ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA
Corso di Laurea in Ingegneria e Scienze Informatiche

PROGETTAZIONE E SVILUPPO DI UN SISTEMA DI TELEMEDICINA INTEROPERABILE

Elaborato in
SISTEMI EMBEDDED ED INTERNET OF THINGS

Relatore
Prof. ALESSANDRO RICCI

Presentata da
LEONARDO MICELLI

Corelatore
Dott. Ing. SARA MONTAGNA

Anno Accademico 2019 – 2020

"Chi non applica nuovi rimedi dev'essere pronto a nuovi mali;
perché il tempo è il più grande degli innovatori."

(Francis Bacon)

Indice

Introduzione

Negli ultimi decenni l'innovazione ha avvicinato sempre di più tecnologia informatica e medicina. Strumenti di analisi permettono diagnosi sempre più accurate, sistemi informativi consentono di gestire grandi quantità di dati relativi ai pazienti, telecomunicazioni permettono alle informazioni di circolare velocemente ed infine, di avvicinare pazienti e medici. È in questo contesto, ed in particolare in quest'ultimo frangente, che si colloca la Telemedicina.

La Telemedicina può essere considerata come l'insieme di tutte le pratiche e tecnologie informatiche che permettono di erogare un servizio medico a distanza, cioè senza necessità che chi eroga il servizio sia nello stesso luogo di chi ne usufruisce.

Secondo il Ministero della Salute italiano, le finalità della telemedicina sono:

- Prevenzione secondaria: si tratta di servizi dedicati alle categorie di persone già classificate a rischio o persone già affette da patologie croniche, le quali, pur conducendo una vita normale devono sottoporsi a costante monitoraggio di alcuni parametri vitali.

- Diagnosi: si tratta di servizi che hanno come obiettivo quello di muovere le informazioni diagnostiche anziché il paziente. La Telemedicina può costituire un completamento all'iter diagnostico o consentire approfondimenti utili al processo di diagnosi e cura, ad esempio, attraverso la possibilità di usufruire di esami diagnostici refertati dallo specialista, presso l'ambulatorio del medico di medicina generale, la farmacia, il domicilio del paziente.

- Cura: si tratta di servizi finalizzati ad operare scelte terapeutiche ed a valutare l'andamento prognostico riguardante pazienti per cui la diagnosi è ormai chiara.

- Riabilitazione: si tratta di servizi erogati presso il domicilio o altre strutture assistenziali a pazienti cui viene prescritto l'intervento riabilitativo come pazienti fragili, bambini, disabili, cronici, anziani.

- Monitoraggio: si tratta della gestione, anche nel tempo, dei parametri vitali, definendo lo scambio di dati tra il paziente in collegamento con una postazione di monitoraggio per l'interpretazione dei dati. [7]

Questa tesi si colloca, in particolare, nell'ambito del telemonitoraggio. In questo contesto, gli attori principali sono il paziente, di cui si vogliono monitorare alcuni parametri vitali, i caretakers, che utilizzano tali dati per erogare servizi sanitari, ed i dispositivi di acquisizione del dato, spesso "connessi" tramite un'applicazione o un data hub. È infatti piuttosto comune al giorno d'oggi trovare sul mercato molti dispositivi di svariate marche, che permettono l'acquisizione di dati sull'utente e, attraverso l'integrazione con applicazioni e servizi web, di condividere queste informazioni con i propri caretakers. Tuttavia questi sistemi prodotto-applicazione sono tendenzialmente chiusi nel loro ecosistema e stentano a comunicare efficientemente con altri sistemi della loro stessa natura, a causa dei diversi protocolli interni di produzione, comunicazione e rappresentazione del dato e ciò porta inevitabilmente ad un rallentamento della loro adozione nella sanità di livello nazionale.

Quindi, partendo da questa problematica, obiettivo di questa tesi sono la progettazione e lo sviluppo di una libreria software, pensata per sistemi mobile, che permetta all'applicazione che ne fa uso di interfacciarsi con vari dispositivi diversi, al fine di ottenere i dati sui parametri vitali del paziente e di esporli formattati secondo lo standard FHIR.

Nel primo capitolo di questa tesi si cercherà di delineare un quadro sull'ampio dominio della telemedicina, partendo dai suoi elementi costitutivi per arrivare alla costituzione di un quadro normativo e tecnologico di riferimento. Nel secondo capitolo si introdurrà con maggiore dettaglio il lettore al problema dell'interoperabilità ed agli standard già esistenti, mentre nel terzo si proporrà una soluzione al problema appena presentato: verrà infatti esposta la progettazione di una libreria software, pensata per dispositivi mobili, che permetta ad un'applicazione mobile di interfacciarsi con dispositivi di acquisizione provenienti da ecosistemi differenti, per ottenere dei dati riguardanti determinati parametri vitali e di ottenerli in un formato standardizzato. Infine, nel quarto capitolo, si forniranno commenti tecnici riguardo al processo di sviluppo ed implementazione della libreria proposta nel terzo capitolo, nonché di una semplice applicazione che si serve di questa libreria, mostrando come essa permetta

di collegarsi a due dispositivi differenti (in particolare, un servizio web ed un dispositivo Bluetooth) e di ottenere dei parametri vitali dell'utente secondo il formato dello standard FHIR.

Capitolo 1

Introduzione alla telemedicina

1.1 Definizione di Telemedicina

La definizione più naturale di telemedicina è di natura etimologica: deriva dalle parole tele(lontano) e medicina, si tratta infatti della prestazione e fruizione di servizi medici a distanza, senza quindi la necessità di un contatto fisico diretto tra medico e paziente. Le prime ragioni che hanno portato alla nascita ed evoluzione della telemedicina sono di natura sociale: si è reso necessario trovare un modo per fornire servizi ed assistenza sanitari ad abitanti di zone rurali e mal collegate con la rete ospedaliera, pazienti in gravi condizioni di disagio o impossibilitati a raggiungere fisicamente l'ospedale più vicino. La telemedicina è nata quindi con l'obiettivo di spostare il centro del trattamento sanitario dall'ospedale al paziente, per garantire un servizio equo e sempre "online".

1.2 Possibilità offerte dalla Telemedicina

Uno scenario di questo tipo offre notevoli possibilità di miglioramento del servizio sanitario, alcune delle quali sono:

- Equità di accesso all'assistenza sanitaria: un servizio efficace di telemedicina può fare da "ponte" tra destinatario di una prestazione sanitaria ed il suo erogatore;
- Garanzia di continuità delle cure: la telemedicina permette portare direttamente a casa del paziente il servizio del medico senza che questi si debba fisicamente recare sul luogo, ad esempio attraverso il telemonitoraggio di vari parametri vitali nell'ambito della cura di un paziente cronico;

- Maggiore efficacia ed efficienza: la telemedicina fornisce un'infrastruttura in grado di raggiungere in poco tempo e tempestivamente il paziente o il medico, in modo da migliorare la gestione della cura in situazioni d'emergenza. Il telemonitoraggio costituisce anche un grande strumento di prevenzione;

A questi vantaggi si frappongono tuttavia delle criticità delle quali è bene tenere presente se si vuole costituire un quadro informato a tutto tondo riguardo alla Telemedicina. Il rischio principale è che i servizi di telemedicina sostituiscano malamente, piuttosto che integrare e potenziare, i servizi offerti dal medico. Questo rischio può venire da entrambe le parti in gioco: il tempo risparmiato grazie alla telemedicina potrebbe portare ad un calo dell'attenzione del medico verso il paziente, tentandolo di preferire la quantità di servizi erogati a scapito della qualità. D'altra parte, il paziente potrebbe contare sempre di più sulle risorse online, vista la facilità di fruizione, preferendole al rapporto diretto col medico. Questa pratica potrebbe portare il paziente a scenari di autoreferenza e autodiagnosi, trasformandolo in un "Self Patient", ovvero paziente di se stesso. [6]

1.3 Attori

Gli attori principali in gioco in un sistema di telemedicina sono:

- Utente: fruisce del serivizio di telemedicina. Può trattarsi sia di un paziente che, nel caso di una telecomunicazione a distanza, anche di un medico;

- Caregiver: colui che eroga il servizio di telemedicina, comunicando col paziente e consultando il database contenente le informazioni critiche a riguardo;

- Sistema Informativo: si occupa di conservare i dati clinici dei pazienti, rendendoli disponibili alla consultazione dei caregivers autorizzati. Occorre porre grande attenzione alla formattazione e alla messa in sicurezza dei dati in quanto estremamente sensibili, nonchè alla sicurezza del protocollo di comunicazione attraverso il quale i dati vengono inviati a o reperiti da tale sistema informativo;

- Personal Health Device (PHD): dispositivo IoT in grado di raccogliere misurazioni di parametri vitali specifici riguardanti il trattamento in corso. I dati raccolti da questi dispositivi andranno a popolare il database;

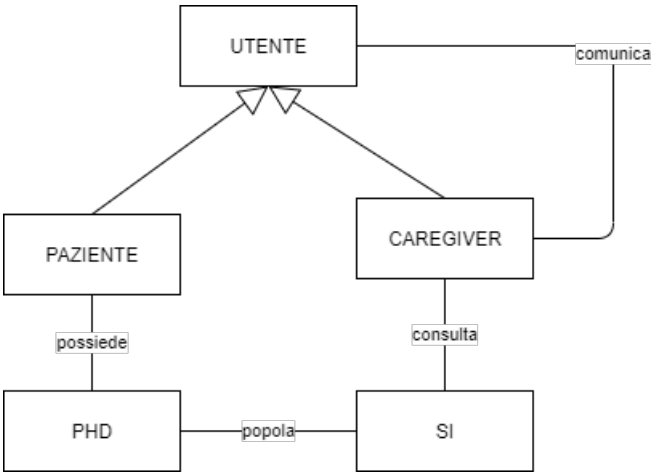


Figura 1.1: Schema degli attori in gioco

1.4 Tassonomie

Possiamo classificare un sistema di telemedicina sulla base di vari criteri:

1.4.1 Modalità

Distinguiamo i sistemi di telemedicina sulla base delle modalità in cui viene erogato il servizio:

- Sincrono: richiede che sia Utente che Caregiver siano attivi, implica una comunicazione real-time tra i due attori in gioco attraverso una rete di telecomunicazione (audio-video);

- Asincrono: implica la raccolta di dati sanitari riguardanti il paziente e la successiva popolazione di un database con tali dati. Il Caregiver potrà consultare tali dati direttamente dal database quando lo riterrà opportuno, ciò può accadere in differita rispetto all'immissione dei dati nel database. La popolazione del database può avvenire in due modi: può essere il paziente parte attiva di tale immissione, oppure può essere un dispositivo IoT in grado di comunicare col database, implicando la passività del paziente;

1.4.2 Luogo

Possiamo contraddistinguere i sistemi di telemedicina anche attraverso il luogo in cui avviene erogato il serivizio:

- Ubiquo: identifica un servizio disponibile in qualsiasi momento ed in qualsiasi luogo. Può indicare sia servizi di consulto di emergenza (quindi sincroni) che di telemonitoraggio (asincroni);
- At-Home: identifica un servizio disponibile presso l'abitazione del paziente, spesso erogati attraverso dispositivi progettati per essere usati a casa dal paziente autonomamente;

1.4.3 Approccio

Ci sono diversi approcci al benessere che beneficiano delle possibilità offerte dalla telemedicina, come ad esempio:

- Benessere Proattivo: implica il monitoraggio dei parametri di un soggetto in salute al fine di promuovere uno stile di vita sano. Questo approccio sposta il focus del trattamento dal "curare" al "prendersi cura";
- Benessere Reattivo: implica il monitoraggio remoto dei parametri vitali di un paziente in stato di cura, in modo da reagire tempestivamente alle situazioni critiche;
- Benessere Ibrido: implica il monitoraggio di un soggetto predisposto allo sviluppo di determinate patologie in modo da prevenirne o arginarne lo sviluppo tramite il continuo controllo medico; [4]

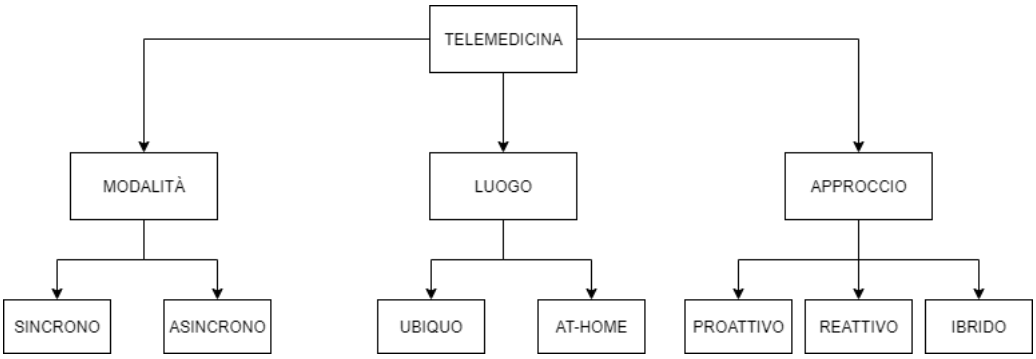


Figura 1.2: Tassonomia completa della telemedicina

1.5 Quadro normativo di riferimento

La telemedicina, dal punto di vista normativo, non è una disciplina medica separata, bensì uno strumento atto ad estendere le possibilità della pratica medica tradizionale. In questo senso si configura come diverso modo di erogare una particolare prestazione socio-sanitaria.

Le fonti di riferimento che compongono il quadro normativo ad oggi sono sviluppate su due ambiti: nazionale ed internazionale. A livello nazionale, le principali norme in materia sono:

- Decreto Legislativo N. 502 del 30.12.1992: "riordino della disciplina in materia sanitaria", dove viene regolamentata l'erogazione dei servizi di telemedicina; [7]
- "Telemedicina: Linee di indirizzo" approvate dall'Assemblea Generale del Consiglio Superiore di Sanità il 10.07.2012;
- "Indicazioni nazionali per l'erogazione di prestazioni in telemedicina", approvate dalla Conferenza permanente per i rapporti tra Stato, Regioni e Province Autonome, Rif. Ministero della Salute del 17.11.2020.

A livello comunitario europeo, la norma più importante è la comunicazione COM(2008) 689 "Telemedicina a beneficio dei pazienti, dei sistemi sanitari e della società", finalizzata a sostenere gli Stati membri nella realizzazione di iniziative di telemedicina.

1.6 Quadro tecnologico di riferimento

1.6.1 Tecnologie abilitanti la telemedicina

Sono le tecnologie alla base delle quali si fondano le tecnologie specifiche della telemedicina, o delle quali viene fatto uso estensivo in contesti applicativi della telemedicina. Queste tecnologie possono essere di vario tipo:

- Reti: reti atte a trasmettere i dati, permettendo di spostare l'informazione piuttosto che il paziente. Nel contesto della telemedicina vengono attivamente impiegati vari tipi di rete, dalle reti cablate e wireless come ADSL e fibra ottica, alle Body Area Network come Zigbee e Bluetooth, alle reti satellitari come LTE e, più recentemente, 5G.
- Standard: hanno il compito di standardizzare il maggior numero di aspetti del dominio della telemedicina possibili, dai protocolli di comunicazione (HTTPS), codifica dei dati (FHIR, DICOM) alla produzione di dispositivi hardware (ISO/IEEE). L'adozione di standard permette ad un sistema di telemedicina di essere interoperabile con altri processi che sono in atto nell'ambiente in cui sono immersi.
- Software: in questo contesto il software costituisce il mezzo attraverso il quale avvengono le interazioni tra paziente, dispositivi e medici. Oltre a

specifici applicativi mediante i quali vengono erogati i servizi di telemedicina, ricoprono un ruolo importante anche i software per la firma digitale e gli applicativi per l'invio e ricezione di posta elettronica certificata.

- Hardware: i dispositivi coinvolti in un tipico scenario di telemedicina sono molteplici e variano da computer resilienti (con parti ridondanti in modo da garantire massima affidabilità in situazioni di disagio), periferiche multimediali come le webcam a dispositivi di acquisizione, dotati cioè di sensori in grado di rilevare un parametro vitale del paziente.

1.6.2 Esplorazione dei dispositivi di acquisizione

Come anticipato nella precedente sotto sezione, la scelta di dispositivi di acquisizione dei dati è un aspetto fondamentale sul quale costruire un sistema di telemedicina robusto. In questa sotto sezione, si vogliono esplorare varie soluzioni di telemedicina presenti sul mercato, in modo da individuare quelle che si prestano meglio ad essere utilizzate per sviluppare progetti di telemedicina. Ai fini dell'esplorazione, si definirà soluzione di telemedicina un'offerta, da parte di un "vendor" di rilevanza nel mercato odierno, costituita da:

- un ecosistema di PHD (costituito eventualmente da un solo dispositivo), in cui ciascun device è in grado di misurare una specifica metrica in un paziente. Si pone come vincolo la presenza di almeno un PHD nell'ecosistema, in grado di monitorare almeno una delle seguenti metriche fondamentali: frequenza cardiaca, temperatura corporea, pressione sanguigna e livelli di O2 presenti nel sangue. Non è richiesta la passività del paziente circa l'utilizzo dei PHD, tuttavia è apprezzata negli scenari di telemonitoraggio;

- un software che permetta all'utente di interfacciarsi e gestire l'ecosistema. Ciò presuppone che i PHD siano connessi, singolarmente o eventualmente tramite un Hub centrale che faccia da coordinatore generale dei dispositivi;

- una API che permetta l'integrazione dell'ecosistema con servizi di terze parti, caratteristica cruciale per garantire l'interoperabilità tra sistemi di vendor differenti.

Ricordiamo ai fini dell'esplorazione che, in questa sede, si definisce PHD un dispositivo IoT in grado di effettuare misurazioni, che è posseduto o utilizzato dal paziente presso la propria abitazione (Telemedicina At-Home) od indossato dal paziente durante la giornata (Telemedicina Ubiqua). Si escludono pertanto dall'esplorazione devices ospedalieri con capacità IoT.

Tra le varie soluzioni presenti sul mercato, si sono distinte tre soluzioni proposte da tre diverse aziende: Withings, Omron e Fitbit.

Withings

Withings è una società francese, con sede ad Issy-les-Moulineaux , in Francia ed è conosciuta nel settore dell'elettronica come la prima società a mettere sul mercato un dispositivo connesso, legato alla salute: la prima bilancia Wi-Fi, messa nel mercato nel 2009. Da quel momento, Withings ha ampliato la gamma di dispositivi smart, arrivando a costituire un ecosistema molto ricco, comprendente smartwatches, bilance bioimpedenziometriche, cuscini per la rilevazione del battito cardiaco e dei movimenti durante il sonno, monitor per la pressione sanguigna e termometri. Tutti i dispositivi sopracitati sono connessi con tecnologia WiFi ed integrati dalla Withings Health Mate App, dalla quale è possibile accedere a tutte le metriche direttamente su smartphone.

Withings offre al professionista il pacchetto Med Pro, consistente in un ecosistema personalizzato di dispositivi Withings sopracitati, da consegnare al paziente, per permettere al personale sanitario professionista di controllare remotamente i parametri d'interesse.

Lato integrazione, Withings mette a disposizione una "Device API" pubblica, che permette di effettuare richieste autenticate ad un server per ottenere le metriche contenute nella Health Mate App, ed una "Advanced API", disponibile dietro stipula di un contratto commerciale, che comprende un SDK per sviluppare programmi per i dispositivi Withings ed accedere a servizi di dropshipping.

L'uso delle API, nella loro forma più semplice, consiste nella comunicazione HTTP con un server REST, scambiando eventuali dati sotto forma di oggetti JSON. L'autenticazione avviene seguendo il workflow stabilito dal protocollo Oauth2.0

Omron

Omron è un'azienda giapponese, attiva nel settore dei componenti per l'automazione industriale, nei prodotti elettromedicali e nei componenti per l'elettronica. La sua divisione in ambito sanitario è chiamata Omron Healthcare. Anche Omron mette a disposizione un vero e proprio ecosistema di prodotti:

monitor per la pressione sanguigna (degno di nota lo "smartwatch" Heart Guide), bilance digitali, rilevatori ECG, apparecchi terapeutici per il rilascio del dolore e dello stress muscolare.

Omron mette a disposizione del medico che vuole effettuare controlli a distanza, il servizio Vital Sight, un servizio di convenzione con Omron che permette al medico di recapitare direttamente all'abitazione del paziente una serie di prodotti connessi Omron, tra cui:

1. Monitor pressione sanguigna
2. Bilancia digitale
3. Data Hub

Il paziente può in questo modo effettuare misurazioni regolarmente e da casa, ed i dati vengono registrati direttamente nel proprio fascicolo elettronico (EMR), in modo che il medico possa agire proattivamente in situazioni critiche o preventivamente.

Lato connettività, i dispositivi Omron sono tutti connettibili tramite bluetooth e sincronizzabili con lo smartphone tramite un'app dedicata ad ogni tipo di prodotto.

Fitbit

Fitbit Inc. è una società americana con sede a San Francisco, California. È conosciuta nel mondo come uno dei brand guida nell'ambito del fitness tracking. All'utente Fitbit offre tre tipi di prodotto: smartwatch, fitness tracker e bilance digitali.

Essendo Fitbit più propriamente un marchio di fitness piuttosto che sanitario, i dispositivi non hanno una validazione clinica. È stata tuttavia presentata come soluzione in questa sede data la grande tendenza alla self-quantification che si sta sviluppando recentemente, di cui Fitbit spicca come un marchio guida, grazie anche alla grande propensione della compagnia all'integrazione e al rendere disponibili e pubbliche le API a sviluppatori terzi, permettendo, qualora lo si desiderasse, di progettare applicativi di telemonitoraggio appoggiati ai servizi Fitbit.

Fitbit mette a disposizione dello sviluppatore una serie di strumenti per lo sviluppo di applicazioni partner per i propri dispositivi. L'intera offerta dell'API è pubblica e richiede solamente la registrazione di un account sviluppatore, nonchè la registrazione di ogni applicazione sviluppata come applicazione partner .

Il primo è il Fitbit SDK, che permette di sviluppare applicazioni per Fitbit OS. Questo kit comprende due componenti principali:

1. Fitbit Studio: il web IDE ufficiale di Fitbit, permette di creare ed eseguire progetti, sincronizzandosi col dispositivo Fitbit o con il simulatore

2. Fitbit OS Simulator: simulatore di un dispositivo Fitbit, è possibile scegliere tra tutti gli smartwatch disponibili all'acquisto, emulandone i sensori ed il comportamento, in modo da testare la propria applicazione senza la necessità di possedere un prodotto fisico.

Il secondo strumento a disposizione dello sviluppatore è la Web API, che permette di accedere alle metriche raccolte per un certo utente, tramite richieste al web server di Fitbit, previo consenso dell'utente target. L'accesso alla Web API è autenticato tramite protocollo Oauth2.0: all'atto della registrazione dell'applicazione su dev.fitbit.com, è richiesto di specificare quale tipologia di applicazione Oauth si desidera registrare tra server, client o personal e di implementare il protocollo oauth corrispondente. Anche in questo caso, lo scambio di informazioni avviene attraverso protocollo HTTP e le informazioni sono rappresentate attraverso oggetti JSON incorporati nel corpo della risposta del server.

1.6.3 Conclusioni riguardo all'esplorazione

Dalla panoramica appena effettuata possiamo trarre alcune conclusioni che aiuteranno a delineare un quadro informato della situazione attuale per quanto riguarda la telemedicina. La prima è che c'è una forte propensione ad offrire un servizio sempre più ubiquo, difatti tutte e tre le soluzioni precedentemente citate offrono almeno un dispositivo wearable.

In secondo luogo, si può registrare una certa tendenza alla chiusura da parte di questi sistemi: benchè siano tutti e tre tecnicamente integrabili attraverso API, non tutte sono open-source o ben documentate. Questo crea una dinamica di disincentivazione all'integrazione con sistemi terzi in favore dell'adozione completa alla soluzione prescelta. Ciò comporta la costituzione di uno scenario

in cui sono sono presenti vari "silos" di informazioni mediche che stentano a comunicare tra loro in maniera ottimale, arginando sensibilmente, come si discuterà nel prossimo capitolo, lo sviluppo e l'adozione della telemedicina come elemento cardine di un sistema sanitario nazionale.

Tuttavia, attraverso varie sperimentazioni delle API offerte al pubblico, si può trovare in Fitbit una buona propensione all'apertura, il che lo rende un buon candidato PHD per quanto riguarda successive sperimentazioni.

Capitolo 2

Il problema dell'interoperabilità

Come emerso dal quadro sviluppato nel primo capitolo, la telemedicina gioca un ruolo fondamentale nella costituzione di un sistema sanitario moderno ed efficiente, ma il potenziale di questo quadro viene a scontrarsi con alcune criticità, in primis, una mancata assunzione della governance dello sviluppo di queste tecnologie da parte dei servizi nazionali di alcuni paesi, che ha causato la proliferazione di iniziative di telemedicina alquanto isolate tra loro [3], sia in termini di progettazione dei dispositivi hardware (PHD) che delle infrastrutture software di gestione dei dati. Questo quadro attuale pone un notevole freno allo sviluppo della telemedicina, in un periodo storico che invece ha evidenziato l'importanza centrale dell'esigenza della gestione di determinate crisi sanitarie. Da quanto detto emerge l'importanza di disporre di un modello di governance condivisa delle iniziative di telemedicina, liberandosi del più grande ostacolo allo sviluppo del settore: la mancata interoperabilità tra le varie soluzioni di telemedicina presenti sul mercato. Questo obiettivo può essere raggiunto attraverso l'adozione di standard robusti ed accreditati, che dettino le linee guida sulla progettazione di hardware e software interoperabili ed integrabili tra loro indipendentemente dalle specifiche interne, ma stabilendo un'interfaccia di comunicazione comune. Con questo obiettivo sono stati sviluppati 3 degli standard prevalenti in materia, che ci apprestiamo ad esplorare e confrontare.

2.1 ISO/IEEE 11073 Personal Health Devices (PHD)

L'ISO/IEEE 11073 PHD è uno standard che si pone l'obiettivo di risolvere le problematiche dell'interoperabilità tra device elettromedicali, con forte focus sui dispositivi ad uso personale del paziente, piuttosto che a quelli ospedalieri.

L'architettura dello standard è la seguente: vi sono due moduli che costituiscono il framework generale (IEEE 11073-2061 e -2061a), e una ulteriore specifica per ogni tipologia di PHD sul mercato, denominate "Device Specialization" (ad es. IEEE 11073-10404 – Device Specialization – Pulse Oximeter).

Lo standard propone la seguente modellazione del dominio di interoperabilità:

- i PHD (cioè devices in grado di raccogliere un set di informazioni circa le metriche di salute del paziente) vengono denominati agenti;
- qualsiasi altro sistema esterno al quale sono destinati i dati raccolti dagli agenti viene definito manager.

Gli agenti vengono decomposti, attraverso modellazione ad oggetti, in unità fondamentali denominate Oggetti, ognuno dei quali rappresenta uno specifico sensore in grado di misurare una determinata metrica. Gli oggetti sono costituiti da una serie di attributi di rilevanza, come ad esempio tipologia e valore assunti da una certa misurazione e rappresentano l'unità base di modellazione di qualsiasi informazione che l'agente è in grado di comunicare al manager.

Lo standard propone anche un "Service Model" che definisce i meccanismi concettuali che regolano lo scambio di dati. Fanno parte del Service Model le seguenti specifiche:

- Upper Layer Frame protocol: stabilisce la struttura dei messaggi, in particolare separando i comandi relativi alla gestione della connessione con quelli relativi agli oggetti (e quindi ai dati);
- Association Service: gestisce l'associazione tra agente e manager, che avviene secondo lo scambio dei seguenti messaggi: Association Request ed Association Response stabiliscono la connessione, mentre Release Request e Release Response servono per terminarla correttamente;
- Object Access Service: comprende tutti i metodi atti a reperire dati riguardanti le specifiche del dispositivo (GET) o delle metriche misurate (EVENT REPORT);

Generalmente la topologia che ci aspettiamo è una serie di agenti che comunicano attraverso connessioni punto a punto con un manager, attraverso delle personal area network (USB, Bluetooth, ZigBee, ecc.). [2]

2.2 HL7 (Health Level Seven)

HL7 è il nome dell'organizzazione internazionale che si occupa di mantenere e gestire l'omonimo linguaggio standard per la gestione e lo scambio di messaggi tra attori in ambito medico. Il suo nome è dovuto al settimo livello dello standard ISO/OSI: quello applicativo.

Esistono varie versioni del linguagio HL7: ad oggi sono in uso le versioni 2.x e 3.x:

2.2.1 HL7 2.x

La specifica HL7 2 è message-based (la comunicazione avviene quindi attraverso messaggi standardizzati) ed adotta una codifica ASCII, che fornisce una certa leggibilità da parte dell'essere umano. La struttura dei messaggi è la seguente:

- ogni messaggio è composto da vari segmenti. Ogni messaggio comincia sempre con il segmento MSH che costituisce l'header del messaggio stesso. Ogni segmento rappresenta una determinata tipologia di comunicazione ed è identificato da un codice di tre lettere e sono separati tra loro in diverse righe (ad esempio MSH è l'header del messaggio, MDM rappresenta la gestione di documentazione medica, ORU rappresenta l'osservazione di una determinata metrica relativa ad un paziente);

- i segmenti sono formati da una sequenza di campi, che contengono le informazioni vere e proprie, solitamente codificati attraverso varie specifiche standard come LOINC. I campi sono separati tra loro da pipes "|" e possono essere ulteriormente suddivisi in sotto campi;

2.2.2 Z-Segment

HL7v2 prevede in totale 140 tipi di messaggio standard, tuttavia, la specifica prevede anche la possibilità di definire dei tipi di messaggio "locali", cioè che hanno un significato specifico alla particolare implementazione in cui vengono impiegati e non sono, pertanto, standardizzati. Per distinguerli dai messaggi standard, HL7 impone che l'iniziale di questi messaggi sia Z. Questo meccanismo permette una notevole flessibilità, a costo di una minore standardizzazione della specifica nel suo complesso: ciò può causare problemi in scenari di interoperabilità.

2.2.3 HL7 3.x

Nel 2010 HL7 ha sviluppato insieme al CEN (Comitato Europeo di Normazione) un framework Object-Oriented per lo sviluppo e la progettazione dei messaggi, utilizzando un modello di riferimento chiamato RIM (Reference Information Model). Il RIM definisce concetti base del dominio come classi (ad esempio Persona o Partecipazione) e template dei messaggi. È sulla base del RIM che è stata costituita la specifica HL7 v3.x.

Questa specifica è considerata la più matura ed affidabile e si differenzia rispetto alla v2 nella codifica adottata: infatti viene preferito XML ad ASCII, e nell'assenza di Z-Segments che indeboliscono lo standard in scenari di interoperabilità. [6]

2.3 IHE (Integrating Healthcare Enterprise

IHE è un'iniziativa internazionale che promuove l'uso coordinato di standard di riferimento, sia per quanto riguarda il lato sanitario che per quello IT, per risolvere varie problematiche di ottimizzazione del trattamento di un paziente. IHE è organizzato in domini clinici ed operazionali, come ad esempio Radiologia, Cardiologia, infrastruttura IT. Ad ogni dominio è assegnato un comitato pianificatore, il cui compito è quello di rilevare i principali problemi di natura interoperazionale, ed un comitato tecnico chiamato a sviluppare e documentare standard risolutivi.

Le soluzioni di interoperabilità proposte da IHE sono documentate in guide chiamate profili. I profili sono specifiche implementabili che descrivono come utilizzare standard già stabiliti e mantenuti da organizzazioni di sviluppo degli standard, come ISO/IEEE, HL7 e W3C, in modo da risolvere i problemi evidenziati in fase di analisi. I profili descrivono in particolare il comportamento degli attori, cioè tutti i sistemi informativi, o parti di essi, che producono o gestiscono informazioni sanitarie. Gli attori scambiano informazioni attraverso delle transazioni, le quali modalità sono definite dagli standard. Gli attori si basano su una profonda comprensione dei dati che vengono scambiati tra loro, infatti alcuni attori sono fortemente orientati alla definizione e modellazione dei dati. Le descrizioni di dati complessi o di grandi dimensioni si trovano nei cosiddetti Content Modules.

Ogni dominio viene incorporato in un "IHE Technical Framework", che propone le soluzioni standardizzate sopra citate. Ogni Technical Framework è costituito da vari volumi:

1. il primo volume fornisce una panoramica di alto livello di ogni profilo, i casi d'uso che va a migliorare, gli attori coinvolti e le transazioni tra di essi;
2. il secondo volume fornisce una dettagliata descrizione delle transazioni;
3. il terzo volume fornisce la specifica dei Content Modules;
4. il quarto volume descrive le estensioni fatte al framework ad opera di istituzioni nazionali. [9]

2.4 HL7 FHIR

2.4.1 Introduzione

FHIR è una nuova specifica HL7. Può essere usata come uno standard di scambio di dati "stand-alone" od in concomitanza con altri standard HL7 già diffusi. L'obiettivo principale di FHIR è quello di semplificare l' implementazione, senza sacrificare l'integrità dell'informazione.

2.4.2 Componenti

L'elemento costitutivo fondamentale di FHIR è la Risorsa (Resource). Tutte le informazioni che possono essere scambiate sono definite come risorse. Ogni risorsa possiede un set di caratteristiche comuni:

- una maniera comune di definirle, rappresentarle e costruirle a partire da Tipi che definiscono pattern di elementi riusabili;
- un set comune di metadati;
- una parte leggibile dall'operatore umano;

Di seguito la struttura di una risorsa rappresentata in formato JSON:

```
{
  "resourceType" : "[Resource Type]",
  // from Source: property0
  "property1" : "<[primitive]>", // short description
  "property2" : { [Data Type] }, // short description
  "property3" : { // Short Description
    "propertyA" : { CodeableConcept }, // Short Description (Example)
  },
```

```
  "property4" : [{ // Short Description
    "propertyB" : { Reference(ResourceType) } // R!  Short Description
  }]
}
```

È bene notare che siccome FHIR come specifica si pone l'obiettivo di essere il più generale possibile, in una determinata risorsa pochi attributi sono realmente obbligatori al fine di poter essere definita una risorsa FHIR "compliant": in questo senso, FHIR offre generalmente una notevole varietà di attributi che possano descrivere vari aspetti di una risorsa, riducendo però al minimo quelli obbligatori, spesso necessari e comuni a tutte le implementazioni e scenari d'uso di una certa risorsa. Non è quindi inusuale trovare risorse FHIR il cui unico attributo obbligatorio il campo "resourceType" che indica il tipo di risorsa che si sta modellando. Questa scelta è particolarmente vantaggiosa negli scenari, spesso comuni, in cui l'informazione è inizialmente incompleta o mancante di certi attributi e solo successivamente completata (come ad esempio nel caso della medicina d'urgenza).

Nonostante questa scelta, FHIR propone per molte risorse dei "Profili", cioè un set predeterminato di attributi consigliati per modellare efficacemente la risorsa in questione in base al particolare dominio applicativo nella quale è utilizzata. Inoltre, vengono proposte anche varie estensioni delle risorse, per poterle meglio adattare al proprio dominio applicativo.

2.4.3 Modellazione

La filosofia di FHIR circa l'approccio alla modellazione dell'informazione, è il costruire un set di risorse fondamentali che, da sole o in combinazione, permettano di soddisfare la maggior parte dei casi d'uso dello standard. Per cui, la modellazione FHIR segue un approccio a composizione: infatti, attraverso FHIR, ogni caso d'uso viene soddisfatto attraverso uso di risorse singole o combinate mediante il meccanismo del Riferimento. È inoltre presente un meccanismo di estensione per coprire i casi d'uso che non verrebbero soddisfatti attraverso la sola composizione, qualora ve ne fossero.

References

Come già specificato, la modellazione FHIR avviene tramite un approccio a "composizione" realizzata attraverso riferimenti tra risorse: ovvero che molti elementi definiti all'interno di una risorsa sono in realtà riferimenti ad altre

risorse. I riferimenti sono sempre definiti e rappresentati in un'unica direzione sorgente (referente)-⟩bersaglio(riferito)(A-⟩B): anche se è vera logicamente la relazione inversa (B-⟩A), essa tende a non essere specificata nella risorsa bersaglio.

I riferimenti vengono creati in base all'identificatore di una risorsa, ci sono diversi identificatori possibili a cui fare riferimento: URL canonici, "Business Identifiers" cioè identificatori del concetto reale piuttosto che del data record che lo sta modellando, permettendo di riconoscere il medesimo contenuto su sistemi diversi.

Le risorse contengono quindi due tipi di riferimento:

- Riferimento a risorsa: generico riferimento tra risorse
- Riferimento canonico: riferimento a risorse tramire il loro URL canonico (indirizzo che la identifica tra tutti i contesti di uso).

Ogni riferimento possiede i seguenti campi:

- riferimento: il riferimento letterale
- identificatore: riferimento logico
- display: descrizione testuale del bersaglio
- tipo: il tipo della risorsa bersaglio

Estensione

La specifica FHIR è basata sulla definizione di una serie di requisiti che deve soddisfare una risorsa (in termini di attributi), comunemente ritenuti importanti nell'industria dell'healthcare. Tuttavia, non è del tutto inusuale che una certa implementazione abbia dei requisiti che non sono tra quelli specificati da FHIR e tuttavia sono perfettamente validi nel contesto in cui vengono richiesti. Siccome l'aggiunta alla specifica di ogni possibile valido requisito renderebbe l'implementazione dello standard difficoltosa e poco pratica, FHIR permette di aggiungere requisiti validi ad una risorsa attraverso il meccanismo dell'estensione, che consente agli elementi di una risorsa di avere elementi di tipo "estensione", che aggiungono attributi estendendo in questo senso le caratteristiche base della risorsa stessa. [1]

Nel capitolo 3 di questa tesi verrà fornito un esempio di modellazione FHIR di un possibile scenario di telemedicina.

2.5 Un confronto tra gli standard in uso ad oggi

A seguito dell'esplorazione svolta, si può apprezzare come il problema dell'interoperabilità sia un problema del quale c'è una coscienza generale, e varie soluzioni vengono proposte per risolverlo. Posto che sia di importanza vitale adottare un approccio standardizzato risolutivo, è bene essere a conoscenza delle differenze tra i vari standard analizzati, sia in termini di possibilità offerte che dei casi d'uso per i quali sono stati pensati.

Per quanto riguarda ISO/IEEE 11073, lo standard utilizza un approccio ad oggetti per progettare l'architettura degli agenti, ed un modello di comunicazione con il manager di tipo punto a punto orientato alla connessione. Questo approccio garantisce flessibilità circa la modellazione dell'ecosistema di sensori a disposizione del paziente per il suo monitoraggio.

L'approccio di IHE è ad un livello di astrazione maggiore, in quanto è presente la nozione di dominio (che solitamente equivale ad uno specifico dipartimento ospedaliero) che funge da contenitore per la progettazione dell'intero sistema informativo, comprensivo di attori produttori di informazioni e di attori consumatori di tali informazioni.

L'approccio di FHIR è quello che più di tutti offre flessibilità nella modellazione del dato, attraverso un approccio a composizione a partire da "mattoni di base" costituiti dalle risorse. Questo tipo di approccio favorisce il riuso di informazione e modularità in fase di progettazione. Le risorse vengono poi scambiate attraverso servizi web REST.

Anche dal punto di vista prestazionale si evidenziano differenze tra questi tre standard. Uno studio del 2018 condotto dalla School of Computer Science and Engineering, Kyungpook National University, Korea [8], ha confrontato i tre standard in uno scenario comune, di misurazione di pressione sanguigna e temperatura corporea. Per quanto riguarda lo standard ISO/IEEE 11073 sono state implementate le device specialization corrispondenti ai due dispositivi. I messaggi scambiati tra questi due agenti ed il manager esterno sono stati codificati in MDER, rappresentati da uno stream di byte esadecimali. Per lo

standard IHE sono stati implementati il Patient Care Device (PCD) per i dispositivi ed il Device Observation Consumer (DEC) per il sistema esterno. Le transazioni sono definite dal DEC e vengono trasportate attraverso un Simple Object Access Protocol (SOAP). Il formato dei messaggi è derivato dallo standard HL7 v2.6, che fa uso di codifica Unicode. Per FHIR sono state utilizzate le risorse Device, Patient ed Observation, accorpate nella risorsa Bundle, gestite da un servizio web REST.

A seguito della sperimentazione, lo studio ha rilevato che le dimensioni dei messaggi nello standard ISO/IEEE 11073 hanno una dimensione fissa dipendente solo dal tipo di misurazione: 202 bytes per la pressione e 182 bytes per la temperatura. I messaggi IHE raggiungono una dimensione di 1143 bytes per la pressione e 733 bytes per la temperatura. Infine, i messaggi FHIR hanno una dimensione di 3237 bytes per la pressione e 1864 bytes per la temperatura. Tuttavia, se il server FHIR contiene già i dati di Device e Patient (vero dal secondo post di informazioni), il protocollo FHIR permette di inviare una risorsa Bundle che contiene solamente i riferimenti (tramite URL nel server) delle risorse Device e Patient, contenendo di fatto solamente le nuove informazioni circa l'ultima misurazione (risorsa Observation). Questo permette di ridurre la dimensione della misurazione della pressione a 2233 bytes e quella della temperatura a 628 bytes.

Un'altro parametro di confronto importante tra i vari standard è il contenuto dei messaggi scambiati. La codifica MDER scelta dallo standard ISO/IEEE 11073 è la più performante in termini di dimensione, ma perde leggibilità da parte di un possibile sviluppatore o agente esterno. La codifica Unicode scelta da IHE risulta più leggibile ad un agente umano, mentre la codifica di FHIR, basata su XML e JSON, è quella maggiormente leggibile. In conclusione, possiamo delineare diversi scenari che permettono di sfruttare i vantaggi offerti da ciascuno standard: lo standard ISO/IEEE 11073 in particolare si presta ad essere utilizzato quando si hanno a disposizione dispositivi dalla scarsa capacità computazionale, o in scenari in cui la modellazione del concetto di paziente non è richiesta. I messaggi inoltre sono trasportati attraverso una PAN, quindi il sistema di comunicazione del dispositivo risulta più leggero ed economico.

I messaggi di IHE contengono informazione sul paziente e sono relativamente più piccoli rispetto a quelli di FHIR. Inoltre possono essere più facilmente convertiti in messaggi ISO/IEEE 11073, pur mantenendo una certa leggibilità grazie alla codifica Unicode. Inoltre i messaggi IHE possono essere trasportati sia su reti ad ampio raggio (WAN) che su PAN, rendendo lo standard IHE particolarmente flessibile in vari scenari di comunicazione.

Infine, i messaggi di FHIR sono i più grandi, rendendo necessario il trasporto attraverso WAN. Tuttavia permettono un buon livello di riutilizzo delle risorse e sono quelli in grado di modellare la più grande varietà di informazioni. Inoltre, la codifica tramite XML o JSON rende le informazioni dei messaggi, previa semplice manipolazione dei dati, particolarmente adatte alla presentazione ad utenti finali. [8]

2.6 Una nota su LOINC

LOINC è uno standard di nomenclatura che si pone l'obiettivo di nominare, identificare e codificare misurazioni ed osservazioni mediche di laboratorio, assegnando un codice differente ad ogni osservazione con valore clinico diverso. Per farlo LOINC individua sei dimensioni, chiamate parti:

1. Component: l'entità che viene misurata
2. Property: la caratteristica del component
3. Time: l'intervallo di tempo in cui viene effettuata la misurazione
4. System (o Specimen): il campione su cui è stata effettuata la misurazione
5. Scale: indica il modo in cui il valore dell'osservazione viene quantificato o espresso (quantitativo, ordinale, nominale)
6. Method: questo campo opzionale descrive ad alto livello d'astrazione il modo in cui l'osservazione è stata effettuata. Questo campo è utile quando il metodo di osservazione è rilevante a fini diagnostici

Il contenuto di LOINC è suddiviso in due scopes principali: Laboratory e Clinical. La divisione Laboratory di LOINC si occupa di tutto ciò che si può osservare e misurare a partire dal campione ed include varie categorie come chimica, ematologia, microbiologia, mentre la sezione Clinical si occupa di ciò che può essere osservato in un paziente senza il prelievo di alcun campione, come ad esempio pressione, frequenza cardiaca, ma anche studi di radiologia e documenti clinici.

LOINC cura un database accessibile gratuitamente che raccoglie tutti i nomi e codici che fanno parte dello standard, permettendo l'integrazione e cooperazione con altri standard, poichè il dominio applicativo di LOINC è la nomenclatura piuttosto che la modellazione di un vero e proprio sistema nel suo complesso. [5]

Capitolo 3

Analisi e progettazione di un sistema interoperabile

Nel primo capitolo abbiamo fornito una panoramica sulla telemedicina come macro-dominio. Nel secondo abbiamo analizzato il problema dell'interoperabilità, sottolineando l'importanza di avere sistemi di telemedicina che si appoggino a strutture di comunicazione e standard che garantiscano l'interoperabilità tra sistemi eterogenei. In questo terzo capitolo si vuole proporre, analizzare e progettare una libreria, utilizzabile da applicazioni mobili, che supporti l'interoperabilità tra dispositivi di acquisizione di dati sanitari relativi ad un paziente, ed agenti esterni che richiedono tali dati per fornire un servizio di telemedicina. In particolare, la libreria dovrà permettere le seguenti operazioni:

1. Connessione ad un dispositivo: questo problema risulta critico in quanto diversi dispositivi possono avere modalità di connessione differenti, come ad esempio connessione Bluetooth o connessione tramite autenticazione ad un web service.

2. Esposizione agli utenti i dati in formato standard.

3.1 Analisi

In questa sezione procederemo ad analizzare i requisiti della soluzione nonchè il dominio nel quale si colloca. Prima di effettuare una vera e propria analisi, riportiamo alcuni termini tecnici relativi al dominio:

- Client: applicativo che usufruisce di un'implementazione della libreria proposta: si tratta tipicamente di un'applicazione mobile utilizzata dal

paziente di cui si vogliono ottenere dati sanitari, per interfacciarsi e raccogliere tali dati dai diversi dispositivi che ha in dotazione;

- Device: termine sinonimo di PHD che verrà adottato in sua sede, indica un qualsiasi dispositivo che permette di effettuare misurazioni;

- Observation: misurazione di un parametro vitale di un paziente, effettuata attraverso un device.

3.1.1 Analisi dei requisiti

Come anticipato in precedenza, gli obiettivi principali della libreria sono il permettere di sviluppare software in grado di interfacciarsi e gestire una gamma eterogenea di devices, ed il fornire una rappresentazione dei dati delle misurazioni standardizzata.

Requisiti funzionali

Attraverso l'implementazione della libreria il software cliente deve essere in grado di:

- registrare uno o più devices;

- connettersi ad uno o più devices;

- ottenere misurazioni da un certo device connesso attraverso una richiesta esplicita (in maniera quindi attiva);

- sottoscriversi ad una particolare misurazione di un particolare device in modo da essere notificati di una nuova misurazione (in maniera quindi passiva).

I requisiti funzionali esposti sono rappresentati graficamente dal seguente diagramma dei casi d'uso:

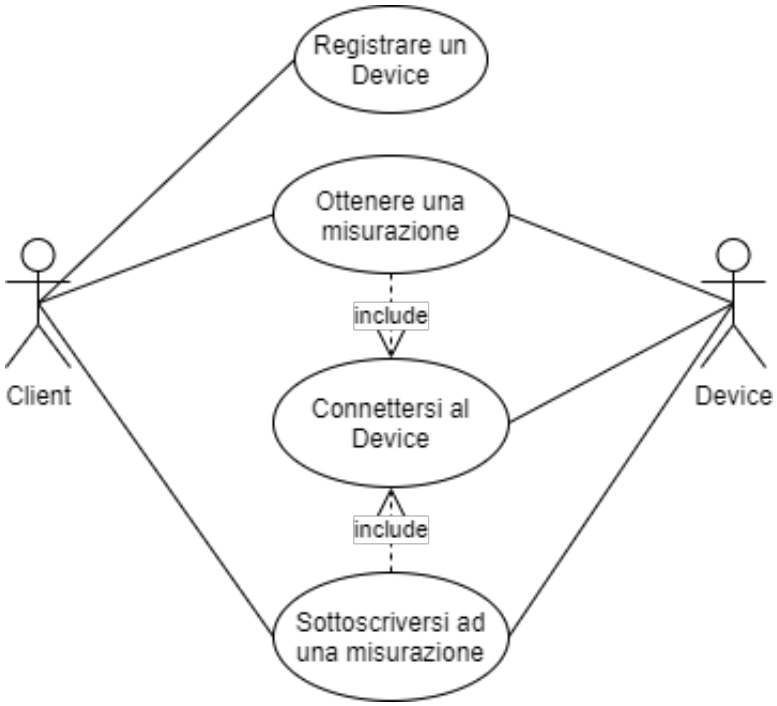


Figura 3.1: Schema UML dei casi d'uso

Requisiti non funzionali

La libreria deve essere riusabile ed estendibile, vale a dire che deve essere sufficientemente generale per permettere l'interfacciamento con devices di produttori diversi, i quali potrebbero avere interfacce di connessione e reperimento del dato profondamente diverse tra loro, come ad esempio API Bluetooth e web.

3.1.2 Analisi del dominio

Si descrive ora il tipico dominio applicativo: un paziente possiede uno o più PHD ed ha uno o più parametri vitali di interesse, che questi PHD sono in grado di misurare. Il paziente si interfaccia ai PHD grazie ad un applicativo, che denominiamo client. Il client, attraverso l'uso della libreria proposta, è in grado di gestire vari PHD, che gli permettono di collegarvisi per raccogliere dati riguardo ai parametri vitali che essi sono in grado di misurare. Lo scenario appena descritto è rappresentato graficamente nella figura 3.2

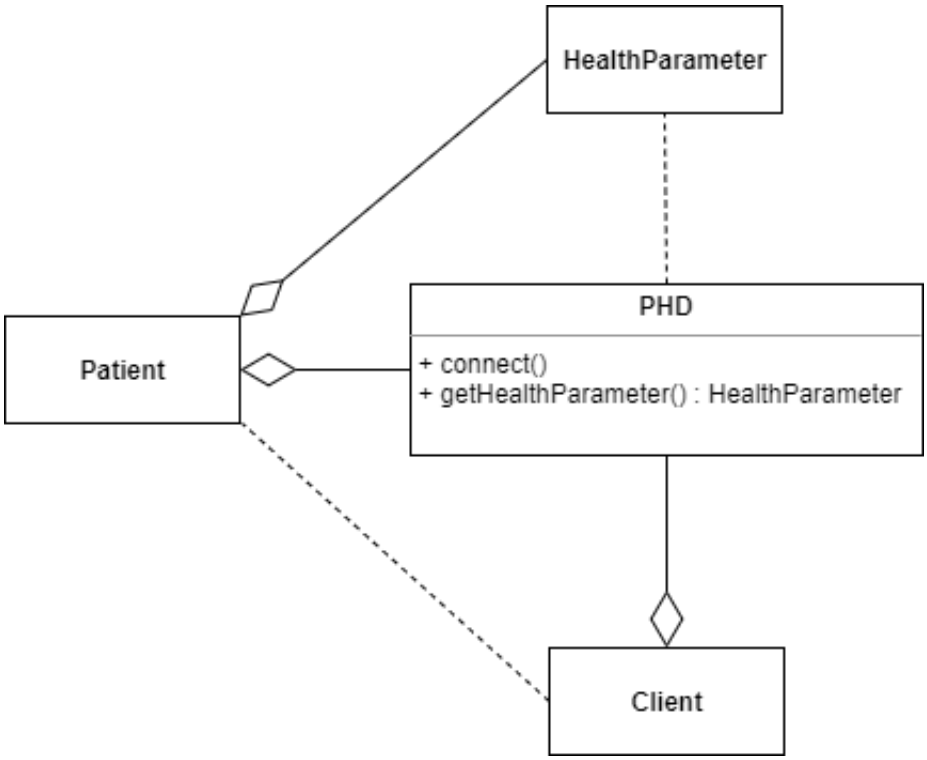


Figura 3.2: Schema UML che modella il dominio applicativo

Come già accennato nel capitolo 2 di questo documento, lo standard healthcare FHIR permette di modellare il dominio applicativo attraverso il concetto di risorsa. Questa modellazione si rivelerà importante in fase di design per meglio progettare una libreria in grado di aderire allo standard efficacemente. Dal punto di vista di FHIR, le entità interessanti del modello del dominio precedentemente descritto sono il paziente, i parametri vitali ed i dispositivi del paziente (rispettivamente Patient, HealthParameter e PHD in figura 3.2). Questi tre concetti sono modellabili attraverso altrettante risorse FHIR: Patient per il paziente, Observation per i parametri vitali e Device per i PHD. Queste tre entità possono eventualmente referenziarsi a vicenda, cioè includere tra le loro proprietà un riferimento ad un'altra risorsa nelle modalità descritte nel capitolo 2, per rappresentare il legame che hanno all'interno del dominio.

N.B. Fhir propone il profilo "Vital Sign" per modellare un'osservazione riferita ad un parametro vitale di un determinato paziente. Nonostante l'adozione di questo profilo possa essere considerata accettabile ai fini della soluzione, è stato scelto un set di attributi più ristretto, in quanto l'adozione del profilo

richiederebbe l'aggiunta di attributi non propri del dominio operativo della libreria proposta.

3.2 Progettazione

3.2.1 Architettura

Raffinando il modello del dominio ottenuto in fase di analisi, otteniamo lo scheletro della libreria che dovrà essere esteso in fase di implementazione. Entry point e controller del sistema è il DeviceManager, che permette una semplice gestione di più Device attraverso l'uso di una struttura a dizionario, che permette l'aggiunta e l'ottenimento del riferimento ai vari Device registrati. Al particolare Device è demandata la logica di connessione, in quanto si è ritenuto oneroso e poco pratico assegnare al Manager il compito di gestire le connessioni di molti dispositivi diversi. Il Device espone poi l'interfaccia di ottenimento dell'Observation. Una observation può essere ottenuta sia in maniera attiva, attraverso esplicita richiesta, che in maniera passiva, attraverso una sottoscrizione, che permette di ottenere una nuova Observation ogniqualvolta si verificano delle circostanze espresse da una particolare Policy. Si vuole inoltre notare che in sede di progettazione non è stata ritenuta necessaria la presenza della risorsa Patient, in quanto non facente strettamente parte del nucleo fondamentale di informazioni che la libreria intende scambiare.

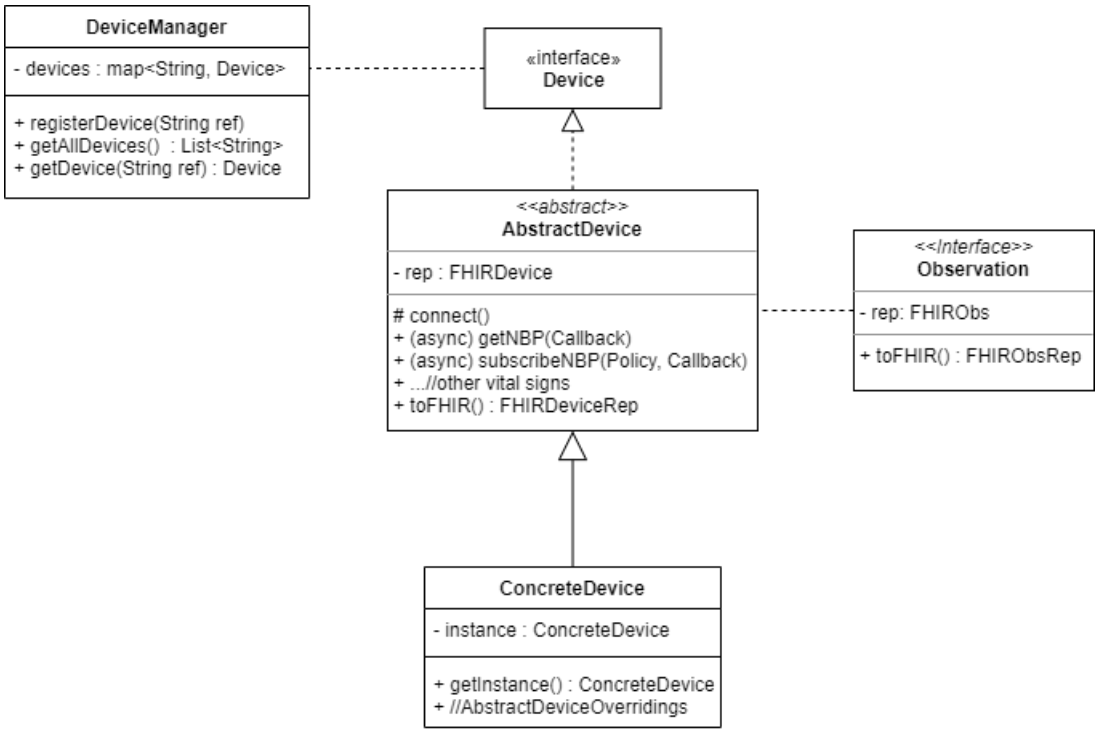


Figura 3.3: Schema UML dello scheletro della libreria

3.2.2 Progettazione esecutiva

Modellazione del Device

La rappresentazione della struttura interna del Device segue la modellazione FHIR della risorsa Device. Una classe astratta di base modella quindi un set essenziale di attributi che facilitano la caratterizzazione di un Device e ne espone all'esterno la rappresentazione FHIR, lasciando comunque la possibilità all'implementatore di specializzare ulteriormente la struttura del Device a seconda del contesto in cui si trova. La classe base espone anche i metodi necessari a richiedere un'osservazione o sottoscriversi ad essa, permettendo all'implementatore dello specifico Device di sovrascrivere i metodi corrispondenti alle Observations trattate dal device in oggetto.

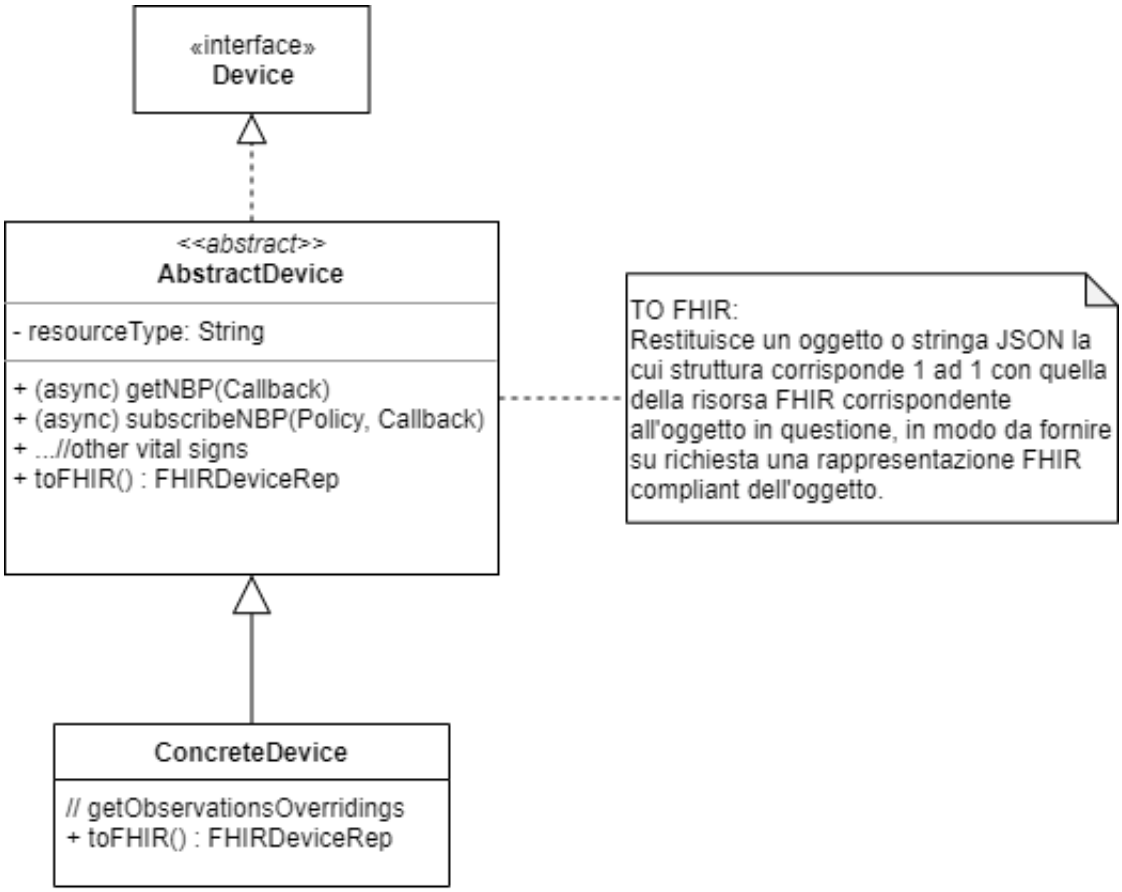


Figura 3.4: Struttura interna base del Device

N.B. Fhir specifica come unico attributo obbligatorio per la risorsa Device l'attributo "resourceType", elemento fondamentale di ogni risorsa Fhir. Nonostante questo attributo sia l'unico necessario affinché la risorsa sia considerabile Fhir-compliant, la maggior parte dei domini applicativi potrebbe avvalersi di altri attributi già a livello di struttura della classe astratta, quali ad esempio "identifier", "deviceName" o "text".

Per quanto riguarda l'istanziamento dei devices specifici, si è adottato il pattern Singleton per garantire che ci sia una unica istanza di Device specifico a run-time.

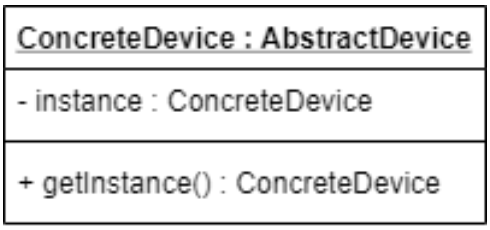


Figura 3.5: Rappresentazione UML del pattern Singleton

In fase di progettazione della logica di connessione al dispositivo, è stato deciso di fornire un metodo astratto nella classe base, lasciando alle varie specializzazioni la responsabilità dell'implementazione vera e propria. Questa scelta si rivela necessaria a causa della eterogeneità tra dispositivi e tecniche di connessione che si assume essere presente in un tipico scenario di interoperabilità tra soluzioni di telemedicina.

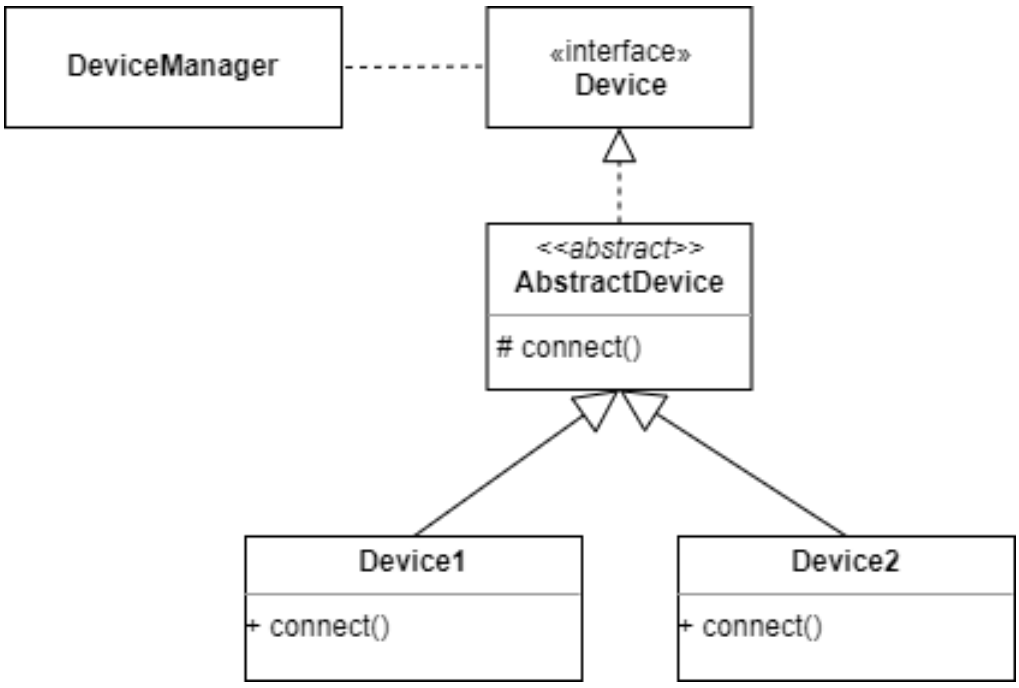


Figura 3.6: La logica di connessione viene specificata solamente dalle estensioni della classe astratta

Observation

Il concetto di osservazione è rappresentato dall'interfaccia Observation. La struttura interna degli attributi rappresenta gli elementi minimi di una risorsa

Fhir di tipo "Observation". In questa fase si è scelto di non modellare un concetto di "valore" dell'Observation in quanto esso per natura può essere sia quantitativo che qualitativo, dipendentemente dallo scenario specifico. Questa variabilità nel concetto di valore si riflette anche nella specifica Fhir, che non impone vincoli sulla modellazione del valore dell'osservazione e anzi, fornisce vari attributi opzionali per modellarlo secondo necessità.

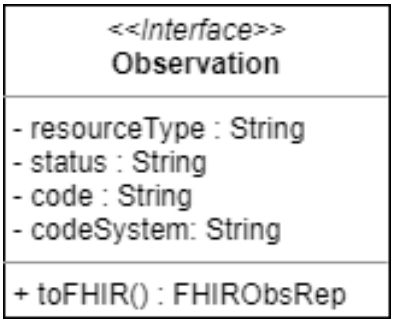


Figura 3.7: Struttura di un'Observation all'interno della libreria

Policy

Per quanto riguarda la verifica della Policy che regola il comportamento della sottoscrizione ad un'osservazione, è necessario che il Device rimanga reattivo alle richieste del Client anche successivamente ad una chiamata a sottoscrizione. Per questo motivo, il concetto di Policy è stato modellato attraverso il pattern Observer. In questa visione, la Policy non è solo un insieme di condizioni da controllare per reagire al loro soddisfacimento, quanto più un subject incaricato di verificare lo stato delle proprie condizioni e notificare tutti i devices sottoscritti (PolicyObserver) del soddisfacimento della Policy stessa.

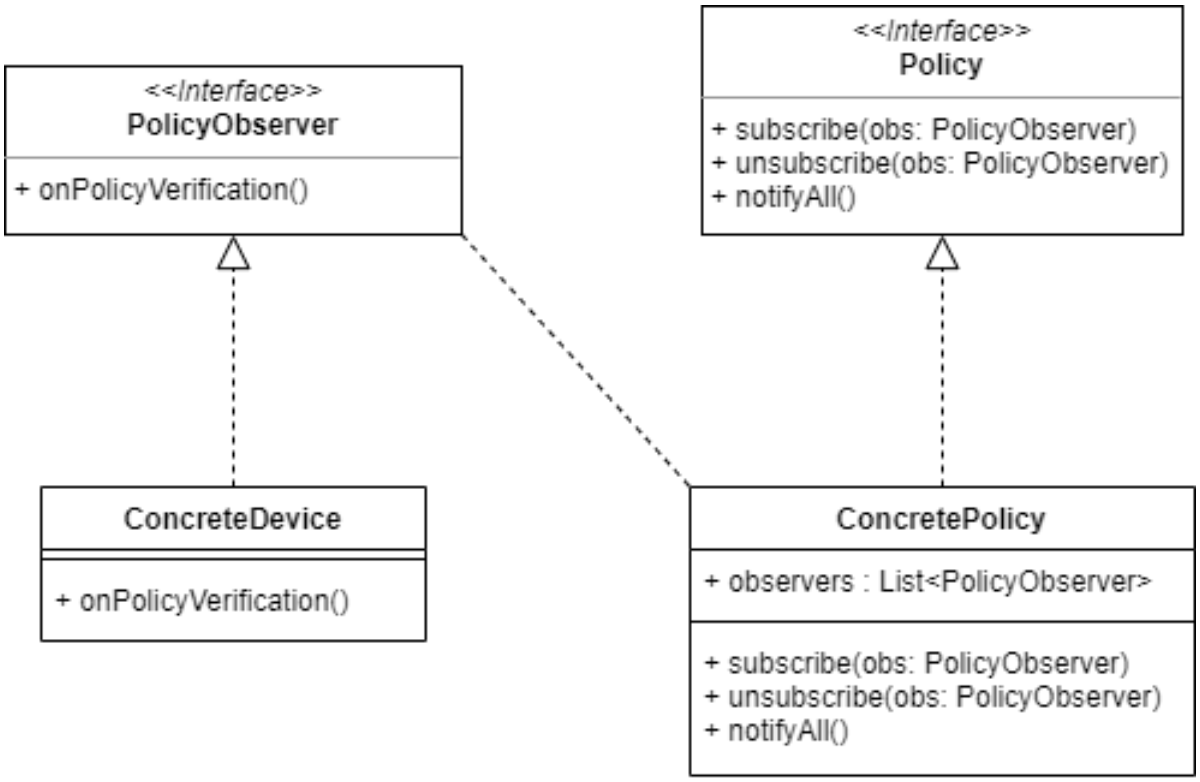


Figura 3.8: Modellazione del concetto di Policy come subject osservabile

Interazione

In questa sotto sezione si vuole mostrare il processo di interazione tra le entità principali in ogni scenario d'uso.

Registrazione di un dispositivo

1. Il Client richiede la registrazione di un nuovo device al DeviceManager;

2. il DeviceManager registra nella sua mappa interna il nuovo tipo di Device, istanziandolo;

3. una volta registrato il Device, il suo riferimento può essere ottenuto richiedendolo al DeviceManager.

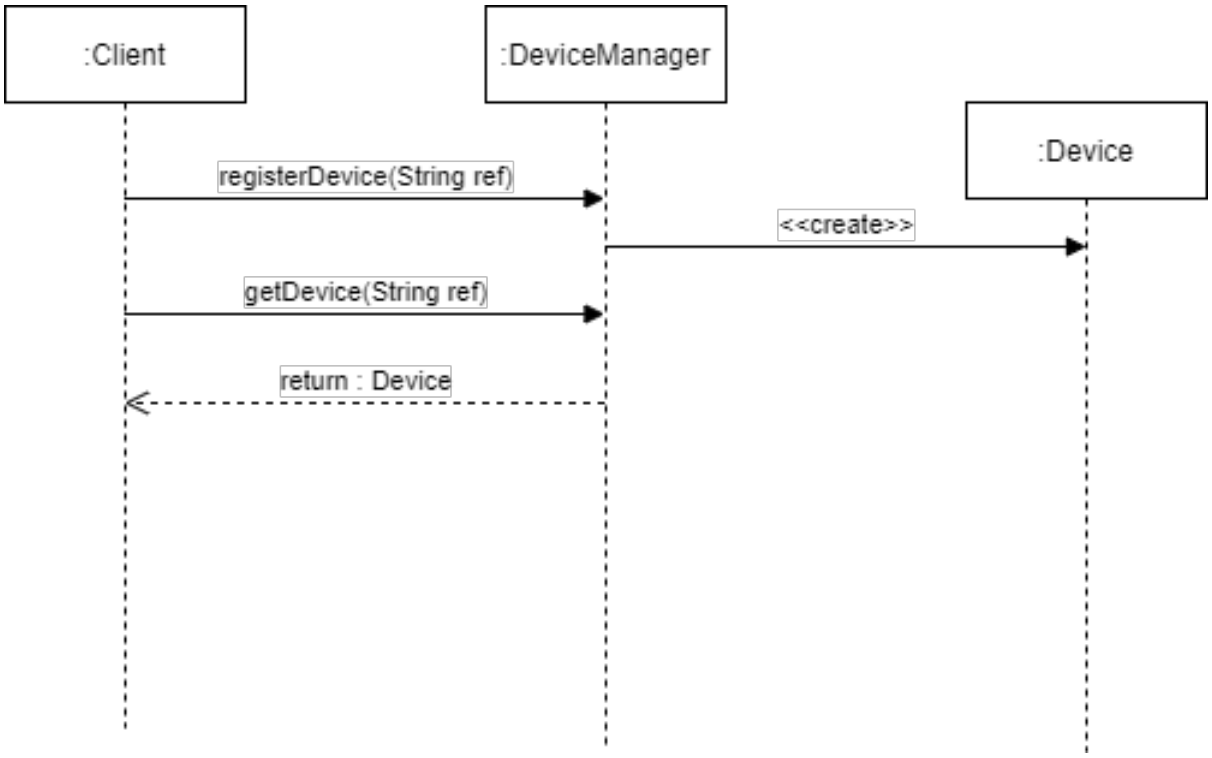


Figura 3.9: Sequence Diagram della registrazione di un nuovo Device

Ottenimento di una Observation

1. Il Client richiede il riferimento al device che intende utilizzare per la richiesta al DeviceManager;

2. ottenuto il riferimento, il Client chiama connect() sul Device per interfacciarvisi;

3. una volta connesso, il Client può richiedere osservazioni in maniera asincrona specificando come parametro della richiesta una callback da chiamare al termine della produzione dell'osservazione;

4. il Device effettua la callback sul Client.

Varianti:

- (3a): Se il Client richiede una Observation non supportata dal Device, vale a dire un metodo non sovrascritto dalla classe astratta di base, verrà eseguito il comportamento specificato dalla classe astratta stesso:

a seconda delle implementazioni della libreria esso potrà essere la generazione di un'eccezione o di un messaggio di errore o un altro tipo di comportamento standard.

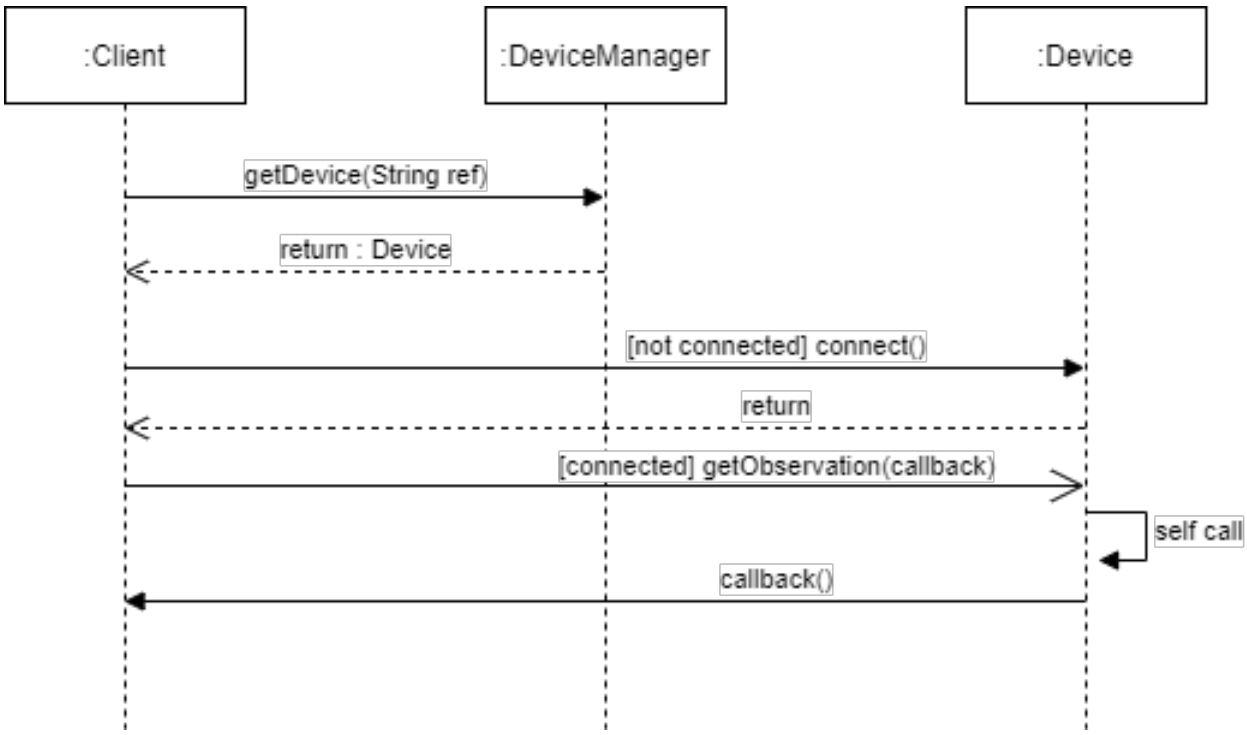


Figura 3.10: Sequence Diagram della richiesta di una osservazione

Sottoscrizione ad una Observation

1. Il Client richiede il riferimento al device che intende utilizzare per la richiesta al DeviceManager;

2. ottenuto il riferimento, il Client chiama connect() sul Device per interfacciarvisi;

3. una volta connesso, il Client può richiedere in maniera asincrona al Device la sottoscrizione alla Observation desiderata, specificando una Policy secondo la quale ottenere l'ossservazione alla quale si è sottoscritto;

4. una volta verificato il set di condizioni espresse dalla Policy della sottoscrizione, il Device chiama la callback sul Client.

Varianti:

- (3a): Se il Client richiede la sottoscrizione ad una Observation non supportata dal Device, vale a dire un metodo non sovrascritto dalla classe astratta di base, anche in questo caso verrà eseguito il comportamento specificato dalla classe astratta stesso: a seconda delle implementazioni della libreria esso potrà essere la generazione di un'eccezione o di un messaggio di errore o un altro tipo di comportamento standard.

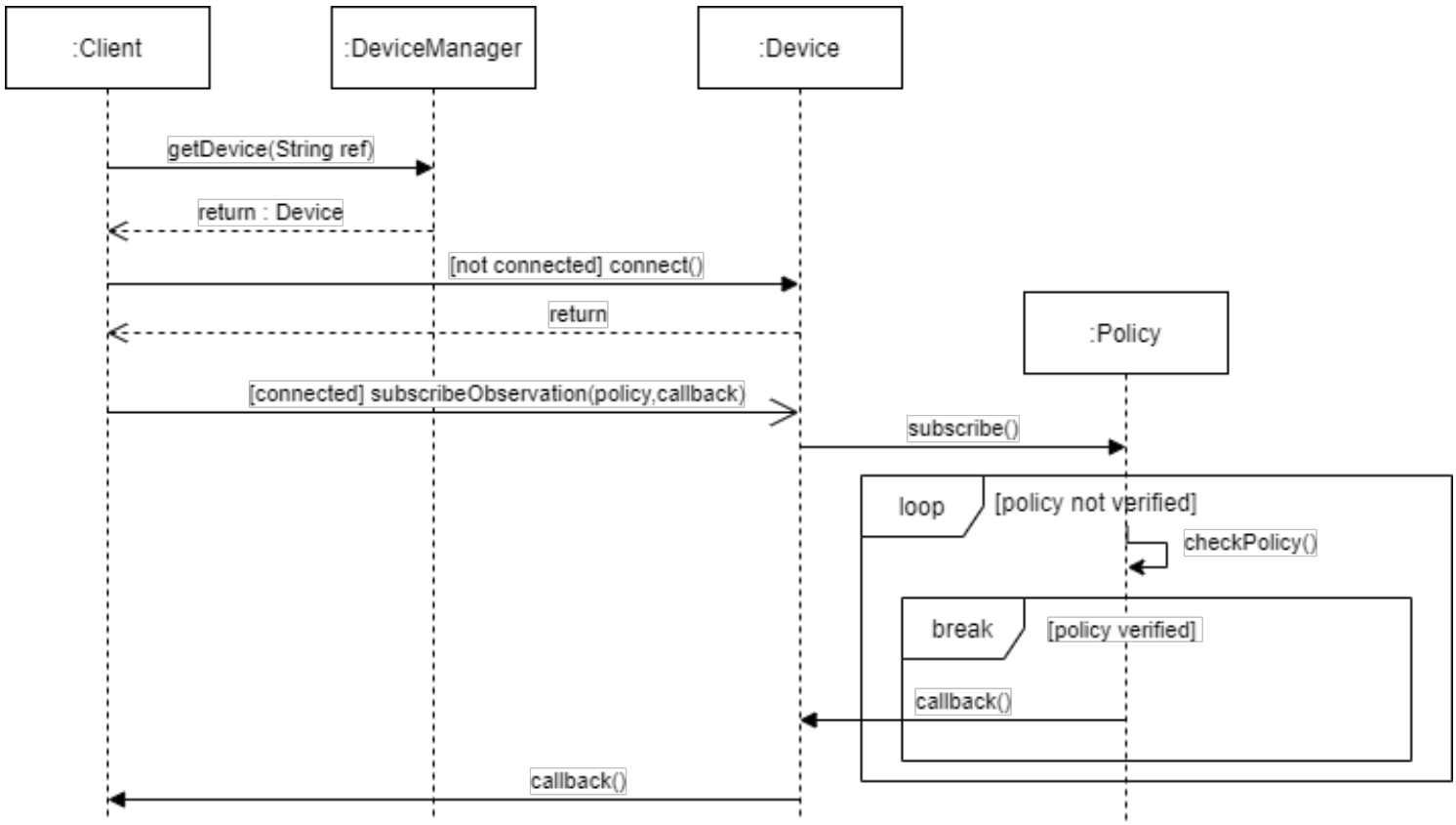


Figura 3.11: Sequence Diagram della sottoscrizione ad una osservazione

Capitolo 4

Implementazione e sviluppo del sistema proposto

In questo capitolo si vuole porre l'accento sulle fasi di implementazione della libreria e di sviluppo di un semplice applicativo che ne faccia uso. L'obiettivo di questa sperimentazione è verificare che la libreria permetta effettivamente di svolgere le operazioni descritte nel capitolo precedente, cioè di interagire in maniera interoperabile con dispositivi di produttori diversi (con tecnologie di connessione diverse) e di esporre le misurazioni ottenute secondo lo standard FHIR.

4.1 Metodologia di lavoro

Il processo di sviluppo adottato in questa sede segue quello del modello di processo a cascata: le fasi di analisi e progettazione sono infatti servite da input per la fase di produzione, che è cominciata una volta definiti tutti gli aspetti di design rilevanti.

In particolare, la fase di produzione si è snodata tra le seguenti sottofasi:

- Implementazione della libreria: in questa sottofase è stato realizzato, a partire dagli schemi di progettazione, un package contenente i moduli che realizzano la libreria.

- Sviluppo di un semplice applicativo Client: seguendo la definizione di Client data nel capitolo precedente in fase di analisi, è stato prodotto un applicativo mobile che facesse uso della libreria per potersi connettere ed ottenere dati da dispositivi differenti.

- Validazione del lavoro svolto: questa sottofase comprende i test automatici e manuali eseguiti per validare il lavoro di sviluppo svolto.

4.2 Tecnologie scelte

In questa sottosezione si vogliono fornire, per ogni sottofase di produzione, le considerazioni fatte circa le tecnologie da impiegare per ognuna.

4.2.1 Implementazione della libreria

Per facilitare il processo di sviluppo, è stato adottato Typescript come estensione del linguaggio Javascript, che aggiunge a quest'ultimo una forte tipizzazione. Ciò abbatte drasticamente la possibilità di commettere errori di tipo, in quanto il compilatore Typescript controlla costantemente che l'uso dei tipi sia coerente in tutto il codice prodotto. Un altro vantaggio di questo linguaggio è quello di introdurre in Javascript molti dei meccanismi tipici della programmazione Object-Oriented fortemente tipizzata come classi astratte ed interfacce, permettendo una più naturale produzione di codice a partire dai documenti di design Object-Oriented prodotti precedentemente. Inoltre, Typescript, basandosi completamente su Javascript, è pienamente interoperabile con esso.

4.2.2 Sviluppo applicativo Client

La libreria proposta è stata pensata per un contesto d'uso tipicamente mobile, per cui il Client sarà rappresentato da un'applicazione mobile. Per quanto riguarda lo sviluppo mobile, si può valutare di sviluppare un'applicazione in linguaggio nativo per ogni sistema operativo mobile diffuso al giorno d'oggi: vale a dire Android ed iOS. In alternativa, esistono dei framework di sviluppo multipiattaforma che permettono di scrivere il codice una sola volta, mantenendo la compatibilità con entrambi i sistemi operativi citati. Attualmente il più diffuso è React Native e rappresenta la scelta tecnologica per lo sviluppo del Client.

React Native è un framework per lo sviluppo di applicazioni mobili open source creato da Facebook, che permette di sviluppare applicazioni native sia Android che iOS appoggiandosi al framework React, che fa uso di Javascript come linguaggio per lo sviluppo di interfacce utente interattive. Questo ci permette di sviluppare una unica applicazione multipiattaforma con la garanzia che possa essere eseguita su entrambi i principali sistemi operativi mobile.

4.2.3 Testing automatico

Per quanto riguarda il testing automatico, si è fatto uso di Jest, un framework di test Javascript gestito da Facebook.

4.3 Validazione

In questa fase sono state testate l'applicazione client, nonché le funzionalità della libreria richieste in fase di analisi dei requisiti.

4.3.1 L'applicativo Client

Come già anticipato, l'applicativo Client gioca un ruolo importante nella validazione della libreria, poichè agisce da mock-up di un possibile processo che usufruisce delle funzionalità da esso esposte. Trattandosi di un semplice mock-up, la sua architettura è piuttosto semplice e funzionale al test manuale della libreria con vari dispositivi. La struttura interna rispecchia la classica struttura di un'applicazione React Native, all'interno della quale sono stati implementati due componenti: DMPanel e DeviceComponent.

DMPanel

Questo componente rappresenta l'entry-point dell'applicazione, quindi è qui che si accede alle funzionalità del DeviceManager. Come mostra la figura 4.1, l'interfaccia del componente è composta da un Textbox, che permette di digitare il codice con cui è stato registrato un dispositivo, ed un bottone submit che avvia la ricerca del device corrispondente al codice immesso, semplicemente invocando il metodo getDevice su un'istanza del DeviceManager. In caso di assenza di un device registrato col codice immesso, un messaggio di errore viene mostrato all'utente. Viceversa, in caso di successo dell'operazione, l'applicazione naviga verso il secondo componente. Il modo in cui avviene la navigazione è il seguente: se getDevice restituisce un oggetto di tipo Device, il router naviga verso DeviceComponent passando come prop l'oggetto Device stesso.

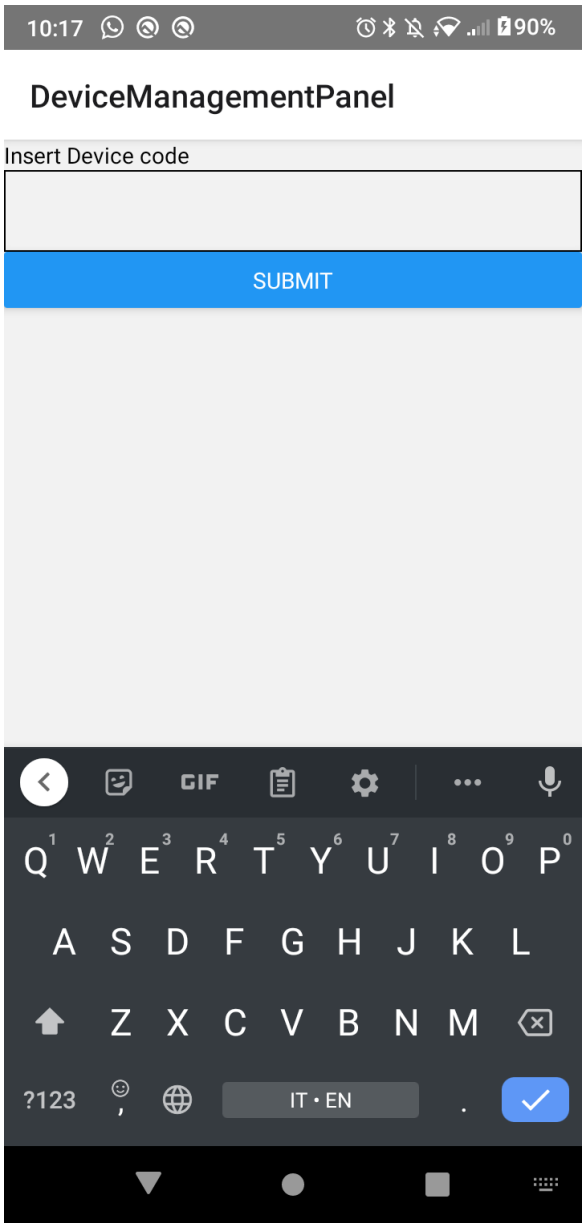


Figura 4.1: Schermata del DeviceManagementPanel

DeviceComponent

Questo componente permette all'utente dell'applicazione client di accedere alle API del Device che ha cercato nel componente precedente. L'interfaccia è costituita da una serie di bottoni (figura 4.2), ognuno dei quali invoca sull'oggetto Device presente nella props il metodo corrispondente. Per mantenere l'UI semplice, i valori di ritorno ed i messaggi provenienti dai metodi invocati

dai bottoni vengono stampati sulla console del React Native Debugger, mentre nell'interfaccia dell'applicazione vera e propria verranno mostrate solo notifiche che mostrano il tipo di esito che ha avuto la chiamata, come mostrato nelle figure 4.3a e 4.3b.

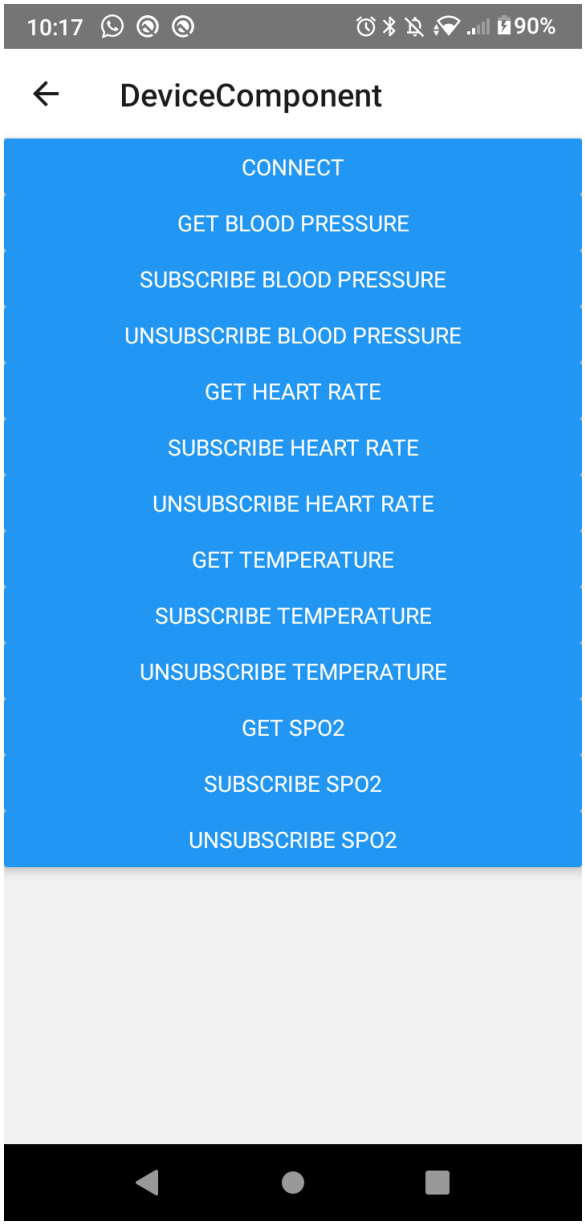
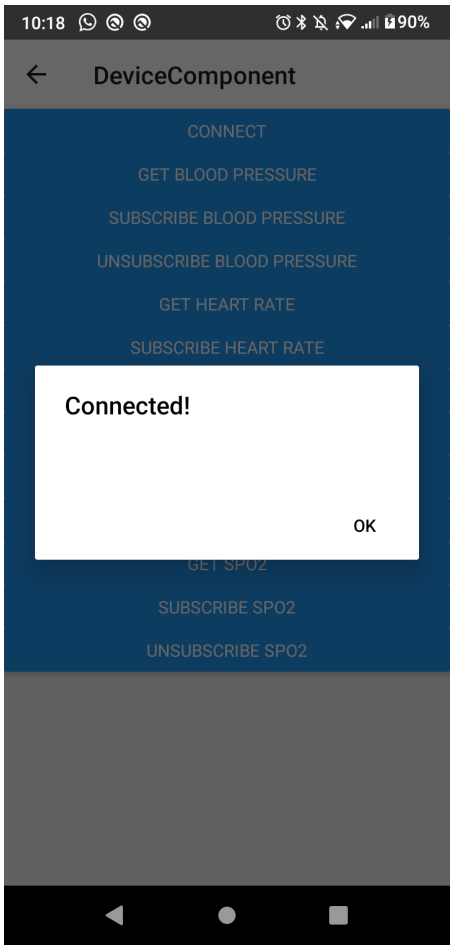
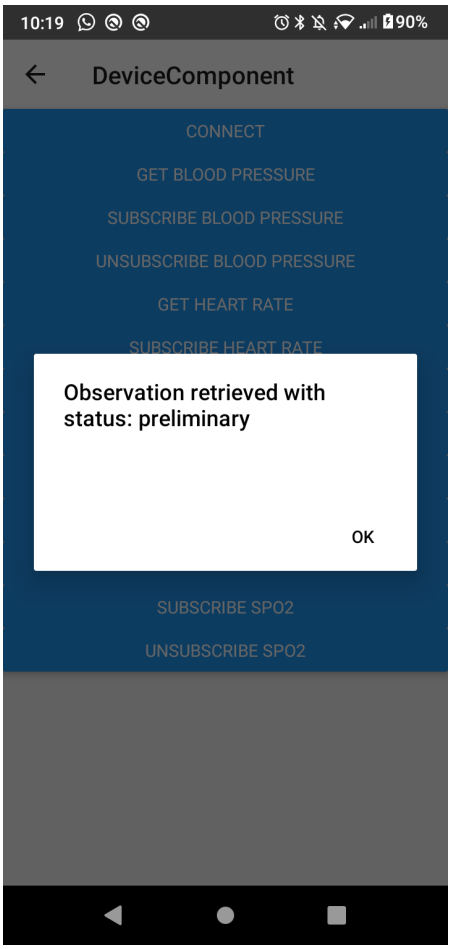


Figura 4.2: Schermata del DeviceComponent



(a) Messaggio di avvenuta connessione



(b) Messaggio che avvisa che l'osservazione è stata recuperata correttamente

4.3.2 Scelta dei dispositivi

La validazione è stata effettuata attraverso l'interfacciamento dell'applicativo client con due dispositivi diversi: un device FitBit ed un device Gima Spot-Check monitor PC300. Questa scelta è stata basata sulla necessità di integrare almeno due dispositivi diversi contemporaneamente, per verificare che l'obiettivo di interoperabilità fosse raggiunto. Inoltre, sono stati scelti due dispositivi le cui modalità di connessione differiscono, in modo da poter sperimentare e validare la libreria a fronte di scenari di interoperabilità vari: per quanto riguarda il dispositivo FitBit, infatti, è possibile ottenere i dati da esso raccolti attraverso l'uso delle FitBit Web API: un servizio Web REST che permette ad operatori autenticati attraverso il protocollo OAuth2.0 di accede-

re alle misurazioni effettuate dal dispositivo di un determinato utente Fitbit, previa autorizzazione di quest'ultimo. In questo senso, la "connessione" al dispositivo richiesta dalla libreria assume più la connotazione di una autenticazione al servizio web di Fitbit. Il dispositivo PC300, d'altra parte, richiede che venga instaurata una connessione Bluetooth (attraverso Bluetooth 2.0) per permettere la comunicazione tra dispositivo e client. Una volta instaurato il canale di comunicazione col dispositivo, è possibile interrogarlo circa le misurazioni effettuate con esso ed ottenerle, sempre tramite Bluetooth.

4.3.3 Validazione attraverso FitBit

L'aspetto principale di cui si è dovuto tener conto sviluppando l'implementazione del Device FitBit è stato quello di autenticazione secondo il workflow del protocollo OAuth2.0. Al fine di risparmiare tempo sull'implementazione del protocollo, è stato deciso di servirsi di una libreria esterna che gestisse l'intero processo di autenticazione: la libreria "react-native-app-auth". Questa scelta permette di riutilizzare una soluzione già testata e validata al problema dell'autenticazione secondo il protocollo OAuth2.0, particolarmente diffuso tra i servizi che raccolgono dati sensibili dell'utente.

Una volta gestito il processo di autenticazione tramite questa libreria, la comunicazione con FitBit API avviene attraverso scambio di messaggi HTTP GET, nei cui headers viene aggiunta una chiave di accesso.

4.3.4 Validazione attraverso Spot-Check PC300

L'aspetto principale di cui si è dovuto tenere conto per imbastire una validazione con PC300, è quello della comunicazione Bluetooth 2.0 col dispositivo: infatti, tra le funzionalità core di React Native, non è presente la gestione della comunicazione Bluetooth. Nonostante la recente diffusione di librerie di gestione del Bluetooth tramite React Native, queste si basano tendenzialmente su tecnologia BLE, rendendo necessaria la gestione a livello nativo delle versioni di Bluetooth più vecchie. È a questo scopo che ci si è avvalsi del meccanismo dei React Native Native Modules.

Native Modules

Durante lo sviluppo di un progetto, ci si può imbattere nella necessità di accedere alle API native della piattaforma in cui ci si trova, ad esempio perchè non è presente un loro corrispettivo in Javascript, o per riusare librerie

Java/Objective-c non implementate in Javascript, o ancora per scrivere codice multi-thread ad alte prestazioni. Per questi motivi il sistema dei Native Modules permette di esporre al codice Javascript delle classi (o parti di esse), scritte in linguaggio nativo, sotto forma di oggetti JS.

Per far sì che una classe nativa sia considerata un modulo nativo, essa deve implementare l'interfaccia NativeModule (o estendere una classe che lo faccia, come ad esempio ReactContextBaseJavaModule). Un modulo nativo può esporre al framework RN un'API nativa attraverso l'annotazione @ReactMethod, che rende il metodo annotato invocabile attraverso l'oggetto JS corrispondente al modulo nativo. Per far sì che un modulo nativo sia "convertibile" in un oggetto JS, occorre che venga aggiunto ad un ReactPackage e che questo package venga registrato all'interno della ReactApplication nel progetto nativo. Allo startup, React Native itererà su tutti i packages, invocando il metodo createNativeModules() su ogni package in modo da registrare tutti i moduli nativi presenti. A questo punto è possibile, all'interno del progetto JS, creare un oggetto corrispondente al modulo nativo ed invocare tutti i metodi annotati @ReactMethod in esso.

N.B. È bene notare che i ReactMethod sono metodi asincroni senza parametri di ritorno. È possibile tuttavia stabilire un metodo di comunicazione asincrono basato su callbacks o Promises per permettere il passaggio di valori dal modulo nativo al processo JS chiamante.

All'interno del Native Module, oltre all'accesso alle API native, ci si è serviti della libreria proprietaria del dispositivo "CreativeHealth-SpotCheck" e nello specifico delle seguenti funzionalità da essa offerte:

- uso di una classe di utility che gestisce la connessione bluetooth al dispositivo;
- uso di una classe di utility che permette, attraverso l'implementazione di un'apposita callback, la comunicazione asincrona con il dispositivo.

Una volta implementato il Native Module è stato quindi possibile, attraverso l'applicazione React Native, collegarsi tramite Bluetooth a PC300 ed interrogarlo per ottenere le osservazioni esposte dalla sua implementazione all'interno della libreria.

4.4 Risultati ottenuti

A seguito delle sperimentazioni della libreria, svolte attraverso l'utilizzo dell'applicativo client che ne fa uso, sono stati ottenuti dei risultati che si allineano agli obiettivi delineati nel capitolo 3. È stato infatti possibile connettersi, durante lo stesso workflow, a due dispositivi con logica di connessione differente: un dispositivo Fitbit attraverso autenticazione OAuth2 al web service, ed un dispositivo multiparametrico connesso attraverso Bluetooth.

Per entrambi i dispositivi è stato possibile mantenere attiva la connessione per tutta la durata della sperimentazione, ed è stato possibile reperire le misurazioni che essi "esponevano" all'applicativo attraverso delle chiamate alla libreria implementata. Inoltre queste ultime sono correttamente rappresentate come risorse FHIR, come possiamo osservare dal seguente esempio:

```
{
    "resourceType":"Observation",
    "status":"preliminary",
    "code":{
        "coding":[{
            "system":"http://loinc.org",
            "code":"8867-4"
        }]
    },
    "valueQuantity":{
        "value":90,
        "unit":"beats/minute"
    }
}
```

Questo oggetto JSON è una tipica stringa ottenuta da una richiesta di osservazione eseguita utilizzando la libreria proposta, in questo esempio mostrata indentata solo a scopo di leggibilità.

Conclusioni

In questa tesi è stato affrontato il problema di interagire con dispositivi di acquisizione, ed in ultima analisi sistemi, di natura diversa, che fanno uso di interfacce e protocolli di comunicazione diversi e che hanno rappresentazioni interne dei dati differenti, per poter ottenere tutti i dati sanitari di cui un caretaker possa avere bisogno nella cura dei suoi pazienti. In questo contesto è stata inoltre discussa la necessità di rendere disponibili tali dati in una forma standard, aspetto critico al fine di rendere la soluzione a sua volta il più possibile interoperabile.

Partendo da questo problema è stata proposta una libreria per dispositivi mobili, che fornisce degli strumenti per interfacciarsi con diversi dispositivi e per ottenere i parametri da essi misurati e formattarli secondo lo standard FHIR. La libreria è stata progettata attraverso l'uso di vari design pattern, tra cui:

- Singleton per l'istanziamento di un solo controller per Device;
- Template Method per la gestione flessibile della logica di connessione ai particolari Device;
- Observer per la gestione delle Policy con cui l'applicazione client si può sottoscrivere ad una osservazione. Questa scelta è stata effettuata per mantenere il Device reattivo a richieste successive a quella di sottoscrizione.

Inoltre, in fase di sviluppo sono state impiegate le seguenti tecnologie:

- Typescript per l'implementazione della libreria;
- React Native per lo sviluppo dell'applicativo client.

La fase di validazione è stata eseguita attraverso l'interfacciamento dell'applicazione client con due dispositivi differenti: un'emulazione di uno smartwatch Fitbit, la cui connessione consiste nell'autenticazione tramite OAuth2.0 alle Fitbit Web API, ed il dispositivo Spot-Check PC300, connesso tramite Bluetooth 2.0. Al termine di questa fase sono stati ottenuti i seguenti risultati: l'applicativo client è stato in grado di collegarsi ad entrambi i dispositivi e ad interrogarli, sia tramite richiesta esplicita che tramite sottoscrizione, per ottenere dei parametri vitali dell'utente dell'applicativo. Questi dati vengono visualizzati formattati secondo lo standard FHIR.

Sviluppi futuri

Questo progetto getta la base di una libreria che può essere utilizzata da una qualsiasi applicazione per integrarsi con dispositivi di acquisizione di parametri vitali, anche differenti da quelli utilizzati durante la fase di validazione. Infatti, per potersi interfacciare con un nuovo dispositivo, è sufficiente implementare correttamente una specializzazione della classe AbstractDevice che gestisca la comunicazione col dispositivo scelto, ed è anche possibile specificare nuovi tipi di Observation attraverso la specializzazione della classe AbstractObservation. Sarebbe inoltre sicuramente interessante sperimentare sulla varietà delle applicazioni che possano fare utilizzo della libreria (dal teleconsulto al telemonitoraggio) per osservare come un'attenta progettazione possa rendere flessibile ed estendibile l'uso del software.

Commento critico sul lavoro svolto

Sono molto soddisfatto del progetto di tesi al quale ho lavorato, perché mi ha permesso di svolgere attività varie e stimolanti. In primo luogo, la ricerca sulla telemedicina mi ha permesso di essere più consapevole circa le ragioni sociali alla base di un fenomeno in grado, potenzialmente, di cambiare il modo in cui si eroga e usufruisce di un servizio. Ho quindi trovato questa fase del lavoro molto utile per costruire una conoscenza di base riguardo ad un aspetto dell'informatica in grande sviluppo attualmente, conoscenza che non può che giovare ai miei studi e progetti futuri. La fase di analisi e progettazione, coadiuvata dai professori Ricci, Croatti e Montagna grazie al confronto aperto e costruttivo, mi ha dato molte lezioni che porterò con me nei miei prossimi studi. Infine, la fase di implementazione e validazione mi ha permesso di approfondire le mie conoscenze di tecnologie con le quali non mi ero mai confrontato in precedenza, dandomi la possibilità di imparare nuove abilità nel momento in cui cercavo di superare le difficoltà che talvolta potevano nascere dal loro uso.

In conclusione, ho trovato l'intero progetto di tesi altamente stimolante ed è stato gratificante poter mettere a frutto le conoscenze maturate in questo percorso triennale attraverso un progetto molto attuale.

Ringraziamenti

Tengo a ringraziare il professor Alessandro Ricci per avermi offerto la possibilità di lavorare ad un progetto calato in un ambito che ritengo molto interessante, attuale ed importante, i professori Angelo Croatti e Sara Montagna per la continua e puntuale disponibilità ad assistermi in ogni passo del mio lavoro, ed infine ringrazio tutte le numerose persone che mi hanno supportato in questo percorso.

Bibliografia

[1] Fhir overview. http://hl7.org/fhir/. Ultimo accesso: 18/04/2021.

[2] Iso/ieee 11073 personal health data (phd) standards. https://en.wikipedia.org/wiki/ISO/IEEE_11073_Personal_Health_Data_(PHD)_Standards. Ultimo accesso: 9/02/2021.

[3] American Telemedicine Association. Telehealth interoperability driving choice, continuity, and scale, 2019.

[4] Hawazin Faiz Badawi, Fedwa Laamarti, and Abdulmotaleb El Saddik. Iso/ieee 11073 personal health device (x73-phd) standards compliant systems: A systematic literature review. *IEEE Access*, 7:3062–3073, 2019.

[5] Regenstrief Institute. Loinc quickstart guide, 2015.

[6] Alberto Rosotti. Informatica medica e reti di telemedicina, lezioni 4, 16 e 17, 2020. Alma mater studiorum università di Bologna, facoltà di Ingegneria.

[7] Ministero Della Salute. Telemedicina - linee di indirizzo nazionali, 2014.

[8] Hyoungho Do Sungkee Lee. Comparison and analysis of iso/ieee 11073, ihe pcd-01, and hl7 fhir messages for personal health devices. https://pubmed.ncbi.nlm.nih.gov/29503752/, 2018.

[9] Integrating the Healthcare Enterprise. Ihe technical frameworks general introduction, 2019.