: https://coreos.com/docs/

Annexos:

Annex I: Codi plantUML dels diagrames.

cas.dus.pml

```
@startuml
title Diagrama de cas d'ús
skinparam componentStyle uml2

:Registrat: as reg
:Anònim: as anom
:Administrador: as adm
  anom <|-- reg

  reg <|-- adm
  adm --> (7: Gestionar Usuaris)
  adm --> (8: Gestionar Productes)
  adm --> (9: Gestionar transaccions)
  reg -up-> (3: Signatura)
  anom --> (2: Llistar productes propers)
  reg -left-> (4: Seleccionar producte)
  reg -up-> (5: Realitzar transacció)
  reg -up-> (6: CRUD Producte)
  anom -> (1: Inscripció)
@end
```

dia.seq.realitzar.transaccio.pml

```
@startuml
title Diagrama de seqüencia de «Realitzar transacció»
actor Comprador as C
actor Venedor as V
entity controladorProducte as PC
entity vistaContacte as CV
entity controladorTransacció as TC
entity vistaMissatge as MV

C -> PC : 1 - Visualitza informació contacte transacció (productID)
PC -> CV : 2 - Recupera vista de contacte del producte (productID)
PC <-- CV : 3 - Vista contacte del producte
C <-- PC : 4 - Vista contacte del producte
opt Opcional
  C -> TC : 5 - Petició per enviar missatge al venedor
  alt no existex conversa oberta
    TC -> TC : 6 - Crear nova conversa establint tipus de missatges
  else
    TC -> TC : 6 - Asignar missatge a última conversa oberta.
  end
  TC -> MV : 7 - Recuperar vista de missatge (nou)
  TC <-- MV : 8 - Vista missatge (nou)
  C <-- TC : 9 - Vista missatge (nou)
  opt Opcional
    TC -> TC : 10 - Adjuntar dades de posició actual
  end
  C -> TC : 11 - Enviar missatge
  note left : Omplir formulari \ndel missatge
  TC -> V : 12 - Avís missatge nou
  V -> TC : 13 - Petició visualització missatge
```

```
  TC -> MV : 14 - Recuperar vista missatge (lectura)
  TC <--MV : 15 - Vista missatge (lectura)
  V <-- TC : 16 - Vista missatge (lectura)
  V -> TC : 17 - Petició resposta missatge (idMissatge)
  TC -> MV : 18 - Petició vista missatge (resposta)
  TC <--MV : 19 - Vista missatge (resposta)
  V <-- TC : 20 - Vista missatge (resposta)
  opt Opcional
    TC -> TC : 21 - Adjuntar dades de posició actual
  end
  V -> TC : 22 - Envia Missatge resposta
  note left : Omplir formulari \ndel missatge
  TC -> C : 23 - Avís missatge nou
  C -> TC : 24 - Petició visualització missatge
  TC -> MV : 25 - Recuperar vista missatge (lectura)
  TC <--MV : 26 - Vista missatge (lectura)
  C <-- TC : 27 - Vista missatge (lectura)
  loop Conversa
    C -> TC : 28 - Petició resposta missatge (idMissatge)
    TC -> MV : 29 - Recuperar vista de missatge (resposta)
    TC <-- MV : 30 - Vista missatge (resposta)
    C <-- TC : 31 - Vista missatge (resposta)
    opt Opcional
      TC -> TC : 32 - Adjuntar dades de posició actual
    end
    C -> TC : 33 - Enviar Missatge resposta
    note left : Omplir formulari \ndel missatge
    TC -> V : 34 - Avís missatge nou
    V -> TC : 35 - Petició visualització missatge
    TC -> MV : 36 - Recuperar vista missatge (lectura)
    TC <--MV : 37 - Vista missatge (lectura)
    V <-- TC : 38 - Vista missatge (lectura)
    V -> TC : 39 - Petició resposta missatge (idMissatge)
    TC -> MV : 40 - Petició vista missatge (resposta)
    TC <--MV : 41 - Vista missatge (resposta)
    V <-- TC : 42 - Vista missatge (resposta)
    opt Opcional
      TC -> TC : 43 - Adjuntar dades de posició actual
    end
    V -> TC : 44 - Envia Missatge resposta
    note left : Omplir formulari \ndel missatge
    TC -> C : 45 - Avís missatge nou
    C -> TC : 46 - Petició visualització missatge
    TC -> MV : 47 - Recuperar vista missatge (lectura)
    TC <--MV : 48 - Vista missatge (lectura)
    C <-- TC : 49 - Vista missatge (lectura)
  end
end
C -> TC : 50 - Senyal de conformitat
TC -> TC : 51 - Tancar conversa / Transacció
TC -> V : 52 - Avís de senyal de conformitat
@enduml
```

esquema_invariant.pml

```
@startuml
title Esquema invariant
skinparam componentStyle uml2
class Dialeg {
    titol: String
    creat: Date
    imatge: String
```

```
    quedada: Boolean
}
class Missatge {
    tipusMissatge: String
    dataEnviament: Date
    numOrdre: Number
    lloc: String
    texte: String
}
class Usuari {
  nom: String
  cognoms: String
  email: String
  usuari: String
  password: String
}
class Producte {
  titol: String
  descripcio: String
  dataRegistre: Date
  lloc: String
  infoContacte: String
}

Usuari "1" -- "0..*" Producte : Publica
Usuari "1" -- "1" Missatge : Remitent
Dialeg "1" -- "1" Usuari : Ven
Dialeg "1" -- "1" Usuari : Compra
Dialeg "1" -- "1..*" Missatge : Conté
@enduml
```

dia.arquitectura.pml

```
@startuml
title Diagrama d'arquitectura del programari

package "Capa Presentació" {
  [PresentacioUsuari] << component >> as CPU
  [PresentacioProducte] << component >> as CPP
  [PresentacioTransaccio] << component >> as CPT
node "AngularJS" {
    [ControladorUsuari] << agularController >> as ACU
    [ControladorProducte] << agularController >> as ACP
    [ControladorComunicació] << agularController >> as ACT

    [VistaRegistre] << angularView >> as AVReg
    [VistaSignatura] << angularView >> as AVSig
    [VistaCataleg] << angularView >> as AVCat
    [VistaProducte] << angularView >> as AVPro
    [VistaContacte] << angularView >> as AVCon
    [VistaMissatge] << angularView >> as AVMis
    [VistaBustia] << angularView >> as AVBus
    [VistaPublicacio] << angularView >> as AVPub
  }

  CPU .up.> ACU: use
  CPP .up.> ACP: use
  CPT .up.> ACT: use

  ACU .up.> AVReg: load
  ACU .up.> AVSig: load

  ACP .up.> AVCat: load
  ACP .up.> AVPro: load
  ACP .up.> AVCon: load
```

```
  ACP .up.> AVPub: load

  ACT .up.> AVMis: load
  ACT .up.> AVBus: load
}

cloud {
  package "Capa Negoci" {
    [NegociUsuari] << component >> as CNU
    [NegociProducte] << component >> as CNP
    [NegociTransaccio] << component >> as CNT
    node "NodeJS"{
      Interface "INegociUsuari" as INU
      Interface "INegociProducte" as INP
      Interface "INegociTransaccio" as INT
      [ControladorUsuari] << expressController>> as ECU
      [ControladorProducte] << expressController>> as ECP
      [ControladorTransaccio] << expressController>> as ECT
    }
    INU -- ECU
    INP -- ECP
    INT -- ECT

    CNU .up.> ECU: use
    CNP .up.> ECP: use
    CNT .up.> ECT: use

    INU -up- CPU
    INP -up- CPP
    INT -up- CPT
  }

  package "Capa Integració" {
    Interface "IIntegracioUsuari" as IIU
    Interface "IIntegracioProducte" as IIP
    Interface "IIntegracioTransaccio" as IIT
    [IntegracioUsuari] << component >> as CIU
    [IntegracioProducte] << component >> as CIP
    [IntegracioTransaccio] << component >> as CIT

    database "MongoDB" {
      [Usuari] <<Collection>> as MCU
      [Producte] <<Collection>> as MCP
      [Transaccio] <<Collection>> as MCT
    }
    IIU -- CIU
    IIP -- CIP
    IIT -- CIT

    CIU ..> MCU: use
    CIP ..> MCP: use
    CIT ..> MCT: use

    IIU -up- CNU
    IIP -up- CNP
    IIT -up- CNT
  }
}
@enduml
```

Annex II: Plantilles de creació de contenidors.

baseuoc.Dockerfile

```
FROM      ubuntu
MAINTAINER Albert Domenech <ado@uoc.edu>

# Set Environment
RUN locale-gen --no-purge es_ES.UTF-8
ENV LC_ALL es_ES.UTF-8
# Let the conatiner know that there is no tty
ENV DEBIAN_FRONTEND noninteractive

# Keep upstart from complaining
RUN dpkg-divert --local --rename --add /sbin/initctl && \
    ln -sf /bin/true /sbin/initctl && \
    # Install dependencies
    apt-get update && apt-get upgrade -y && apt-get install -y -q --no-
install-recommends \
      git build-essential openssl libssl-dev pkg-config curl supervisor
python-pip  &&\
    rm -rf /var/lib/apt/lists/* && \
    echo "supervisor-stdout" > base_reqs.txt && \
    pip install -r base_reqs.txt && \
    rm -rf ~/.pip/cache && \
    rm -rf build && \
    mkdir -p /var/log/supervisor

ADD supervisord.conf /etc/supervisor/conf.d/supervisord.conf

CMD ["/usr/bin/supervisord"]
```

baseuoc.supervisor.conf

```
[supervisord]
nodaemon=true

[eventlistener:stdout]
command = supervisor_stdout
buffer_size = 100
events = PROCESS_LOG
result_handler = supervisor_stdout:event_handler
```

mongodb.Dockerfile

```
FROM adomenech73/baseuoc
MAINTAINER Albert Domenech <ado@uoc.edu>

RUN apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 7F0CEB10
&& \
    echo 'deb http://downloads-distro.mongodb.org/repo/ubuntu-upstart dist
10gen' \
    | sudo tee /etc/apt/sources.list.d/mongodb.list && \
    apt-get update && apt-get upgrade -y && \
    apt-get install -y -q --no-install-recommends mongodb-org


# Open to connections other than localhost
RUN sed -e '/bind_ip/ s/^#*/#/' -i /etc/mongod.conf
# copy supervisor conf
ADD supervisor/mongodb.conf /etc/supervisor/conf.d/mongodb.conf
```

```
# start supervisor
CMD ["/usr/bin/supervisord"]

# Expose ports
EXPOSE 27017
```

supervisor.mongodb.conf

```
[program:mongodb]
command=/usr/bin/mongod --config /etc/mongod.conf
autorestart=true
```

nodejs.Dockerfile

```
FROM adomenech73/baseuoc
MAINTAINER Albert Domenech <ado@uoc.edu>

ENV NVM_DIR /usr/local/nvm
ENV NODE_VERSION 0.10.38

# Replace shell with bash so we can source files
RUN rm /bin/sh && ln -s /bin/bash /bin/sh
# Install nvm with node and npm
RUN curl https://raw.githubusercontent.com/creationix/nvm/v0.24.1/install.sh
| bash \
    && source $NVM_DIR/nvm.sh \
    && nvm install $NODE_VERSION \
    && nvm alias default $NODE_VERSION \
    && nvm use default

ENV NODE_PATH $NVM_DIR/v$NODE_VERSION/lib/node_modules
ENV PATH      $NVM_DIR/v$NODE_VERSION/bin:$PATH

# copy supervisor conf
ADD supervisor/nodejs.conf /etc/supervisor/conf.d/nodejs.conf
# start supervisor
CMD ["/usr/bin/supervisord"]

EXPOSE 3000 3001
```

supervisor.nodejs.conf

```
[program:npm]
directory=/srv
command=npm install

[program:nodejs]
directory=/srv
environment=NODE_ENV=production
command=node /srv/server/server.js
autostart=true
autorestart=true
startretries=3
stderr_logfile=/var/log/node.err.log
stdout_logfile=/var/log/node.out.log
```

Annex III: Plantilles de operativa dels serveis.

Fleetctl.mongodb.1.service

```
[Unit]
Description= Mongodbis an open-source document database.
# Requirements
Requires=etcd.service
Requires=docker.service
# Dependency ordering
After=etcd.service
After=docker.service
Before=mongodb-discovery.1.service

[Service]
# Let processes take awhile to start up (for first run Docker containers)
TimeoutStartSec=0
Restart=always
RestartSec=10s
# Change killmode from "control-group" to "none" to let Docker remove
# work correctly.
KillMode=none

ExecStartPre=-/usr/bin/docker kill mongodb
ExecStartPre=-/usr/bin/docker rm mongodb
ExecStartPre=-/usr/bin/docker pull adomenech73/mongodb
ExecStart=/usr/bin/docker run --rm --name mongodb -p 27017:27017
adomenech73/mongodb
ExecStop=/usr/bin/docker stop mongodb

[Install]
WantedBy=local.target

[X-Fleet]
# Don't schedule on the same machine as other instances
X-Conflicts=mysql.*.service
```

Fleetctl.nodejs.1.service

```
[Unit]
Description=Node.js is a platform built on Chrome's JavaScript runtime for \
            easily building fast, scalable network applications.
After=mongodb.1.service
Requires=mongodb.1.service
Before=nodejs-discovery.1.service

[Service]
# Let processes take awhile to start up (for first run Docker containers)
TimeoutStartSec=0
Restart=always
RestartSec=10s
# Change killmode from "control-group" to "none" to let Docker remove
# work correctly.
KillMode=none
ExecStartPre=-/usr/bin/docker kill nodejs
ExecStartPre=-/usr/bin/docker rm nodejs
ExecStartPre=-/usr/bin/docker pull adomenech73/nodejs

ExecStart=/usr/bin/docker run --rm --name nodejs -v /home/core/share:/srv \
          --link mongodb:mongodb -p 3000:3000 -p 3001:3001 -e "NODE_ENV=test"
\           adomenech73/nodejs

ExecStop=/usr/bin/docker stop nodejs
```

```
[Install]
WantedBy=local.target

[X-Fleet]
# Don't schedule on the same machine as other instances
X-Conflicts=nodejs.*.service
```