Treball Fi de Carrera

ENGINYERIA TÈCNICA EN INFORMÀTICA DE SISTEMES

Facultat de Matemàtiques
Universitat de Barcelona

VISUALITZACIÓ DE XARXA DE SANEJAMENT SOBRE GOOGLE MAPS

Albert Ortiz Mirabete

Director: Sergio Escalera
Realitzat a: Departament de Matemàtica Aplicada i Anàlisi UB
Barcelona, 10 de juny de 2013

Agraïments

A Sergio Escalera, el meu tutor del projecte, per la seva ajuda i paciència a l'hora de realitzar el projecte.

A José Pedro i Manel, perquè quan em vaig plantejar començar els estudis em van donar totes les facilitats a la feina per poder assistir a classe.

A Víctor, perquè una tarda em va convèncer per tornar a estudiar.

A David Planas, cap de producció d'Aigües de Castellbisbal, pel seu suport durant el projecte.

A tots els professors que he tingut durant els estudis, per la seva dedicació.

I per últim, als companys de classe, família i amics que m'han recolzat durant aquests anys.

Resum

Els sistemes d'informació geogràfica (GIS) son un punt clau per una companyia d'aigües. I amb la implantació de noves tecnologies disposen d'una eina molt eficient per optimitzar la seva xarxa. El que es pretén amb aquest projecte és que el treball de camp sigui una mica més eficient.
A més, volem promoure l'ús de sistemes operatius gratuïts, com Android, que faciliten l'accés a la tecnologia als usuaris.
Tot això ho farem recolzant-nos en 2 eines: una aplicació de camp desenvolupada per dispositius Android i un Web Service per subministrar les dades a aquesta aplicació. A més, Aigües de Castellbisbal vol que la seva Àrea Técnica supervisi tot el procés. És per això que els canvis que es realitzin a camp quedaran pendents de la supervisió final dels tècnics.

Los sistemas de información geográfica (GIS) son un punto clave para una compañía de agua. Y con la implantación de nuevas tecnologías disponen de una herramienta muy eficiente para optimizar su red. Lo que se pretende con este proyecto es que el trabajo de campo sea un poco más eficiente.
Además, queremos promover el uso de sistemas operativos gratuitos, como Android, que facilitan el acceso a la tecnología a los usuarios.
Todo esto lo haremos apoyándonos en 2 herramientas: una aplicación de campo desarrollada para dispositivos Android y un Web Service para suministrar datos a esta aplicación. Además, Aigües de Castellbisbal quiere que su Área Técnica supervise todo el proceso. Es por esto que los cambios que se realizarán en el trabajo de campo quedarán pendientes de la supervisión de los técnicos.

Geographic Information Systems (GIS) are a key point for a water company. And with the introduction of new technologies have a very efficient tool to optimize their network. The aim with this project is turn the job more efficent.
Furthermore, we promote the use of free operating systems such as Android, withi provide access to the technology to users.
All this we will do by relying on two tools: a client application developed for Android devices and a Web Service to provide data to this application. Furthermore, Aigues de Castellbisbal want their Technical Area supervise the whole process. This is why the changes to be made in the client application will be pending for supervision of technicians.

Índex

3. Situació actual

Aigües de Castellbisbal es troba en un procés de revisió i digitalització de la seva xarxa de sanejament. Aquest procés es divideix en les següents fases:

1- Treball de camp
   a. Localització topogràfica dels pous.
   b. Anàlisi de les característiques dels pous.

2- Treball a les oficines
   a. Dibuixar amb Autocad els pous localitzats durant la fase 1ª.
   b. Definició de les característiques amb NETSanea, que és una extensió sobre Autocad que permet definir característiques dels elements i guardar-les en bases de dades.

La fase 1b es realitza amb una aplicació que els operaris porten instal·lada en un portàtil (a partir d'ara, SanejamentApp). SanejamentApp fa servir una base de dades independent de NETSanea i que s'executa off-line. L a seqüència de presa de dades de camp és aquesta:

1- Abans de sortir a camp, han d'actualitzar les taules de parametrització de SanejamentApp al portàtil per si hi ha hagut canvis (per exemple, s'ha definit un nou estat dels pous, un nou material, etc…).
2- L'operari revisa els pous i actualitza les dades a SanejamentApp.
3- Quan l'operari torna a les oficines, el personal de l'Àrea Técnica actualitza la base de dades de SanejamentApp al servidor
4- L'Àrea Tècnica farà una validació de les dades abans de fer la sincronització amb la base de dades de NETSanea.

3.1 Problemes detectats

Aquest procés ha presentat els següents problemes:

1- Econòmics: el cost d'un portàtil es considera elevat i actualment només es disposa d'un portàtil destinat a aquestes tasques. Això limita els recursos humans que es destinaran a digitalitzar perquè només un equip pot sortir simultàniament.

2- Operatius, que podríem dividir en 3 apartats:
   a. Els operaris necessiten que algú d'Àrea Técnica els actualitzi el portàtil abans de poder anar a camp i per la natura de les seves feines, es troben que molts cops son fora de les oficines.
   b. Quan es troben a camp, poden trobar-se que hi ha noves característiques que no estaven parametritzades i no poden tancar la recollida de dades d'aquest pou fins que tornin a les oficines i s'actualitzin les taules de parametrització.
   c. Els operaris saben quin pou estan actualitzant orientant-se amb un plànol que els han imprès. En alguns casos, el pou no es troba en una zona de fàcil accés i si hi ha diversos pous junts, hi ha possibilitats d'error durant la identificació.

3- Higiènics: el treball de camp és un treball dur i els operaris estan exposats a elements nocius. Aquesta exposició fa que hagin de prendre unes mesures higièniques altes i, tot i prendre mesures, han de desinfectar el portàtil cada cop que surten a camp. Netejar a fons un portàtil és un procés delicat.

3.2 Proposta de millora

El que aquest projecte proposa és l'ús de tablets per realitzar el treball de camp. Per fer-ho, caldrà desenvolupar 2 aplicacions:

1- Web Service de sincronització: aquest web service s'encarregarà de :
   a. Proporcionar a les tablets dades actualitzades de les taules de parametrització.
   b. Rebre les dades del pou i actualitzar la base de dades de SanejamentApp al servidor.

2- Aplicació de tablets: aquesta aplicació s'encarregarà de:
   a. Dibuixar els pous sobre Google Maps i geoposicionar a l'operari sobre el mapa.
   b. Fent clic a un pou, recollirem les seves dades del Web Service i l'operari pot modificar-les.
   c. Un cop modificades, les envia al Web Service per actualitzar-les a SanejamentApp.

La proposta soluciona tots els problemes que s'havien detectat:

1- Per un preu molt competitiu es poden adquirir tablets amb pantalla d'11" amb 3G i GPS, requisits imprescindibles per complir els objectius.
2- A nivell operatiu, cada cop que revisem un pou estem rebent les dades actualitzades i si hi ha canvis, només hem de demanar a Àrea Técnica que ho registri a SanejamentApp i les rebrem a les tablets. El GPS ajudarà a l'operari a millorar la localització dels pous.
3- I a nivell higiènic, netejar una tablet és un procés molt més senzill que un portàtil.

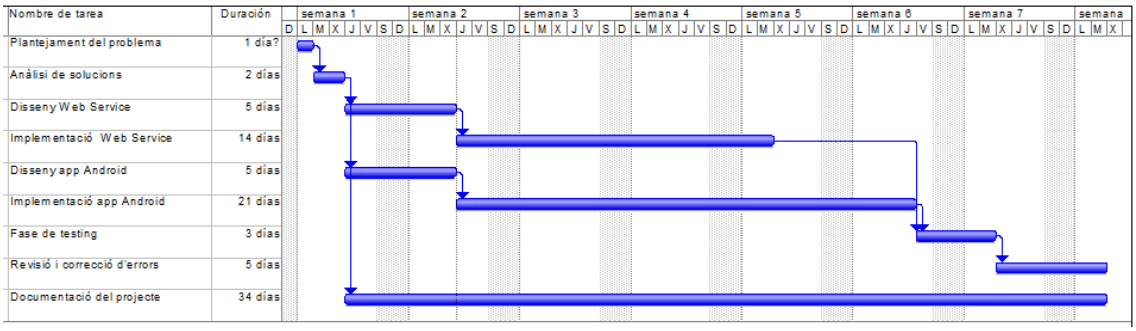
4. Anàlisi

4.1 Planificació

Les fases del projecte son:

- Plantejament del problema
- Anàlisi i avaluacions de possibles de solucions
- Disseny del Web Service
- Implementació del Web Service
- Disseny de la aplicació d'Android
- Implementació de l'aplicació d'Android
- Fase de testeig de la solució
- Revisió i correcció d'errors
- Documentació del projecte

A la imatge següent podem veure la durada i dependència de cadascuna d'aquestes fases:



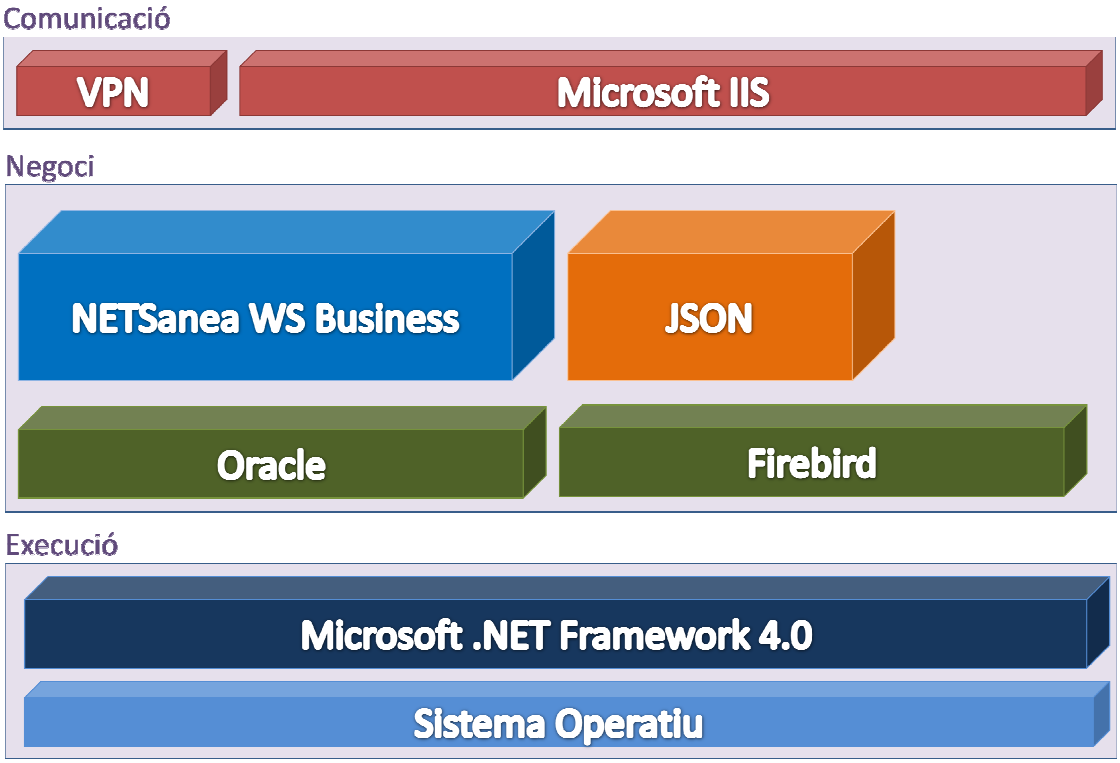
4.2 Arquitectura de la solució i requisits

La solució s'ha dividit en 2 elements:

- **Web Service**
- **Aplicació per a tablet**

4.2.1 Web Service

El **Web Service** es desenvoluparà amb el llenguatge C# i s'executarà sobre el Framework .NET de Microsoft. La versió del Framework amb que s'ha testejat és la 4.0. Aquesta és l'arquitectura completa del Web Service:



Execució

Com a infraestructura d'execució entenem tots els productes des de la capa de sistema operatiu fins al servidor d'aplicacions. Els productes triats son:

- **Sistema Operatiu**: Microsoft Windows Server 2008. Hem de tenir en compte que caldrà configurar el tallafocs del sistema perquè permeti la comunicació amb el Web Service.
- Servidor aplicacions: **Microsoft .NET Framework 4.0** .

Negoci

La infraestructura de negoci es composa dels components que pertanyen a la capa de serveis empresarials. Els components que formen la infraestructura de negoci son:

- Bases de dades: **Oracle 9i / Firebird 2.5**. La versió 9i és una versió que ja no està suportada per Oracle, però es manté perquè és la que Aigües de Castellbisbal té llicenciada. Firebird és una base de dades Open Source que per entorns mitjans dona un gran rendiment.

- **NETSanea WS Business**: Compon el bastidor de serveis comuns i horitzontals de l'aplicació NETSanea WS. Aquests serveis son emprats des de qualsevol punt de la solució. La seva funció és reutilitzar funcionalitat i permetre la integració amb eines externes. Aquesta és l'aplicació que desenvoluparem.

- JSON: És la notació que farem servir per enviar / rebre les dades. JSON és l'acrònim de JavaScript Object Notation, i és un format d'intercanvi de dades, com pot ser XML.

Comunicació

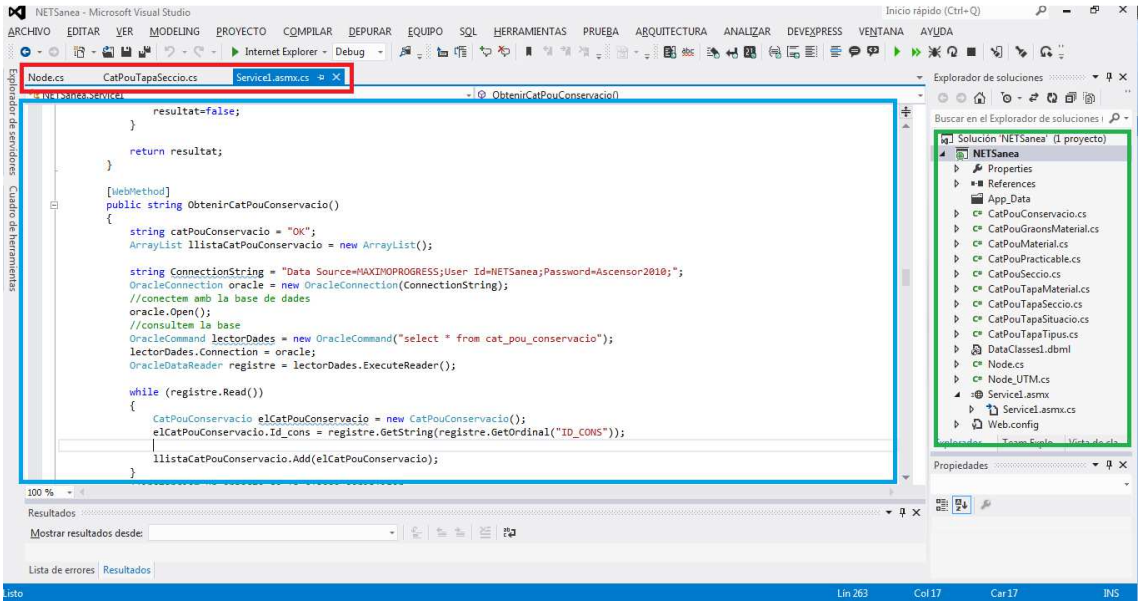
La infraestructura de comunicació dona un únic punt d'accés, tant a usuaris interns com externs, a més de definir una zona de seguretat.

- **IIS**: Servidor d'http de Microsoft. Per no interferir amb altres possibles aplicacions web, canviarem el port del nostre Web Service pel 8070.

- VPN: Les comunicacions amb les tablets es faran sempre sota una VPN. Això ens estalvia afegir seguretat sobre el Web Service i simplifica el desenvolupament. La VPN la establirem amb el software Sonicwall VPN Client, que disposa d'una versió gratuïta per Android. S'ha triat aquest software perquè Aigües de Castellbisbal fa servir un tallafocs Sonicwall.

Entorn de desenvolupament del Web Service

Per desenvolupar el Web Service es farà servir Visual Studio 2012. Visual Studio és un IDE (Entorn de Desenvolupament Integrat) de Microsoft, que suporta diversos llenguatges de programació. La versió *Express Edition* és gratuïta i no inclou característiques avançades. He triat aquest IDE perquè ja estava familiaritzat amb ell per l'ús en l'entorn laboral i per les eines que aporta per a la posada en producció dels Web Service (apèndix XXX). A la imatge següent veiem l'aspecte de l'IDE:



A la part superior (quadre vermell) podem veure els arxius que tenim oberts.

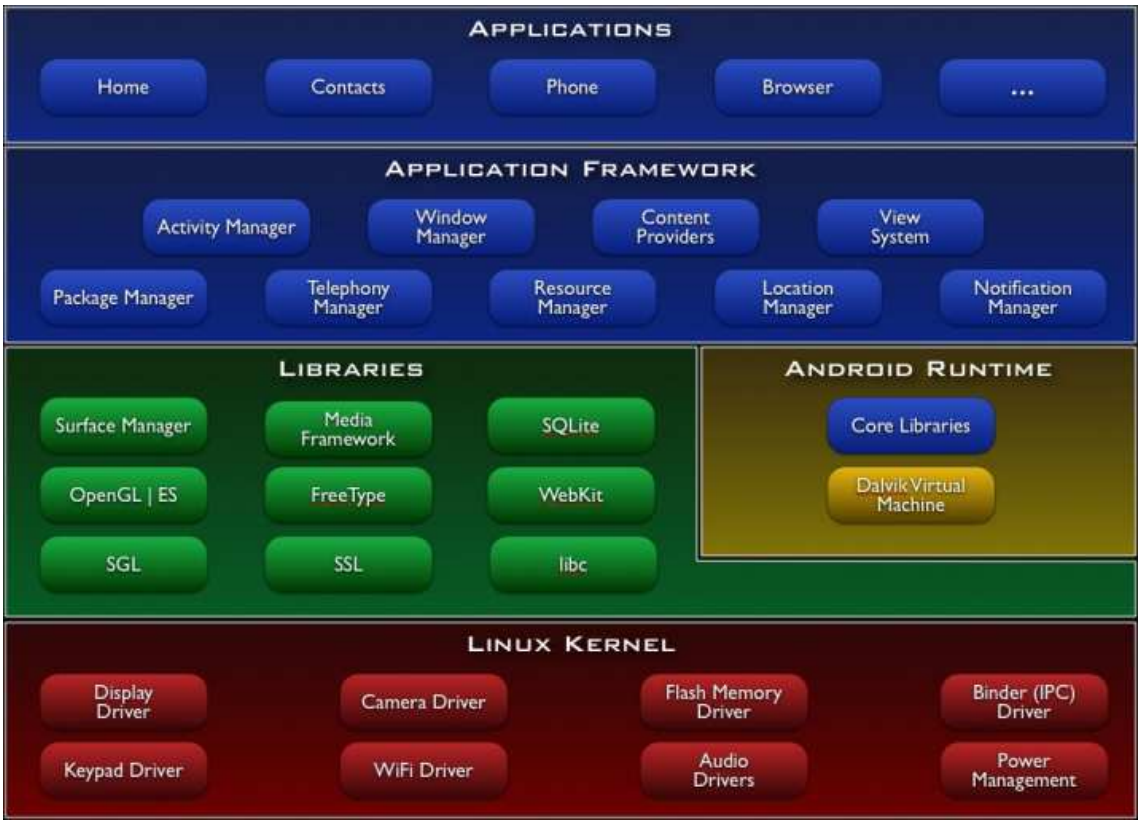
A la part central (quadre blau) visualitzem el codi de l'arxiu

A la part dreta (quadre verd) podem veure els arxius que formen el nostre projecte.

4.2.2 Aplicació per tablet

L'**aplicació per tablet** es desenvoluparà amb l'Android Developer Tools (ADT) sobre Eclipse. El llenguatge amb que es desenvoluparà és Java. A continuació expliquem l'arquitectura dels dispositius Android:

L'arquitectura d'Android està formada per diferents capes. Per accedir a les capes més baixes farem servir llibreries. Cadascuna de les capes fa servir elements de les capes inferiors per realitzar les seves funcions (arquitectura pila). Aquest és el diagrama de les capes:



Kernel de Linux: El núcli del sistema operatiu Android està basat en un kernel de Linux adaptat a les característiques del hardware sobre el que s'executarà Android. El programador no accedeix directament a aquesta capa, sinó que fa servir les llibreries disponibles a capes superiors. Amb això ens estalviem haver de conèixer les característiques físiques de tots els models de dispositius. Per cada element del telèfon (càmera, tarja SD, gps, ...) existeix un driver que permet fer-lo servir des del software. El kernel també s'encarrega de gestionar la comunicació entre processos.

Llibreries: La següent capa es situa just a sobre del kernel la componen les biblioteques natives d'Android, també anomenades llibreries. Estan escrites en C o C++ i compilades per l'arquitectura específica del telèfon. Normalment les desenvolupa el fabricant del dispositiu. L'objectiu de les llibreries és proporcionar funcionalitat a les aplicacions per tasques que es repeteixen amb freqüència, evitant haver de codificar-les cada vegada. Entre les llibreries més important trobarem OpenGL (gràfics), Webkit (navegador), SQLite (base de dades).

Entorn d'execució: Com podem veure al diagrama, l'entorn d¡execució d'Android no es considera una capa en si mateix, donat que també està format per llibreries. Aquí trobarem les llibreries amb funcionalitats habituals de Java i d'altres específiques d'Android. El component principal de l'entorn d'execució d'Android és la màquina virtual Dalvik. Les aplicacions es codifiquen en Java I son compilades en format específic perquè aquesta màquina virtual les executi. Això permet la compatibilitat entre dispositius sempre que les versions d'Android siguin iguals.
Dalvik és una variació de la màquina virtual de Java I no és compatible amb el bytecode Java. Java es fa servir només com a llenguatge de programació i els executables es generen amb extensió .dex.

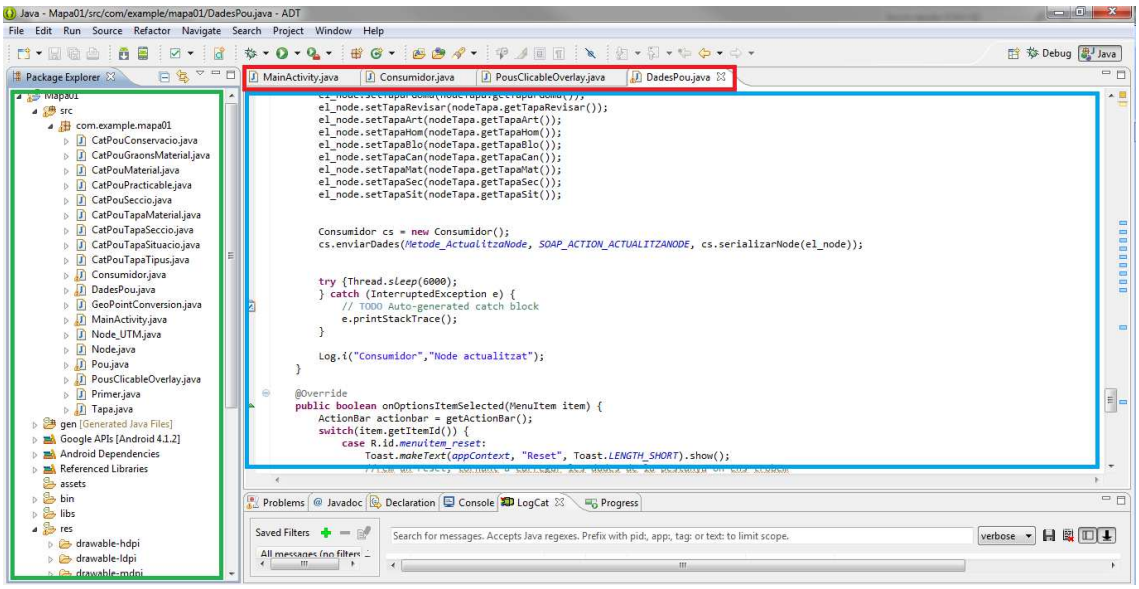
Framework d'aplicacions: La següent capa està formada per totes les classes I serveis que fan servir directament les aplicacions per realitzar les funcions. La majoria dels components d'aquesta capa son llibreries que accedeixen a recursos de les capes anterior a través de la màquina virtual Dalvik. Aqui trobarem:

1. Activity Manager: administra la pila d'activitats de l'aplicació i el seu cicle de vida.
2. Windows Manager: organitza el que es mostrarà a la pantalla
3. Content Provider: crea una capa que encapsula les dades que es compartiran entre aplicacions, per tenir control sobre com s'accedeix a aquesta informació.
4. Views: les vistes son els elements que ens ajuden a construir les interfícies d'usuari: botons, quadres de text, llistes, visor de Google Maps, …
5. Notification Manager: serveis per notificar l'usuari : barra d'estat, sons, leds, vibració…
6. Package Manager: permet obtenir informació sobre els paquets instal·lats al dispositiu i instal·lar nous paquets.
7. Telephony Manager: permet enviar/rebre trucades, sms, mms.
8. Resource Manager: podem gestionar els elements que formen part de l'aplicació I estan for a del codi: texts, imatges, sons, layouts…
9. Location Manager: permet determinar la posició geogràfica del dispositiu mitjançant el GPS o xarxes mòbils. També permet treballar amb mapes.
10. Sensor Manager: permet manipular els sensors: acceleròmetre, giroscopi, sensor de lluminositat, pressió, temperatura, etc…
11. Càmera: permet fer servir la càmera
12. Multimèdia: permet reproduir i visualitzar vídeo, àudio o imatges.

Entorn de desenvolupament de l'aplicació de tablet

Com ja hem comentat, per desenvolupar l'aplicació farem servir Eclipse amb el pluguin ADT d' Android.

A la imatge següent podem veure l'aspecte de l'IDE:



A la part superior (quadre vermell) podem veure els arxius que tenim oberts.

A la part dreta(quadre blau) visualitzem el codi de l'arxiu

A la part esquerra (quadre verd) podem veure els arxius que formen el nostre projecte.

ADT disposa d'un simulador per testejar les aplicacions, però en aquest cas s'ha optat per provar-ho amb una tablet Archos 10i

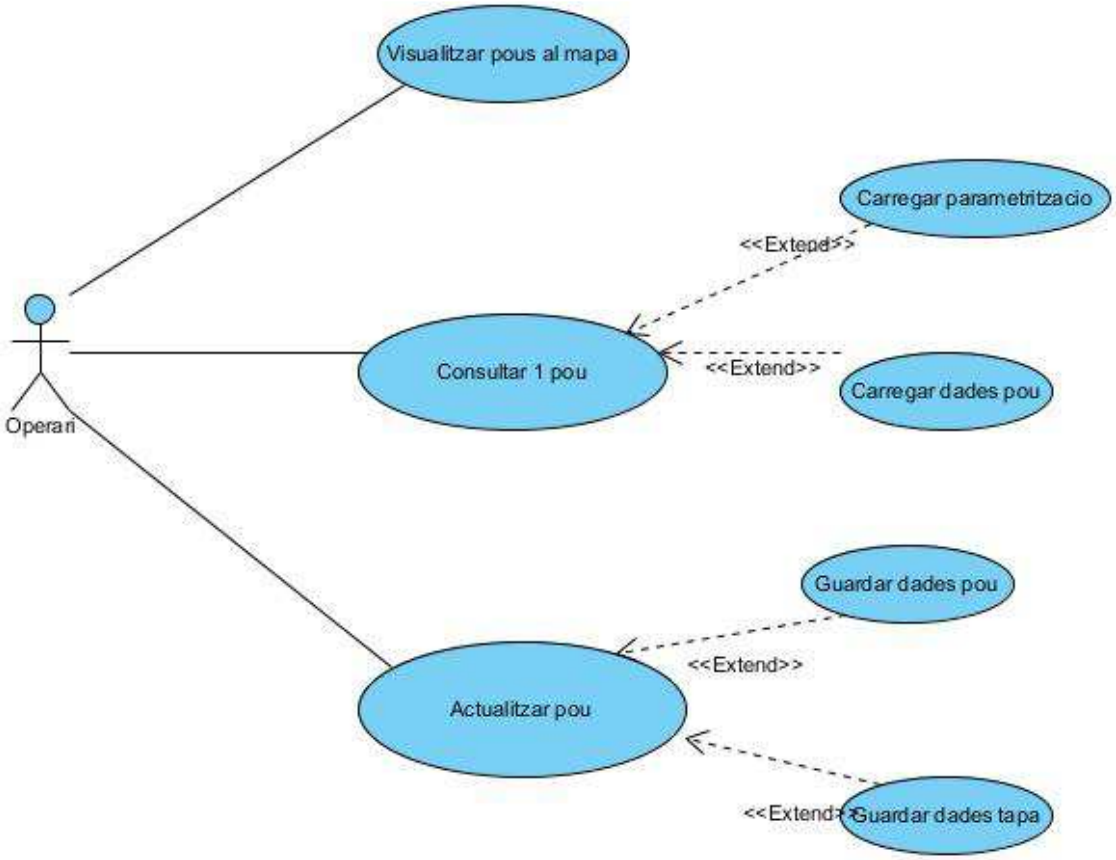
4.3 Requeriments funcionals

4.3.1 Casos d'ús

L'aplicació té els següents casos d'ús:

- Visualitzar pous al mapa
- Consultar 1 pou
- Actualitzar pou

Els podem veure al següent diagrama de casos d'ús:



4.3.1.1 UC1: Visualitzar pous al mapa

Nom:	UC1-Visualitzar pous al mapa
Actors:	Usuari
Descripció:	L'usuari inicia l'aplicació de la tablet i el sistema dibuixa els pous sobre el mapa
Precondicions:	
Flux normal:	1. L'usuari arrenca el programa 2. L'aplicació dibuixa el mapa amb Google Maps 3. L'aplicació sol·licita al web service el llistat de pous 4. El web service retorna el llistat de pous 5. L'aplicació crea els pous 6. L'aplicació afegeix una capa al mapa 7. L'aplicació dibuixa els pous sobre aquesta capa
Flux alternatiu:	3b. Si després de 3 intents no rebem els pous, l'aplicació finalitza

4.3.1.2 UC2: Consultar 1 pou

Nom:	UC2-Consultar 1 pou
Actors:	Usuari
Descripció:	L'usuari visualitza les dades d'un pou
Precondicions:	L'usuari fa clic a un pou del mapa
Flux normal:	1. L'aplicació crea la Activity que mostrarà les dades del pou 2. L'aplicació demana al web service les taules de parametrització 3. El web service retorna les taules de parametrització 4. L'aplicació envia al web service el nº de pou i li demana les dades de l'element 5. El web service retorna les dades del pou 6. L'aplicació carrega els desplegables de les taules de parametrització 7. L'aplicació carrega les dades del pou
Flux alternatiu:	2b. Si després de 3 intents no obtenim les dades, finalitzem l'Activity 3b. Si després de 3 intents no obtenim les dades, finalitzem l'Activity

4.3.1.3 UC3: Actualitzar pou

Nom:	UC3-Actualitzar pou
Actors:	Usuari
Descripció:	L'usuari modifica les dades d'un pou
Precondicions:	L'usuari visualitza les dades del pou
Flux normal:	1. L'usuari guarda els canvis a les dades de la tapa / pou 2. L'usuari pitja el botó per enviar les dades al Web Service 3. L'aplicació envia les dades del pou al Web Service 4. El Web Service actualitza les dades a la BBDD 5. El Web Service retorna el resultat de l'actualització
Flux alternatiu:	3b. Si després de 3 intents no pot contactar amb el Web Service, rebem un avís

4.3.1.4 UC2.1: Carregar parametrització

Nom:	UC2.1-Carregar parametrització
Actors:	Activity de visualització de dades pou
Descripció:	L'usuari visualitza 1 pou i l'aplicació carrega les taules auxiliars
Precondicions:	L'Activity de visualització de dades s'ha creat
Flux normal:	1. L'aplicació fa N consultes al Web Service, 1 per cada taula 2. El Web Service retorna un JSON amb les dades de la taula 3. L'aplicació transforma el JSON en el tipus de dada del desplegable 4. L'aplicació omple el desplegable amb els diferents valors possibles
Flux alternatiu:	1b. Si després de 3 intents no obtenim les dades, finalitzem l'Activity

4.3.1.5 UC2.2: Carregar dades pou

Nom:	UC2.2-Carregar dades pou
Actors:	Activity de visualització de dades pou
Descripció:	L'usuari visualitza 1 pou i l'aplicació carrega les dades del pou
Precondicions:	Hem rebut les taules de parametrització (UC 2.1)
Flux normal:	1. L'aplicació consulta al Web Service les dades d'1 pou (li envia el codi) 2. El Web Service retorna un JSON amb les dades del pou 3. L'aplicació transforma el JSON en un objetce de tipus Node 4. L'aplicació omple els camps de la tapa i el pou amb les dades rebudes
Flux alternatiu:	2b. Si després de 3 intents no obtenim les dades, finalitzem l'Activity

4.3.1.6 UC3.1: Guardar dades del pou

Nom:	UC3.1-Guardar dades pou
Actors:	Usuari
Descripció:	L'usuari fa clic al botó per guardar les dades del pou
Precondicions:	Hem carregat les dades del pou i les mostrem per pantalla
Flux normal:	1. L'aplicació actualitza l'objecte Node amb les dades actuals del pou que visualitzem a la pantalla
Flux alternatiu:	

4.3.1.7 UC3.2: Guardar dades de la tapa

Nom:	UC3.2-Guardar dades tapa
Actors:	Usuari
Descripció:	L'usuari fa clic al botó per guardar les dades de la tapa
Precondicions:	Hem carregat les dades de la tapa i les mostrem per pantalla
Flux normal:	1. L'aplicació actualitza l'objecte Node amb les dades actuals de la tapa que visualitzem a la pantalla
Flux alternatiu:	

4.4 Dades econòmiques del projecte

A continuació es detallen les dades econòmiques del projecte

Quantitat	Article	Preu	Import total
	Desenvolupament aplicacions		
36	Hores programació Web Service	38,50 €	1.386,00 €
60	Hores programació aplicació tablet	38,50 €	2.310,00 €
10	Hores analista	52,50 €	525,00 €
	Infraestructura		
1	Servidor HP Proliant DL360 G8	2.630,00 €	2.630,00 €
1	Windows Server 2008 R2 x64	650,00 €	650,00 €
3	Tablet Archos 10i XS	300,00 €	900,00 €
	Total projecte		8.401,00 €
	IVA 21%		1.764,21 €
	Total		**10.165,21 €**

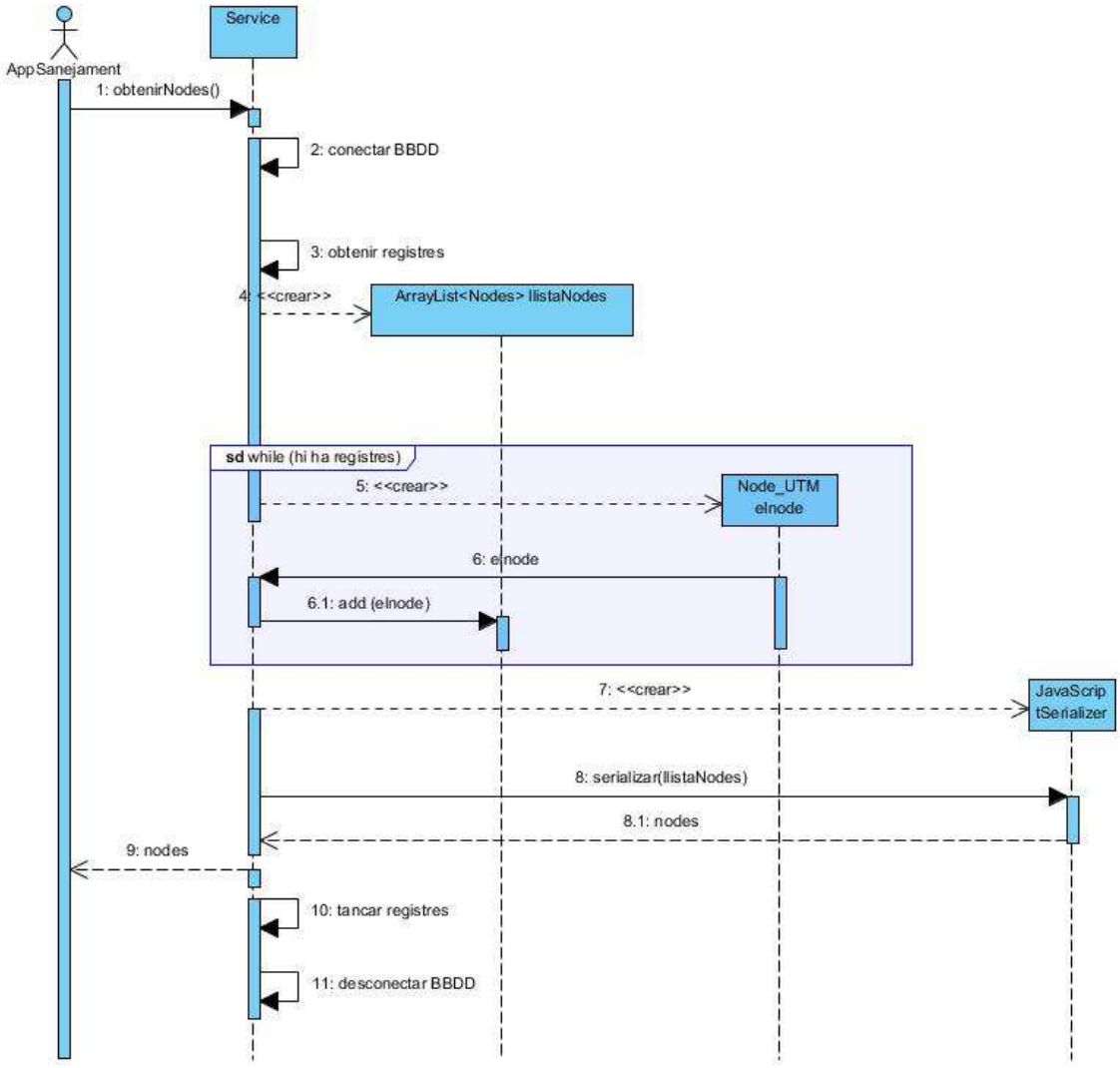
5 Anàlisi

5.1 Diagrames de seqüència

A continuació tenim els diagrames de seqüència de les principals operacions de l'aplicació, tant pel Web Service com per l'aplicació de tablet.

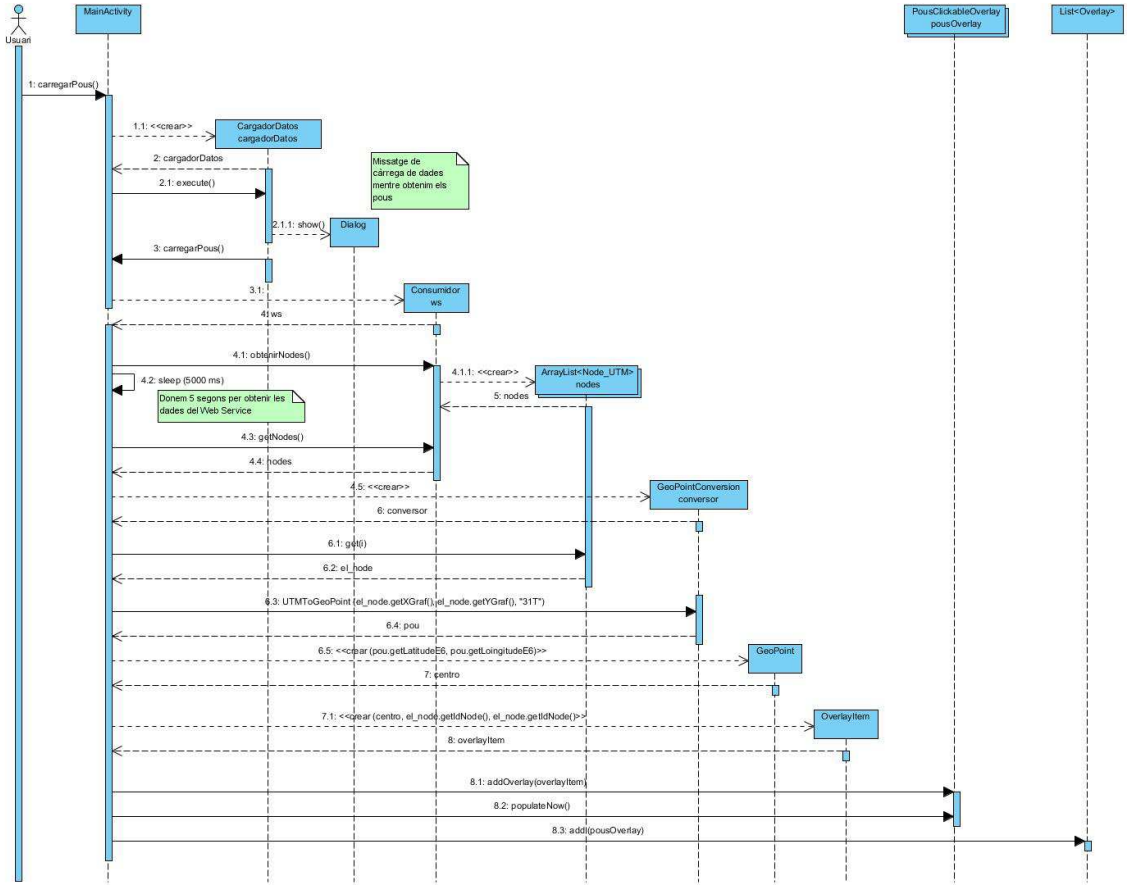
5.1.1 Visualitzar pous al mapa: quan l'usuari inicia l'aplicació, es visualitzen els pous dibuixats sobre Google Maps. L'aplicació connecta amb el WebService i li sol·licita la informació UTM de tots els pous. Aquest és el diagrama de seqüència des del Web Service:

5.1.1.1 DS1.1 WebService:visualitzarPous



I aquest és el diagrama de seqüència de l'aplicació de tablet :

5.1.1.2 DS1.2 AppTablet:visualitzarPous:

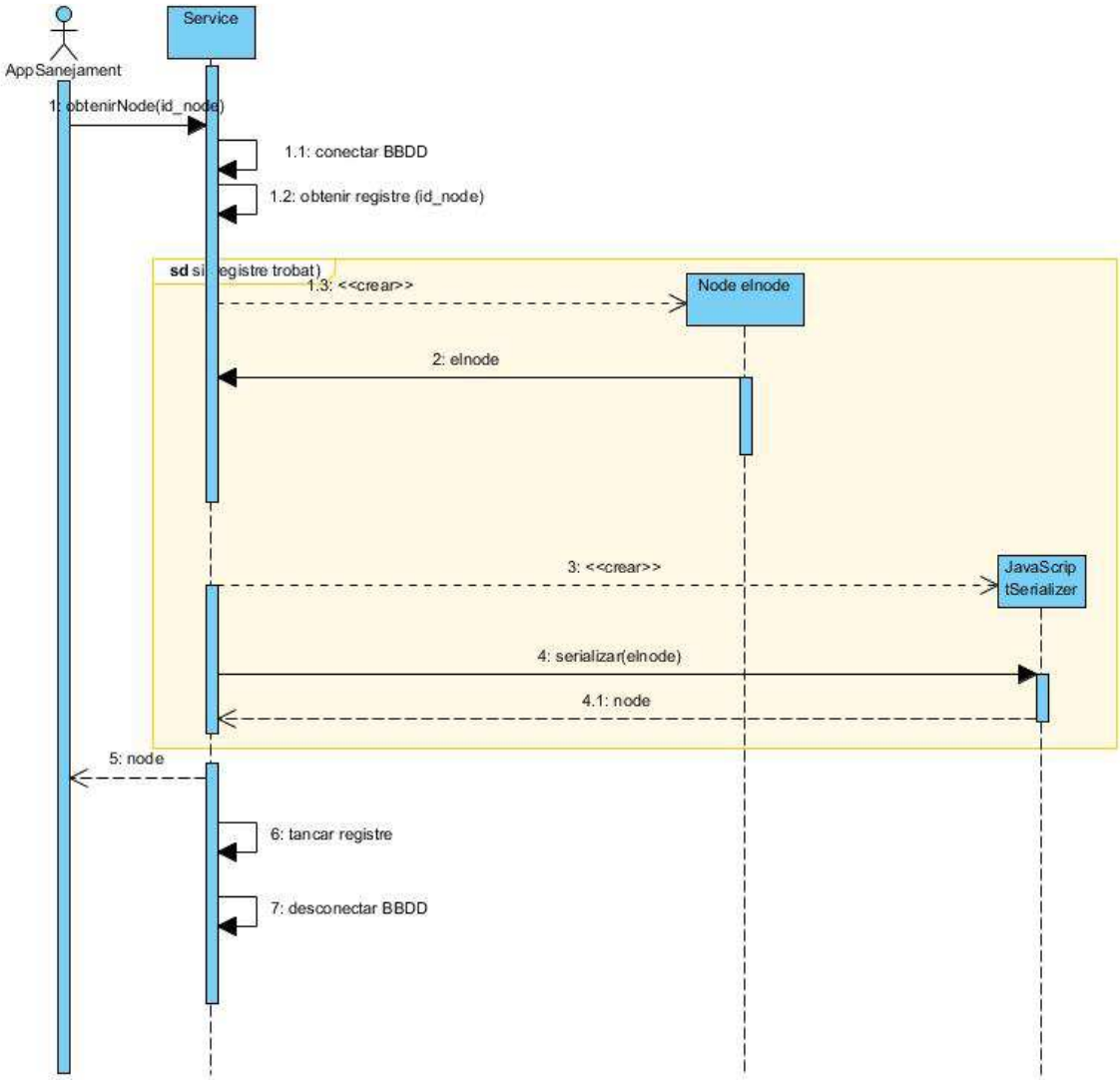


5.1.2 Consultar 1 pou

Quan l'usuari fa clic sobre un pou, es visualitzen les característiques d'aquest pou. Per fer-ho, l'aplicació envia al WebService la petició per obtenir les dades, indicant el codi de pou que volem visualitzar.

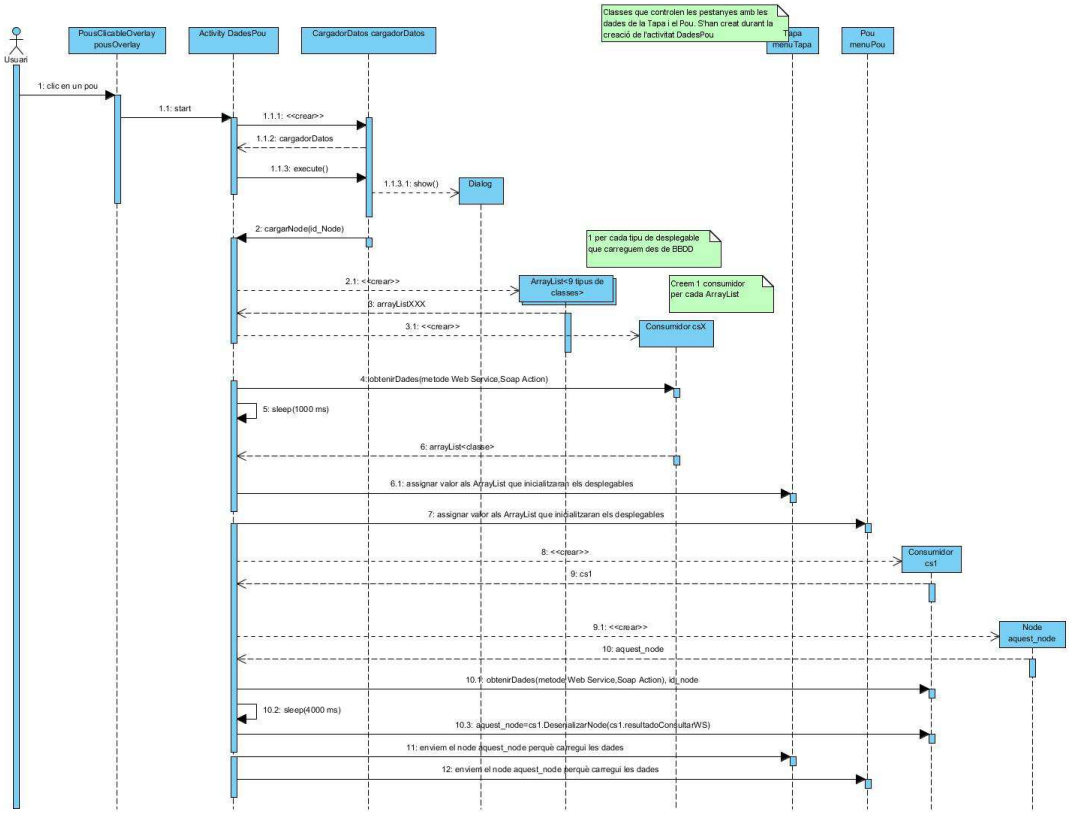
Aquest és el diagrama de seqüència des del Web Service:

5.1.2.1 DS2.1 WebService:Consultar1Pou



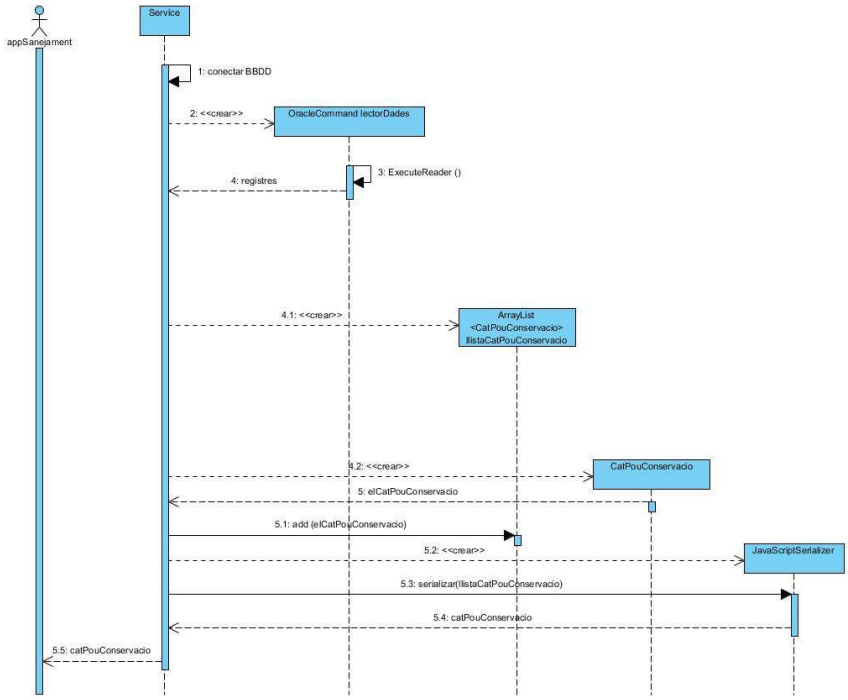
I aquest el diagrama de seqüència de l'aplicació de tablet:

5.1.2.2 DS2.2: AppTablet:Consultar1Pou



Per visualitzar les dades del pou també rebrem les taules de parametrització, que ens serviran per a omplir els desplegables de la fitxa. A continuació podem veure el diagrama de seqüència per obtenir les dades d'1 de les taules de parametrització, des del Web Service:

5.1.2.3 DS2.3: WebService:ObtenirTaulesAuxiliars

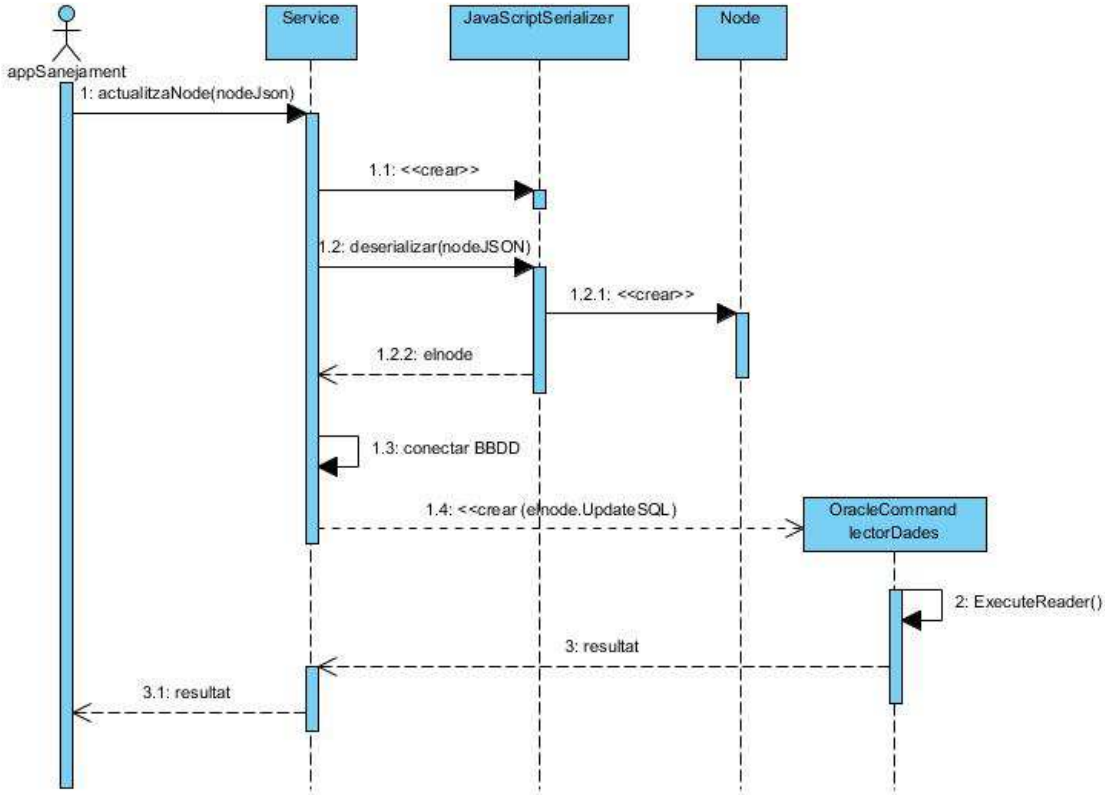


5.1.3 Actualitzar pou

5.1.3.1 DS3.1: WebService:ActualitzarPou

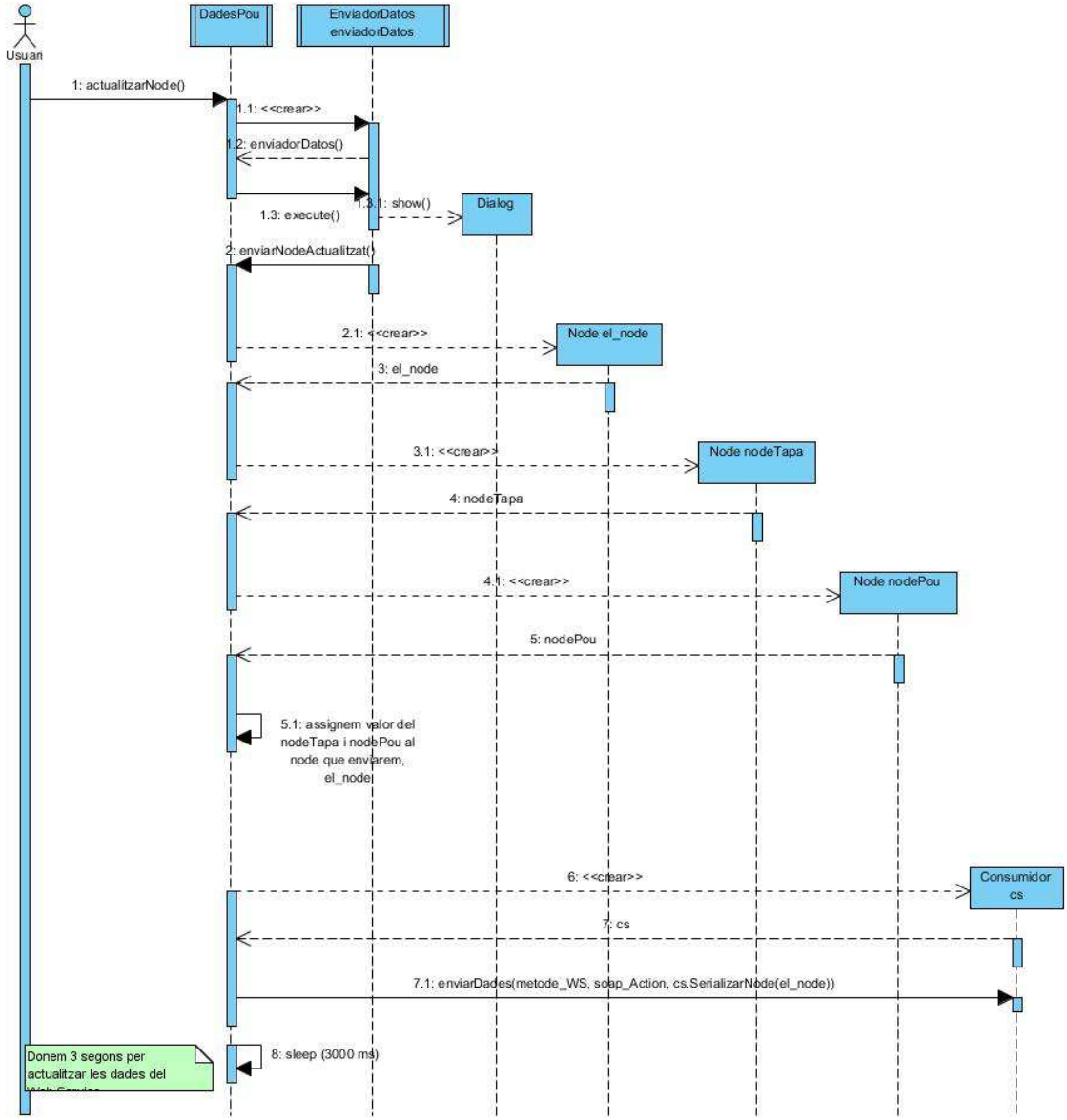
Quan l'usuari modifica i guarda les dades del pou, ha d'enviar-les al servidor perquè les actualitzi. Envia els objectes serialitzats en notació JSON al WebService i rep el resultat de l'actualització també en notació JSON.

A continuació podem veure el diagrama de seqüència del Web Service:



I aquest és el diagrama de seqüència de l'aplicació de tablet:

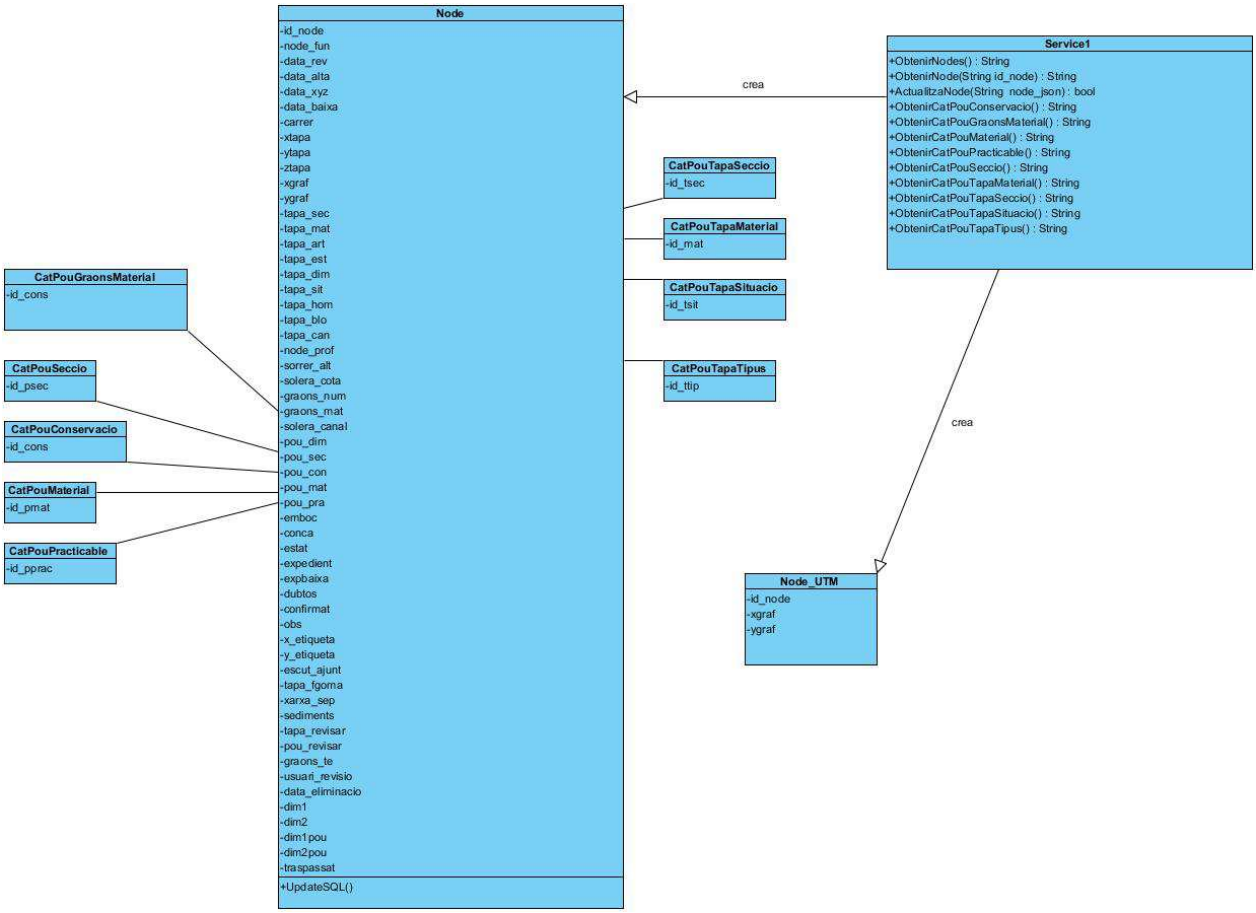
5.1.3.2 DS3.2: WebService:ActualitzarPou



5.2 Diagrama de classes

El Web Service i l'aplicació de tablet tenen en comú diverses classes. Això és necessari perquè s'estan transmetent els objectes serialitzats entre les aplicacions i han de tenir els mateixos atributs.

5.2.1 Diagrama de classes del Web Service:



5.2.1.1 Descripció de classes Web Service

A continuació tenim una petita descripció de cadascuna de les classes del Web Service:

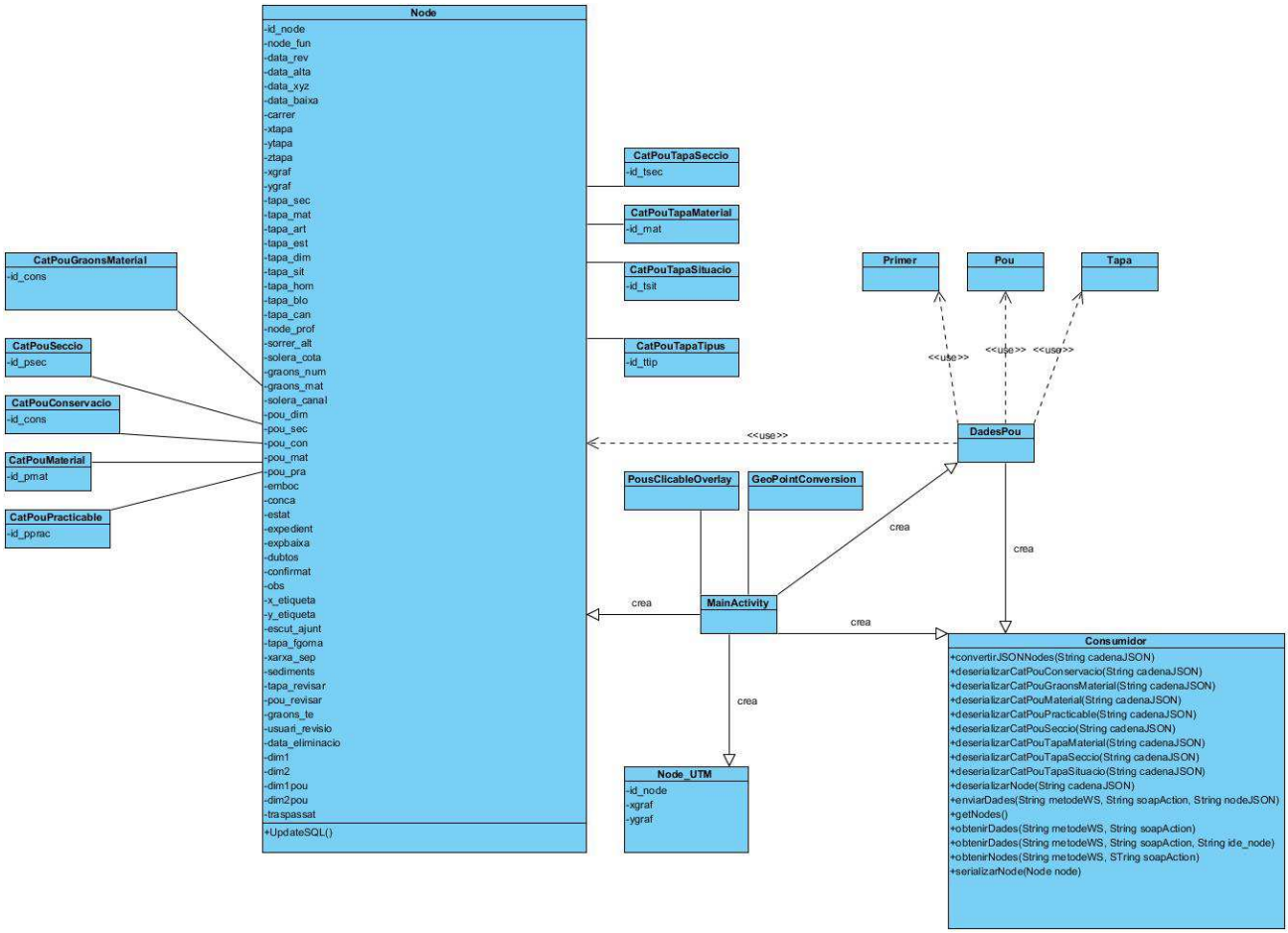
Service1: És la classe principal del Web Service. S'encarrega d'atendre les peticions que se li fan. Té mètodes definits com [WebMethod], que son els que es publiquen al servidor web.

Node: És la classe que conté totes les dades d'un node

Node_UTM: És una versió reduïda del node, que només conté el codi de node i les coordenades UTM. Es fa servir per minimitzar el tràfic entre el Web Service i l'aplicació de sanejament quan demanem tots els pous

CatPouConservacio: Aquesta classe defineix els diferents tipus de conservació d'un pou.
CatPouGraonsMaterial: Aquesta classe defineix els diferents materials dels graons d'un pou.
CatPouMaterial: Aquesta classe defineix els diferents materials d'un pou.
CatPouPracticable: Aquesta classe defineix els diferents nivells de practicable d'un pou.
CatPouSeccio: Aquesta classe defineix les diferents seccions d'un pou.
CatPouTapaMaterial: Aquesta classe defineix els diferents materials de la tapa del pou.
CatPouTapaSeccio: Aquesta classe defineix els diferents tipus de secció de la tapa del pou.
CatPouTapaSituacio: Aquesta classe defineix les diferents situacions de la tapa del pou.
CatPouTapaTipus: Aquesta classe defineix els diferents tipus de tapa del pou.

5.2.2 Diagrama de classes de l'aplicació de tablet:



5.2.2.1 Descripció de classes de l'aplicació de sanejament

A continuació tenim una petita descripció de cadascuna de les classes de l'aplicació de sanejament:

Node: És la classe que conté totes les dades d'un node

Node_UTM: És una versió reduïda del node, que només conté el codi de node i les coordenades UTM. Es fa servir per minimitzar el tràfic entre el Web Service i l'aplicació de sanejament quan demanem tots els pous

CatPouConservacio: Aquesta classe defineix els diferents tipus de conservació d'un pou.
CatPouGraonsMaterial: Aquesta classe defineix els diferents materials dels graons d'un pou.
CatPouMaterial: Aquesta classe defineix els diferents materials d'un pou.
CatPouPracticable: Aquesta classe defineix els diferents nivells de practicable d'un pou.
CatPouSeccio: Aquesta classe defineix les diferents seccions d'un pou.
CatPouTapaMaterial: Aquesta classe defineix els diferents materials de la tapa del pou.
CatPouTapaSeccio: Aquesta classe defineix els diferents tipus de secció de la tapa del pou.
CatPouTapaSituacio: Aquesta classe defineix les diferents situacions de la tapa del pou.
CatPouTapaTipus: Aquesta classe defineix els diferents tipus de tapa del pou.

MainActivity: És l'activitat principal de l'aplicació. Aquí carreguem Google Maps i la capa que conté els pous.

PousClicableOverlay: Aquesta classe és la que es pot superposar com a capa de Google Maps. Sobre ella dibuixem els pous i ens permet fer clic en qualsevol d'ells i controlar quina acció volem fer.

GeoPointConversion: És una classe auxiliar que fem servir per convertir les coordenades UTM en latitud i longitud.

Consumidor: És l'encarregat de comunicar-se amb el Web Service, amb qui intercanviarà les dades dels nodes i les taules de parametrització.

DadesPou: És una Activity (aplicació). S'inicia quan fem clic a un pou per visualitzar les dades. Fa de contenidor per carregar les pestanyes que contenen les dades del Pou.

Primer: Primera pestanya que carreguem a DadesPou. Conté un text explicatiu del funcionament de l'aplicació.

Pou: Segona pestanya que carreguem a DadesPou. Conté les dades del pou.

Tapa: Tercera pestanya que carreguem a DadesPou. Conté les dades de la tapa.

5.3 Decisions tècniques: el perquè de tot plegat

A continuació exposarem els aspectes més destacats del desenvolupament de l'aplicació de tablet i el perquè d'algunes decisions.

5.3.1 Consumidor

La classe consumidor s'ha creat per centralitzar totes les comunicacions de l'aplicació i per un imperatiu en el disseny d'Android. Si intentem accedir a Internet des del fil principal de l'Activity (en aquest cas, MainActivity), Android no ens ho permet i genera una excepció del tipus ***android.os.NetworkOnMainThreadException***. Això és perquè des del fil principal no podem fer aquest tipus d'operació. Per solucionar-ho, la classe Consumidor crea un nou fil d'execució (Thread) i des d'aquest fil es connecta amb el Web Service.

5.3.2 CargadorDatos

La classe CargadorDatos es fa servir per mostrar la finestra de càrrega quan fem operacions que ens porten uns segons (com per exemple, rebre dades del Web Service). Aquesta classe és una extensió de la classe AsynkTask. Aquesta classe s'executa en 4 fases:

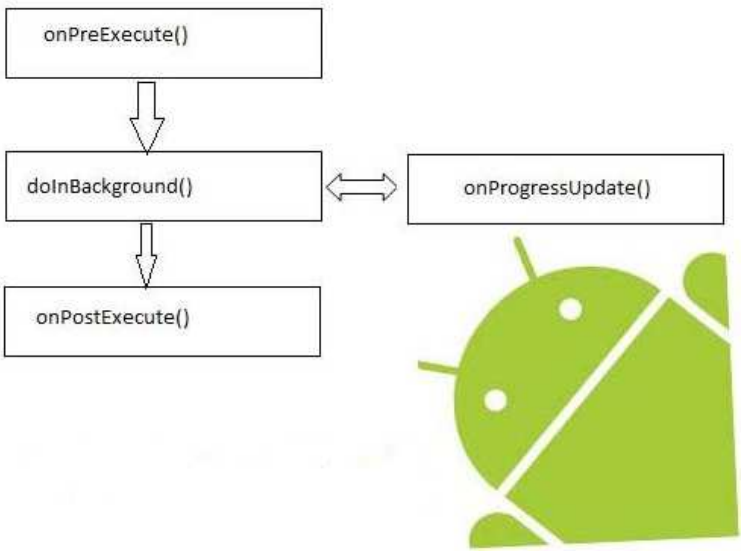
OnPreExecute() : Abans de l'execució, podem personalitzar algunes característiques, com el missatge que mostrarem.

doInBackground(Params...): aquesta és la fase principal, on executarem les feines que volem dur a terme quan mostrem la finestra de càrrega.

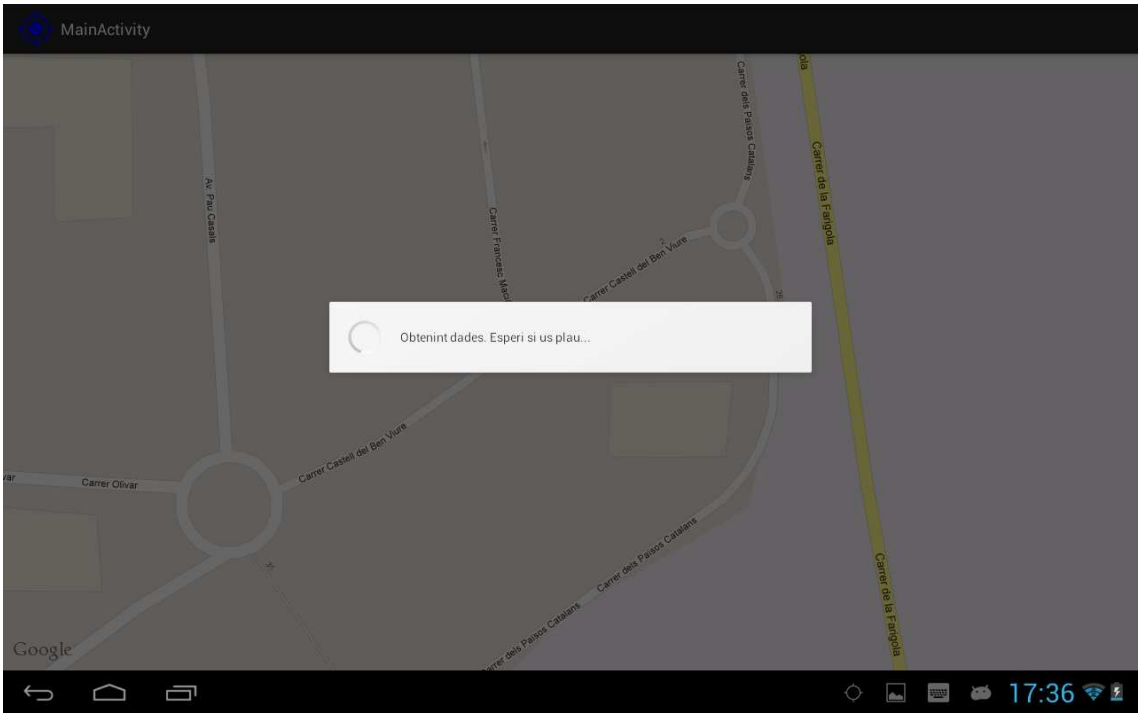
onProgressUpdate(Progress...): aquest mètode es crida durant l'execució de doInBackground() i ens permet actualitzar la informació de la finestra. Normalment es fa servir per actualitzar una barra de progrés.

onPostExecute(Result): s'executa quan acaba l'execució de doInBackground(). En el nostre cas, en aquest mètode és on tanquem la finestra de càrrega.

Aquest és el cicle de vida de la classe AsynkTask

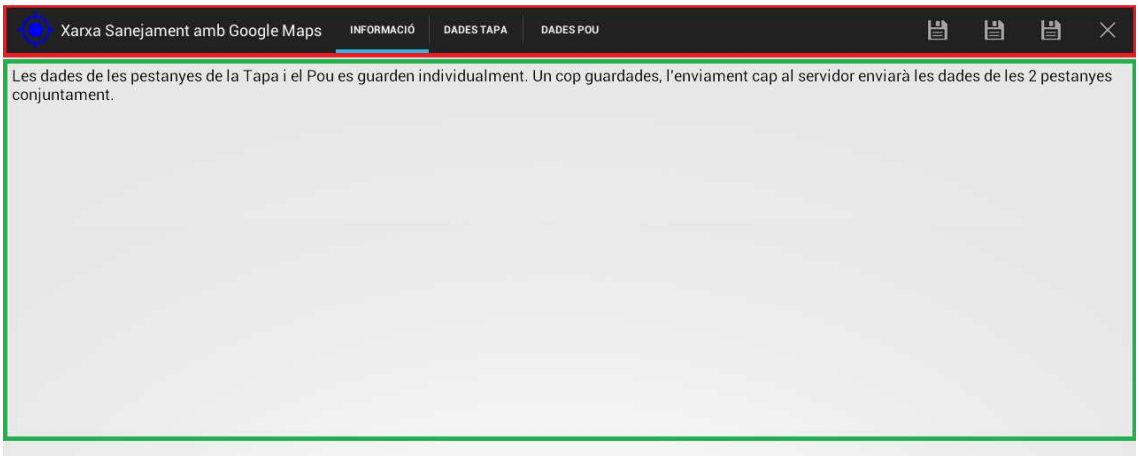


Aquest és l'aspecte de la finestra de càrrega de dades:



5.3.3 **Pestanyes: les carreguem a DadesPou**

La classe DadesPou és la que ens mostra les dades del pou. Quan l'executem, crea els contenidor necessaris per carregar les pestanyes. A la següent imatge veurem els diferents contenidors que necessitem:



ActionBar (zona vermella): Aquí ubicarem les diferents pestanyes (Tab). En el nostre cas, les pestanyes son :

- Informació
- Dades Tapa
- Dades Pou

Cadascuna d'aquestes pestanyes té associat un "escoltador"(TabListener) que, quan li fem clic, canvia el contingut del contenidor (zona verda) pel que volem. En el nostre cas, el més destacable és que quan fem clic a DadesTapa mostrarem la classe Tapa i quan fem clic a DadesPou, mostrarem la classe Pou.

A l'ActionBar també podem ubicar botons que facin altres accions. A la part dreta de la barra s'han col·locat els botons per :

- Guardar les dades del pou
- Guardar les dades de la tapa
- Enviar el node actualitzat
- Sortir de l'aplicació

5.3.4 Disseny de les pantalles de Dades Pou i Dades Tapa

A Android, les diferents finestres d'una aplicació s'anomenen layout. Es poden dissenyar amb un editor WYSIWYG (What You See Is What You Get) o directament amb codi XML. Per mostrar les dades del pou i de la tapa, hem fet servir el control GridLayout per organitzar les finestres. GridLayout és una graella, que facilita la estructuració dels controls i garanteix que el posicionament és fix independentment de la mida de la pantalla del dispositiu.

En els 2 casos s'han definit 9 files i 9 columnes per organitzar els diferents controls. Aquest és l'esquema de les dades de la tapa:

UBICACIÓ DELS CONTROLS DE LA PESTANYA "TAPA POU"

		Column								
		0	1	2	3	4	5	6	7	8
Row	0	CotaLbl	<-- 36 dp -->	CotaEdit	<-- 36 dp -->	EscutCb	<-- 36 dp -->	XTapaLbl	<-- 36 dp -->	XTapaEdit
	1	SituacioLbl		SituacioSpin		FaltaCb		YTapaLbl		YTapaEdit
	2	MaterialLbl		MaterialSpin		RevisarCb		TipusLbl		TipusSpin
	3	SeccioLbl		SeccioSpin		ArticuladaCb				
	4	DiametreLbl		DiametreEdit		HomologCb				
	5	AmpleLbl		AmpleEdit		BloqueigCb				
	6					CandauCb				
	7									
	8									

I aquest l'esquema de les dades del pou:

UBICACIÓ DELS CONTROLS DE LA PESTANYA "DADES POU"

		Column								
		0	1	2	3	4	5	6	7	8
Row	0	ProfunditatLbl	<-- 36 dp -->	ProfEdit	<-- 36 dp -->	XarxaCb	<-- 36 dp -->	GraonsMatLbl	<-- 36 dp -->	GraonsSpin
	1	AlçadaLbl		AlçadaEdit		PouCb		GraonsLbl		GraonsEdit
	2	MaterialLbl		MatSpin		SedimentsCb		ConservLbl		ConservSpin
	3	SeccioLbl		SeccioSpin		GraonsCb		PractLbl		PractSpin
	4	DiametreLbl		DiamSpin		DubtosCb		EstatLbl		EstatSpin
	5	AmpleLbl		AmpleEdit		ConfirmatCb				
	6	CotaLbl		CotaEdit		EmbocCb				
	7									
	8									

Els principals controls que hem fet servir son:

- TextView: és una etiqueta de text. S'ha fet servir pels títols dels camps
- Checkbox: és una casella de verificació on podem fer clic per activar/desactivar
- Spinner: és un desplegable on podem escollir diverses opcions. S'ha fet servir per carregar les opcions d'alguns atributs que tenen els valors definits en alguna taula.
- TextField: és un quadre de text per escriure-hi. S'ha fet servir pels camps on l'usuari pot escriure lliurement algun valor.

Cadascun dels controls havien de tenir informats 2 camps per posicionar-los bé:

- Layout_column: A quina columna es col·loca
- Layout_row: A quina fila es col·loca

Per deixar espais horitzontals entre els controls el que he fet és col·locar uns controls de tipus Space dins de la casella corresponent. Aquest és el codi XML:

```
<Space
android:layout_column="1"
android:layout_width="36dp"
/>
```

Només té 2 atributs:

Layout_column: A quin nº de columna col·loquem l'espai.
Layout_width: Amplada de l'espai

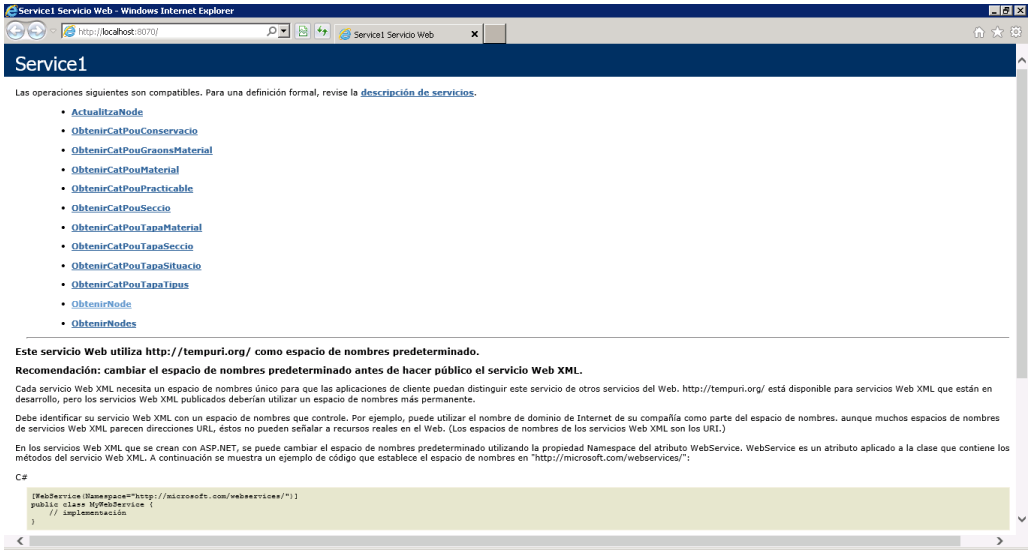
6 Resultats

Per validar els resultats i testejar els diferents mòduls s'han realitzat dividit les proves en 2 fases:

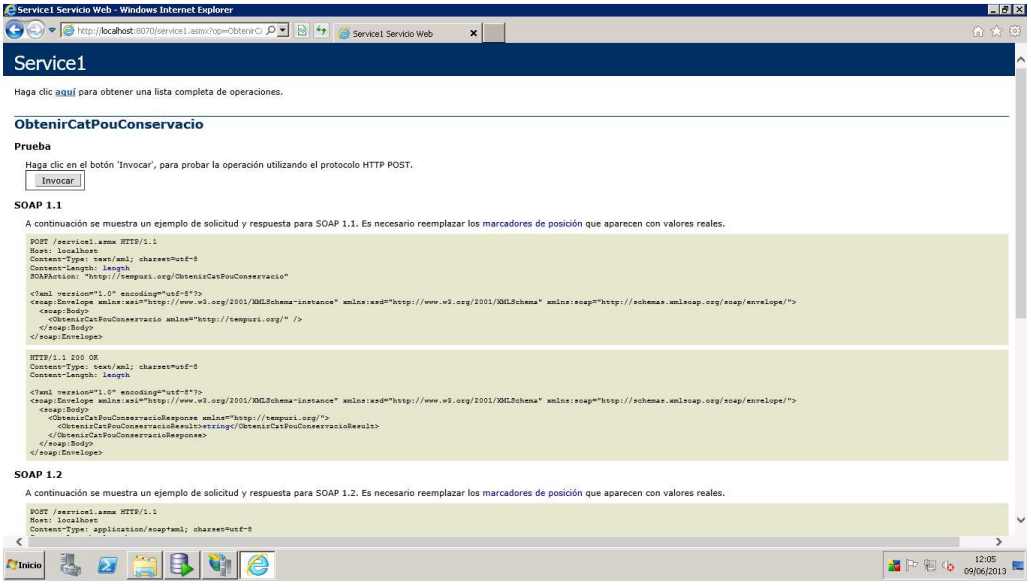
1- Proves Web Service
2- Proves Aplicació Sanejament

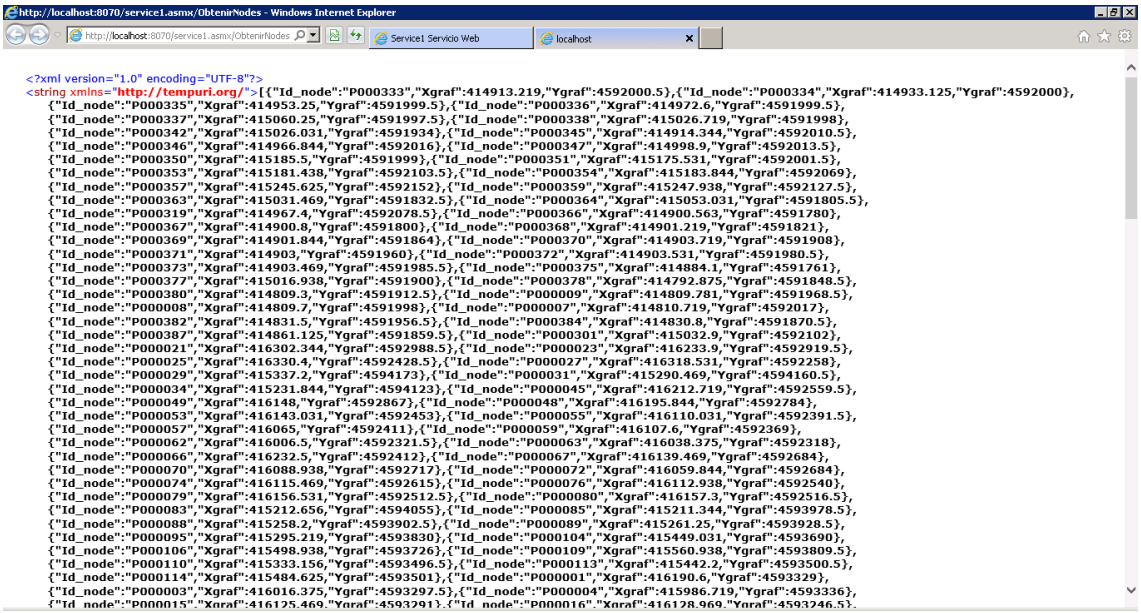
6.1 Proves Web Service

Durant aquesta fase, s'ha fet servir l'entorn de pre-producció que proporciona Visual Studio. Aquest entorn ens mostra la pàgina web del Web Service i podem testejar els diferents mètodes públics.



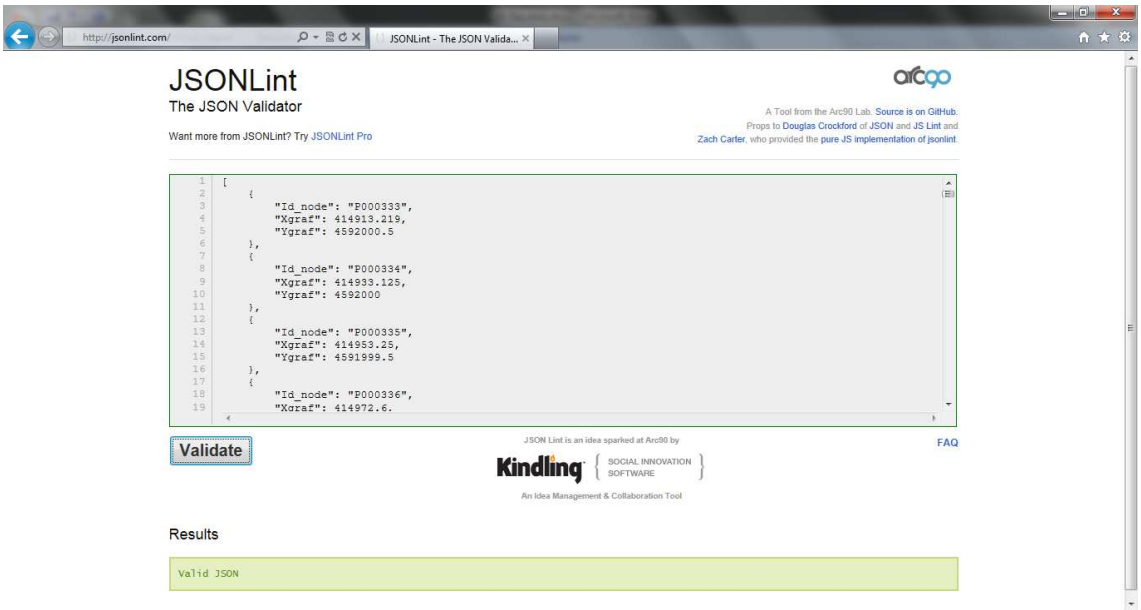
Si fem clic a qualsevol dels mètodes, podem veure les seves característiques i si pitgem el botó "Invocar", veurem el resultat





En aquest cas el mètode ens torna una cadena JSON amb els nodes. Per validar que està ben formada s'ha fet servir un validador de JSON des de l'adreça web http://jsonlint.com/ .

Com podem veure a la imatge següent, el validador ens confirma que la cadena JSON és correcta:



6.2 Proves Aplicació Sanejament

Per validar l'aplicació de sanejament el que he fet és separar el desenvolupament en 2 mòduls:

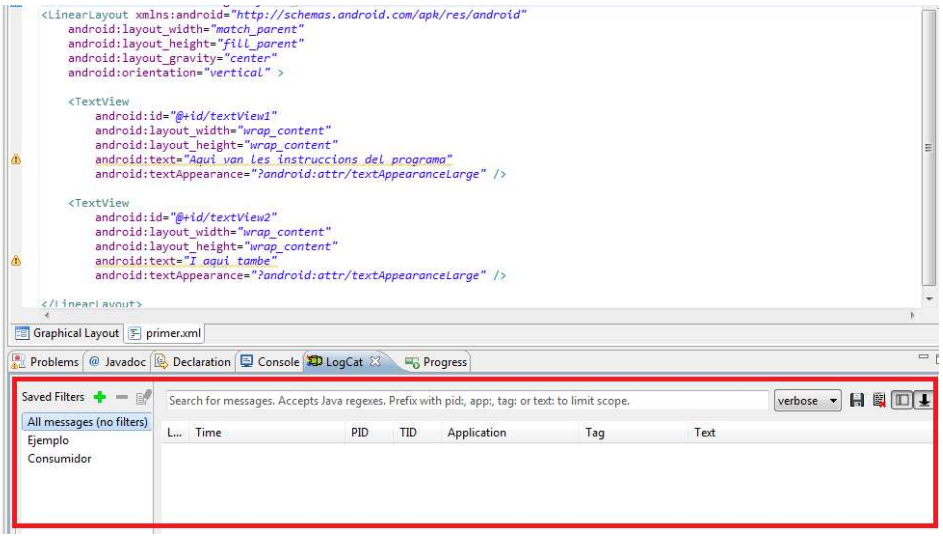
- Google Maps
- Fitxa de consulta/modificació de dades

Aquests 2 mòduls s'han desenvolupat per separat i s'han unit en la fase final del projecte.

El mòdul de Google Maps només s'encarregava de consulta el Web Service i rebre la informació dels pous. Un cop fet, dibuixava la xarxa sobre el mapa.

El mòdul de fitxa de consulta/modificació de dades demanava al Web Service les taules de parametrització i la informació d'un node.

Per controlar els possibles errors i fer una traça del procés, s'ha fet servir la eina de LogCat que incorpora ADT per Eclipse.



Des de l'aplicació es registrava informació durant diverses fases (al rebre les taules de parametrització o la informació del node, per exemple) i es podia visualitzar aquí.

Aquesta eina es complementava amb una execució en mode de "Debug", per poder aturar el programa en els punts crítics.

Per enregistrar informació des de l'aplicació es fa servir la següent instrucció:

```
Log.i("Categoria","Text");
```

El primer camp, Categoria, ens permetrà aplicar filtres sobre LogCat on només veiem uns determinats events. Per exemple, podem voler filtrar només els events que generi la classe Consumidor, per verificar les comunicacions.

El segon camp, Text, és la descripció de l'event que estem enregistrant.

6.3 Errors detectats

Els errors detectats es poden dividir en 2 grups:

- Comunicacions
- Validació

6.3.1 Errors de comunicacions

Si la tablet no té accés a Internet o no pot connectar amb el Web Service, l'aplicació donava errors greus i es penjava. S'ha solucionat afegint un control en la recepció de les dades del Web Service. Si aquest control detecta un error, la part de Google Maps no dóna error però no carrega cap pou al mapa. I si aquest error es produeix durant la càrrega del pou a la fitxa, l'aplicació mostra un text informatiu i s'atura, impedint a l'usuari que consulti les pestanyes de dades de la tapa i el pou

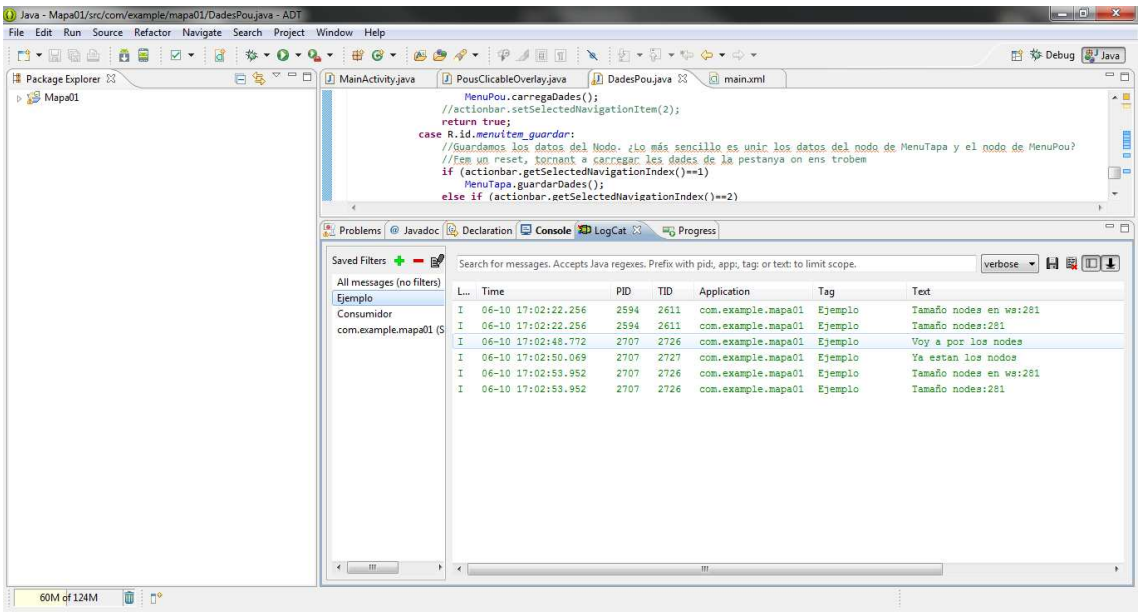
6.3.2 Errors de validació

Ha quedat pendent de resoldre el problema de les validacions de les dades introduïdes al formulari. Per exemple, es podria introduir una cadena de text en un camp numèric. Si això succeeix, l'aplicació de tablet dóna un error i es tanca. Aquest punt es solucionarà abans de que l'aplicació entri en fase de producció a Aigües de Castellbisbal.

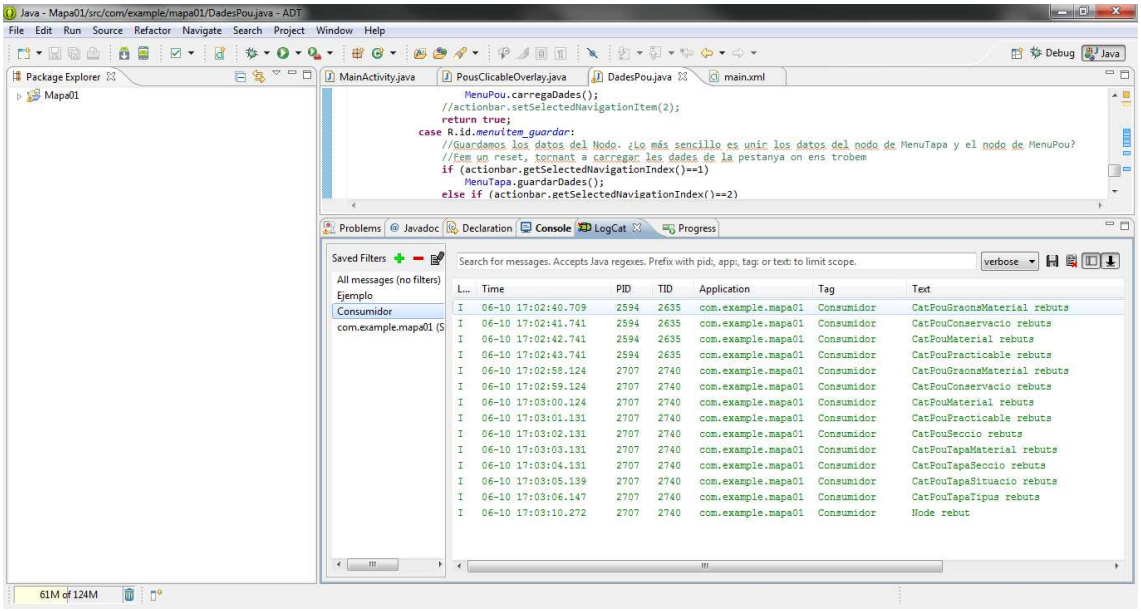
6.4 Prova completa

S'han realitzat proves completes del circuït i s'ha anat validant pas a pas. Aquest és el resultat:

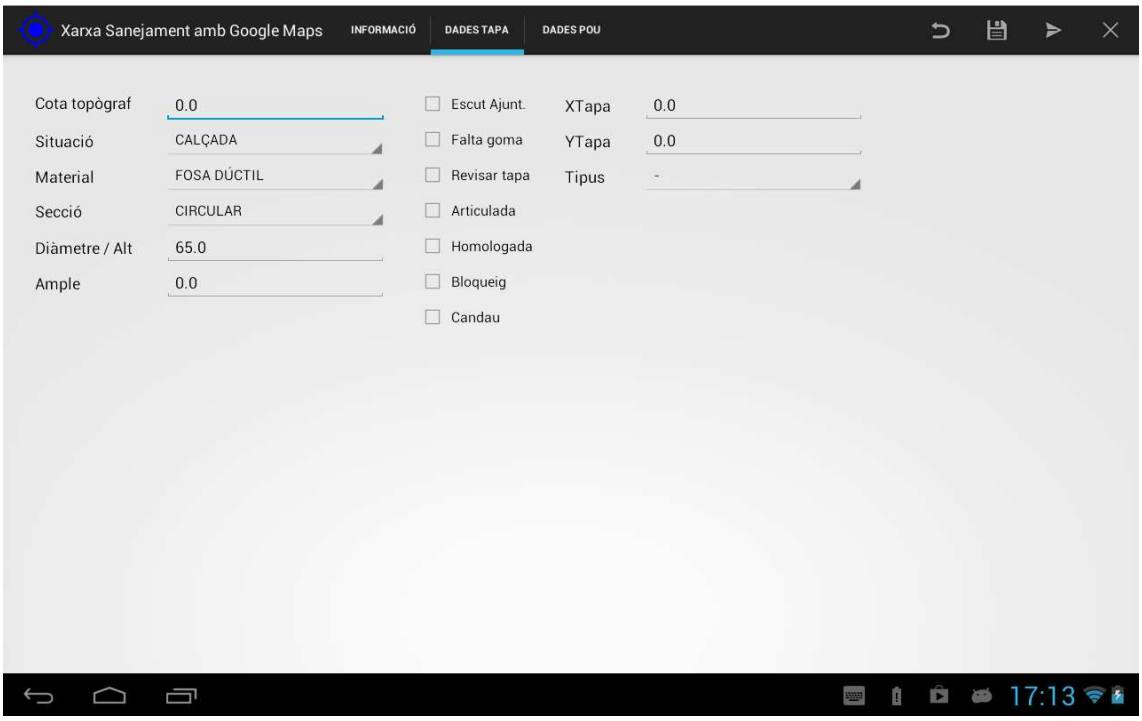
Primer s'han carregat els pous. Al LogCat de l'ADT per Eclipse podem observar que s'han rebut 281 nodes.



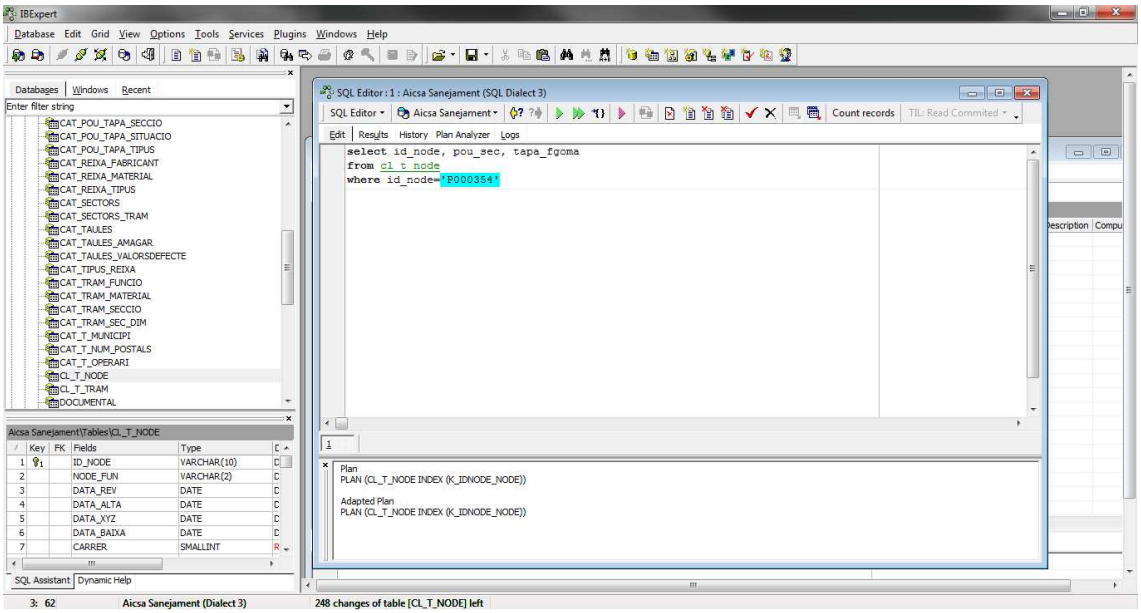
A continuació hem fet clic en un node. El seu ID és el P000354. Al LogCat podem observar que s'han rebut les taules de parametrització i les dades d'aquest node:

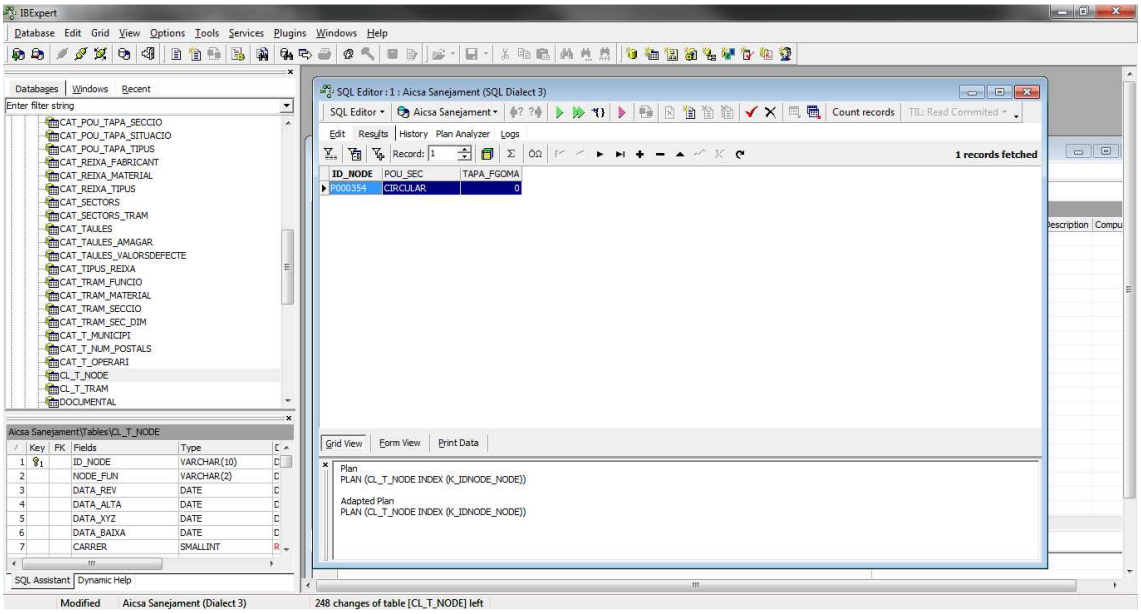


A l'aplicació d'Android visualitzem aquestes dades a la pestanya de dades de la tapa:

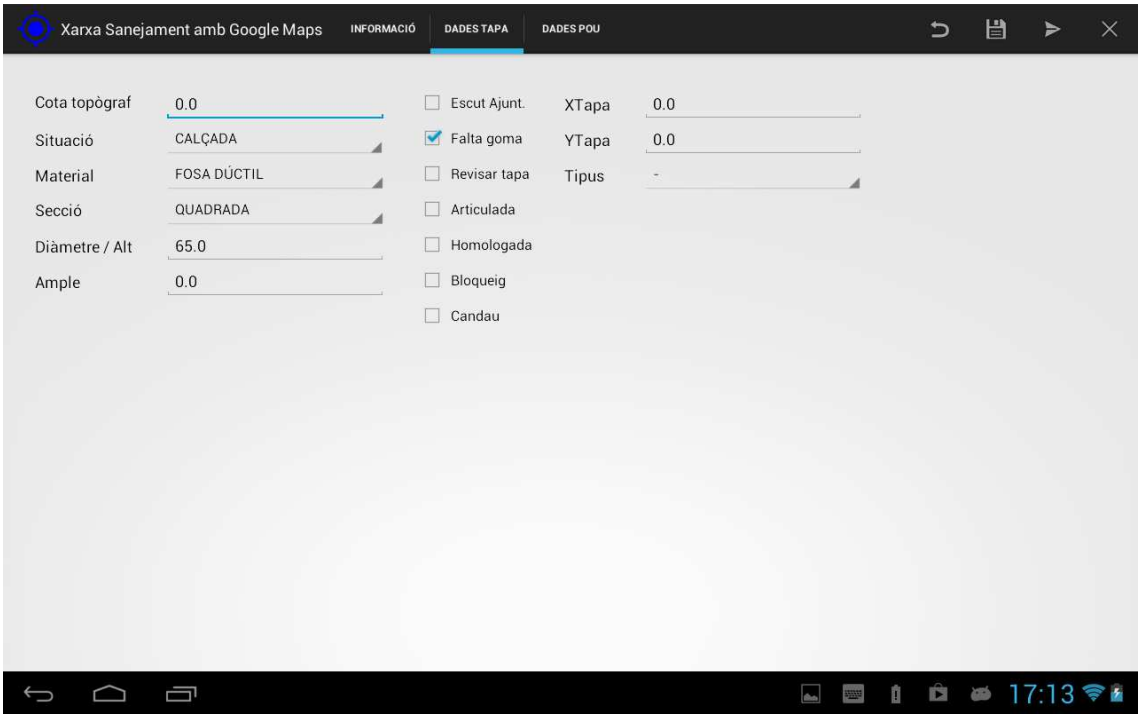


Si fem una consulta a la base de dades dels camps Seccio i Tapa_Fgoma, aquests son els resultats:

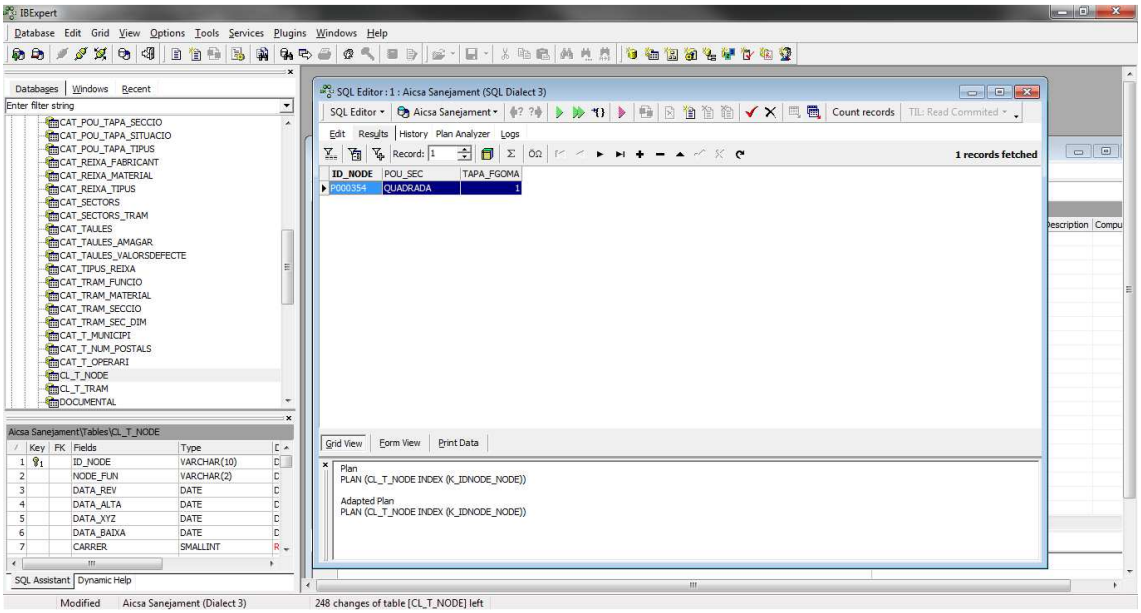




Ara, modifiquem a l'aplicació de la tablet aquests 2 camps i enviem el node actualitzat al servidor.



Si tornem a fer la consulta, observem que els valors han canviat:



7 Referències

7.1 Pàgines web

Pàgina oficial desenvolupament Android

http://developer.android.com/reference/packages.html

Article amb arquitectura d'Android

http://androideity.com/2011/07/04/arquitectura-de-android/

Pàgina oficial Microsoft .NET Framework

http://www.microsoft.com/net

Recursos c# Microsoft

http://msdn.microsoft.com/es-es/vstudio/hh341490.aspx

Exemples de programació amb Google Maps

http://www.androidcompetencycenter.com/2009/01/android-location-api

http://www.sgoliver.net/blog/?p=2004

Coordenades i zones UTM

http://www.elgps.com/documentos/utm/coordenadas_utm.html

Classe GeoPointConversion (que s'ha fet servir al projecte)

http://www.java2s.com/Open-Source/Android/Map/mapdroid/org/mapdroid/utils/GeoPointConversion.java.htm

Exemple de consumidor de Web Services des d'Android

http://androideity.com/2011/11/16/consumiendo-web-service-soap-json-con-android-ii/

Explicació de l'error android.os.NetworkOnMainThreadException

http://android-developers.blogspot.in/2009/05/painless-threading.html

Wikipedia: format JSON

http://es.wikipedia.org/wiki/JSON

Validador JSON

http://jsonlint.com/

7.2 Llibres

"Así es Microsoft.NET" - Ed. McGraw-Hill - ISBN 84-481-3251-3

"Profesional ASP.NET 2.0" – Ed. Anaya – ISBN 84-415-2100-X

"Desarrollo de aplicaciones para Android (ed. 2013) – Ed. Anaya – ISBN 9788441533257

8 Conclusions

Podem concloure que l'aplicació s'ha finalitzat exitosament. Les proves han estat satisfactòries i la corba d'aprenentatge per a l'usuari ha estat poc pronunciada. Inclús perfils d'usuaris poc acostumats a treballar amb les noves tecnologies, com els operaris de taller, han estat capaços d'aprendre a fer servir l'aplicació molt fàcilment. Les expectatives de l'aplicació són altes i Aigües de Castellbisbal espera que els treballs per inventariar la xarxa es tornaran més eficients quan l'aplicació estigui en producció.

Per desenvolupar aquest projecte he necessitat adquirir coneixements de programació per Android i C#. D'altres, com la gestió i consulta amb bases de dades Oracle o Firebird i l'administració de servidors web, ja els tenia assolits. El personal d'Aigües de Castellbisbal m'ha tramés la informació necessària per entendre els sistemes GIS i estructurar l'aplicació de la manera que els fos més senzill l'ús. També m'han fet entendre com es gestiona una xarxa de sanejament i com es realitza un inventari detallat dels elements.

El desenvolupament d'aplicacions amb Android pot ser senzill o complicat, depenent de l'abast d'un projecte, però sens cap mena de dubte disposem d'un nombre enorme de recursos a Internet per documentar-nos i adquirir formació en qualsevol aspecte del desenvolupament. Hi ha moltes comunitats de desenvolupadors que comparteixen els seus coneixements i això fa que el llenguatge sigui molt viu.

Quant al llenguatge C# i .NET, Microsoft va estar a punt de desestimar el Framework en versions anteriors, però va saber resoldre alguns problemes i avui dia és un entorn consolidat dins de la programació. El punt feble és que es necessita fer una inversió en llicències per servidors que d'altres llenguatges i entorns proporcionen gratuïtament.

Google Maps ha revolucionat el món i avui en dia és un recurs importantíssim en camps tant diversos com la mobilitat, la indústria i l'oci. Hi ha infinites aplicacions que fan servir els seus mapes i les seves eines.

Voldria destacar l'assignatura de Programació per a la Xarxa com la que més ha influït en l'arribada a bon terme del projecte, perquè és on vam aprofundir més en els fils (Threads) i la programació client/servidor.

El projecte és millorable i durant els propers mesos s'afegiran elements al mapa com els tubs de la xarxa i altres elements que no son pous (embornals, desguàs, etc...) . També s'activarà la localització de l'usuari, que s'ha desactivat durant el desenvolupament perquè no es feia a Castellbisbal i complicava les proves.

Annex I : Contingut del CD

00 – Memòria: Aquesta carpeta conté la memòria en format PDF

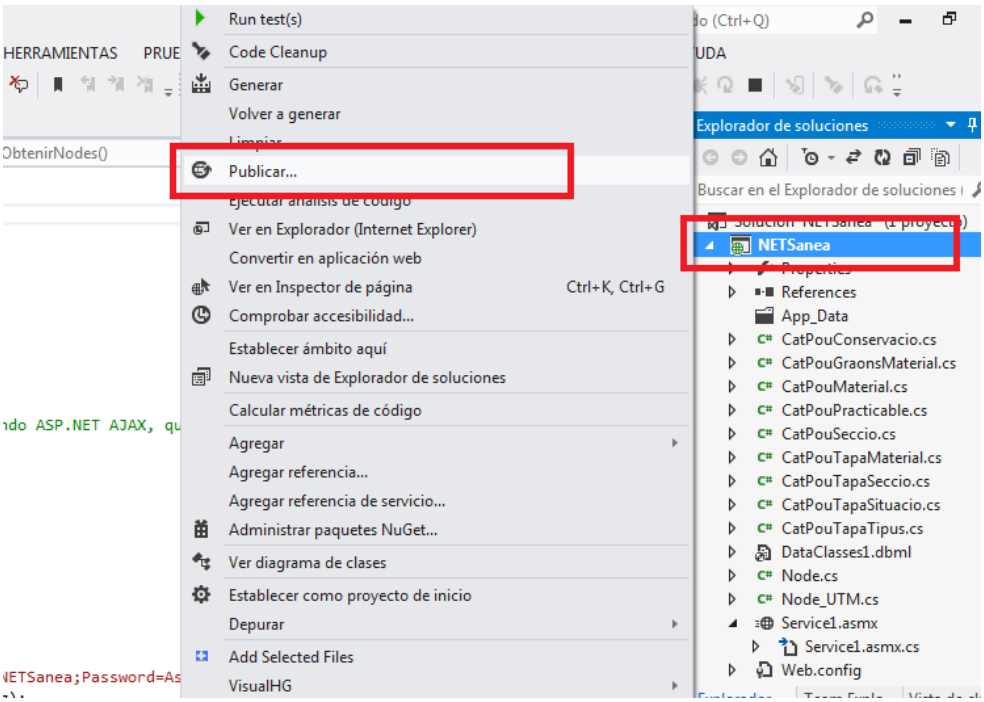
01 – Web Service : Aquesta carpeta conté el codi font del Web Service

02 – Android: Aquesta carpeta conté el codi font de l'aplicació d'Android

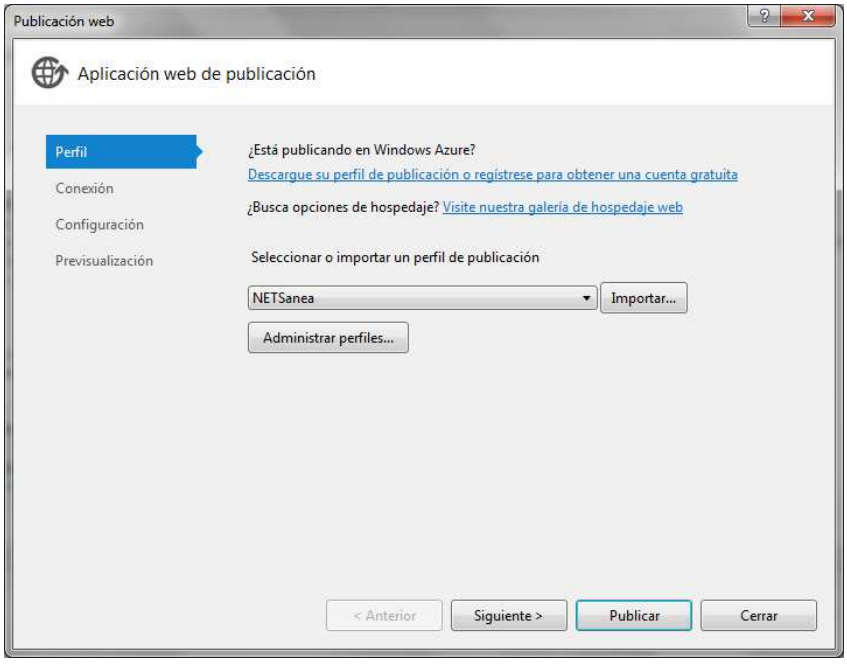
03 – Llibreries Android: Aquesta carpeta conté llibreries necessàries per compilar l'aplicació d'Android.

Anex II: Publicació del Web Service

Per publicar el Web Service i poder-lo instal·lar a un servidor web, hem de fer clic amb el botó dret sobre el projecte NETSanea i escollir la opció "Publicar…"

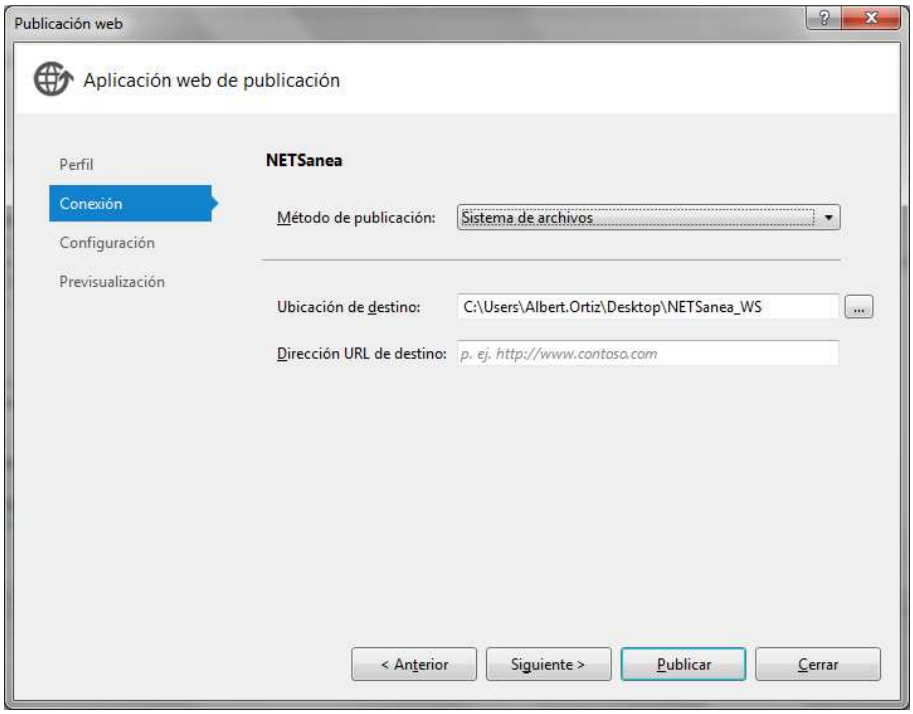


A la finestra següent podrem configurar el perfil de publicació. Aquest perfil ens servirà per guardar les opcions per les següents publicacions de l'aplicació. A la primera finestra podrem triar el nostre perfil o crear-ne un de nou. Només haurem d'indicar el nom del perfil.

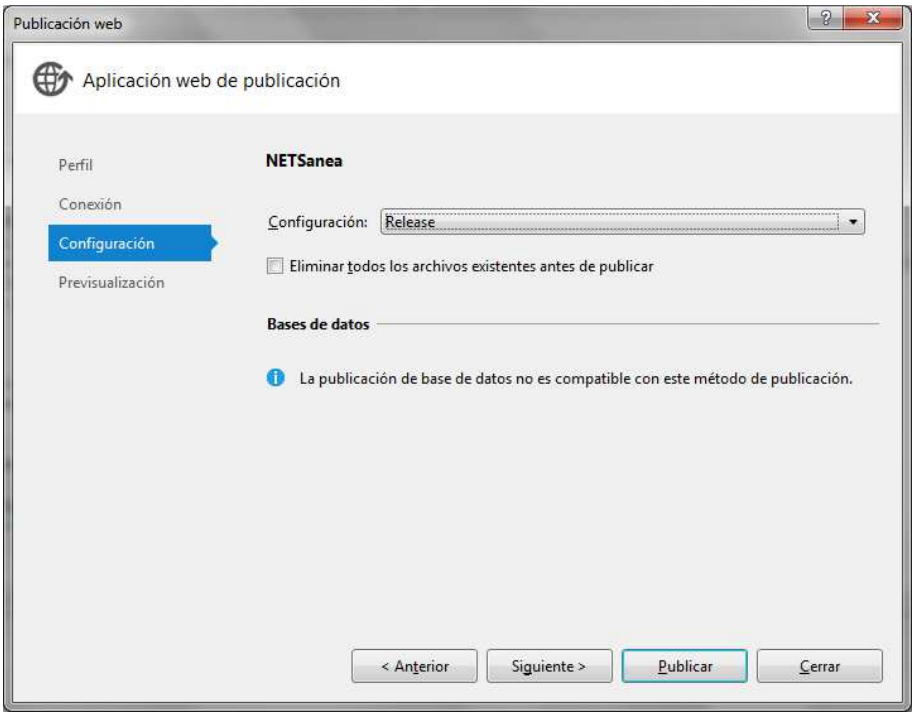


A la segona finestra, “Conexión”, definirem el mètode de publicació. En el meu cas, he triat “Sistema de archivos”. L’assistent de publicació guardarà el lloc web en una carpeta.

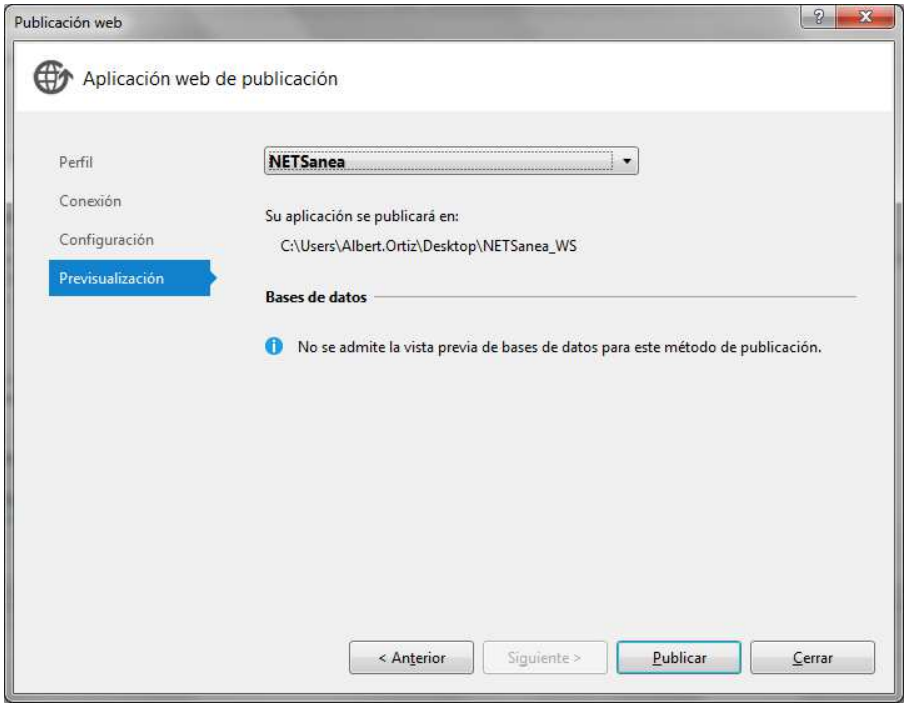
A “Ubicación de destino” indiquem a quina carpeta volem guardar els arxius i fem clic a “Siguiente”.



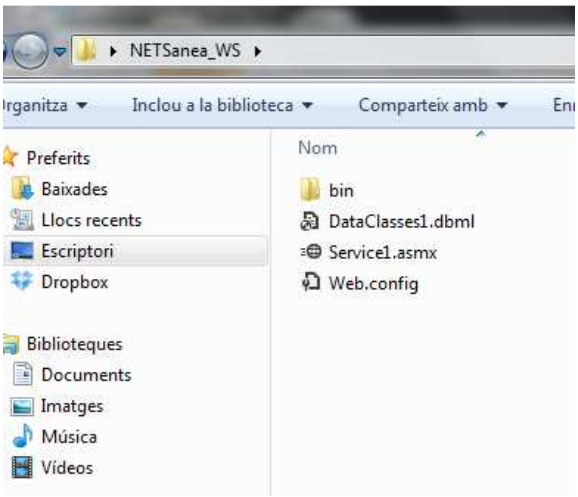
A la finestra següent, “Configuración”, definim si la versió és de Debug o Release.



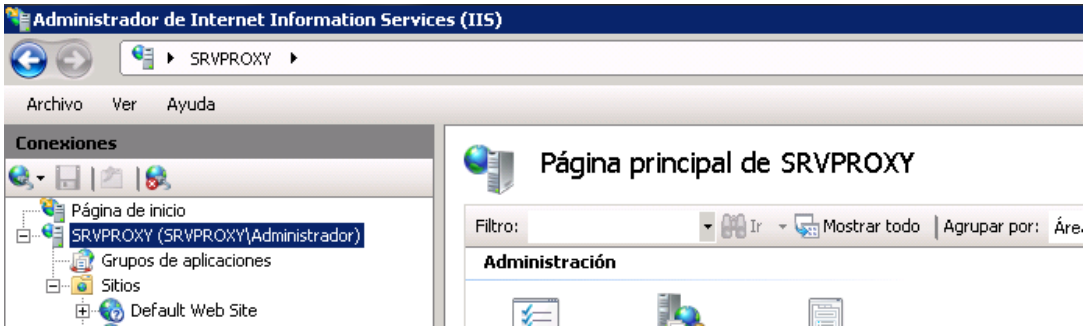
A la darrera finestra, “Previsualización”, podem veure un resum de les opcions que hem triat.



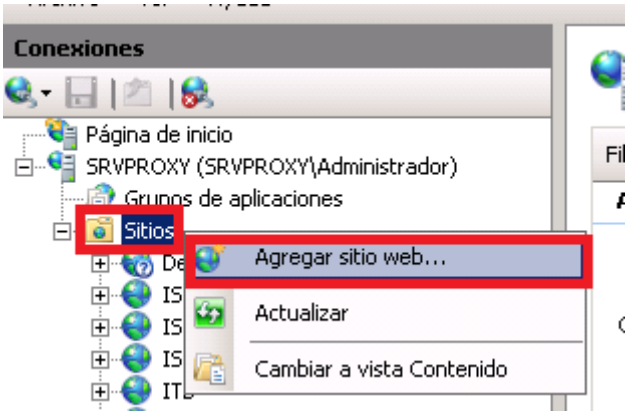
Ara fem clic a “Publicar” i, si anem a la carpeta que hem triat, tenim els arxius necessaris per instal·lar el Web Service en un servidor web.



Ara ens queda instal·lar l’aplicació al servidor web. Anem a la consola d’administració d’Internet Information Services (IIS).

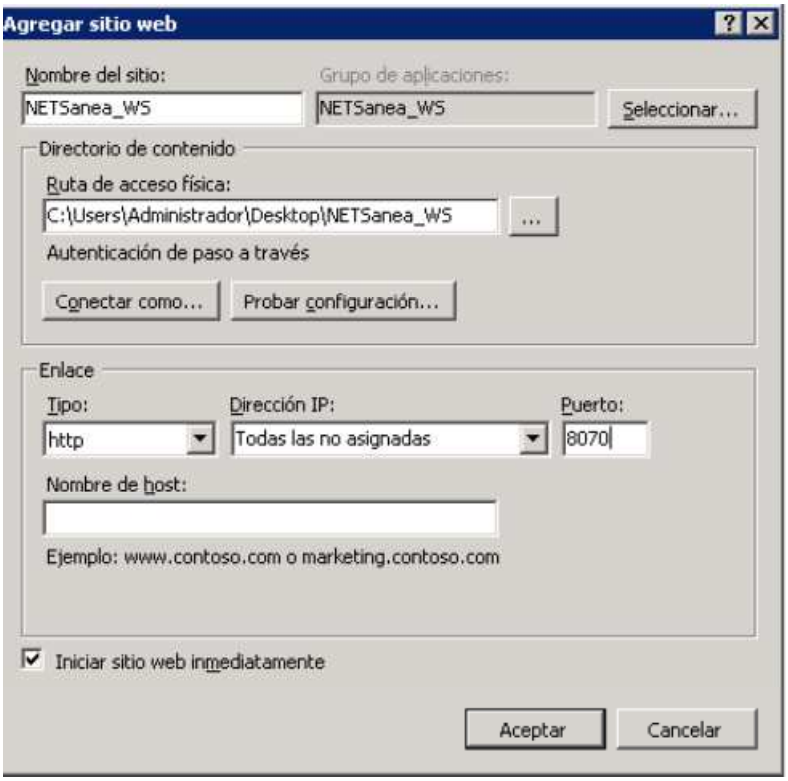


Per instal·lar l'aplicació, fem clic amb el botó dret sobre "Sitios" i escollim "Agregar sitio web"

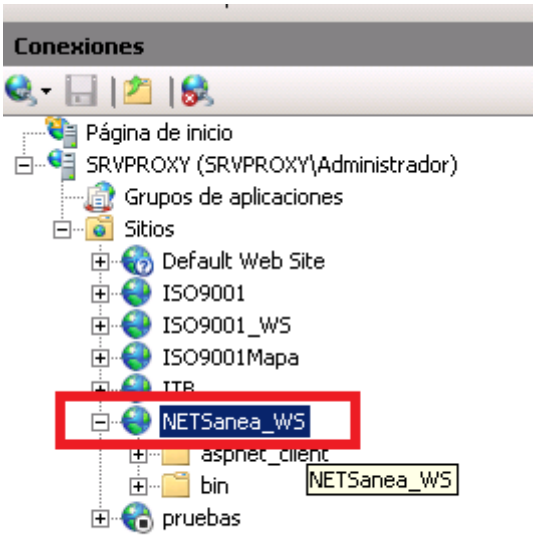


A la finestra següent hem de configurar les opcions de la web. Les més importants son:

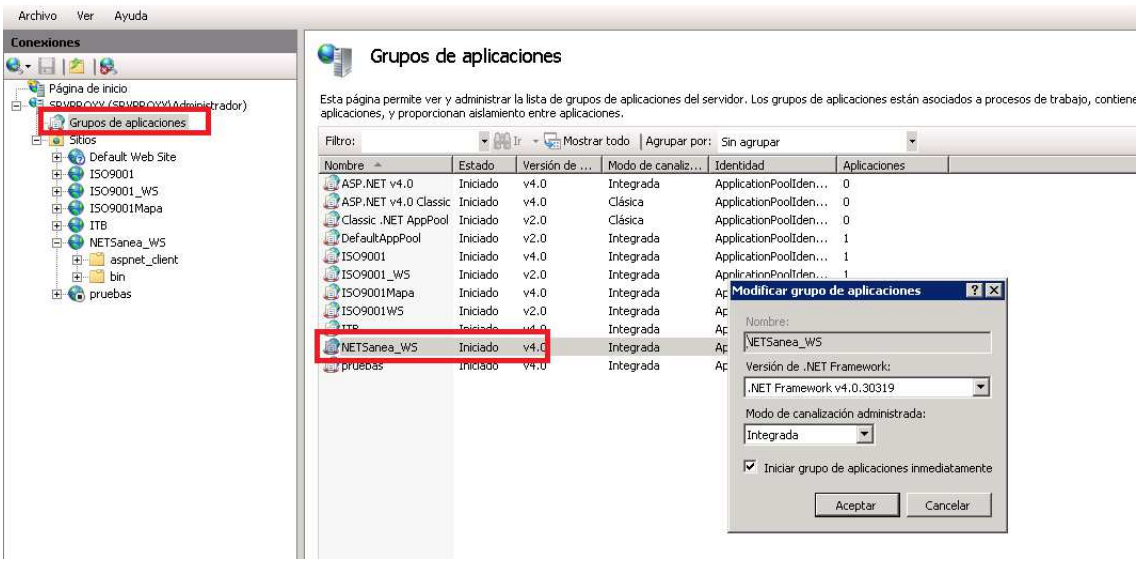
- Nombre del sitio: nom que tindrà dins de IIS, i amb el que identificarem la web
- Ruta de acceso física: La carpeta on es troba l'aplicació
- Grupo de aplicaciones: Nom del grup d'aplicacions de la web. Per defecte, IIS crea un nou grup. Quan acabem de configurar la web, verificarem la configuració de l'aplicació.
- Puerto: Port que farà servir la web. En el nostre cas, la hem configurat amb el port 8070
- Nombre de host: URL de la web. Si no posem cap, totes les peticions que arribin pel port 8070 seran tractades per aquesta web.



Després de pitjar el botó “Aceptar”, ja tenim la web instalada

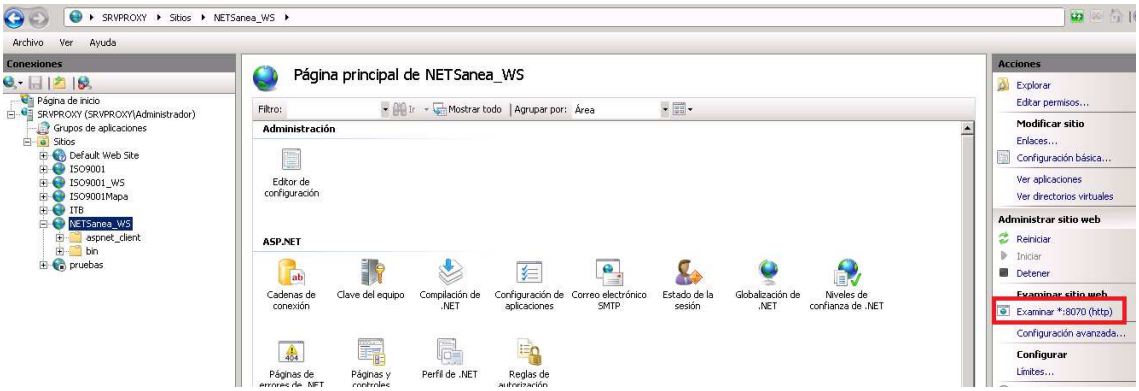


Ara només ens queda revisar les opcions de l’aplicació NETSanea_WS. Per fer-ho, anem a “Grupos de aplicaciones” i fem doble clic sobre NETSanea_WS

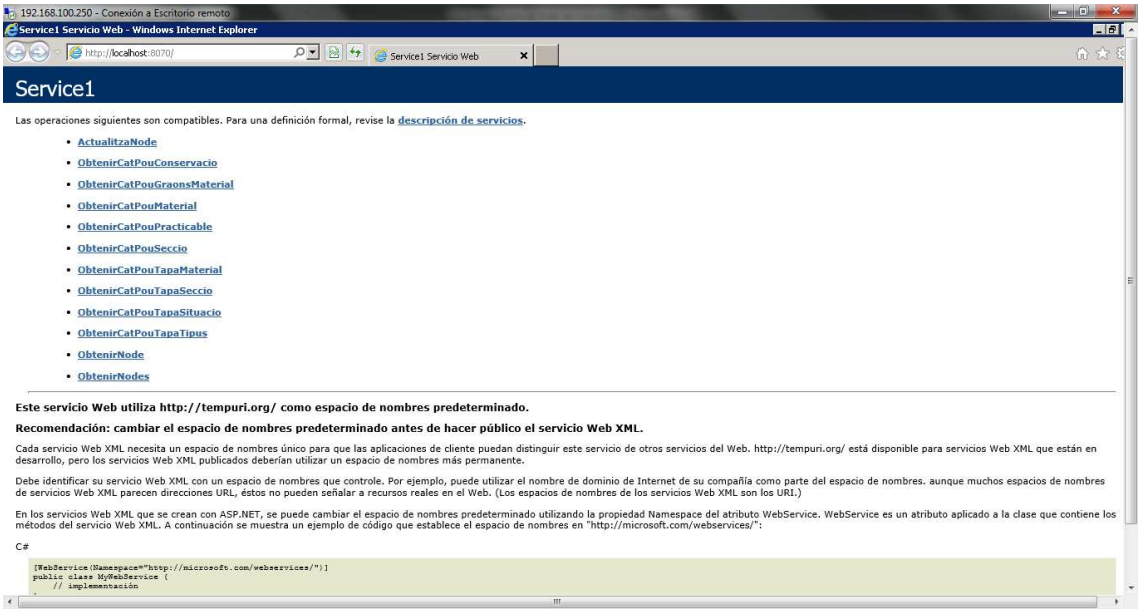


Aquí podem configurar la versió del Framework .NET i el mode de canalització. En el nostre cas la versió del Framework és la 4.0 i la canalització, “Integrada”.

Per comprovar que tot funciona correctament, tornem al sitio web “NETSanea_WS” i fem clic a “Examinar *:8070”

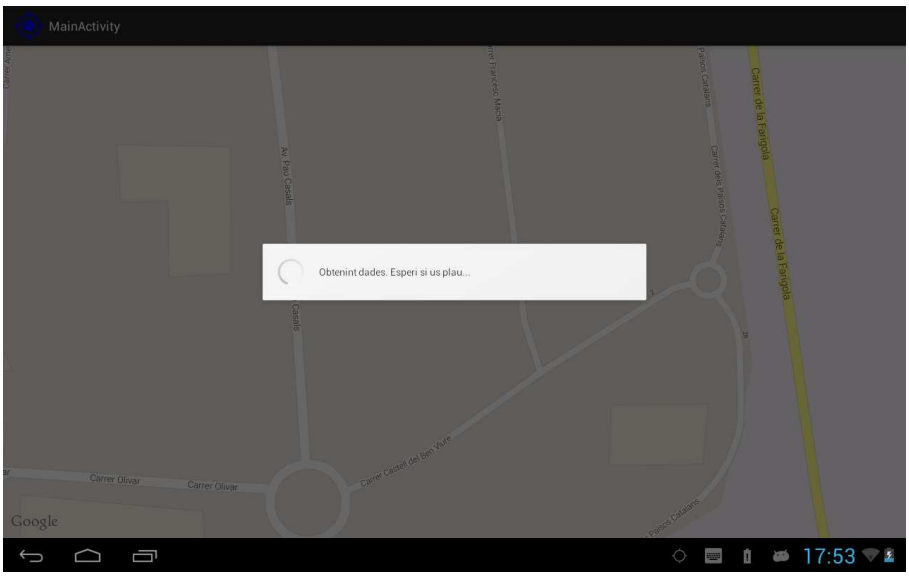


Si tot està ben configurat, veurem la pàgina web principal del Web Service

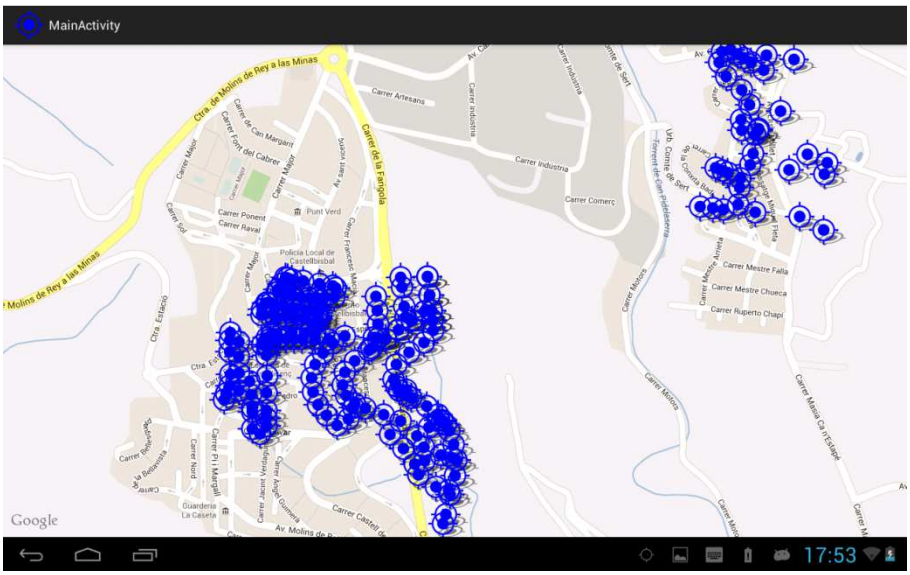


Annex III: Manual de l'aplicació

Quan executem el programa, el primer que fa és rebre les dades dels pous.

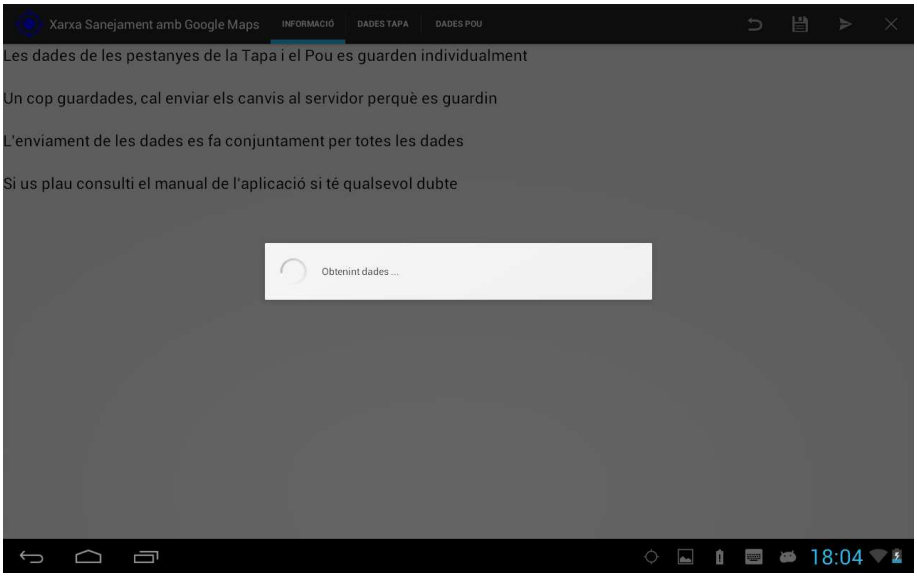


Un cop la rebem, ens apareixeran tots dibuixats sobre el mapa. Podem fer zoom amb 2 dits, seleccionant 2 punts de la pantalla i arrossegant-los cap a la part exterior.



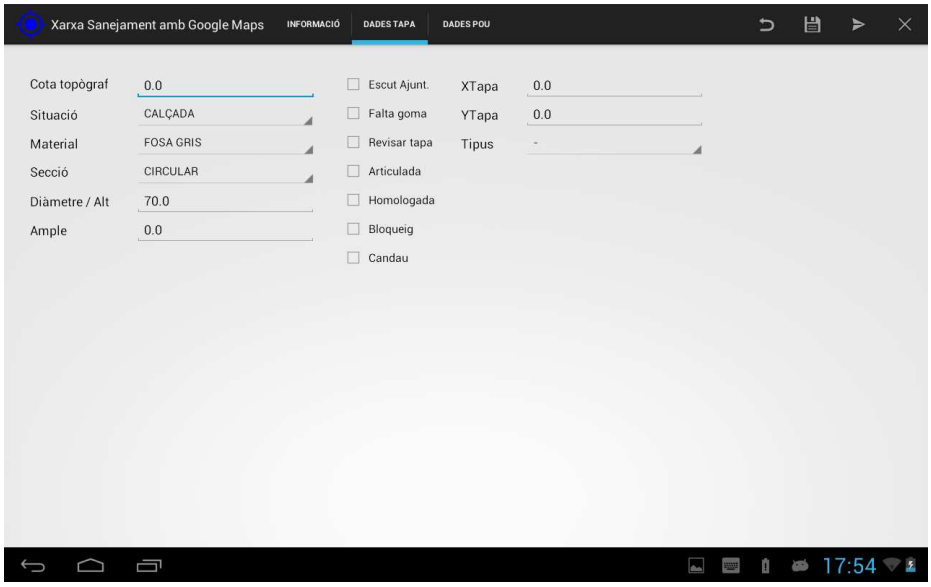
Un cop hem fet el zoom, tenim més precisió per triar el pou que volem modificar (nota: per fer aquest manual no m'he desplaçat a Castellbisbal i no es pot mostrar la posició de l'usuari al mapa, cosa que definitivament facilita la tria del pou)

Quan fem clic a un pou, l'aplicació torna a connectar amb el Web Service per rebre les dades.

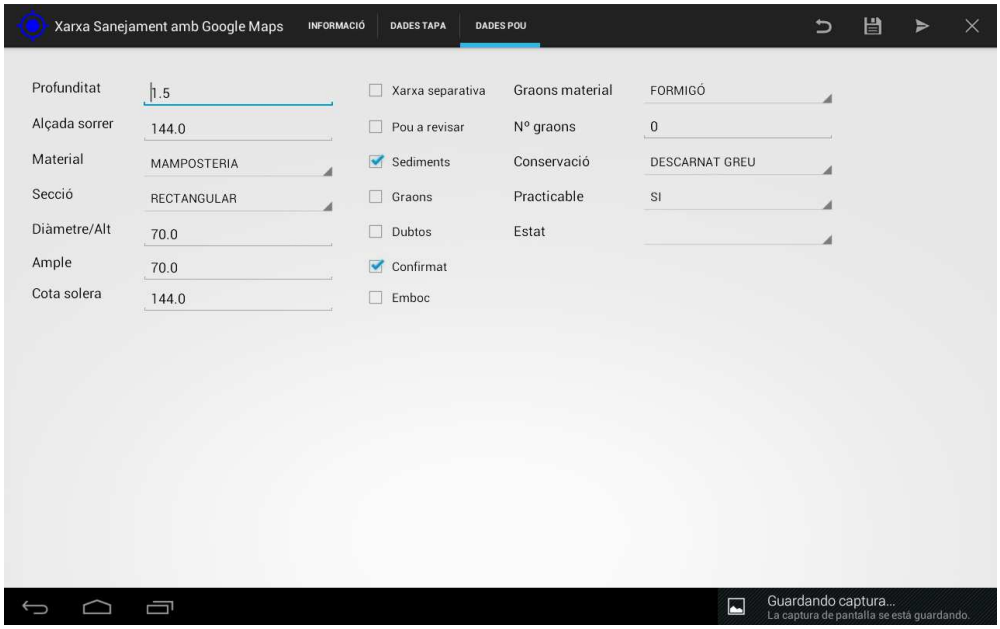


Un cop les rebem, a la primera pestanya tenim unes instruccions bàsiques de l'aplicació.

Si fem clic a la pestanya "Dades Tapa", visualitzem les dades de la tapa del pou.



Si fem clic a la pestanya “Dades pou”, visualitzem les dades del pou.



A la part dreta tenim la barra d’accions

Les accions son:

	Reset: desfà els canvis que hem fet a les dades *de la pestanya on ens trobem*
	Save: guarda les dades *de la pestanya on ens trobem*
	Send: Envia les dades actualitzades al servidor. Envia totes les dades, independentment de la pestanya on ens trobem
	Back: Tanca la finestra i tornem al mapa