Sistemes Encastats

Memòria del PFC

Pedro Andreo García
Universitat Oberta de Catalunya
Juny 2011

Índex de continguts

1. Introducció

El present document detalla la motivació, definició, planificació, disseny e implementació del PFC de Sistemes Encastats, així com l'anàlisi dels resultats obtinguts.

El capítol inicial s'inicia amb una descripció exhaustiva de la motivació i objectius del projecte, al temps que proporciona una introducció als sistemes encastats en general i a l'entorn *tinyOS* en particular. Així mateix, ofereix una enumeració detallada dels casos d'us identificats i els actors involucrats.

Respecte al capítol dedicat a la planificació, en primer lloc s'identifiquen i enumeren les activitats que compondran el projecte i el seu ordre d'execució. A continuació es defineix el calendari de treball amb dates concretes i s'estableixen les principals fites externes del projecte.

El següent capítol, dedicat al disseny del sistema, detalla totes les decisions de disseny preses i ofereix una descripció exhaustiva del disseny de cadascun dels mòduls que conformen la solució, oferint també una visió global de l'arquitectura i una especificació dels diferents protocols de comunicació involucrats.

A continuació, el capítol relatiu a la implementació del sistema descriu les estratègies que s'han utilitzat a l'hora de codificar els diferents mòduls així com les divisions lògiques i físiques que s'han establert pel codi font.

Els capítols 6 i 7 incorporen la documentació del projecte tant a nivell tècnic (construcció e instal·lació del sistema) com a nivell d'usuari.

Finalment, el capítol 8 plasma les conclusions extretes del desenvolupament del projecte així com els punts de millora i/o complementació no adreçats en el PFC. L'ultim capítol ofereix una llista de la bibliografia consultada durant tot el decurs de l'assignatura.

2. Descripció del projecte

2.1. Sistemes encastats i xarxes de sensors inal·làmbrics.

Ja sigui formant part de grans aplicacions com ara els sistemes de control d'un avió de passatgers o d'una central nuclear, o bé dintre de petits dispositius com ara reproductors de música o forns microones, el sistemes encastats estan presents en molts dels aspectes quotidians de les societats actuals.

Un sistema encastat es pot definir com un conjunt de maquinari i programari dissenyat específicament per un propòsit concret ([11]). Tot i que la frontera que defineix els sistemes encastats no està perfectament marcada, tradicionalment se'ls ha relacionat amb dispositius petits, que disposen d'una certa autonomia, que sovint son parts d'un sistema més complexe, que no ofereixen pràcticament interacció amb usuaris i que contenen programari molt específic.

A més de la seva característica principal, l'especificitat de la seva funció, els sistemes encastats n'acostumen a mostrar d'altres trets comuns, com podrien ser:

- Operació en temps real
- Disseny simplificat, que afavoreix l'eficiència i sovint permet reduccions significatives de costos, especialment en economies d'escala.
- Disponibilitat reduïda de recursos de maquinari. Us de microcontroladors i altres dispositius com ara conversors analògics/digitals. Alt nivell d'integració.
- Programari implementat en llenguatges de baix nivell, poc canviant, i sovint emmagatzemat en memòria de només lectura.

Les xarxes de sensors inal·làmbrics, per la seva banda, representen una aplicació concreta de la utilització de sistemes encastats de forma distribuïda i col·laborativa. Normalment incorporen dispositius que tenen la capacitat de monitoritzar determinats paràmetres del seu entorn (com ara temperatura, composició química d'una substància,...) i de comunicar-se amb d'altres dispositius per tal de distribuir la informació recopilada.

Entre les seves avantatges es troben el seu baix cost de producció, la seva tolerància a fallades i la seva capacitat de mobilitat i escalabilitat. Aquestes característiques han propiciat que aquest tipus de xarxes es despleguin en solucions que van des de l'anàlisi de la contaminació de l'aire fins a la detecció de moviment de tropes en aplicacions militars.

2.2. Justificació i context

Les xarxes de sensors inal·làmbrics estan adquirint un alt grau d'implantació gracies a que les seves característiques abans citades les fan aptes per nombroses i molt diverses aplicacions.

Els dispositius mòbils de consum, per una altra banda, estan experimentant també un alt grau de penetració al mercat gracies a sistemes com ara l'Android de Google o l'iOS d'Apple, i estan transformant la forma en que la societat es comunica i obté la informació que necessita en cada moment.

El present PFC pretén analitzar l'us d'ambdues tecnologies, amb tendència clarament ascendent, per tal de demostrar la seva idoneïtat per a una aplicació de domòtica.

2.2.1. Punt de partida i aportació del PFC

Com a punt de partida es disposarà de dos entorns tecnològics clarament establerts i madurs, com son l'entorn nesC/tinyOS per al desenvolupament i desplegament d'aplicacions sobre xarxes de sensors, i l'entorn Android, una de les principals plataformes de desenvolupament d'aplicacions per a dispositius mòbils d'última generació. Així mateix es disposarà de una plataforma de maquinari dissenyada per la UOC, les característiques de la qual propicien el seu us en aplicacions domòtiques.

L'aportació del PFC pretén ser la unió de totes aquestes tecnologies per tal de resoldre un problema concret, com es la gestió de la eficiència energètica, el confort i la seguretat a una llar.

2.2.2. Aplicació real

Donada la naturalesa eminentment pràctica amb que s'ha enfocat el PFC, les aplicacions reals del projecte descrit son immediates.

La solució proposada pot, amb un nombre relativament reduït de modificacions i ampliacions, implantar-se a una llar per tar de propiciar un control eficient i automatitzat de la seva gestió energètica, així com un control complet de seguretat del seu perímetre.

2.3. Objectius

A un nivell general, el sistema proposat tindrà la següent funcionalitat

- Control del perímetre de la llar, mitjançant la detecció de l'apertura de portes i finestres
- Control de persianes i regulació de la il·luminació artificial en funció de la intensitat de la llum natural present
- Control i regulació de la temperatura interior de la llar
- Interacció àgil i flexible de l'usuari amb el sistema mitjançant una aplicació disponible sobre un telèfon mòbil.

Per donar resposta a aquestes funcionalitats, es desenvoluparà un sistema que constarà dels següents components

1. Sensors de monitorització.

   Idealment n'hi hauria diversos sensors repartits estratègicament a les diferents estàncies de la llar. A efectes del present projecte, però, es disposarà d'un sol sensor que controlarà els paràmetres de lluminositat, temperatura i detecció d'apertura a una suposada finestra.

   Aquest sensor se n'encarregarà de:

a) Monitoritzar els paràmetres mencionats i comunicar-los a la estació central.

b) Sota demanda de la estació central, accionar un hipotètic motor d'una persiana, un hipotètic termòstat de regulació de temperatura, i/o un hipotètic regulador de llum artificial, que tendrien com a objectiu reduir o augmentar el nivell de temperatura i/o d'il·luminació natural/artificial a la estància. (la interacció amb aquests hipotètics dispositius externs es simularà mitjançant els tres leds del mateix sensor)

2. Estació central

Es tractarà d'un subsistema que permetrà la visualització dels valors actuals dels paràmetres de lluminositat, temperatura i detecció d'apertura que reporta el sensor de monitorització i, en el cas de la lluminositat o la temperatura, interactuarà automàticament amb ell per tal de corregir els valors si aquests no estan dintre del rang preestablert per l'usuari (lliurant-li ordres de pujada/baixada de persiana, increment/decrement de il·luminació artificial i pujada/baixa de termòstat per que les passi al corresponent dispositiu extern.

Aquest subsistema constarà de 2 components:

1. *Aplicació de monitorització*

Permetrà la visualització dels valors actuals dels paràmetres monitoritzats, la introducció del rang de lluminositat desitjat i interactuarà amb el sensor d'enllaç per tal que aquest retransmeti al sensor de monitorització les ordres correctives corresponents.

Aquesta aplicació estarà, a la seva vegada, composta per dos mòduls separats:

- Modul de Servidor: S'executarà com a servei a un PC. Realitzarà les tasques de comunicació amb el sensor de monitorització a través del sensor d'enllaç i mantindrà en memòria les lectures rebudes.

- Mòdul de presentació: Gestionarà la interacció amb l'usuari, mostrant-li les lectures obtingudes directament del modul de servidor.

El motiu d'aquesta separació es el d'oferir una separació respecte a la tecnologia de presentació.

2. *Sensor de enllaç*

Dispositiu que farà de pont entre l'aplicació de PC i el sensor de monitorització, rebent lectures i enviant ordres via radio.

El sistema queda representat a alt nivell al diagrama de la figura 2.1:

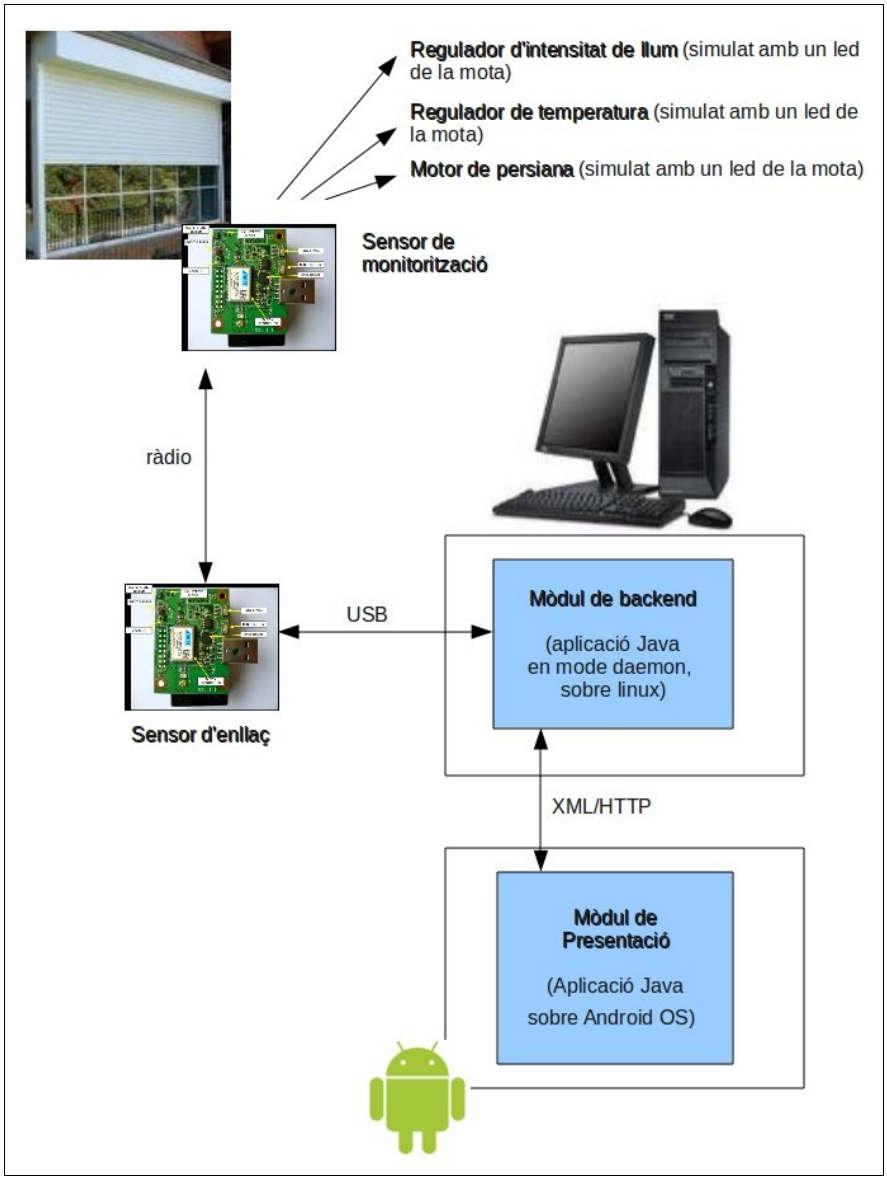

Figura 2.1: Diagrama d'alt nivell del sistema a construir

2.4. Definició de casos d'us

El present apartat detalla la relació de casos d'usos que representaran la funcionalitat oferta per la solució proposta.

En primer lloc, es defineixen els diferents actors que interactuaran amb el sistema:

- **Usuari**: La persona que visualitzarà l'estat del sistema i el configurarà mitjançant l'aplicació de presentació.
- **Punt de monitorització:** Finestra/porta física a la qual estarà connectada el detector magnètic del sensor de monitorització per tal de detectar la seva apertura/tancament.
- **Regulador d'apertura de persiana**: Sistema extern aliè al projecte, associat a un punt de monitorització, i capaç de processar comandes de pujada/baixada de la persiana,

accionant el corresponent motor. A efectes del PFC, les comandes es simularan mitjançant la visualització de polsos a un dels leds del sensor de monitorització.

- **Regulador de temperatura**: Sistema extern aliè al projecte, associat a un punt de monitorització, i capaç de processar comandes de augment/disminució de temperatura, accionant el corresponent termòstat. A efectes del PFC, les comandes es simularan mitjançant la visualització de polsos a un dels leds del sensor de monitorització.
- **Regulador d'intensitat de llum artificial**: Sistema extern aliè al projecte, associat a un punt de monitorització, i capaç de processar comandes de augment/disminució de lluminositat, accionant el corresponent potenciòmetre. A efectes del PFC, les comandes es simularan mitjançant la visualització de polsos a un dels leds del sensor de monitorització.

Un cop establerts els actors, i segons els objectius exposats pel projecte, s'identifiquen a continuació els casos d'us que defineixen la funcionalitat del sistema:

- UC01 - Introducció de rang de temperatura a mantenir
- UC02 - Introducció de rang de lluminositat a mantenir
- UC03 - Visualització de temperatura actual a l'estància mesurada
- UC04 - Visualització de nivell de lluminositat actual a l'estància mesurada
- UC05 - Visualització de l'estat del sensor magnètic de la finestra de l'estància mesurada.
- UC06 - Control automàtic de temperatura dintre del rang establert a l'estància mesurada
- UC07 - Control automàtic de nivell lluminositat dintre del rang establert a l'estància mesurada

A continuació es detalla cadascun d'ells:

Codi	UC01
Nom	**Introducció de rang de temperatura a mantenir**
Actors	Usuari
Descripció	El sistema ha de permetre que l'usuari especifiqui el rang mínim/màxim de temperatura que s'ha de mantenir a l'estància monitoritzada
Precondicions	L'aplicació de presentació ha establert comunicació amb l'aplicació de servidor
Flux	L'usuari introdueix els valors desitjats i aquests son enviats a l'aplicació de servidor, que els enregistra com a nous valors objectiu.
Postcondicions	L'aplicació de ha enregistrat el nou rang i procedeix a donar les ordres pertinents als sensors de monitorització per tal de mantenir-lo. (veure UC06)

Codi	UC02
Nom	**Introducció de rang de lluminositat a mantenir**
Actors	Usuari
Descripció	El sistema ha de permetre que l'usuari especifiqui el rang mínim/màxim de lluminositat que s'ha de mantenir a l'estància monitoritzada
Precondicions	L'aplicació de presentació ha establert comunicació amb l'aplicació de servidor
Flux	L'usuari introdueix els valors desitjats i aquests son enviats a l'aplicació de servidor, que els enregistra com a nous valors objectiu.
Postcondicions	L'aplicació de servidor ha enregistrat el nou rang i procedeix a donar les ordres pertinents als sensors de monitorització per tal de mantenir-lo. (veure UC07)

Codi	UC03
Nom	**Visualització de la temperatura actual de l'estància monitoritzada**
Actors	Usuari
Descripció	El sistema ha de permetre que l'usuari pugui visualitzar la temperatura actual de l'estància monitoritzada.
Precondicions	L'aplicació de presentació ha establert comunicació amb l'aplicació de servidor
Flux	L'aplicació de servidor va interrogant periòdicament el sensor de monitorització per tal d'obtenir la temperatura actual que aquest mesura, i l'enregistra. L'aplicació de presentació, a la seva vegada, va interrogant periòdicament l'aplicació de servidor per tal d'obtenir l'últim valor de temperatura que aquesta ha enregistrat.

	L'usuari pot en tot moment visualitzar a l'aplicació de presentació aquest darrer valor de temperatura mesurat.
Postcondicions	-

Codi	UC04
Nom	**Visualització de la lluminositat actual de l'estància monitoritzada**
Actors	Usuari
Descripció	El sistema ha de permetre que l'usuari pugui visualitzar el nivell de lluminositat actual de l'estància monitoritzada.
Precondicions	L'aplicació de presentació ha establert comunicació amb l'aplicació de servidor
Flux	L'aplicació de servidor va interrogant periòdicament el sensor de monitorització per tal d'obtenir la lluminositat actual que aquest mesura, i l'enregistra. L'aplicació de presentació, a la seva vegada, va interrogant periòdicament l'aplicació de servidor per tal d'obtenir l'últim valor de lluminositat que aquesta ha enregistrat. L'usuari pot en tot moment visualitzar a l'aplicació de presentació aquest darrer valor de lluminositat mesurat.
Postcondicions	-

Codi	UC05
Nom	**Visualització de l'estat del sensor magnètic del punt de monitorització.**
Actors	Punt de monitorització, Usuari
Descripció	El sistema ha de permetre que l'usuari pugui visualitzar l'estat del sensor magnètic del punt de monitorització.
Precondicions	L'aplicació de presentació ha establert comunicació amb l'aplicació de servidor
Flux	El sensor de monitorització notifica l'aplicació de servidor quan es produeixi un canvi en l'estat del detector magnètic. Aquesta procedeix a enregistrar el nou valor. L'aplicació de presentació, a la seva vegada, va interrogant periòdicament l'aplicació de servidor per tal d'obtenir l'últim valor del detector magnètic que aquesta ha enregistrat. L'usuari pot en tot moment visualitzar a l'aplicació de presentació aquest darrer valor mesurat.
Postcondicions	-

Codi	UC06
Nom	**Control automàtic de temperatura dintre del rang establert a l'estància mesurada**
Actors	-
Descripció	El sistema ha de controlar de forma automàtica que la temperatura de l'estància es manté dintre del rang mínim/màxim establert per l'usuari.
Precondicions	-
Flux	L'aplicació de servidor va interrogant periòdicament el sensor de monitorització per tal d'obtenir la temperatura actual que aquest mesura, i l'enregistra. En cas que la temperatura estigui fora de rang, envia una comanda al sensor de monitorització per tal que aquest ordeni al Regulador de Temperatura que procedeixi a incrementar/disminuir la temperatura. Quan la temperatura torna a estar dintre del rang establert, envia una comanda de cancel·lació al sensor de monitorització per tal que aquest deixi d'indicar al Regulador de Temperatura que modifiqui la temperatura.
Postcondicions	-

Codi	UC07
Nom	**Control automàtic de lluminositat dintre del rang establert a l'estància mesurada**
Actors	-
Descripció	El sistema ha de controlar de forma automàtica que la lluminositat de l'estància es manté dintre del rang mínim/màxim establert per l'usuari.
Precondicions	-
Flux	L'aplicació de servidor va interrogant periòdicament el sensor de monitorització per tal d'obtenir la lluminositat actual que aquest mesura, i l'enregistra. En cas que la lluminositat estigui fora de rang, envia una comanda al sensor de monitorització per tal que aquest ordeni als Reguladors de persiana i/o llum artificial que actuïn. Quan la lluminositat torna a estar dintre del rang establert, envia una comanda de cancel·lació al sensor de monitorització per tal que aquest deixi d'indicar als reguladors que actuïn.
Postcondicions	-

2.5. Tecnologies involucrades

A continuació es detallen les tecnologies utilitzades als diferents mòduls del sistema.

2.5.1. Aplicació encastada (sensors de monitorització i enllaç)

Es desenvoluparà mitjançant el llenguatge de programació nesC, i s'executarà sobre el sistema operatiu encastat tinyOS.

2.5.2. Aplicació de servidor

Es desenvoluparà íntegrament en Java 1.6 (probablement amb ajuda de diferents protocols estàndard i llibreries de codi obert, tot i que aquests aspectes s'analitzaran a la fase de disseny), i podrà executar-se indistintament sobre sistemes operatius Windows o Linux. A efectes del present PFC, tot el desenvolupament es realitzarà utilitzant Linux.

2.5.3. Aplicació de presentació

Es desenvoluparà íntegrament en Java, sobre sistema operatiu Android. A efectes del present PFC, tot el desenvolupament i validació es realitzarà sobre l'emulador d'Android de l'Android SDK.

3. Planificació del projecte

El present capítol detalla la planificació prevista per a l'execució del projecte, tenint en compte les fites externes establertes al pla de l'assignatura.

3.1. Relació d'activitats

A continuació es descriuen en detall totes les activitats que compondran el projecte.

1. Definició

Compren les activitats necessàries per tal de definir i documentar els objectius i abast del projecte, en funció de les possibilitats de l'entorn tecnològic prefixat al pla del PFC.

Aquest grup inclou les següents activitats:

Codi	3	**Predecessores**	-
Nom	**Estudi de tecnologia i possibilitats**		
Desc.	Estudi tecnològic de la plataforma cou24 subministrada i del seu funcionament a l'entorn tinyOS. Identificació de aplicacions potencials per aquest entorn tecnològic		
Entrades	Documentació Web PFC. Documentació tinyOS		
Sortides	-		

Codi	4	**Predecessores**	3
Nom	**Elaboració de proposta de projecte**		
Desc.	Elaboració de document de proposta de projecte en funció de l'estudi realitzat		
Entrades			
Sortides	Document de Proposta de Projecte		

2. Planificació

Compren les activitats necessàries per tal d'establir la planificació del cicle de vida complert del projecte.

Aquest grup inclou les següents activitats:

Codi	7	**Predecessores**	5
Nom	**Elaboració del document de planificació**		
Desc.	Identificació i planificació de les activitats i recursos necessaris per a l'execució del projecte. Definició de mecanismes de control e identificació de fites i riscos del projecte.		
Entrades	Document de Proposta de Projecte		

Sortides	Document de Planificació de Projecte

3. Disseny

Compren les activitats necessàries per a l'elaboració del disseny tècnic de la solució proposada a la fase de Definició.

Aquest grup inclou les següents activitats:

Codi	10	**Predecessores**	8
Nom	**Arquitectura del sistema**		
Desc.	Disseny tècnic a alt nivell del l'arquitectura del sistema e identificació dels seus components principals i les interaccions entre ells.		
Entrades	-		
Sortides	-		

Codi	11	**Predecessores**	10
Nom	**Aplicació nesC encastada**		
Desc.	Disseny tècnic detallat de l'aplicació o aplicacions nesC que s'executaran sobre la plataforma cou24, així com del protocol de comunicació via ràdio entre els diferents dispositius cou24. Disseny de la interfície de comunicació entre aquesta aplicació i l'aplicació de servidor		
Entrades	-		
Sortides	-		

Codi	12	**Predecessores**	11
Nom	**Aplicació de servidor**		
Desc.	Disseny tècnic detallat de l'aplicació de servidor, així com del protocol de comunicació amb l'aplicació de presentació.		
Entrades	-		
Sortides	-		

Codi	13	**Predecessores**	12
Nom	**Aplicació de presentació**		
Desc.	Disseny tècnic detallat de l'aplicació de presentació, i de la seva interfície d'usuari.		
Entrades	-		
Sortides	-		

Codi	14	**Predecessores**	13
Nom	**Elaboració del Document de Disseny**		
Desc.	Document de disseny englobant els resultats de les activitats de Disseny anteriors		
Entrades	-		
Sortides	Document de Disseny		

4. Implementació

Compren les activitats necessàries per a la implementació dels diferents components de la solució.

Aquest grup inclou les següents activitats:

Codi	17	**Predecessores**	15
Nom	**UC03 – Visualització temperatura actual**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC03, a l'aplicació nescC/tinyOS, a l'aplicació de servidor i a l'aplicació de presentació.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada, servidor i presentació		

Codi	18	**Predecessores**	17
Nom	**UC01 – Introducció del rang de temperatura objectiu**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC01, a l'aplicació nescC/tinyOS, a l'aplicació de servidor i a l'aplicació de presentació.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada, servidor i presentació		

Codi	19	**Predecessores**	18
Nom	**UC06 – Control automàtic de temperatural**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC06, a l'aplicació nescC/tinyOS.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada		

Codi	20	Predecessores	19
Nom	**UC05 – Visualització de l'estat del sensor magnètic**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC05, a l'aplicació nescC/tinyOS, a l'aplicació de servidor i a l'aplicació de presentació.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada, servidor i presentació		

Codi	21	Predecessores	20
Nom	**UC04 – Visualització lluminositat actual**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC04, a l'aplicació nescC/tinyOS, a l'aplicació de servidor i a l'aplicació de presentació.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada, servidor i presentació		

Codi	22	Predecessores	21
Nom	**UC02 – Introducció del rang de lluminositat objectiu**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC02, a l'aplicació nescC/tinyOS, a l'aplicació de servidor i a l'aplicació de presentació.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada, servidor i presentació		

Codi	23	Predecessores	22
Nom	**UC07 – Control automàtic de lluminositat**		
Desc.	Desenvolupament de la funcionalitat indicada al cas d'us UC07, a l'aplicació nescC/tinyOS.		
Entrades	-		
Sortides	Codi font d'Aplicació encastada		

5. Lliurament

Compren les activitats necessàries per a la validació i preparació del lliurament final del projecte.

Aquest grup inclou les següents activitats:

Codi	22	Predecessores	20
Nom	**Documentació final**		
Desc.	Tot i que la documentació tècnica es desenvoluparà en paral·lel a les tasques de		

	planificació, disseny e implementació del projecte, s'estableix un període addicional de temps per tal de completar i unificar els documents i generar la Memòria del projecte.
Entrades	-
Sortides	Memòria del Projecte

Codi	23	**Predecessores**	22
Nom	**Validació final**		
Desc.	Fase final de proves funcionals, validació del programari i de la documentació del projecte.		
Entrades	-		
Sortides	-		

3.2. Calendari de treball

La duració estimada del projecte es de tres mesos, compresos al període març-maig de 2011.

La Figura 3.1 mostra el diagrama de Gantt que representa les activitats identificades, amb la seva duració i ordre d'execució.

Donada la naturalesa, duració del projecte i fites externes preestablertes, s'ha optat per dissenyar una planificació basada en un cicle estàndard de desenvolupament en cascada. No obstant això, la fase d'implementació estarà dirigida pels casos d'us identificats, amb l'objectiu d'avançar al màxim la disponibilitat de parts operatives de la solució.

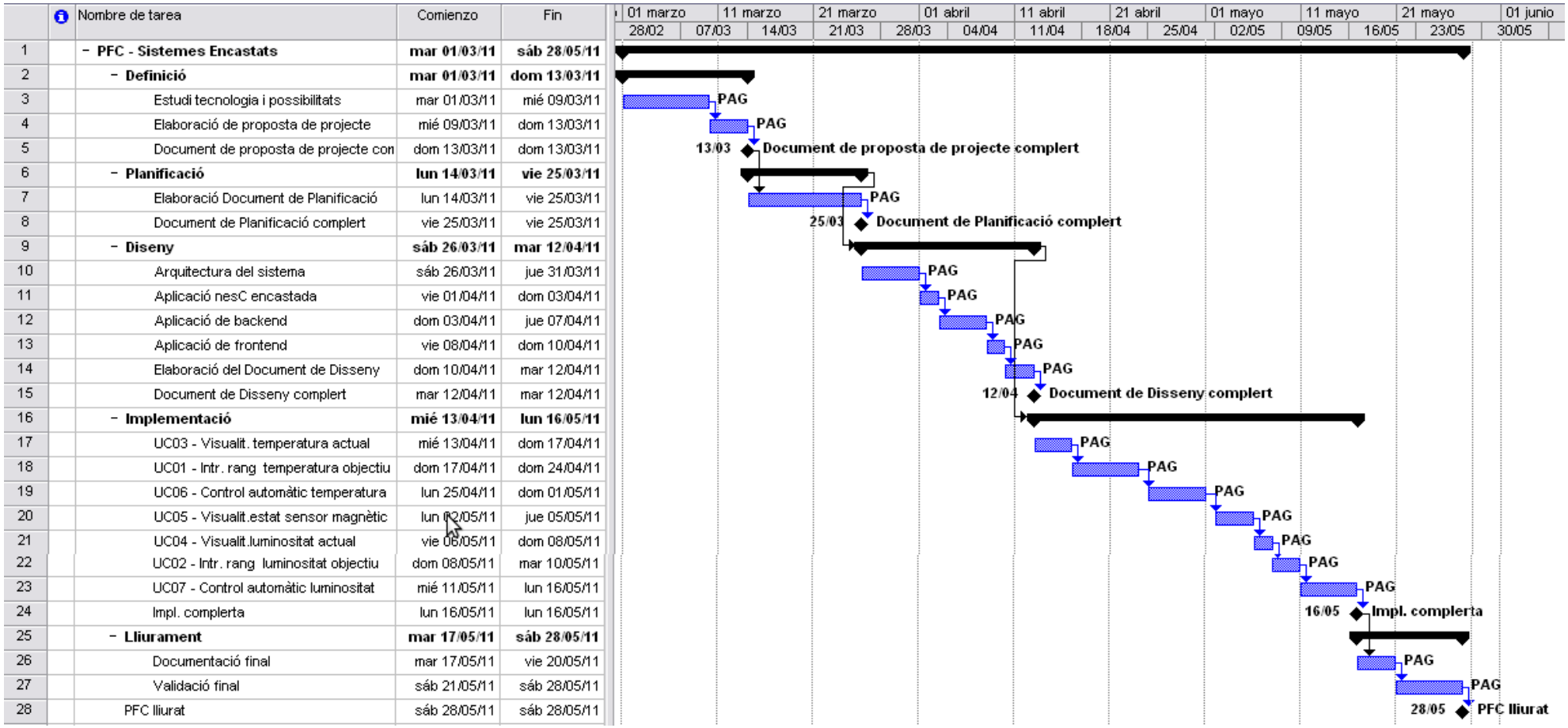

Figura 3.1: Planificació del projecte (Diagrama de Gantt)

3.3. Fites principals

A partir de la Planificació de l'apartat anterior, com a Fites més rellevants es consideren les següents:

Data	Descripció Fita
13/03/11	**Document de Proposta de Projecte lliurat (fita externa)** Un cop validat, haurà de permetre l'inici del procés de planificació.
25/03/11	**Document de Planificació de Projecte lliurat (fita externa)** Aquest document establirà les bases de les activitats posteriors i donarà pas a l'inici del disseny tècnic.
12/04/11	**Document de Disseny de Projecte lliurat (fita externa)** Aquest document detallarà l'arquitectura general del sistema i de cadascun dels seus components, i donarà pas a l'inici de la fase d'implementació. Tot i que hi existeix un requeriment d'entrega d'aquest document en una data posterior, es considerarà la data aquí indicada com a fita, obtenint així un període de temps major a la fase d'implementació.
17/04/11	**Cas d'us UC03 disponible (fita interna**) Funcionalitat relativa al cas d'us disponible per a verificació
24/04/11	**Cas d'us UC01 disponible (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
01/05/11	**Cas d'us UC06 disponible (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
05/05/11	**Cas d'us UC03 disponible (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
08/05/11	**Cas d'us UC04 disponible (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
10/05/11	**Cas d'us UC02 disponible (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
16/05/11	**Cas d'us UC07 disponible / Desenv. finalitzat (fita interna)** Funcionalitat relativa al cas d'us disponible per a verificació
28/05/11	**Projecte operatiu i lliurat (fita externa)** Darrera fita del projecte, que implica el lliurament del projecte complet i la seva corresponent documentació.

4. Disseny del projecte

El present capítol detalla el disseny de la solució a construir. En primer lloc s'exposen una serie de consideracions generals que han marcat les decisions de disseny i es mostra un detall de l'arquitectura a alt nivell. Posteriorment s'analitza el disseny detallat de cadascun dels mòduls que componen la solució.

4.1. Consideracions generals de disseny

A l'hora de dissenyar la solució, s'ha partit sempre de les següents premises a l'àmbit funcional:

1. **Punt únic d'interacció amb l'usuari, basat en l'ús d'un dispositiu mòbil**, amb la intenció de proporcionar a l'usuari una experiència de "comandament a distancia" del sistema. D'aquesta forma, l'usuari té un accés immediat a totes les funcionalitats del sistema des d'un dispositiu que habitualment estarà sempre molt a prop d'ell i podrà utilitzar el sistema de forma idèntica tant si es troba a la seva llar com si es a la feina o al carrer, sense la necessitat de tenir accés a un ordinador.

Així mateix, des del punt de vista tecnològic, s'han tingut presents els següents aspectes:

- **Modularitat, desacoblament i escalabilitat,** amb l'objectiu d'obtenir una solució reaprofitable i ampliable. Es per això que s'ha considerat imperatiu separar completament la interfície d'usuari de la lògica de negoci (control de les motes).

- **Us de estàndards**, tant a nivell de entorns tecnològics com de protocols de comunicació, per tal d'obtenir un sistema basat en components coneguts i de robustesa contrastada.

4.2. Disseny general del sistema

El disseny a alt nivell del sistema queda reflectit al diagrama de la figura 4.1, que mostra els següents elements:

- **Dispositiu mòbil – Aplicació client**
  Representa l'únic punt d'interacció amb l'usuari. Es tracta d'una aplicació Java sobre sistema operatiu *Android*.

- **Servidor – Aplicació servidor**
  Representa el punt d'intercomunicació entre la mota de monitorització i l'aplicació client. Manté en tot moment una còpia de les darreres lectures rebudes de la mota de monitorització i dels valors objectiu dels paràmetres monitoritzats que ha indicat l'usuari. Es tracta d'una aplicació Java basada en *Servlets* sobre *Apache Tomcat*.

- **Mota d'enllaç – Aplicació encastada**
  Permet la comunicació entre l'aplicació servidor i la mota de monitorització. Es tracta d'una aplicació *nesC* sobre *tinyOS* que comunica via port serie/USB amb el servidor i

via ràdio amb la mota de monitorització.

- **Mota de monitorització – Aplicació encastada**
  Aplicació *nesC* sobre *tinyOS* encarregada de monitoritzar els diferents paràmetres mitjançant els sensors, així com d'actuar sobre els reguladors (simulats mitjançant leds) quant els valors estan fora de rang.

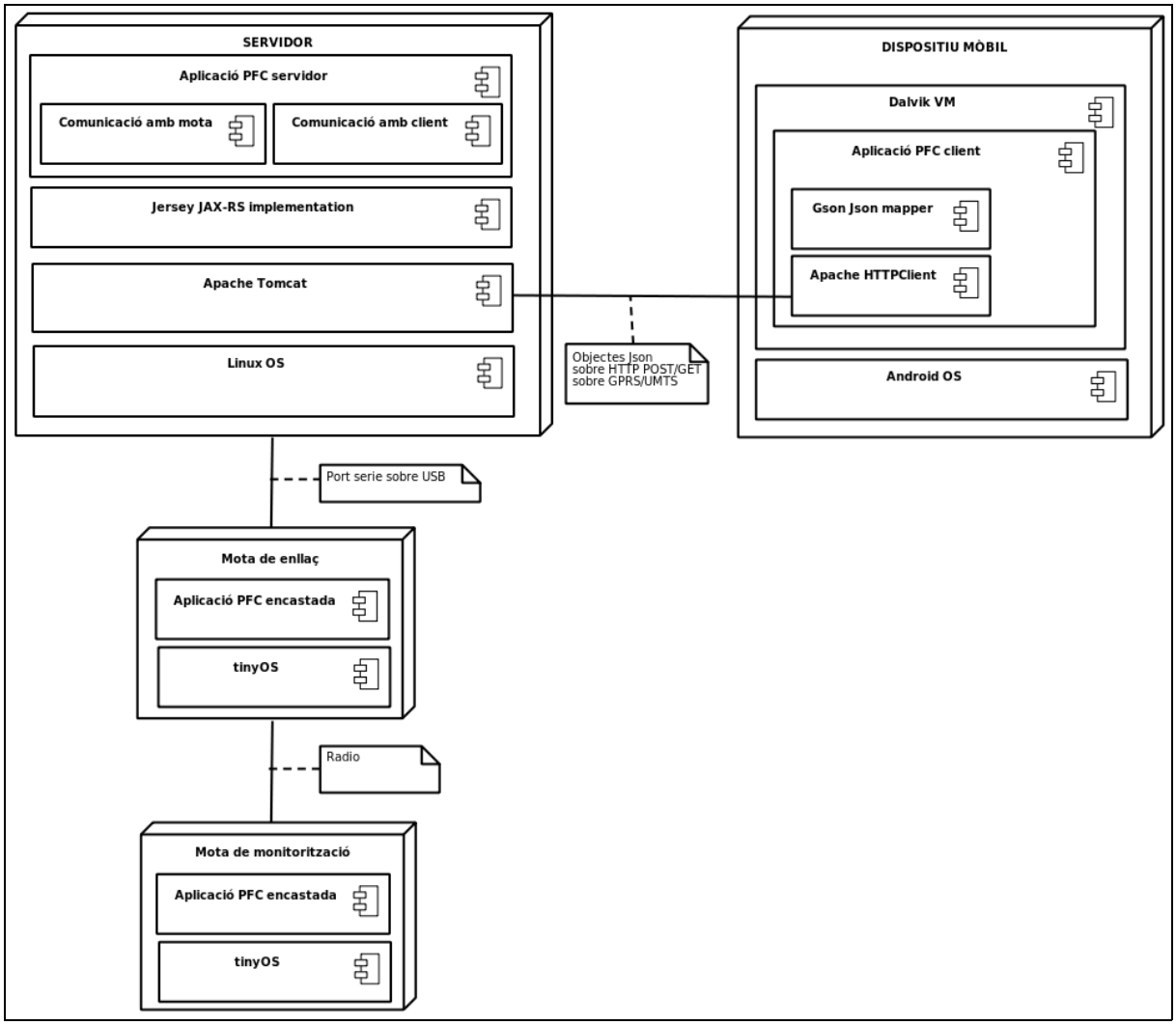

Figura 4.1: Diagrama de desplegament UML reflectint l'arquitectura del sistema a alt nivell

El disseny detallat de cadascun d'aquests elements, així com dels protocols de comunicació mitjançant els quals s'intercomuniquen, s'exposa als següents apartats.

4.3. Aplicació client al dispositiu mòbil

Tal i com s'ha mencionat anteriorment, l'aplicació client es desenvoluparà en Java sobre la plataforma *Android*.

4.3.1. Disseny de l'aplicació

El disseny de l'aplicació client ha tingut en compte dos factors fonamentals:

1. **Simplicitat/usabilitat de la interfície d'usuari:** Donat que aquesta aplicació representa l'únic punt de contacte entre l'usuari i el conjunt del sistema, i amb la dificultat afegida que representa el fet que s'executi sobre un dispositiu amb capacitats de visualització limitades,resulta fonamental oferir una interfície usable, pràctica i eficient.

2. **Gestió de la comunicació:** Les característiques inherents als dispositiu mòbils, com ara pèrdues temporals de connexió o filtratges agressius del tràfic de xarxa per part dels operadors, fan necessari el disseny d'una capa de comunicació flexible i robusta, basada en protocols de comunicació estàndard.

Respecte al primer punt, l'aplicació contempla tres pantalles (*activitats*, en terminologia *Android*) que permeten de forma simple tenir accés a la configuració del servidor, la visualització dels valors actuals dels paràmetres monitoritzats i l'establiment dels valors objectiu a regular.

En lo respectiu al segon punt, s'ha optat per dissenyar una capa de comunicació amb l'aplicació servidor basada en l'arquitectura *Representational State Transfer* (REST) [3]. Es tracta d'un mecanisme que permet posar a disposició d'un client una serie de recursos d'un servidor (cadascun dels quals es referenciable mitjançant un identificador o URI) utilitzant un protocol estàndard com es HTTP. Per mitjà d'aquests recursos el client pot subministrar i rebre informació de negoci del servidor, així com sol·licitar-li la execució d'accions determinades.

S'ha triat l'enfocament basat en REST en front d'altres aproximacions com per exemple XML-RPC, RMI o SOAP pels següents motius:

- Es tracta d'un mecanisme lleuger, perfectament suportat a la banda de servidor, i amb diverses implementacions sobre *Android* que el fan viable en aquesta plataforma.

- Utilitza HTTP com a protocol de comunicació. Aquest fet garanteix la correcta comunicació entre client i servidor per mitjà de qualsevol xarxa mòbil, eliminant el risc de bloqueig de la comunicació degut a filtratges de ports que els ISP mòbils consideren “no estàndard”.

L'apartat 4.6.1. reflecteix en detall la forma en que s'ha dissenyat aquest paradigma REST a la present solució.

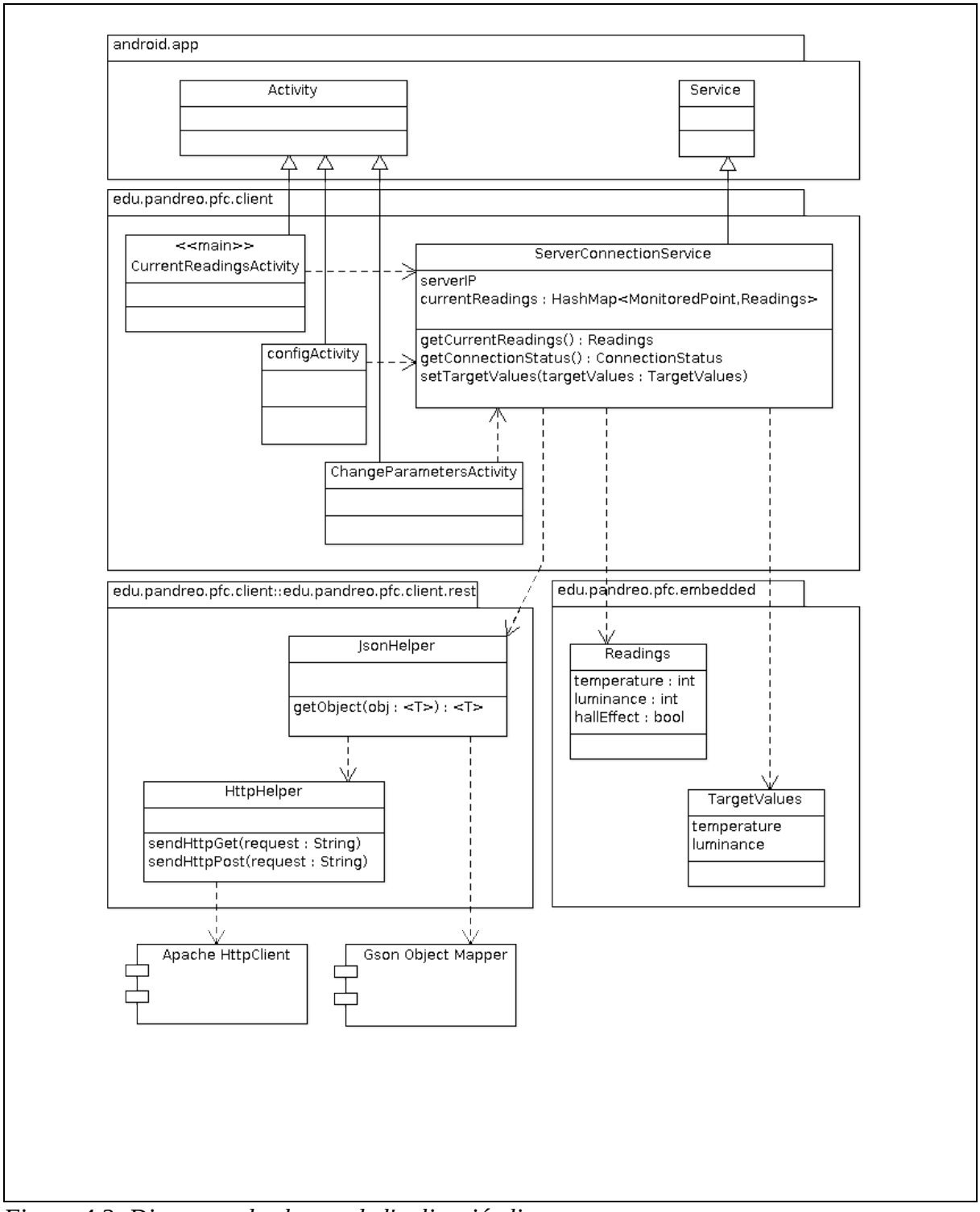

Figura 4.2: Diagrama de classes de l'aplicació client

4.3.2. Disseny de la interfície d'usuari

La interfície d'usuari es compondrà de les tres pantalles corresponents a les classes d'UI definides al punt anterior.

Aquestes pantalles oferiran la funcionalitat bàsica de configuració del servidor, visualització

dels valors actuals dels paràmetres monitoritzats i establiment de valors objectiu.

La figura 4.3 representa el contingut i distribució de les tres pantalles. Es tracta, però d'una representació conceptual que es veurà refinada a l'etapa de desenvolupament, en funció de les possibilitats de presentació de la plataforma *Android*.

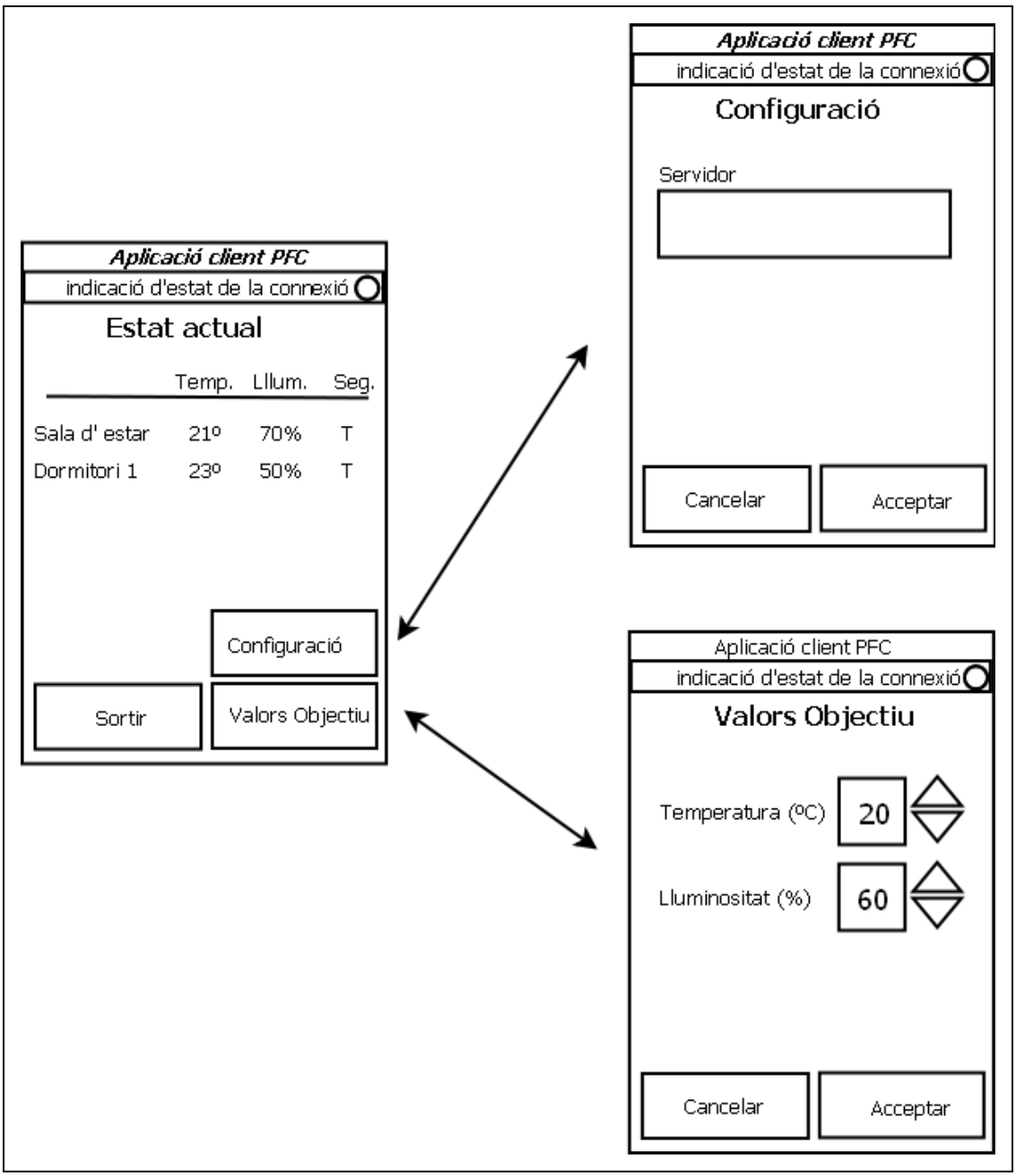

Figura 4.3: Representació conceptual de les pantalles de l'aplicació client

Tal i com es pot apreciar a la figura, totes tres vistes inclouen una barra d'estat que indicarà la disponibilitat o no de connexió al servidor.

4.4. Aplicació servidor al PC

La aplicació de servidor té com a objectiu servir de pont entre la mota de monitorització i l'aplicació client al dispositiu mòbil. No ha de incloure cap funcionalitat de presentació donat que no interacciona amb l'usuari.

El disseny d'aquesta aplicació està fortament condicionat per les decisions de disseny efectuades amb relació als protocols de comunicació.

Per una banda, la decisió d'implementar una arquitectura REST per la comunicació client/servidor implica que la aplicació ha de suportar el protocol HTTP. En aquest sentit, s'ha optat per aprofitar un component conegut i robust com es A*pache Tomcat*, ideal per al processament de peticions HTTP dinàmiques, i desenvolupar una aplicació web dinàmica.

Per altra banda, la aplicació encastada BaseStation que se n'encarregarà de fer de pont entre la mota d'enllaç i el PC (veure apartat 4.5.3.) es comunica amb aquest ultim via port sèrie/USB. Es desenvoluparà doncs el codi necessari per interaccionar amb aquesta aplicació com a part de l'aplicació web Tomcat. Aquest codi s'executarà a la mateixa maquina virtual Java.

La figura 4.4 mostra un diagrama de classes de l'aplicació del servidor. El conjunt de classes englobades al paquet *edu.pandreo.pfc.embedded* tindrà la responsabilitat de controlar la comunicació via port sèrie/USB amb la mota d'enllaç. La informació referent a les peticions i respostes es modelarà mitjançant objectes Java generats per *tinyOS* (veure apartat 4.6.2.).

La classe `MonitoringManager` (situada a la part central del diagrama) representarà el nucli de la lògica de negoci de l'aplicació del servidor. Es tracta, de fet, del veritable punt d'unió entre la informació a transmetre/rebre cap a les motes i la informació a transmetre/rebre cap a l'aplicació client. Dissenyada com a un patró de disseny *singleton* [4], implementarà la interfície `IMoteLinkListener`, per la qual podrà rebre les dades provinents de les motes i oferirà diversos mètodes als recursos REST, per tal que aquest puguin servir les peticions de l'aplicació client.

Els recursos REST, dintre del paquet *edu.pandreo.pfc.server.rest*, son classes anotades que el *framework Jersey* [6] interpretarà automàticament com a recursos REST accessibles des de *Tomcat* (veure apartat 4.6.1.)

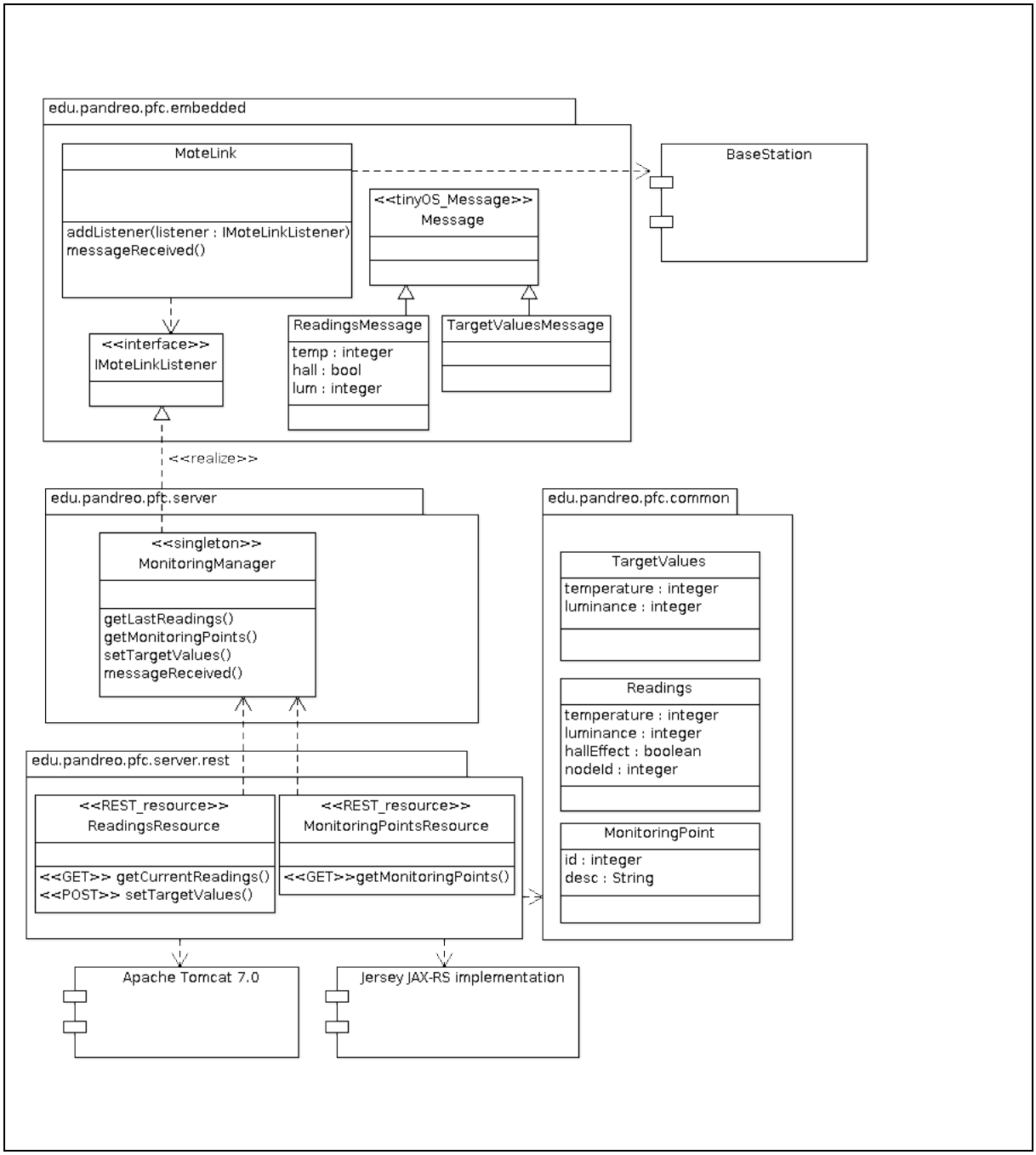

Figura 4.4: Diagrama de classes. Aplicació de servidor

4.5. Aplicacions encastades

4.5.1. Disseny genèric

D'una banda, l'aplicació de la **mota d'enllaç** té com a únic objectiu l'encaminament dels missatges que representen el protocol de comunicació entre la mota de monitorització i l'aplicació de servidor. En aquest sentit, aquesta aplicació haurà de tenir accés simultani a les interfícies de ràdio i port serie.

D'altra banda, l'aplicació de la **mota de monitorització** ha de realitzar tres tasques bàsiques:

- Control de l'aplicació encastada e intercanvi de missatges amb l'aplicació servidor (mitjançant la mota d'enllaç).
- Accés al reguladors que actuarien sobre elements externs a la mota per tal d'aconseguir els valors objectiu indicats per a la temperatura i luminància (simulats mitjançant leds)
- Accés als sensors de temperatura, luminància i efecte Hall, per tal d'obtenir els valors actuals d'aquests paràmetres.

La figura 4.5 mostrada a continuació reflecteix el disseny genèric d'ambdues aplicacions.

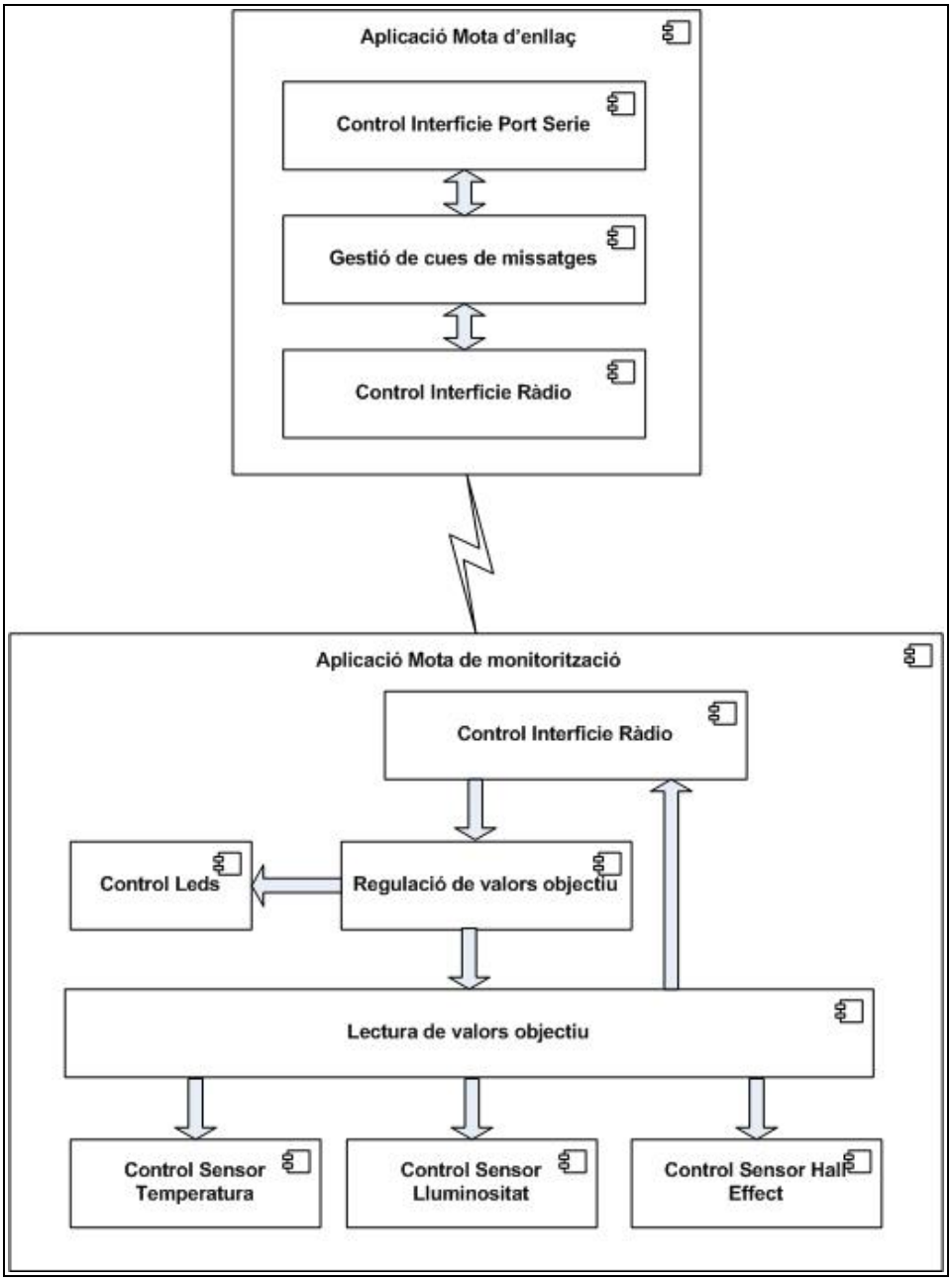

Figura 4.5: Disseny genèric de les aplicacions encastades.

4.5.2. Adaptació a tinyOS

TinyOS es un entorn especialment dissenyat per a ser utilitzat en xarxes de sensors inal·làmbrics. Tot i que, estrictament parlant, no es un sistema operatiu com tal, proporciona un conjunt de funcionalitats essencials que permeten el desenvolupament d'aplicacions específiques sobre ell, com son un planificador i un conjunt de components que faciliten l'abstracció del maquinari.

Entre les seves característiques fonamentals, convé destacar les següents:

- La seva arquitectura està basada en un model orientat a la gestió d'events. Aquest model s'implementa en dos nivells, representats per events i tasques.
- Es basa en un model d'execució simple: no incorpora nucli, ni gestió de processos ni memòria virtual.
- Les aplicacions es construeixen mitjançant components, cadascun dels quals especifica mitjançant interfícies les seves dependències (altres components que consumeix) i funcionalitats exportades (que proporciona a altres components).

La figura 4.6 mostra una versió molt simplificada de l'arquitectura de *tinyOS* ([12]).

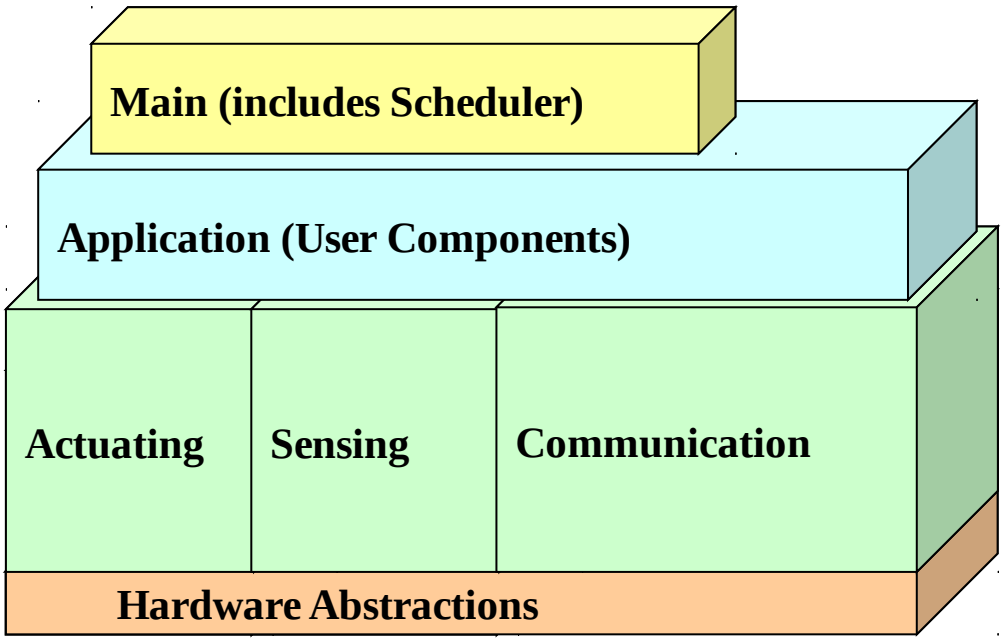

Figura 4.6: Arquitectura simplificada de tinyOS

4.5.3. Aplicació encastada a la mota d'enllaç

S'ha optat per utilitzar una aplicació ja existent que forma part de la distribució de *tinyOS*. Aquesta aplicació, anomenada BaseStation, farà de pont entregant els missatges que provenen via ràdio de la mota de monitorització cap al port sèrie del PC de l'aplicació de servidor, i viceversa.

Es tracta d'una aplicació robusta que incorpora el tractament de cues de missatges d'entrada i sortida per tal de proporcionar una gestió eficient de possibles pics de càrrega.

4.5.4. Aplicació encastada a la mota de monitorització

El disseny de l'aplicació s'ha articulat en base als tres grups de funcionalitats identificats al disseny genèric, mantenint aquesta jerarquia funcional i donant com a resultat un component especialitzat per a cada grup.

El disseny genèric d'aquesta aplicació ha estat adaptat a l'entorn *tinyOS* tenint en compte les següents consideracions:

- Per no complicar en excés les interdependències entre components *tinyOS*, el component que modela els reguladors es farà servir també per a l'obtenció de les lectures actuals per part de les capes superiors i encapsularà al component que modela als sensors, retransmetent-li les peticions de lectura.
- L'accés als diferents sensors a *tinyOS* es portarà a terme mitjançant el component que modela el convertidor analògic/digital de la mota.
- A diferència del control directe de la interfície de ràdio proposat al disseny genèric, *tinyOS* proporciona un nivell d'abstracció que permet gestionar la comunicació a nivell de missatges.

Aquestes particularitats, han donat com a resultat el model reflectit al diagrama de la figura 4.7. Aquest diagrama detalla els diferents components *tinyOS* i la seva interacció[1]. De més alt a més baix nivell:

1. El punt d'entrada de l'aplicació es el component **`PfcEmbeddedC`**, encarregat del control l'aplicació i de la gestió de la comunicació amb l'aplicació de servidor.
2. Aquest element fa servir el component **`RegulationsControllerC`**, que proporciona una abstracció complerta tant de la lectura dels paràmetres monitoritzats com de l'actuació sobre els reguladors que han de mantenir-los dintre dels valors objectiu establerts.
3. Finalment, `RegulationsControllerC` utilitza el component **`SensorsControllerC`** que proporciona una abstracció de la lectura dels valors de temperatura, lluminositat i efecte Hall.

Component PfcEmbeddedC

Es tracta del component principal i punt d'entrada de l'aplicació i tindrà com a funció principal l'enviament i recepció de missatges mitjançant la interfície de ràdio proporcionada per la mota.

Utilitzarà els components `ActiveMessageC`, `AMSenderC` i `AMReceiverC` proporcionats per l'entorn tinyOS per tal de rebre missatges de la interfície de ràdio i notificar periòdicament (mitjançant el component `Timer`) les darreres lectures efectuades sobre els sensors, que obtindrà del component `RegulationsController`.

S'encarregarà també d'Indicar a aquest component `RegulationsController`, prèvia recepció del missatge pertinent via ràdio, quins son els valors objectiu que aquest ha d'assolir.

1 El diagrama utilitza una notació UML adaptada a l'entorn tinyOS, on els elements de tipus configuració es consideren especificacions, i els mòduls d'implementació es consideren realitzacions [1].

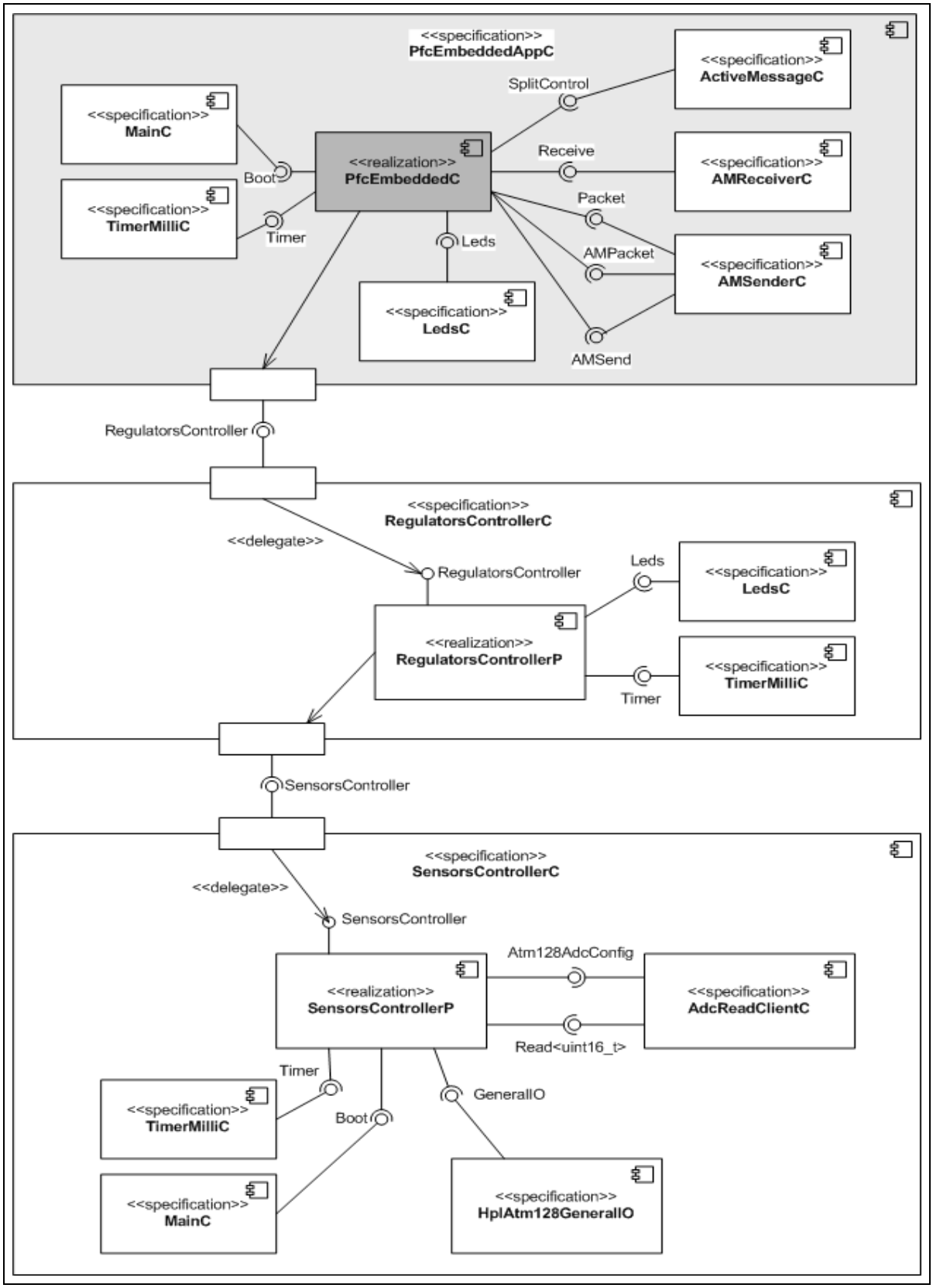

Figura 4.7: Aplicació encastada de la mota de monitorització. Diagrama de components UML del disseny adaptat a tinyOS.

Component RegulationsControllerC

Es tracta del component encarregat de actuar sobre els reguladors per tal d'assolir els valors objectiu indicats per als paràmetres sota monitorització.

Com s'ha indicat amb anterioritat, la interacció amb uns possibles reguladors de temperatura i lluminositat es limitarà a una simulació utilitzant els leds de la mota.

Aquesta simulació es realitzarà de la següent forma:

- **Regulació de temperatura**: El component `RegulationsControllerC` provocarà un *blinking* lent (transicions entorn a 1000ms) del led 1 (taronja) de la mota per indicar una ordre de disminució de temperatura, i un *blinking* més ràpid (transicions entorn a 200ms) per indicar una ordre d'augment de temperatura.

- **Regulació de lluminositat**: El component `RegulationsControllerC` provocarà un *blinking* lent (transicions entorn a 1000ms) del led 2 (verd) de la mota per indicar una ordre de disminució d'intensitat al sistema d'il·luminació de la llar, i un *blinking* més ràpid (transicions entorn a 200ms) per indicar una ordre d'increment d'intensitat.

- **Efecte Hall**: El component `RegulationsControllerC` mantindrà el led 0 (vermell) de la mota encès mentre el sensor d'efecte Hall reporti activació[2] per indicar una ordre sostinguda d'activació d'un hipotètic sistema d'alarma.

Es per aquest motiu que aquest component només utilitza els components `Timer` i `Leds` de l'entorn tinyOS, a més del component `SensorsControllerC`, del qual obtindrà les lectures actuals dels sensors.

Component SensorsControllerC

Aquest component tindrà com a únic objectiu obtenir les lectures dels sensors i emmagatzemar-los per tal d'entregar-los immediatament quan els altres components li sol·licitin.

Per aconseguir-ho accedirà als components que proporcionen la capa d'abstracció del microcontrolador *Atmel* i del seu ADC (components `HplAtm128GeneralIO` i `AdcReadClientC` respectivament).

Definició d'interfícies dels components

D'una banda, tal i com s'observa al diagrama anterior, el component principal `PfcEmbeddedC` interacciona amb el component `RegulationsControllerC` per mitjà de la interfície `RegulationsController`. D'altra banda, `RegulationsControllerC` utilitza el component `SensorsControllerC` mitjançant la seva interfície `SensorsController`.

La figura 4.8 descriu les operacions (*commands*, en terminologia *tinyOS*) que proporcionen aquests dos mòduls mitjançant les seves respectives interfícies.

2 A efectes del projecte, i per facilitar la visualització de la simulació, es considerarà que la seguretat del punt de monitorització ha estat compromesa quan el sensor d'efecte Hall reporta activació. En una aplicació real, el sensor reportaria activació mentre el punt de monitorització està correctament tancat i, per tant, no compromès.

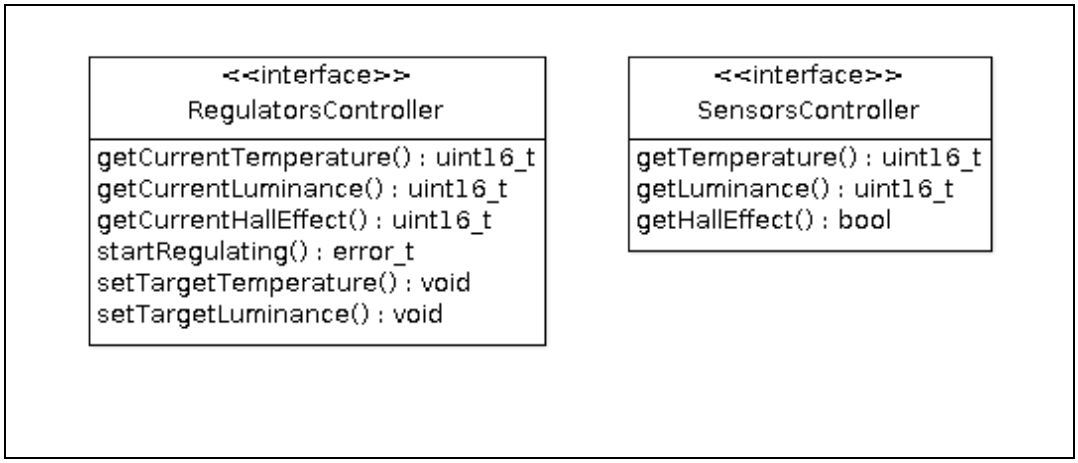

Figura 4.8: Interfícies dels components RegulationsController i SensorsController

4.6. Protocols de comunicació

El present apartat detalla els protocols de comunicació entre els components que composen el projecte, així com els elements (estàndards, llibreries, etc...) que es faran servir per implementar-los.

4.6.1. Comunicació entre l'aplicació client i l'aplicació servidor

Tal i com s'ha comentat amb anterioritat, la comunicació entre l'aplicació client i l'aplicació servidor es portarà a terme mitjançant una implementació particular del paradigma REST.

A continuació es detalla la seqüència complerta d'accions que tindran lloc durant una petició de l'aplicació client a l'aplicació servidor.

1. L'aplicació client, mitjançant la llibreria *Gson* [8], converteix l'objecte que contindrà els paràmetres de la petició en un objecte compatible amb el protocol JSON [7].

2. Amb aquest objecte JSON, i amb la URI que identifica el recurs REST al servidor que ha de donar servei a la petició, l'aplicació client utilitza la llibreria *Apache HTTPClient* [10] per tal de generar una petició HTTP GET o POST estàndard.

3. Aquesta petició es processada per *Apache Tomcat* al servidor, i adreçada al servlet que fa de punt d'entrada al framework Jersey [6] Aquest, a la seva vegada, l'encamina cap a la classe anotada que representa el recurs REST indicat a la URI.

4. El mètode corresponent d'aquesta classe es invocat automàticament per *Jersey*. Aquest mètode, habitualment, farà servir els objectes de negoci residents al servidor per obtenir la resposta necessària, que retornarà com a objecte POJO bàsic.

5. El *framework Jersey* se n'encarregarà automàticament de transformar l'objecte POJO a una representació JSON que encapsularà a una resposta HTTP.

6. De nou al dispositiu mòbil, *HttpClient* rebrà la resposta HTTP i l'aplicació client, mitjançant *Gson* , transformarà la notació JSON obtinguda en un objecte POJO bàsic que contindrà la informació de negoci sol·licitada.

Així doncs, el protocol de comunicació queda reflectit a la figura 4.4 per les classes POJO del paquet *edu.pandreo.pfc.common* (que representen l'intercanvi d'informació) així com les classes anotades del paquet *edu.pandreo.pfc.server.rest* (que representen les possibles peticions).

4.6.2. Comunicació entre l'aplicació servidor i la mota d'enllaç

Tal i com reflecteix la figura 4.4, la comunicació via port sèrie/USB amb l'aplicació encastada a la mota d'enllaç es realitzarà, per part del servidor, mitjançant les classes del paquet *edu.pandreo.pfc.embedded*.

La clau en aquest protocol de comunicació serà la utilització de l'eina MIG de l'entorn *tinyOS*, la qual, a partir de les estructures de missatges definides al codi *nesC* de l'aplicació encastada, generarà de forma automàtica les classes Java equivalents que encapsularan a la banda del servidor els missatges rebuts/enviats pel port serie/USB.

4.6.3. Comunicació entre la mota d'enllaç i la mota de monitorització

Els missatges que es transmetran ambdues motes mitjançant els seus canals de ràdio son, a nivell conceptual, els representats per les classes `ReadingsMessage` i `TargetValuesMessage` a la figura 4.4.

L'aplicació de la mota d'enllaç farà, de fet, de retransmissor entre el servidor i la mota de monitorització sense tenir constància real de la semàntica dels missatges que distribueix.

5. Implementació del Projecte

El present apartat detalla l'estructura física dels diferents projectes que conformen el sistema i reflecteix les decisions d'implementació i desviacions respecte el disseny inicial.

5.1. Aplicació client

L'aplicació client s'ha desenvolupat de forma fidel al disseny previ tot i que n'hi han hagut diverses desviacions, la principal de les quals ha estat la implementació del servei de gestió de la connexió amb el servidor mitjançant una classe *singleton* estàndard en lloc d'un servei Android.

Així mateix, a nivell d'interfície d'usuari, s'ha omès la existència d'un botó explícit per a l'accés a la vista de Valors Objectiu. L'accés a aquesta vista es farà fent click directament sobre la fila que representa el punt de monitorització a modificar, essent aquest un mecanisme més àgil.

L'estructura del projecte `pfc.client` es la següent:

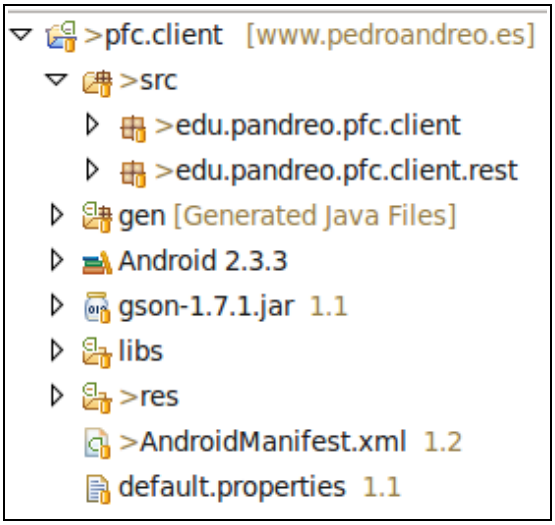

Figura 5.1: Estructura del projecte pfc.client

El projecte ha estat optimitzat específicament per dispositius Android 2.x amb pantalla de baixa resolució QVGA de 2.7i.

5.2. Aplicació servidor al PC

El codi font que representa la implementació d'aquesta aplicació s'ha estructurat en diversos projectes Eclipse, que es descriuen a continuació:

5.2.1. Projecte pfc.embedded.link

Conté la lògica necessària per a accedir a la mota d'enllaç per tal d'enviar i rebre missatges a les motes de monitorització. La seva implementació ha estat fidel al disseny previ. La seva estructura es reflecteix a la figura següent:

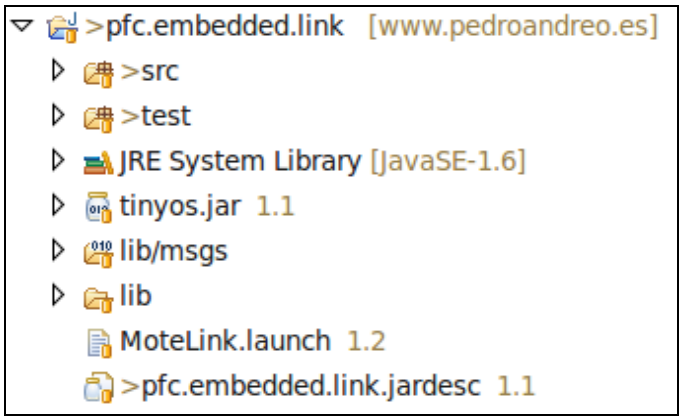

Figura 5.2: Estructura del projecte pfc.embedded.link

El resultat del projecte es una llibreria JAR de Java, que s'incorporarà al projecte `pfc.server.restserver`.

5.2.2. Projecte pfc.server.restserver

Conté la lògica necessària per a rebre peticions de l'Aplicació Client i encaminar-les cap a les motes mitjançant el codi font del projecte anterior. S'ha implementat com una aplicació Web desplegable sobre *Apache Tomcat* 7.x, i s'ha codificat amb l'ajut del suport WEB que integra l'IDE Eclipse en la seva versió per a J2EE.

La seva estructura es reflecteix a la figura següent:

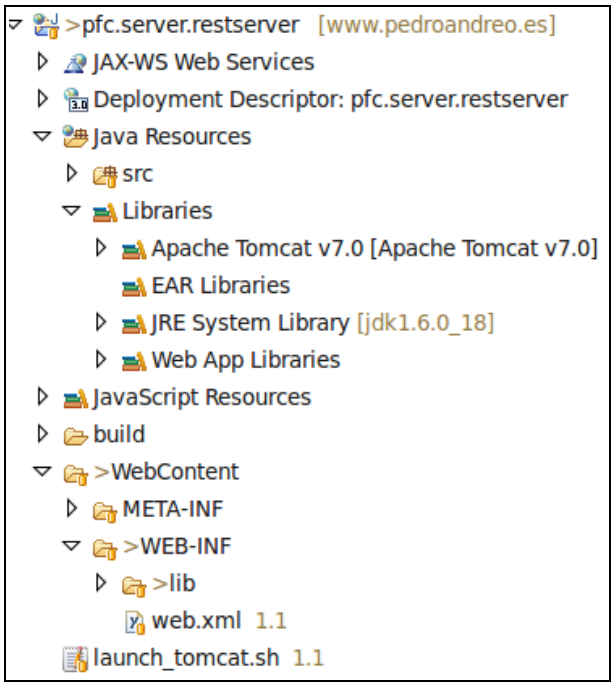

Figura 5.3: Estructura del projecte pfc.server.restserver

La implementació ha estat fidel al disseny previ. En destaquen les classes Java anotades que representen els recursos REST disponibles per a l'Aplicació Client, que el *framework* Jersey reconeix i gestiona automàticament un cop desplegats sobre *Tomcat*:

```
@Path("/readings")
public class ReadingsResource {

    @GET
    @Produces( {MediaType.APPLICATION_JSON })
    public Readings getCurrentReadings() {
        return MonitoringManager.getInstance(
    }

    @POST
    @Consumes( {MediaType.APPLICATION_JSON })
    public PostResult updateTargetValues(Read
        PostResult ret = new PostResult();
        if(MonitoringManager.getInstance().se
            ret.setSuccess(true);
        else
            ret.setSuccess(false);
        return ret;
    }
```

Figura 5.4: Example d'anotacions REST

5.3. Aplicacions encastades

L'aplicació de la mota de monitorització ha estat implementada de forma fidel al disseny previ (en especial al indicat a la figura 4.7), tot i que cal indicar diverses desviacions:

- El led 0 (vermell) de la mota s'utilitza finalment per a indicar l'enviament per part d'aquesta d'un missatge cap al servidor. S'ha optimitzat l'enviament de missatges mota->PC per tal que només es produeixin quan hi ha hagut un canvi real a algun dels valors monitoritzats. El led 0, doncs, hauria de reflectir aquesta optimització.

- El sensor Hall, per tant, no està simulat mitjançant leds sinó que el seu estat es visible a l'aplicació client.

L'estructura del projecte `pfc.embedded.mote` es la següent:

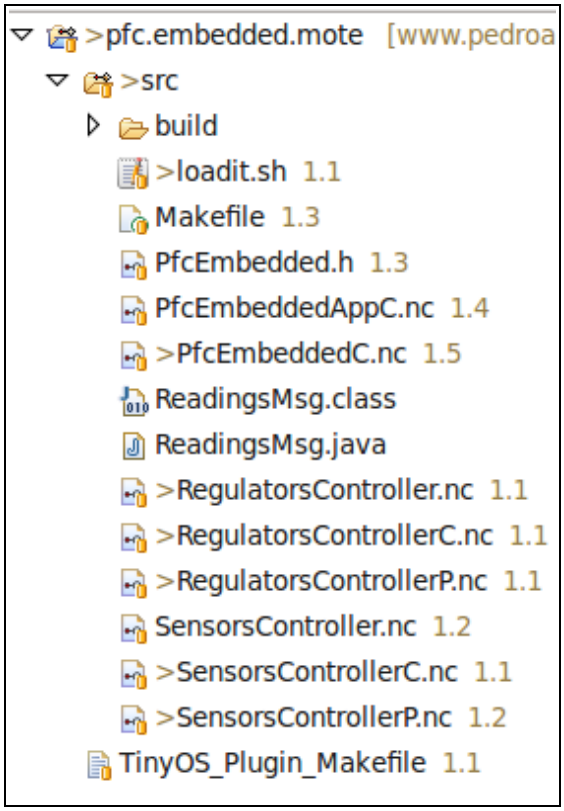

Figura 5.5: Estructura del projecte pfc.embedded.mote

5.4. Codi font comú a les aplicacions client i servidor

Com a part de la comunicació entre l'Aplicació Client i l'Aplicació Servidor, ha estat necessària la implementació d'un conjunt de classes Java de tipus POJO.

Aquestes classes han estat agrupades al projecte `pfc.common`, del qual depenen tant el projecte `pfc.client` com el projecte `pfc.server.restserver`.

6. Documentació tècnica del projecte

El present apartat agrupa la documentació generada com a part del projecte a nivell tècnic (construcció e instal·lació).

6.1. Construcció de la solució

El present apartat detalla els requeriments i els passos necessaris per a construir els diferents components del sistema.

6.1.1. Estructura de lliurament

El lliurament que compren els diferents components de la solució està estructurat segons l'arbre de directoris reflectit en la figura 6.1.

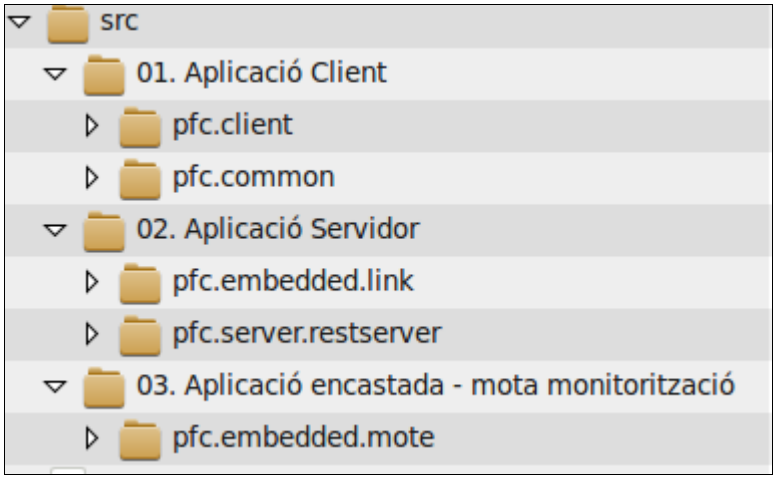

Figura 6.1: Estructura de lliurament del codi font

6.1.2. Aplicacions encastades

Les aplicacions encastades han estat implementades en llenguatge *nesC*. Per tal de procedir a la seva construcció, serà necessari disposar d'un equip que tingui instal·lats i correctament configurats els següents requeriments:

- Sistema operatiu Linux (preferentment Ubuntu 10.04 o superior)
- Entorn de compilació de C/C++ gcc
- Distribució tinyOS, versió 2.1 amb les modificacions necessàries per tal de suportar la plataforma de maquinari COU24 de la UOC.

La construcció de l'aplicació encastada corresponent a la mota de monitorització es porta a terme mitjançant la línia de comandes de Linux, escrivint la comanda:

```
:~$ make cou24
```

dintre del directori `pfc.embedded.mote/src`.

El resultat de la construcció serà l'arxiu `main.srec`, ubicat al directori `pfc.embedded.mote/build/cou24`. Aquest arxiu està llest per ser carregat a la mota corresponent, tal i com es descriu a l'apartat relatiu a la instal·lació del sistema.

La construcció de l'aplicació corresponent a la mota d'enllaç es fa de idèntica forma (el codi font d'aquesta aplicació no està inclòs al lliurament, donat que es tracta de l'aplicació *BaseStation* pertanyent a la distribució de *tinyOS*).

6.1.3. Aplicació de Servidor

L'aplicació de servidor ha estat implementada en llenguatge Java. Com es pot apreciar a la figura 6.1, aquesta aplicació està composta per dos mòduls. El primer d'ells, el mòdul `pfc.server.restserver`, té com a objectiu la comunicació amb l'aplicació client mitjançant la publicació d'un servei REST sobre *Apache Tomcat* (el resultat de la seva construcció es, per tant, un arxiu WAR). El segon, el mòdul `pfc.embedded.link`, té com a objectiu la comunicació amb la mota d'enllaç i el resultat de la seva construcció es una llibreria de classes JAR,

La construcció d'ambdues aplicacions es farà mitjançant Eclipse. Per tal de procedir-hi, serà necessari disposar d'un equip que tingui instal·lats i correctament configurats els següents requeriments:

- Sistema operatiu Linux (preferentment Ubuntu 10.04 o superior)
- Entorn de desenvolupament Java JDK 1.6.x
- Eclipse Helios per a desenvolupament J2EE
- Distribució tinyOS, versió 2.1 amb les modificacions necessàries per tal de suportar la plataforma de maquinari COU24 de la UOC.
- Apache Tomcat 7.x

Pas 1. Construcció del projecte pfc.embedded.link

Un cop verificats el requeriments, des d'Eclipse caldrà:

- Importar el projecte dintre d'Eclipse (mitjançant *File->Import->Existing Projects into Workspace*)
- En cas que no ho estigui, serà necessàri activar l'opció *Project->Build Automatically*)
- Finalment, seleccionar l'opció *Project->Clean*.

Pas 2. Construcció del projecte pfc.server.restserver

Els passos a seguir son els mateixos que els indicats pel projecte anterior. A més, per tal d'obtenir l'arxiu WAR final, caldrà triar l'opció d'Eclipse: *File->Export->Web->WAR file*, que donarà pas a un diàleg com el reflectit a la següent figura, on es triarà el projecte i la ubicació de l'arxiu de destí. Aquest arxiu està llest per ser desplegat a *Apache Tomcat*, mitjançant la seva còpia al corresponent directori `webapps` d'aquest servidor d'aplicacions.

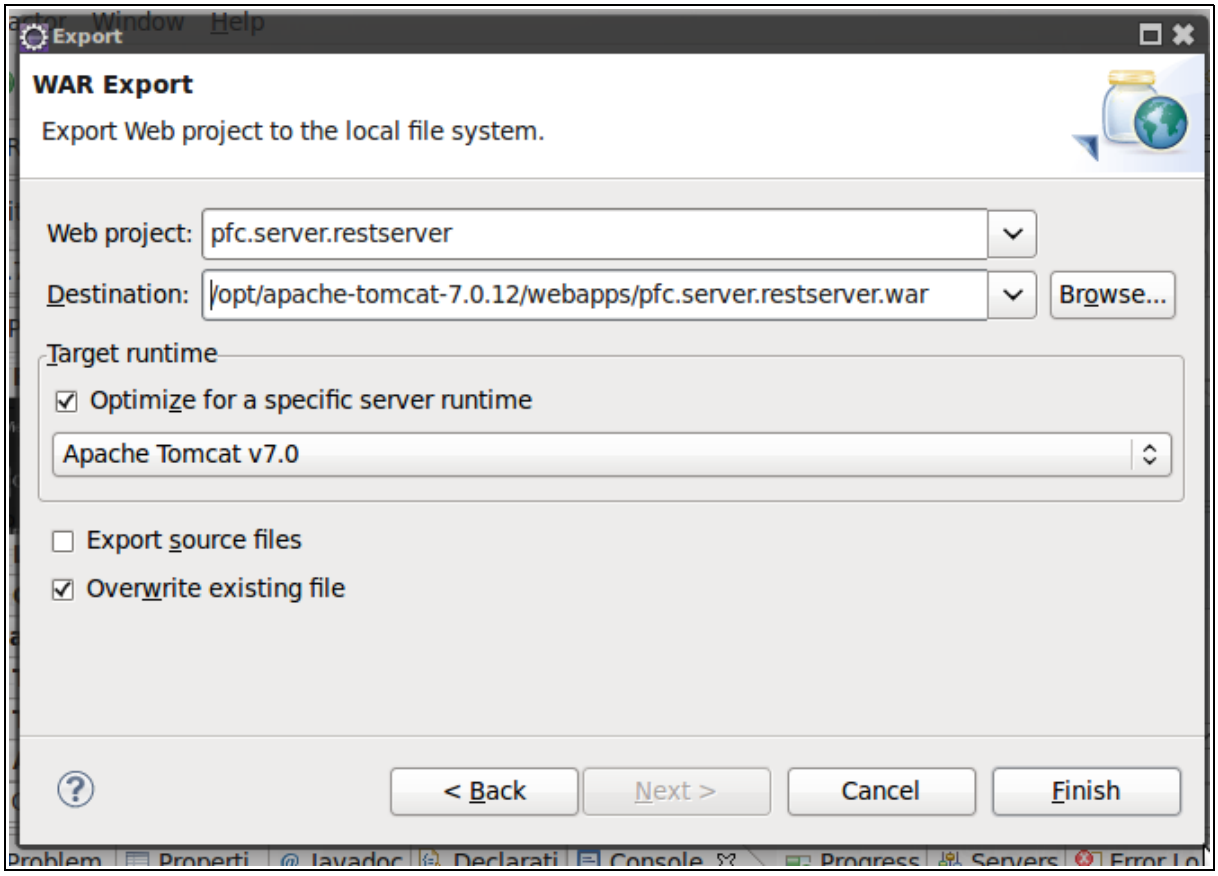

Figura 6.2: Exportació d'arxiu WAR corresponent al projecte pfc.server.restserver

6.1.4. Aplicació Client

L'aplicació client s'ha implementat en Java, mitjançant l'entorn de desenvolupament Android SDK, basat en Linux i Eclipse.

Per tal de procedir a la construcció de l'aplicació, serà necessari disposar d'un equip que tingui instal·lats i correctament configurats els següents requeriments:

- Sistema operatiu Linux (preferentment Ubuntu 10.04 o superior) o Windows.
- Entorn de desenvolupament Java JDK 1.6.x
- Entorn de desenvolupament Android SDK
- Eclipse IDE amb el corresponent plugin de desenvolupament per a Android

La construcció de l'aplicació es farà des d'Eclipse. A continuació es descriuen els passos necessaris:

- Importar el projecte `pfc.client` dintre d'Eclipse (mitjançant *File->Import->Existing Projects into Workspace*)
- En cas que no ho estigui, serà necessàri activar l'opció *Project->Build Automatically*)
- Finalment, seleccionar l'opció *Project->Clean*.

El resultat de la construcció serà l'arxiu `pfc.client.apk`, ubicat al directori

`pfc.client/bin`. Aquest arxiu està llest per ser instal·lat al dispositiu mòbil Android (a efectes del present projecte, aquest arxiu es desplegarà automàticament a l'emulador Android mitjançant l'opció *Project->Run*).

6.2. Instal·lació de la solució

El present apartat detalla els requeriments i els passos necessaris per a instal·lar els diferents components del sistema.

Tot i que poden ser construïts tal i com es detalla a l'apartat anterior, s'han inclòs com a part del lliurament del PFC els binaris corresponents als diferents mòduls del sistema. La següent figura detalla l'estructura de directoris corresponent:

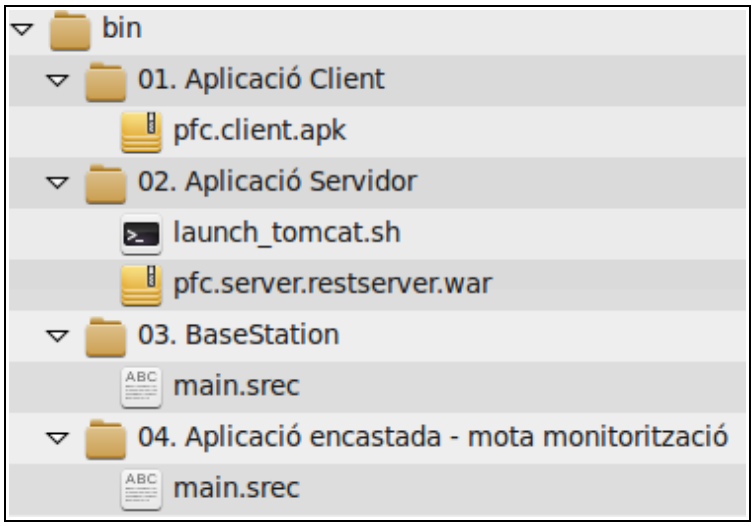

Figura 6.3: Estructura de lliurament dels arxius binaris

6.2.1. Aplicacions encastades

La instal·lació de les aplicacions encastades es realitza mitjançant la utilitat *meshprog*. Un cop connectada la mota al port USB de l'ordinador es necessari, mitjançant la línia de comandes, situar-se al directori on es troba el binari corresponent a l'aplicació construïda i executar la comanda:

```
:~$ meshprog -t/dev/ttyUSB1 -f./main.srec
```

La utilitat *meshprog* es quedarà esperant la senyal de *reset* per part de la mota (per la qual cosa cal prémer el boto de *reset* d'aquesta) i començarà la transferència del arxiu binari.

Un cop finalitzada la transferència, la mota es reiniciarà automàticament i l'aplicació tot just carregada començarà a executar-se.

6.2.2. Aplicació de Servidor

La aplicació de Servidor requereix un sistema operatiu Linux (preferentment Ubuntu 10.4 o

posterior) així com la corresponent distribució de tinyOS i un *Apache Tomcat* ubicat, idealment, a `/opt`.

S'adjunta amb el WAR un *script* útil per a iniciar *Tomcat* (serà necessari revisar els directoris referenciats a aquest *script*), donat que es necessari que les llibreries natives de *tinyOS* siguin accessibles per l'aplicació.

El procés d'instal·lació es limita, doncs, a copiar l'arxiu WAR dintre del directori `webapps` de *Tomcat* e iniciar-lo.

Serà necessari, no obstant, garantir l'accés extern al port 8080 de l'ordinador on s'ha d'executar aquesta aplicació per tal que l'Aplicació Client pugui accedir-hi, així com dotar-lo directament o mitjançant NAT d'una adreça IP pública.

6.2.3. Aplicació Client

L'aplicació Client pot ser instal·lada a un dispositiu mòbil Android físic mitjançant l'Android SDK. A efectes del present PFC, però, la instal·lació es portarà a terme sobre l'emulador de l'SDK (veure apartat 6.1.4.).

7. Manual d'usuari

El present sistema té com a doble objectiu l'augment del confort i la reducció del cost energètic a la llar gracies a la regulació automàtica de les condicions de temperatura i lluminositat. Així mateix pot ser utilitzat com un sistema eficient de detecció d'intrusions.

El sistema està composat per tres mòduls diferenciats:

- L'**Aplicació client** que s'executa sobre un dispositiu mòbil i ofereix l'usuari la possibilitat de gestionar les condicions a aplicar a cada punt de monitorització.
- El **servidor** que rep les ordres de l'aplicació client i gestiona els sensors corresponents als diferents punts de monitorització.
- Els **sensors**, situats a cada punt de monitorització, que capten les condicions actuals de temperatura, lluminositat i detecció d'intrusions i actuen sobre els reguladors externs per tal de mantenir-les constants en base als valors establerts per l'usuari.

Tant el servidor com els sensors operen de forma autònoma i no requereixen cap acció per part de l'usuari per tal de funcionar correctament.

L'aplicació client, per altra banda, representa l'únic punt d'interacció del sistema amb l'usuari. Així doncs, el present apartat incideix en la funcionalitat oferta per aquesta aplicació, així com les diferents pantalles que posa a disposició de l'usuari.

7.1.1. Inici de l'aplicació

L'aplicació client s'iniciarà de del seu accés directe al menú d'aplicacions del dispositiu mòbil, tal i com mostra la següent figura:

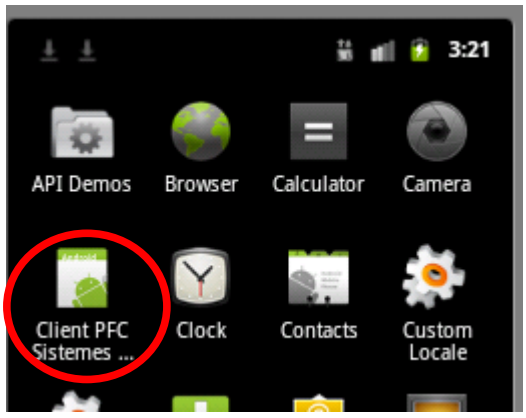

Figura 7.1: Inici de l'aplicació

7.1.2. Vista principal

Un cop iniciada, l'aplicació mostra una vista de la situació actual dels diferents punts de monitorització. Per a cadascun d'ells s'indica la temperatura i nivell de lluminositat actuals així com l'estat del detector d'intrusions.

Figura 7.2: Vista principal de l'aplicació

Per altra banda, la part superior de la pantalla indica en tot moment l'estat de la connexió amb el servidor del sistema, tal i com mostra la imatge a continuació.

 Connexió disponible amb el servidor	 Connexió no disponible amb el servidor

En cas que l'aplicació no disposi de connexió amb el servidor, no serà possible la visualització dels valors actuals de temperatura, lluminositat o detecció d'intrusions. Així mateix, tampoc serà possible l'establiment de nous valors per a aquests paràmetres.

No obstant això, tot i no disposar de connexió des de l'aplicació client, el servidor continuarà monitoritzant l'estància i ajustant les seves condicions en base als últims valors introduïts per l'usuari

Les opcions disponibles des d'aquesta pantalla son:

- **Accés a la pantalla d'introducció de valors objectiu** per a un punt de monitorització. Per tal d'accedir a aquesta pantalla únicament caldrà una pulsació sobre la fila corresponent al punt de monitorització que es vol modificar.
- **Accés a les opcions de configuració**. Per tal d'accedir a la configuració es polsarà el botó de menú del dispositiu mòbil i es triarà l'opció de configuració, representada mitjançant la icona de la següent figura:

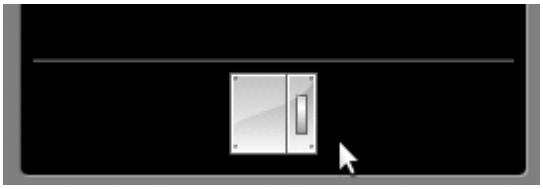

Figura 7.3: Opció de menú corresponent a la configuració

- **Sortida de l'aplicació**, mitjançant la pulsació del botó estàndard del dispositiu mòbil *Android* a tal efecte.

7.1.3. Introducció de valors objectiu

Mitjançant aquesta pantalla, es possible establir els valors desitjats de temperatura i lluminositat a una estància concreta:

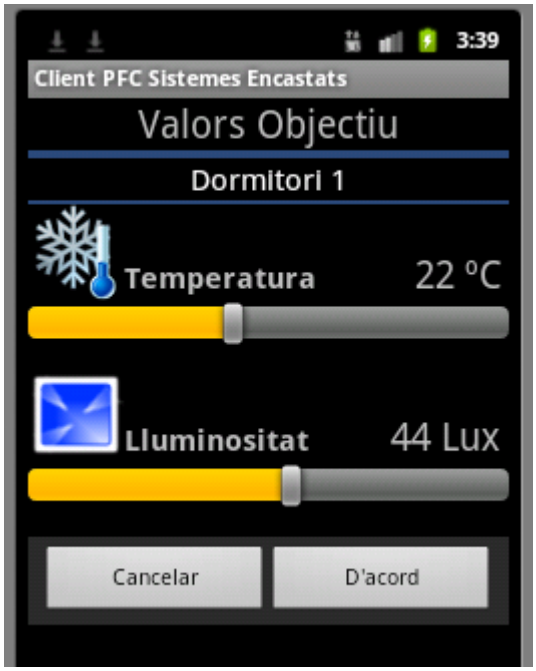

Figura 7.4: Introducció de valors objectiu per a un punt de monitorització

Un cop introduïts i validats mitjançant el botó "D'acord", aquests valors son enviats al servidor del sistema qui se n'encarregarà, mitjançant els corresponents sensors, de mantenir les condicions de temperatura i lluminositat de l'estància constants en funció d'aquests paràmetres.

El botó "Cancel·lar" permet tornar a la pantalla principal descartant qualsevol canvi fet.

7.1.4. Configuració de l'aplicació

Mitjançant aquesta pantalla, es possible establir els valors de configuració de l'aplicació. Actualment, l'únic paràmetre configurable es l'adreça on serà accessible l'Aplicació Servidor, tal i com es mostra en la següent figura:

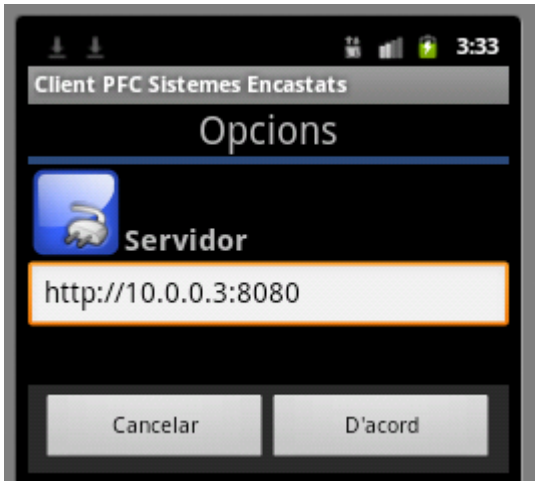

Figura 7.5: Opcions de configuració

Es necessari tenir precaució a l'hora d'establir aquest valor, donat que ha de mantenir el format indicat a la imatge. Es a dir, l'adreça haurà d'estar formada pel literal “http://”, seguit de l'adreça IP o nom DNS del servidor, i finalment el port en format “:<num_port>”.

8. Avaluació de resultats del projecte

El present capítol detalla els reptes, problemes i l'aprenentatge que ha representat el desenvolupament del PFC, i ofereix un conjunt de reflexions a mode de conclusions del projecte.

8.1. Reptes i problemes

Els principals reptes del projecte han estat el disseny i el desenvolupament de les aplicacions encastades a les motes així com la interfície amb aquestes des de l'Aplicació de Servidor. Això ha estat degut principalment a la utilització d'un entorn per a desenvolupament encastat com es *tinyOS*, que representa un paradigma de desenvolupament substancialment diferent al model tradicional.

El desenvolupament sobre les motes requereix un nivell especial d'atenció i precisió, donat que, en no disposar d'interfície d'usuari ni sistema d'arxius, les aplicacions encastades resulten molt difícils de depurar. A més, el model d'execució de *tinyOS* requereix que es tingui molt en compte la duració i càrrega de cadascuna de les tasques que s'implementen a les aplicacions. Si aquestes excedeixen un temps de computació raonable, el sistema sencer es torna inestable.

La resta d'elements del projecte no han introduït problemes significatius. Alguns d'aquests elements representen tecnologies relativament recents, com el sistema operatiu *Android*. D'altres en representen tecnologies amb més antiguitat al mercat, com el desenvolupament web sobre *Apace Tomcat*. En qualsevol cas, tots ells tenen en comú el fet que son més properes a la filosofia tradicional de desenvolupament d'aplicacions i son molt mes fàcils de depurar.

8.2. Aprenentatge

El present PFC, donada la seva naturalesa i les tecnologies amb que ha estat implementat, ha permès l'assoliment d'un alt grau d'aprenentatge en diferents àmbits.

En primer lloc, i més important, ha propiciat l'estudi i aprenentatge del concepte i filosofia de les xarxes de sensors inal·làmbrics i dels sistemes encastats en general, així com dels processos de disseny e implementació d'un sistema encastat en una aplicació particular de domòtica.

Al llarg del projecte s'ha pogut analitzar i experimentar amb l'entorn *tinyOS*, s'ha vist com explicitar un disseny genèric sobre aquesta plataforma, i s'ha aprés com desenvolupar les aplicacions encastades com a part fonamental d'una solució completa, amb la qual interactuen per portar a terme els objectius prefixats.

Així mateix, el PFC ha facilitat també l'aprenentatge d'altres conceptes i tecnologies que compten amb un alt grau de penetració al mercat. Una d'elles son el serveis REST sobre un contenidor web dinàmic com *Apache Tomcat.* Una altra es el sistema operatiu mòbil *Google Android* amb el seu entorn de desenvolupament basat en Java, mitjançant el qual s'ha desenvolupat l'aplicació client. Aplicació que, si més no, té una certa ressemblança amb un sistema encastat i representa un contrapunt significatiu respecte al programari i maquinari present a les motes.

8.3. Planificació real respecte a la inicial

Durant el decurs de la implementació del projecte, s'ha pres la decisió de deixar fora del lliurament del PFC el cas d'us UC05 relatiu a la visualització de l'estat del sensor Hall de les motes. Tot i que el cas d'us ha estat implementat a nivell d'Aplicació de Servidor i d'Aplicació Client, la implementació a l'Aplicació de la mota de monitorització estava introduint un endarreriment que posava en risc el lliurament del projecte. Així doncs, aquest cas d'us es completaria en una segona fase del projecte.

Al marge d'aquesta decisió, la resta del projecte no ha experimentat desviacions significatives respecte la planificació inicial proposada al capítol 3. Això ha estat així gràcies principalment als fets que el projecte ha sigut relativament curt en termes de duració, ha estat acotat a nivell funcional i no ha tingut dependències externes.

L'únic risc capaç de introduir potencials desviacions ha estat la incertesa tecnològica respecte a l'entorn *tinyOS* existent al moment d'establir la planificació inicial. Aquest entorn però, donat la seva maduresa i amplia documentació, no ha presentat obstacles seriosos durant el projecte.

8.4. Punts de millora

Donat que el present PFC pretén ser una prova de concepte a nivell tecnològic i funcional, son nombrosos els punts de millora que es poden identificar sobre els materials generats. Aquest punts, tenint en compte els objectius i duració del PFC, no s'han inclòs a l'abast del mateix.

En primer lloc, seria útil substituir la simulació dels reguladors de temperatura i lluminositat (realitzada mitjançant leds a la mota) per una interconnexió amb reguladors reals que actuïn sobre elements reals (una caldera, un conjunt de llums).

A nivell funcional, seria útil la implementació de determinades funcionalitats addicionals com per exemple una millor detecció de fallades al sistema, la visualització de l'estat de les bateries de les motes, la possibilitat que les motes puguin retransmetre missatges d'unes altres motes cap a l'estació base, o la possibilitat d'oferir un manteniment dels punts de control sota monitorització.

A nivell tècnic, el procés de construcció dels diferents projectes que formen el sistema podria ser millorat mitjançant la introducció d'*scripts Ant* o *Maven* que permetessin una construcció automàtica sense necessitat d'utilitzar l'entorn de desenvolupament Eclipse.

D'igual manera, la instal·lació dels components de la solució, en especial el servidor, també podria ser simplificada amb una distribució autocontinguda que incorporés a l'hora l'aplicació de servidor i el servidor *Apache Tomcat* que ha de contenir-la.

8.5. Conclusions

El concepte de xarxa de sensors inal·làmbrics ha encaixat perfectament en l'aplicació de domòtica proposada, on cal monitoritzar nombrosos punts de forma distribuïda i a la vegada autònoma.

Aquest fet ha quedat patent a les motes utilitzades durant el PFC. Gràcies a la seva configuració de maquinari, s'han mostrat molt indicades per a l'obtenció i distribució de la informació monitoritzada al temps que se'ls ha incorporat funcionalitat (en aquest cas, la regulació continua en base als valors objectiu prefixats) que els hi permet operar de forma aïllada.

L'entorn *tinyOS*, per la seva banda, s'ha mostrat com una opció molt interessant pel desenvolupament de sistemes encastats, principalment gràcies al seu model de programació i al nivell d'abstracció que aquest proporciona.

Un cop adaptada la plataforma de maquinari a *tinyOS*, els components d'aquest entorn s'encarreguen de garantir la correcta execució de les tasques de més baix nivell (com ara l'accés als diversos elements de maquinari de la mota o l'enviament/recepció de missatges) i permeten, doncs, que el desenvolupador es focalitzi en la funcionalitat concreta de l'aplicació encastada.

Finalment, l'aplicació client sobre *Android* ha acomplit els objectius inicials de simplicitat i uniformitat i s'ha revelat com un punt de interacció amb l'usuari intuïtiu i molt accessible.

En definitiva, el PFC ha posat de manifest la versatilitat de les xarxes de sensors inal·làmbrics i la seva idoneïtat per a aplicacions domòtiques, al temps que ha demostrat la potència de l'entorn *tinyOS* per al desenvolupament de sistemes encastats, i ha deixat patent que els dispositius mòbils representen un complement ideal per a aquest tipus de solucions, en oferir un punt de control únic, disponible a l'abast de la mà en qualsevol situació.

9. Bibliografia

[1] Philip Levis (2006, Juny), TinyOS Programming, http://www.tinyos.net/tinyos-2.x/doc/pdf/tinyos-programming.pdf, data d'accés 2011-03-18

[2] Sebastian A. Bachmaier (2009), UML 2.0 for modeling TinyOs components. In Proceedings of the FGSN '09 Conference, Universität Stuttgart , http://www.ti5.tu-harburg.de/events/fgsn09/proceedings/fgsn_087.pdf, data d'accés 2011-04-22

[3] Roy Thomas Fielding (2000), Architectural Styles and the Design of Network-based Software Architectures. Tesis doctoral, University of California, http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm, data d'accés 2011-04-08

[4] Partha Kuchana (2004), Software Architecture Design Patterns in Java . CRC/Auer-bachs, Publications

[5] (autor no especificat) (2009), JSR-000311 JavaTM API for RESTful Web Services, Sun Microsystems, http://jcp.org/aboutJava/communityprocess/mrel/jsr311/index.html, data d'accés 2011-04-08

[6] (autor no especificat), Jersey Home Page, Sun Microsystems, http://jersey.java.net/, data d'accés 2011-04-10

[7] (autor no especificat), JSON Home Page, http://www.json.org/, data d'accés 2011-04-10

[8] (autor no especificat), google-gson Home Page, http://code.google.com/p/google-gson/, data d'accés 2011-04-10

[9] (autor no especificat), Android Developer's Site, Google, http://developer.android.com/index.html, data d'accés 2011-04-12

[10] (autor no especificat), HttpClient Home Page, The Apache Software Foundation, http://hc.apache.org/httpcomponents-client-ga/, data d'accés 2011-04-12

[11] (autor no especificat), Embedded System. Wikipedia. http://en.wikipedia.org/wiki/Embedded_system , data d'accés 2011-05-27

[12] Matteo Cessana, Enabling Networked Sensors. Advanced Network Technologies Labo-ratory, Politecnico di Milano. http://home.dei.polimi.it/cesana/TinyOS/TinyOS.pdf, data d'accés 2011-05-29