

ALMA MATER STUDIORUM · UNIVERSITÀ DI
BOLOGNA

SCUOLA DI SCIENZE

Corso di Laurea Magistrale in Informatica

**Progettazione e realizzazione
di servizi e applicativi con architettura REST
in ambito sanitario**

Relatore:
Chiar.mo Prof.
Alessandro Amoroso

Presentata da:
Margherita Lazzarini

Sessione II
Anno Accademico 2013-2014

*Dedico questa tesi alla mia famiglia,
senza la quale nulla sarebbe mai stato possibile*

Introduzione

L'analisi dello scenario evolutivo che si presenta in relazione all'aumento della vita media della popolazione mostra chiaramente, con l'avanzare dell'età, l'insorgenza di patologie croniche, quali nefropatie, cardiopatie, diabete e obesità. Queste malattie, nei primi stadi del loro sviluppo, non presentano sintomi rilevanti, pertanto i pazienti intraprendono solitamente un percorso di cura in fasi avanzate della patologia. Favorire la diagnosi precoce può rivelare eventuali cause ancora reversibili della patologia e rallentarne il decorso, applicando trattamenti terapeutici che possano migliorare la qualità della vita del paziente.

In questo contesto si colloca *Viewpoint*, un applicativo sviluppato dal gruppo *NoemaLife* nell'ambito del progetto *Smart Health*, che si pone come obiettivo lo sviluppo di servizi ICT per la sanità pubblica al fine di migliorare la qualità del servizio alla cittadinanza. È stato sviluppato un primo prototipo, la cui validazione e sperimentazione è prevista per dicembre 2014 presso il Policlinico di Bari. In questa versione, le funzionalità di *Viewpoint* sono legate all'individuazione e analisi di potenziali nefropatie, mentre in futuro verrà esteso a supporto di altre patologie croniche asintomatiche. Il sistema sviluppato permette l'applicazione di appositi algoritmi indicatori di rischio sulla somma dei dati provenienti da differenti sorgenti informative, la visualizzazione di grafici, tabelle e mappe che mostrino i dati aggregati sulla popolazione presa in esame e la possibilità di effettuare simulazioni sul futuro in modo da valutare aspetti clinici ed economici. Inoltre, il software permette di identificare ogni singolo paziente e di contattare il medico di base

o il paziente stesso - in funzione del consenso alla privacy fornito - andando a migliorare il paradigma di notifica dal medico di medicina generale allo specialista, che spesso è fonte di anomalie organizzative all'interno del SSN.

Per favorirne la portabilità, il sistema software è un applicativo web, che utilizza servizi REST e sfrutta l'architettura EJB. Per la maggior parte sono state utilizzate tecnologie open source, quali Pentaho CE per la logica di business intelligence, JBoss Application Server per il deploy dell'applicazione, HTML5, JQuery, Bootstrap e Angular JS per l'implementazione del front end. La parte da me curata è stata la sezione *Analytics*, la quale implementa le funzionalità che consentono di visualizzare dati aggregati sui pazienti del reparto di nefrologia tramite grafici e tabelle, e di effettuare simulazioni sui potenziali futuri pazienti.

Indice

Introduzione	i
1 Il contesto	1
1.1 L'organizzazione del SSN	2
1.1.1 Le Aziende Sanitarie Locali	3
1.2 Il progetto <i>Smart Health 2.0</i>	4
1.3 L'insufficienza renale cronica	6
1.3.1 Le cause	6
1.3.2 La stadiazione dei pazienti	7
1.3.3 Le terapie dialitiche	8
1.3.4 Impatto clinico e sociale	11
1.3.5 Attori coinvolti nella gestione della patologia	15
2 Il sistema <i>Viewpoint</i>	17
2.1 Lo scopo del progetto	19
2.2 Requisiti funzionali	22
2.2.1 Basi di dati collegate	24
2.2.2 Algoritmi e regole cliniche	25
2.3 Architettura applicativa	29
2.3.1 Servizi REST	31
2.3.2 L'architettura EJB	33
2.4 Tecnologie utilizzate	35
2.5 Interfaccia utente	42

3	La sezione <i>Analytics</i>	45
3.1	La configurazione dei parametri	47
3.2	La sezione <i>Input data</i>	51
3.3	La sezione <i>Simulation</i>	55
3.3.1	<i>Actuals and trend</i>	55
3.3.2	<i>Costs</i>	59
3.3.3	<i>Resource</i>	62
	Conclusioni	67
	Bibliografia	69

Elenco delle figure

1.1	Logo del progetto <i>Smart Health 2.0</i>	4
1.2	Schematica del funzionamento dell'emodialisi ospedaliera . . .	9
1.3	Schematica di funzionamento per la dialisi peritoneale	10
1.4	Schematica del trapianto di rene	11
2.1	Schema per l'architettura applicativa di <i>Viewpoint</i>	30
2.2	Architettura di <i>Viewpoint</i> e relative tecnologie utilizzate . . .	35
2.3	Diagramma a stella che rappresenta il data warehouse utilizzato	38
2.4	Organizzazione delle schermate utente dell'applicazione	43
3.1	Screenshot della sezione <i>Input data</i>	51
3.2	Screenshot della sezione <i>Simulation</i>	56
3.3	Screenshot della sezione <i>Simulation</i>	57
3.4	Screenshot della sezione <i>Simulation</i>	57
3.5	Screenshot della sezione <i>Costs</i>	60
3.6	Screenshot della sezione <i>Costs</i>	60
3.7	Screenshot della sezione <i>Resource</i>	62

Elenco delle tabelle

1.1	Mortalità per tutte le cause al ridursi del GFR (dato fornito da [3])	13
3.1	Mapping livelli - unità organizzative afferenti al SSN	47
3.2	Sintassi e semantica delle regole di configurazione che è possibile impostare dalla sezione <i>Configuration</i> . La colonna Aggregazione indica con quale operazione vengono aggregati i dati delle unità di livello superiore rispetto ai dati impostati nelle foglie dell'albero.	50

Capitolo 1

Il contesto

La rete organizzativa dei servizi socio-sanitari è molto complessa: le risorse, in disponibilità limitata, devono rispondere alle esigenze estremamente diversificate di un'utenza estesa, e devono quindi essere utilizzate con razionalità ed evitando sprechi. In quest'ottica è quindi molto importante un'integrazione fra servizi sanitari e sociali al fine di attivare interventi per la promozione della salute, tali da prevenire, individuare e minimizzare gli esiti degenerativi di alcune patologie sulle quali è possibile intervenire. In questo contesto si colloca il progetto *Smart Health 2.0* che si pone come obiettivo lo sviluppo di servizi ICT per la sanità pubblica al fine di migliorare la qualità del servizio alla cittadinanza. Il gruppo *NoemaLife*, fondato in Italia nel 1996, è divenuto leader europeo nel mercato dell'informatica clinica ospedaliera grazie allo sviluppo di software destinati alle organizzazioni sanitarie sia pubbliche che private che mira ad ottimizzare il flusso di lavoro delle strutture sanitarie a livello dipartimentale, ospedaliero e territoriale. *NoemaLife* è tra i responsabili scientifici del progetto, in collaborazione con enti di ricerca, istituti universitari e imprese private.

1.1 L'organizzazione del SSN

Il **Sistema Sanitario Nazionale (SSN)** ha il compito principale di garantire l'assistenza sanitaria a tutti i cittadini senza distinzioni di genere, residenza, età, reddito e lavoro. Esso assicura quindi un accesso ai servizi nel rispetto dei principi della dignità della persona, dei bisogni di salute, di equità, qualità, appropriatezza delle cure e economicità nell'impiego delle risorse. I cittadini effettuano la libera scelta del luogo di cura e dei professionisti nell'ambito delle strutture pubbliche e private accreditate ed esercitano il proprio diritto alla salute per ottenere prestazioni sanitarie, inclusive della prevenzione, della cura e della riabilitazione.

Il SSN è articolato secondo due livelli di responsabilità di governo:

- il livello **centrale**: lo Stato deve garantire a tutti i cittadini il diritto alla salute
- il livello **regionale**: le Regioni hanno una competenza esclusiva nella regolamentazione ed organizzazione di servizi e di attività destinate alla tutela della salute e dei criteri di finanziamento delle Aziende Sanitarie Locali (ASL) e delle aziende ospedaliere.

Si è infatti individuato il decentramento a livello regionale dei servizi sanitari come strumento per rispondere in modo più efficace ai bisogni della popolazione. In quest'ottica, gli enti pubblici coinvolti nella gestione del SSN sono:

il Ministero della Salute: gli sono attribuite le funzioni, prima spettanti allo Stato, di tutela della salute pubblica, di coordinamento del SSN, di sanità veterinaria, di tutela della salute nei luoghi di lavoro, di igiene e sicurezza degli alimenti.

enti e organi di livello nazionale: quali il **Consiglio Superiore di Sanità (CSS)**, l'**Istituto superiore di Sanità (ISS)**, **Istituto Superiore per la Prevenzione e Sicurezza del Lavoro (ISPESL)**, gli

Istituti di Ricovero e Cura a Carattere Scientifico (IRCCS),
etc.

enti e organi di livello regionale: fra le quali le **Aziende Sanitarie Locali (ASL)**, l'**Ufficio Relazioni con il Pubblico (URP)**, il **Centro Unico di Prenotazione (CUP)**, le **Aziende Ospedaliere**, etc.

1.1.1 Le Aziende Sanitarie Locali

L'Azienda Sanitaria Locale (ASL) provvede all'erogazione dell'assistenza sanitaria attraverso i dipartimenti di prevenzione, i distretti sanitari di base e i presidi ospedalieri. L'organizzazione dipartimentale è il modello ordinario di gestione operativa di tutte le attività delle aziende sanitarie. In particolare, il dipartimento di prevenzione è una struttura dotata di autonomia organizzativa e contabile ed organizzata in centri di costo e di responsabilità.

La legge regionale disciplina l'articolazione in distretti dell'Unità Sanitaria Locale. Il distretto è individuato garantendo una popolazione minima di almeno sessantamila abitanti e assicura i servizi di assistenza primaria relativi alle attività sanitarie e sociosanitarie, nonché il coordinamento delle proprie attività con quella dei dipartimenti e dei servizi aziendali, inclusi i presidi ospedalieri. Il distretto è dotato di autonomia tecnico-gestionale ed economico-finanziaria, e gli sono attribuite risorse definite in rapporto agli obiettivi di salute della popolazione di riferimento.

Gli ospedali che non hanno natura aziendale costituiscono i presidi della ASL. Ai presidi ospedalieri è attribuita autonomia economico finanziaria, secondo le disposizioni previste per le aziende ospedaliere in quanto applicabili.

1.2 Il progetto *Smart Health 2.0*

Smart Health 2.0 è un progetto della durata di 30 mesi cofinanziato dal Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) finalizzato alla promozione dell'innovazione del Sistema Sanitario attraverso tecnologie ICT, che consentano lo sviluppo di nuovi servizi di sanità digitale, insieme ad una migliore organizzazione territoriale delle risorse. *NoemaLife*, gruppo internazionale che opera nel mercato dell'informatica per la sanità con sede a Bologna, è tra i responsabili scientifici del progetto. L'obiettivo è di creare di un'infrastruttura tecnologica innovativa sulla quale sviluppare diversi servizi ad alto valore aggiunto, per consentire l'attivazione di nuovi modelli di attività nell'area della salute e del benessere. I fondi sono destinati al sostegno delle attività di ricerca e innovazione in Puglia, Calabria, Sicilia e Campania al fine di promuovere l'avanzamento tecnologico del Mezzogiorno d'Italia.



Figura 1.1: Logo del progetto *Smart Health 2.0*

Smart Health 2.0 è articolato in tre ambiti: infrastrutturale, applicativo e sperimentale. Tra gli obiettivi principali, vi è lo sviluppo del Fascicolo Sanitario Elettronico di seconda generazione, che supporterà non solo dispositivi mobili interoperabili e nuove modalità di fruizione dei servizi, ma diventerà un punto unico di aggregazione delle informazioni sullo stato di salute del cittadino, la sua storia clinica e le cure in corso. Altro obiettivo è la riduzione dei tempi di attesa dei pazienti e l'ottimizzazione dell'erogazione dei servizi sanitari. Infine, sono previste attività di ricerca su temi di tele-monitoraggio (ad esempio di pazienti dialitici, ma anche diabetici, cardiopatici, malati di cancro), tele-assistenza e *ambient assisted living*.

La sperimentazione verrà attuata in diverse regioni, collegate attraverso un laboratorio che coordinerà virtualmente le attività di ricerca e i siti di sperimentazione. Tale piattaforma integrata permetterà l'ottimizzazione della gestione dei processi sanitari nelle diverse realtà territoriali, in particolare saranno analizzati i flussi di pazienti all'interno dell'ospedale e saranno definite tecniche e modelli innovativi per incrementare l'efficienza e l'efficacia delle prestazioni erogate nei singoli reparti ospedalieri e massimizzare la soddisfazione dei pazienti. All'interno del progetto integrato si affronteranno anche studi relativi a modelli matematici e statistici per analisi predittive affidabili; metodi di elaborazione dei parametri clinici per l'interscambio dei dati eterogenei provenienti da vari sistemi clinico-ospedalieri, diagnostici, ecc.; standardizzazione dei protocolli clinici, anche attraverso un'integrazione con il Fascicolo Sanitario Elettronico.

Il software prodotto verrà sottoposto a diverse attività di testing che ne verifichino la correttezza funzionale, la correttezza di sistema e la correttezza di integrazione. Dopo essere state testate in isolamento, le componenti verranno integrate e testate in maniera funzionale per assicurare che restituiscano i risultati desiderati. Il testing di integrazione è un processo che definisce e verifica l'interoperabilità del software e la sua cooperazione con le applicazioni sviluppate nei vari domini applicativi.

Da un lato la piattaforma permetterà quindi il passaggio da dato a conoscenza, attraverso un processo di strutturazione semantica, garantendo interoperabilità e possibilità inferenziali; dall'altro, le sue possibilità classificatorie permetteranno di individuare e interpretare i dati relativi allo specifico paziente, garantendo un miglioramento globale del processo di cura, attraverso le attività di monitoraggio e diagnostica, oltre all'analisi integrata dei fattori di rischio e delle possibilità di occorrenza di eventi avversi nei percorsi di cura interni ed esterni alle strutture ospedaliere.

1.3 L'insufficienza renale cronica

I reni sono organi che costituiscono l'apparato urinario. Si occupano principalmente di filtrare dal sangue i prodotti di scarto del metabolismo per l'espulsione tramite l'urina, di regolare i livelli critici di sale, potassio e sostanze acide nel corpo, e producono ormoni che incidono sulle funzionalità degli altri organi, ad esempio ormoni che stimolano la produzione di globuli rossi, o che regolano la pressione sanguigna. I reni agiscono quindi sugli equilibri chimici del corpo, rimuovendone sostanze di scarto, riequilibrando i fluidi, rilasciando ormoni e producendo vitamina D, funzionale allo sviluppo osseo.

Si definisce **malattia renale cronica** la presenza di danno renale da almeno tre mesi, rilevabile attraverso una riduzione del filtrato glomerulare ($GFR < 60ml/min/1.73m^2$), o tramite anomalie rilevate ecograficamente. Essa può essere di tipo strutturale, se legata ad alterazioni anatomopatologiche, o di tipo funzionale se collegata ad alterazioni biochimiche.

1.3.1 Le cause

Vi sono molte cause legate all'insufficienza renale, alcune delle quali sono ereditarie, altre congenite. Il **diabete** è fra queste: causa una scarsa produzione di insulina o rende il corpo non in grado di utilizzarla adeguatamente, portando ad alti livelli di glucosio nel sangue, pericolosi per la salute del corpo. L'**ipertensione** è un'altra delle cause più comuni di insufficienza renale e di altre complicazioni cardiache. L'eccessiva pressione sanguigna fa aumentare la forza esercitata del sangue contro le pareti delle arterie. Quando la pressione è contenuta, il rischio di complicazioni renali diminuisce. La **glomerulonefrite** è una patologia che causa infiammazioni dei glomeruli, le unità di filtraggio contenute nei reni. In certi soggetti, questa patologia compare improvvisamente e si ripresenta solo in maniera intermittente. Tuttavia, la malattia può svilupparsi lentamente nel corso degli anni e portare alla progressiva perdita della funzionalità renale. Il **rene policistico** è la for-

ma più comune di malattia renale ereditaria, caratterizzata dalla formazione di cisti all'interno del rene che si dilatano col tempo, fino a causare gravi danni all'apparato urinario. Altre nefropatie ereditarie sono la sindrome di Alport, l'iperossaluria e la cistinuria. I **calcoli renali** sono molto comuni, e causano dolori alla schiena e nelle parti laterali del corpo. Fra le cause, vi è un disordine ereditario che porta all'eccessivo assorbimento di calcio dalle sostanze nutritive, e infezioni del tratto urinario o ostruzioni. Dieta e cure adeguate possono aiutare a prevenire la formazione ricorrente dei calcoli renali. Le **patologie congenite** solitamente riguardano malformazioni a livello di apparato urinario che si sviluppano durante la crescita nel ventre materno. Anche particolari farmaci e tossine possono causare problemi ai reni. Ad esempio, l'utilizzo prolungato di antidolorifici, certi tipi di pesticidi e droghe, ad esempio l'eroina, contribuiscono ad alterare la funzionalità renale.

1.3.2 La stadiazione dei pazienti

La stadiazione dei pazienti che sono affetti da malattia renale cronica prende in considerazione la riduzione della funzione renale, intesa come velocità di filtrazione glomerulare (VFG o GFR), ossia il volume di liquido filtrato dai glomeruli nell'unità di tempo. La patologia è diagnosticabile da esami di laboratorio che evidenziano presenza di albumina, proteine o tracce di sangue nelle urine, ecografie renali in grado di individuare alterazioni patologiche o tramite biopsia. La presenza di tali segni consente di diagnosticare la malattia renale cronica anche quando la velocità di filtrazione glomerulare è ancora normale o aumentata (stadio 1) o solo lievemente ridotta (stadio 2).

Stadio 1 Segni di danno renale con GFR normale o aumentato ($\text{GFR} > 90$ ml/min)

Stadio 2 Segni di danno renale con lieve riduzione del GFR (GFR tra 60 e 89 ml/min)

Stadio 3 Riduzione moderata del GFR (GFR tra 59 e 30 ml/min). Lo stadio viene suddiviso ulteriormente in 3A e 3B.

Stadio 4 Grave riduzione del GFR (GFR tra 29 e 15 ml/min)

Stadio 5 Insufficienza renale terminale o uremia (GFR < 15 ml/min o paziente in terapia sostitutiva)

1.3.3 Le terapie dialitiche

Il quinto stadio è detto terminale poichè i reni hanno perso quasi del tutto la loro funzione: nel paziente questo comporta ritenzione di acqua e di altre sostanze che si accumulano fino a dar luogo alla cosiddetta sindrome uremica. Ad essa conseguono manifestazioni ematologiche, cardiocircolatorie, nervose, endocrine, ossee, etc. In questa fase lo specialista nefrologo deciderà una strategia terapeutica sostitutiva della funzione renale. Le soluzioni possibili sono:

Emodialisi, HD È un trattamento che prevede l'utilizzo di un'apposita macchina chiamata rene artificiale, la quale utilizza un filtro dializzatore attraverso cui viene depurato il sangue, che viene restituito al paziente privo di scorie e dell'acqua in eccesso, accumulatasi a causa della nefropatia. La possibilità di collegamento alla macchina per la dialisi avviene tramite un accesso vascolare chiamato fistola artero-venosa o tramite protesi o catetere venoso centrale, che richiedono un piccolo intervento chirurgico per l'apposizione. La seduta di emodialisi dura in media 4 ore o più, ed è necessario eseguirla 3 volte la settimana. Le tre principali tipologie di emodialisi sono:

Bicarbonato, Bic Si usa un liquido dialitico a concentrazioni stabili di bicarbonato, che porta ad una migliore stabilità cardiovascolare e ad una riduzione dei sintomi dialitici.

Emodiafiltrazione, HDF Metodica che combina l'emodialisi con l'emofiltrazione, una tecnica che consente di eliminare macromolecole dal sangue.

sostanze tossiche e i liquidi in eccesso. Distinguiamo due tipologie di PD:

Dialisi peritoneale continua ambulatoriale, CAPD In questo tipo di dialisi si mantiene la soluzione dialitica nella cavità peritoneale per sottoporre il sangue a depurazione continua. La soluzione, solitamente in quantitativi intorno ai 2 litri o più, viene sostituita a intervalli regolari nel corso della giornata con una frequenza di 4 o 5 volte al giorno. Lo scambio dura circa 30/40 minuti e, una volta riempito l'addome, il paziente può svolgere liberamente le attività quotidiane.

Dialisi peritoneale automatizzata, APD È un trattamento che si avvale di una macchina programmabile dal medico che si occupa di eseguire gli scambi durante il sonno notturno a seconda delle esigenze depurative. Il principio e la depurazione sono identici alla CAPD ma in questo caso gli scambi vengono eseguiti automaticamente dalla macchina.

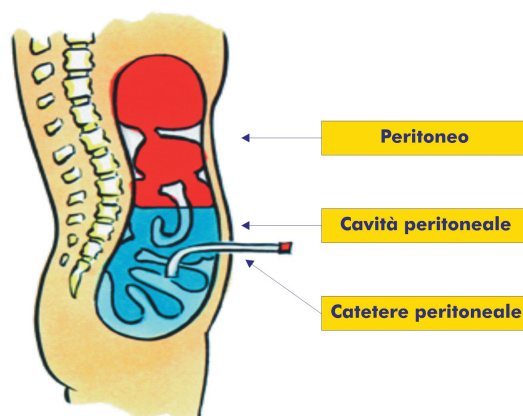


Figura 1.3: Schematica di funzionamento per la dialisi peritoneale

Trapianto, TR Il trapianto renale si esegue con un intervento chirurgico che consiste nell'inserire un rene sano, prelevato da un donatore cadavere o vivente, nella parte anteriore dell'addome del paziente. L'intervento di

trapianto dura dalle 2 alle 4 ore e può essere eseguito su pazienti di età compresa tra pochi mesi di vita fino a oltre 75 anni di età. Nella maggior parte dei casi, i reni nativi non vengono rimossi, a meno che non si ritenga che possano essere causa di complicazioni cliniche successive. È capace di restituire una normale funzionalità renale e permettere alla maggior parte dei pazienti il ritorno a una vita socialmente produttiva. È sufficiente un solo rene per condurre una vita normale, senza comunque un eccesso di abusi alimentari che possano provocare un rischio di insufficienza renale. Il paziente trapiantato dev'essere sottoposto a periodici controlli: dopo la dimissione, si eseguono 3 controlli settimanali per un mese, dopodiché la frequenza dei controlli gradualmente diminuisce fino ad arrivare a un controllo al mese dopo il primo anno e ancora meno frequenti successivamente.

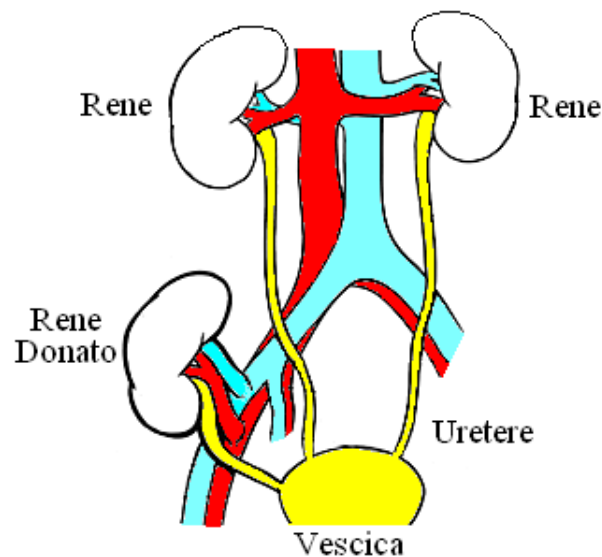


Figura 1.4: Schematica del trapianto di rene

1.3.4 Impatto clinico e sociale

La gestione a lungo termine dei nefropatici richiede un attento monitoraggio della percezione soggettiva dello stato di salute. Le misure standard

di qualità della vita consentono di identificare specifici problemi e compromissioni della funzionalità globale dei pazienti, che possono ostacolarne la capacità di adattamento alla malattia e ai trattamenti richiesti. Lo studio *ANQoL (Apulia Nephrology Quality of Life)* [4] ha dimostrato come il sesso femminile, l'avanzata scolarizzazione e la giovane età si associno alla percezione soggettiva di bassa qualità della vita, sia nelle componenti fisiche che mentali, già dalle fasi iniziali della malattia renale cronica. I dati riportati in letteratura dimostrano come la percezione soggettiva della qualità della vita da parte dei pazienti in terapia dialitica sia significativamente peggiore di quella dei soggetti normali.

Se oltre alla insoddisfacente qualità della vita del paziente uremico, si valutano anche le consistenti risorse economiche impegnate per garantire il programma nazionale di dialisi, stimate intorno a 35.000,00 €/ paz / anno [5], ci si rende conto quanto sia necessaria l'adozione di un progetto assistenziale che miri alla riduzione della progressione della malattia renale cronica verso l'uremia. A tal proposito, è dimostrato che la frequenza di consultazioni nefrologiche in un anno si associa ad una minore incidenza di uremia, e che la terapia con farmaci attivi è in grado di **ritardare l'inizio della terapia dialitica di due anni in un paziente ogni 28 pazienti trattati**. Tale approccio programmatico risulta vantaggioso oltre che per motivi di ordine etico e sociale, anche dal punto di vista organizzativo ed economico per il SSN, che riuscirebbe così a garantire un maggior numero di risorse ospedaliere e assistenziali.

La World Health Organization (WHO) sottolinea che l'epidemia che l'umanità deve affrontare prioritariamente non è più tanto connessa alle malattie infettive, quanto invece alle malattie croniche, quali il cancro, l'infarto del miocardio, la malattia cerebro-vascolare e la malattia vascolare periferica. La WHO tra queste malattie non cita la malattia renale cronica, nonostante questa patologia ne costituisca un'importante componente, dal momento che colpisce circa il 10% della popolazione mondiale.

Dopo anni in cui il fenomeno è stato ignorato, ora tra i nefrologi è un fatto

gfr stimato	Mortalità per tutte le cause (Adjusted Hazard Ratio)
$\geq 60\text{ml/min/1.73m}^2$	1.00
$45 - 59\text{ml/min/1.73m}^2$	1.2 (1.1 - 1.2)
$30 - 44\text{ml/min/1.73m}^2$	1.8 (1.7 - 1.9)
$15 - 29\text{ml/min/1.73m}^2$	3.2 (3.1 - 3.4)
$< 15\text{ml/min/1.73m}^2$	5.9 (5.4 - 6.5)

Tabella 1.1: Mortalità per tutte le cause al ridursi del GFR (dato fornito da [3])

assodato che l'insufficienza renale moderata è molto frequente nella popolazione generale. In Italia, solo negli ultimi anni l'attenzione dello specialista nefrologo e degli amministratori si è rivolta verso le misure di frequenza della malattia renale cronica nella popolazione generale italiana. La insufficiente attenzione a tale problema e, quindi, alle possibili misure di prevenzione, può essere chiamata in causa, insieme ad altre concause, per spiegare l'aumento di circa 5% dell'incidenza dell'uremia cronica terminale, ovvero la necessità di iniziare la terapia dialitica, registrata negli ultimi dieci anni.

Lo studio *CARHES* (*CARDiovascular risk in Renal patients of the Health Examination Survey*) [2] ha rilevato in Italia la prevalenza di malattia renale cronica nell' 8% circa della popolazione, si può quindi ragionevolmente ipotizzare che oltre due milioni e mezzo di soggetti ne siano affetti. I risultati preliminari dello stesso studio evidenziano una minore prevalenza in Italia rispetto ad altre realtà europee ed extraeuropee (10-12%). Inoltre, essa è più elevata negli stadi iniziali (circa 9%) della malattia rispetto a quella registrata negli stadi più avanzati della stessa (circa 6%).

L'impatto clinico è di notevole portata. Uno studio inglese [3] ha dimostrato che, indipendentemente da altri fattori di rischio, alla progressiva riduzione della funzione renale si associa in maniera diretta un aumento del rischio di morte per ogni causa fino a circa 6 volte (v. Tabella 1.1).

L'uremia cronica terminale rappresenta invece, l'intervento clinico più fre-

quente nei pazienti giunti negli stadi avanzati (quarto e quinto). L'incidenza dell'uremia cronica terminale nella regione Puglia nell'ultimo quinquennio è in tendenziale aumento, fino a valori che si aggirano intorno ai 700 nuovi pazienti all'anno. Dal programma di dialisi escono per decesso circa il 12% dei pazienti nell'arco di un anno. Tenendo conto anche delle uscite dei pazienti per trapianto renale e il loro ritorno al trattamento dialitico nella misura del 15%, l'analisi di tali dati evidenzia che, comunque, si determina un saldo attivo di almeno 200 uremici. Se questi nuovi pazienti dovessero inserirsi nel programma di dialisi ospedaliera, occorrerebbe l'istituzione di nuovi centri, investendo in nuove risorse umane ed economiche. Infatti i centri ospedalieri attualmente esistenti sono oberati dal lavoro di routine e spesso operano su tre turni giornalieri. Anche da queste considerazioni sorge la necessità di deospedalizzare la dialisi. L'azione programmata sull'assistenza ai nefropatici cronici prevede infatti, tra le finalità, il potenziamento dell'attività di dialisi in sedi non ospedaliere, incentivando la dialisi peritoneale. Attualmente sono in dialisi peritoneale 292 su 3940 pazienti (dato 2011) pari al 7,4 %.

Lo studio condotto da [3] ha inoltre rilevato un'associazione fra rischio di eventi cardiovascolari e di ospedalizzazione. Questi infatti, si incrementano rispettivamente fino a 3,4 e a 3,1 volte man mano che la patologia renale progredisce verso gli stadi più avanzati. Nel triennio 2009-2011 si è registrato un incremento di ricoveri ordinari per insufficienza renale cronica dell'11% (da 66.811 a 73.964), determinando un incremento del tasso di ospedalizzazione da 13 a 15 ricoveri ogni 10000 abitanti. Si può ipotizzare che tale incremento vicari la riduzione del numero delle prestazioni ambulatoriali, a sua volta, conseguente alle perduranti difficoltà economiche vissute dalla popolazione. In Regione Puglia, nel triennio sopra considerato, l'andamento del numero di ricoveri per la stessa patologia è pressoché stazionario (circa 7.300), a fronte, però, di un tasso di ospedalizzazione costantemente più elevato (18 ricoveri/10000 ab). Questo può essere anche legato alla mancata attuazione del processo di integrazione delle attività assistenziali ospedaliere con quelle ter-

ritoriali. Essa si realizza attraverso la strutturazione di un modello gestionale che, attraverso la creazione di una efficiente rete di collegamento tra le strutture sanitarie del SSN, sia in grado di garantire la continuità assistenziale a pazienti a rischio di nefropatie o con ipertensione arteriosa e/o diabete.

1.3.5 Attori coinvolti nella gestione della patologia

Il follow-up di pazienti con MRC agli stadi 1, 2, 3A è affidato al medico di medicina generale, mentre quello dei pazienti agli stadi 3B, 4 e 5 è affidato allo specialista nefrologo ospedaliero o territoriale secondo protocolli condivisi, nei quali sono previsti anche i percorsi per le eventuali urgenze o complicanze intercorrenti.

Le Unità Organizzative di nefrologia e dialisi si avvalgono di altre figure professionali e operano coordinate dal nefrologo in team multidisciplinare con infermieri, farmacisti, psicologi, dietisti e assistenti sociali. Per competenza svolgono interventi individuali e collettivi di prevenzione delle nefropatie, ricoveri e terapie in degenza ospedaliera, attività di day-hospital nefrologico, attività di assistenza domiciliare in collaborazione con il Distretto e il medico di medicina generale, attività ambulatoriali, selezione pazienti per l'inserimento e permanenza in lista di attesa per trapianto renale.

Il medico di medicina generale o pediatra di libera scelta opera in campo preventivo, diagnostico e terapeutico per quanto di sua competenza, in particolare è attivo nella individuazione dei principali fattori di rischio delle malattie renali e nella diagnosi precoce attraverso l'utilizzo di semplici test diagnostici e valutazioni cliniche quali misurazione della pressione arteriosa, valutazione del BMI, misurazione della circonferenza addominale. Collabora inoltre con lo specialista nefrologo e altri attori nell'elaborazione ed esecuzione di protocolli per la gestione delle complicanze della malattia renale in tutte le sue fasi all'interno di una logica di continuità terapeutica.

Infine il distretto sanitario locale attua interventi di prevenzione primaria esaminando i possibili determinanti sociali ed economici delle malattie

renali (ad es. attraverso questionari somministrati alle fasce di popolazione a rischio.

Capitolo 2

Il sistema *Viewpoint*

Le patologie croniche quali cardiopatie, nefropatie, obesità e diabete sono difficilmente individuabili in quanto inizialmente asintomatiche. Tuttavia, è possibile prevedere con un certo anticipo gli individui potenzialmente a rischio tramite l'applicazione di algoritmi specifici ai risultati di analisi di laboratorio.

L'applicazione *Viewpoint* nasce dall'esigenza di migliorare il processo di instradamento dei pazienti verso il percorso di cura più appropriato, rilevando le potenziali criticità e notificandole al medico di medicina generale, che potrà così valutare il percorso di cura migliore per il proprio paziente. Il progetto si pone quindi come scopo ultimo il miglioramento della qualità della vita, permettendo anche il risparmio di tempo e risorse pubbliche messe a disposizione dal SSN.

Il primo prototipo di *Viewpoint* è stato sviluppato da un team di Noema-Life con sede a Bari, coordinato con un team a Bologna. La prima sperimentazione e validazione avverrà presso il Policlinico di Bari, ed è prevista per dicembre 2014. In questa versione le funzionalità di *Viewpoint* sono legate all'individuazione e analisi di potenziali nefropatie, mentre in futuro verrà esteso per poter individuare ulteriori patologie croniche asintomatiche.

Il sistema software è costituito da un applicativo web per favorirne la portabilità, che utilizza servizi REST e sfrutta l'architettura EJB. Si struttura

su tre macro-livelli gerarchici:

- il livello **presentazione**, costituito dall'interfaccia grafica operante su piattaforma client
- il livello **applicazione**, lo strato in cui si realizza la logica applicativa del sistema e in grado di comunicare con lo storage. Risiede sull'application server, che può essere una macchina logicamente e/o fisicamente distinta
- il livello **dati**, lo strato di persistenza delle informazioni costituito dal database.

Viewpoint è stato sviluppato per la maggior parte ricorrendo a tecnologie open source, quali Pentaho CE per la logica di business intelligence, JBoss Application Server per il deploy dell'applicazione, HTML5, JQuery, Bootstrap e Angular JS per l'implementazione del front end.

2.1 Lo scopo del progetto

L'analisi dello scenario evolutivo che si presenta in relazione all'aumento della vita media della popolazione mostra chiaramente, con l'avanzare dell'età, l'insorgenza di patologie croniche. In particolare, sono in costante aumento pazienti affetti da nefropatie, cardiopatie, diabete e obesità, patologie croniche accomunate dalle forti limitazioni nello stile di vita dei pazienti affetti dalla malattia e dall'asintomaticità, nonché da un alto impatto economico sul SSN. In particolare il focus è rivolto verso la malattia renale cronica (MRC). Poiché asintomatica, riveste particolare importanza tentare di limitare quanto più possibile il problema del *late referral*, per cui si arriva a conoscenza della patologia solo in uno stadio avanzato per il quale l'unica alternativa è il trattamento con terapia sostitutiva (emodialisi) che nei confronti della persona ha un forte impatto sullo stile di vita e sul SSN ha un forte impatto in termini di costi e risorse. Favorire la diagnosi precoce delle malattie asintomatiche può rivelare eventuali cause ancora reversibili della patologia e rallentare il decorso della MRC, in modo da migliorare la qualità della vita del paziente ritardando quanto più possibile un trattamento invasivo.

La prassi prevede che, all'insorgere di un sintomo, il paziente richieda l'intervento del proprio medico di base, il quale può richiedere accertamenti mediante un'impegnativa per la prenotazione ed esecuzione degli esami di laboratorio ritenuti adeguati. Il paziente, una volta ricevuto il proprio referto, ha in carico la notifica al proprio medico che, valutata la situazione, decide se richiedere un approfondimento specialistico tramite una nuova prenotazione e conseguente visita, a seguito della quale lo specialista valuterà l'opportunità o meno di instradare il paziente verso un percorso di cura. Tale impostazione prevede quindi che il processo parta dal paziente e/o dal suo medico di base verso il medico specialista, approccio che può non essere sempre efficace per arrivare ad una diagnosi precoce della patologia. Fermo restando la validità del modello che prevede un punto di riferimento per il paziente rappresentato dal medico di base, si ritiene possibile fornire un ausilio all'attuale

organizzazione finalizzato ad una sistematica e più efficace segnalazione di potenziali problemi legati a specifiche patologie croniche e asintomatiche. In questo modello infatti si prevede che il medico di base nella valutazione degli accertamenti richiesti tenga conto di un quadro generale delle condizioni, andando ad evidenziare anche problematiche non strettamente correlate alla richiesta. Questo punto è forzatamente disomogeneo e strettamente legato al fattore umano cui è lasciato il compito di intravedere anche altri problemi rispetto al motivo che ha scatenato l'indagine diagnostica. Inoltre, molteplici altri fattori condizionano questo percorso, ad esempio l'assenza di una regola che impone l'esecuzione di controlli periodici. In ogni caso il percorso prevede una grossa attività in carico sia al paziente che al medico di medicina generale.

Tutto ciò porta ad un'anomalia organizzativa che spesso si verifica nei servizi di nefrologia e dialisi italiani, nei quali l'ambulatorio, punto di ingresso e di primo contatto con i nuovi potenziali pazienti, osserva numerosi falsi positivi, ovvero pazienti erroneamente segnalati ed invitati ad effettuare una visita nefrologica di controllo, a discapito di coloro che invece hanno una effettiva necessità di essere instradati verso un percorso di cura in nefrologia.

In questo contesto un sistema automatico può migliorare il paradigma di notifica dal medico di medicina generale a specialista automatizzando l'analisi di grandi moli di dati con uniformità di interpretazione. È qui che nasce l'applicativo *Viewpoint* il quale, sfruttando i dati già presenti in grande quantità nei laboratori analisi presso le strutture ospedaliere, elabora ed estrae indicatori in grado di supportare i clinici nel loro lavoro.

Una volta operativo, il sistema indagherà immediatamente lo storico disponibile, segnalando tutti i casi problematici rilevati grazie a specifici indicatori di patologia; dopodiché, al progredire degli esami archiviati e richiesti, segnalerà nuovi potenziali casi sospetti da verificare. Il coinvolgimento del medico e soprattutto del paziente avverrà solamente a valle di un doppio controllo (specialista - medico di base) che in team decideranno se coinvolgere il paziente e approfondire una potenziale problematica. Gli elementi innovativi

di tale approccio sono da ricercarsi nell'elaborazione di dati già esistenti per la segnalazione, nella collaborazione tra specialista e medico di base, e infine nella doppia verifica prima di coinvolgere il paziente.

2.2 Requisiti funzionali

I presupposti principali su cui l'applicazione è orientata sono:

- Il carattere generale: l'applicazione è generalizzabile, ovvero configurabile in modo che possa analizzare differenti ambiti clinici (nefrologia, ipertensione, ...)
- Separazione fra i diversi ambiti: gli ambiti di analisi rimangono comunque nettamente separati (*Viewpoint* per nefrologia sarà distinto da *Viewpoint* per ipertensione) così come, i parametri di configurazione necessari per le analisi what-if
- L'applicativo è sviluppato con una struttura dati progettata per gestire differenti lingue (di default l'italiano).

Per quanto concerne le **funzionalità**, il sistema permette:

- la definizione di appositi algoritmi indicatori su specifiche patologie (nefrologia in primis)
- l'applicazione di tali indicatori sulla somma dei dati provenienti da differenti sorgenti informative quali:
 - sistemi *Electronical Medical Record (EMR)* di NoemaLife (Galileo, DX Care, etc.)
 - sistemi di laboratorio di NoemaLife (DNLab, Powerlab, etc.)
 - referti documentali
- la visualizzazione di grafici, tabelle e mappe che mostrino i dati aggregati sulla popolazione presa in esame dagli indicatori specifici
- la possibilità di andare in drill down fino al singolo paziente
- la possibilità di notificare il medico di base o il paziente stesso in funzione del consenso alla privacy indicato

- la possibilità di effettuare simulazioni sul futuro sulla base dei dati storici, in modo da valutare aspetti clinici ed economici (analisi *what if*)

Il sistema ha come primo obiettivo l'identificazione della malattia renale cronica, senza dimenticare la possibilità di estensione per effettuare screening anche su altre patologie quali diabete, ipertensione ed obesità. Il primo prototipo di *Viewpoint* ha come prima finalità la stadiazione dei pazienti in classi di rischio a seconda dello stadio di evoluzione della nefropatia, in modo tale da analizzare la situazione patologica sul territorio. Ulteriore elemento abilitante della piattaforma è la possibilità di effettuare simulazioni a partire dallo storico a disposizione su quello che potrebbe essere lo scenario evolutivo della malattia: le simulazioni hanno come finalità l'individuazione delle necessità di programmazione sanitaria per l'area di competenza.

La validazione del sistema consiste nel verificare il decorso clinico nel tempo per un set di pazienti conosciuti ed affetti da insufficienza renale allo stadio 5, quindi sottoposti a trattamento dialitico sostitutivo, sfruttando i dati storici a disposizione nella base dati indagata e ricostruendo la curva di progressione della malattia. È inoltre prevista una fase in cui verrà modificata la procedura con la quale viene acquisito il consenso informato per il trattamento dei dati del paziente. Al momento della prescrizione degli esami, verrà introdotta una formula per richiedere al paziente la volontà di aderire allo screening dei dati, con la manifestazione della volontà di indagare sui dati storici e su quelli disponibili per il futuro. Il consenso informato prevederà anche la possibilità di essere informati direttamente dallo specialista che individuerà potenziali criticità, e di essere informati tramite il proprio medico di base, che rimarrà comunque parte attiva nel percorso e sarà comunque informato dello stato di salute del proprio assistito una volta rilevata una potenziale problematica.

2.2.1 Basi di dati collegate

I dati necessari all'applicativo si possono attingere dalle basi di dati legate agli applicativi oggi diffusi nelle strutture ospedaliere, quali dati anagrafici, dati dai laboratori di analisi, etc.

La prima sperimentazione di *Viewpoint* avverrà presso il Policlinico di Bari, utilizzando il sistema informativo relativo. Pertanto sono state prese in considerazione le seguenti basi dati:

Database Oracle di Powerlab e DNLab di NoemaLife Sistemi informativi per la gestione dei laboratori di analisi cliniche e microbiologiche.

Database Oracle di Galileo di NoemaLife Sistema informativo ospedaliero che contiene informazioni sui pazienti sia di tipo anagrafico e amministrativo, che di tipo medico (ordini di visite, prescrizioni, dati clinici, etc.)

Database di terze parti

Il sistema è comunque stato progettato per essere indipendente dalla sorgente dati di estrazione. Sarà poi possibile estendere le informazioni di partenza aggiungendo dati di altra provenienza, tramite la creazione di un processo ETL apposito.

La politica di utilizzo del sistema prevede:

1. Installazione e primo caricamento del DWH con tutti i dati storici disponibili
2. Certificazione, validazione e normalizzazione del DWH
3. Elaborazione dei dati secondo gli algoritmi predefiniti
4. Aggiornamento e rielaborazione con frequenza predefinita

Si ritiene sufficiente un aggiornamento del DWH effettuato con frequenza giornaliera, tipicamente durante le ore notturne, per consentire l'esecuzione delle procedure di ETL quando il carico di lavoro nelle strutture ospedaliere è minore.

2.2.2 Algoritmi e regole cliniche

Per arrivare a generare un quadro clinico d'insieme che riassume le condizioni cliniche del paziente, il sistema prende in considerazione i seguenti esami:

- Creatinina sierica
- Esame delle urine
- Glicemia
- Albuminuria
- Proteinuria
- Colesterolo
- Trigliceridi
- C3
- C4
- uACR
- uPCR

I valori recuperati sono normalizzati rispetto all'unità di misura considerata e utilizzati per l'applicazione di formule validate scientificamente che permettono l'individuazione dei pazienti a rischio.

Algoritmi per la stadiazione dei pazienti

La **creatininemia**, ossia la concentrazione di creatinina nel sangue (valori normali compresi fra 0.7 e 1.3 mg/dl), è un comune indicatore della funzione renale: un suo aumento è indice di un esteso danno ai nefroni. Tuttavia non è un parametro affidabile per la misura della funzione renale: in quanto prodotta dai muscoli ed eliminata dal rene, cresce al crescere delle masse muscolari. Pertanto, un valore di 1,2 mg/dl, espressione di una normale funzione renale in un soggetto con masse muscolari particolarmente sviluppate, può al contrario mascherare un'insufficienza renale in un soggetto con scarsa massa muscolare. Per questi motivi, la valutazione della funzione renale non può basarsi solo sulla creatininemia, ma è necessario determinare la VFG, ossia calcolare la velocità di filtrazione glomerulare.

Il metodo classico è il calcolo della **clearance della creatinina**, ossia il tasso di smaltimento della creatinina, basato sul dosaggio delle creatinina plasmatica e della creatinina urinaria delle ultime 24 ore.

$$\text{Creatinina Clearance (VFG)} = \frac{\text{Creatininuria} * \text{Diuresi delle 24 ore}}{\text{Creatininemia} * 1440} \quad (2.1)$$

Il limite della affidabilità del valore della clearance misurata in laboratorio risiede nell'elevata probabilità di un errore pre-analitico dovuto al paziente (inadeguata raccolta delle urine delle 24 ore, approssimativa determinazione del volume urinario).

Quando non è possibile o risulta poco affidabile la raccolta delle urine da parte del paziente, o comunque per verificare l'affidabilità della clearance misurata in laboratorio, è opportuno ricorrere alla **formula di Cockcroft-Gault**:

$$\text{Uomini VFG} = \frac{(140 - \text{Età}) * \text{Peso ideale}}{\text{Creatininemia} * 72} \quad (2.2)$$

$$\text{Donne VFG} = \text{VFG Uomini} * 0.85 \quad (2.3)$$

Nella formula è specificato peso ideale (calcolato tramite l'indice di massa corporea BMI), in quanto in soggetti magri od obesi la mancanza o l'eccesso

di grasso determina un errore di stima. Nei soggetti normopeso può essere invece utilizzato il peso reale.

La formula di Cockcroft-Gault necessita di informazioni aggiuntive sui pazienti, quali il peso e l'età, non sempre facilmente reperibili dalle base di dati a disposizione. Si può pertanto ricorrere al calcolo della VFG tramite la **formula MDRD (Modification of Diet in Renal Disease)**, che nella sua formula semplificata (sMDRD) risulta essere:

$$\text{VFG} = 186 * \text{Creat}^{-1.154} * \text{età}^{-0.203} * K_1 * K_2 \quad (2.4)$$

dove $K_1 = 1$ per individui di razza bianca, 1.21 per individui di colore. $K_2 = 1$ negli uomini, 0.742 nelle donne.

La MDRD semplificata è una formula affidabile (a parte gravide ed anziani) poichè non influenzata dai dati antropometrici e dalla variabilità della massa muscolare o lipidica. Inoltre il VFG calcolato è già rapportato alla superficie corporea, quindi non richiede altri calcoli.

Tuttavia ne esiste una versione più avanzata e accurata, che è quella attualmente utilizzata da *Viewpoint* per la stadiazione dei pazienti. La formula **CKD-EPI (Chronic Kidney Disease Epidemiology Collaboration)** (versione 2009), nota in letteratura, costituisce la più diffusa in ambito clinico per la grande accuratezza dei risultati ottenuti.

$$\text{VFG} = 141 * \min\left(\frac{\text{Cre}}{\text{Den}}, 1\right)^a * \max\left(\frac{\text{Cre}}{\text{Den}}, 1\right)^{-1.209} * 0.993^{\text{età}} [*1.018 \text{ se femmina}]$$

$$[*1.159 \text{ se di colore}] \quad (2.5)$$

dove Cre = Creatinina sierica, Den = 0,9 se maschio e 0,7 se femmina, $a = -0,329$ se femmina e $-0,411$ se maschio.

Algoritmi per la classificazione dei pazienti in base al trattamento dialitico

La definizione degli algoritmi per classificare i pazienti in base al trattamento dialitico a cui sono sottoposti è stato difficoltoso, in quanto nelle

basi di dati a disposizione non era presente il dato esplicito. È stato necessario dedurre questa informazione dall'applicazione di procedure sviluppate in collaborazione con i medici del policlinico di Bari, che tenessero conto non solo della tipologia di esami effettuati, ma anche della frequenza degli stessi. Tuttavia, i risultati prodotti da questi algoritmi non sono esatti, ma approssimativi e probabilistici, in quanto non sempre tutti i pazienti rispettano lo stesso iter di controlli ematici.

La classificazione ha tenuto conto delle seguenti regole:

- **Pazienti in emodialisi ospedaliera:** pazienti con almeno 6 esami mensili richiesti dal reparto di nefrologia e dialisi e contenenti creatinina, albuminuria, proteinuria
- **Pazienti in emodialisi domiciliare:** esami mensili effettuati al domicilio del paziente, dei quali il laboratorio riceve solo i risultati, e contenenti creatinina, albuminuria, proteinuria
- **Pazienti trapiantati:** pazienti con esami che contengano il controllo della ciclosporina almeno due volte nello stesso anno
- **Pazienti in dialisi peritoneale:** pazienti con profilo di esami mensili contenenti il controllo del liquido peritoneale

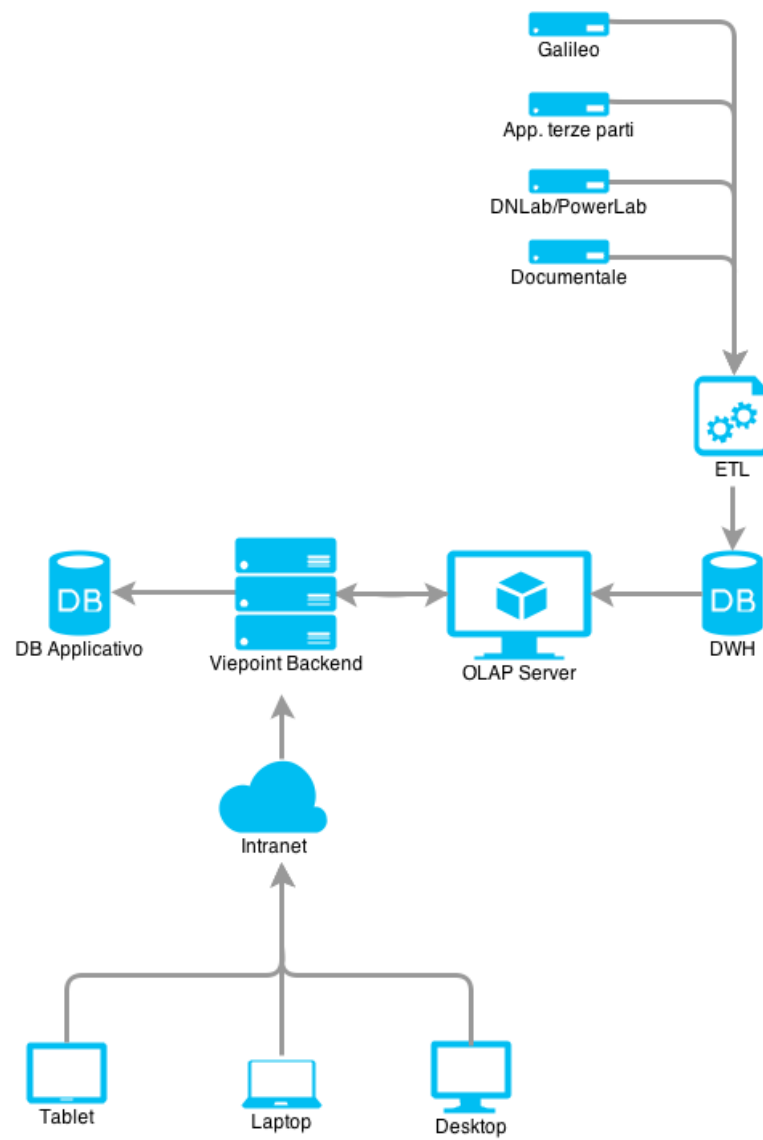
2.3 Architettura applicativa

La soluzione proposta è basata su un'architettura web per consentire una diffusione capillare dell'applicativo. L'applicazione si struttura nelle seguenti parti principali:

- Un servizio di **ETL** per estrazione, la trasformazione e il caricamento dei dati sul DWH
- un **data warehouse (DWH)** per la centralizzazione delle informazioni da sorgenti eterogenee
- un **OLAP server** per le interrogazioni multidimensionali sui contenuti del DWH
- un **application server** che esporrà la web application di *Viewpoint* (supportato da un apposito database applicativo) assieme ai servizi di business analytics, security, audit, configurazione e il gestore per le notifiche

Il sistema recupera risultati di laboratorio e informazioni relative agli episodi clinici dalle basi di dati a disposizione e, dopo un processo di omogeneizzazione operato tramite opportune trasformazioni, li carica sul data warehouse. Sfruttando quindi il supporto dell'OLAP Server, le informazioni vengono elaborate secondo un approccio multidimensionale, in grado di supportare le valutazioni del contenuto informativo attraverso meccanismi di drill down o drill up su specifiche aree di analisi.

I risultati ottenuti sono visualizzabili e manipolabili attraverso un'interfaccia web in grado di mostrare grafici e tabelle dei dati aggregati secondo precisi criteri di selezione, permettere la navigazione verso il dettaglio dello specifico paziente (abilitato in base al consenso al trattamento dei dati personali del paziente), gestire le operazioni di notifica verso il medico di base (abilitato in base al consenso al trattamento dei dati personali del paziente) ed effettuare simulazioni cliniche ed economiche sugli anni futuri, in base ai

Figura 2.1: Schema per l'architettura applicativa di *Viewpoint*

dati storici a disposizione. Per fare ciò, il frontend sfrutta l'architettura di tipo **REST**, effettuando richieste al backend tramite un URI che identifica la risorsa desiderata. Il backend si avvale di diversi **service** che elaborano la richiesta e restituiscono un risultato interrogando le componenti di *Security*, *Audit*, *Configuration* utilizzando le specifiche **EJB**, oppure interrogando il DWH grazie all'OLAP server Mondrian.

2.3.1 Servizi REST

REST è un insieme di vincoli architetturali che cercano di minimizzare la latenza di comunicazione, ma allo stesso tempo massimizzano l'indipendenza e la scalabilità dei componenti implementativi. Fa uso di meccanismi di caching, sostituzione dinamica dei componenti e riuso delle interazioni per soddisfare le esigenze di un sistema distribuito inserito all'interno dell'architettura Internet. In questo senso, l'insieme delle pagine web che formano un'applicazione è vista come una sorta di stato virtuale: ogni azione dell'utente ha come risultato una transazione al prossimo stato che avviene tramite il trasferimento di una rappresentazione dello stato dell'applicazione all'utente.

L'insieme di vincoli architetturali che impone REST caratterizzano l'interazione fra i vari componenti del sistema.

Architettura di tipo client-server Utilizzando questo modello di comunicazione, vengono disgiunti l'interfaccia utente dalla memorizzazione dei dati, migliorando la portabilità del front-end su più piattaforme e la scalabilità delle componenti back-end. Questa separazione consente alle due componenti di evolvere in maniera indipendente, supportando i requisiti del web.

Comunicazione *stateless* Il server non mantiene informazioni aggiuntive, bensì ogni richiesta contiene tutti i parametri necessari per reperire una determinata risorsa. Le informazioni sulla sessione vengono mantenute interamente in memoria dal client. Questo vincolo, come ogni scelta ar-

chitetturale, rappresenta un *tradeoff*: viene incrementata l'affidabilità e la scalabilità del sistema, a scapito dell'efficienza di rete causato da un overhead di comunicazione. Va inoltre considerato che la condizione di *statelessness* consente di processare richieste in parallelo senza la necessità di abilitare un meccanismo che mantenga la coerenza semantica fra gli stati del server ad ogni richiesta proveniente dal client.

Utilizzo di cache Per migliorare l'efficienza, si ricorre all'uso di meccanismi di caching delle risorse, che permettono di eliminare totalmente o parzialmente alcune interazioni client-server, riducendo la latenza media di comunicazione del sistema.

Organizzazione gerarchica Organizzare il sistema su più livelli gerarchici permette ai vari componenti di comunicare esclusivamente con i livelli adiacenti, rendendo invisibile l'implementazione sottostante o sovrastante. In questo modo viene posto un limite alla complessità totale del sistema a favore dell'indipendenza di ogni substrato.

REST quindi semplifica il modello di interazione fra le varie componenti, grazie alla separazione fra le implementazioni e i servizi che esse stesse forniscono. Il *tradeoff* è sull'efficienza del sistema: l'interfaccia uniforme fornita da REST comporta una perdita di efficienza in sistemi specifici, ma è ottimizzata per il Web, in cui vengono scambiate risorse eterogenee in grandi quantità.

Gli elementi del sistema vengono classificati a seconda del ruolo che svolgono all'interno dell'architettura:

1. i *data elements*, ovvero i dati, rappresentati da risorse
2. i *processing elements*, ovvero i componenti
3. i *connecting elements*, ovvero i connettori fra risorse e componenti

REST ignora i dettagli implementativi dei componenti per focalizzarsi sul ruolo che essi hanno all'interno del sistema, i vincoli sull'interazione con

gli altri componenti e la loro interpretazione dei dati significativi. La definizione astratta di risorsa fornisce genericità includendo fonti di informazioni eterogenee senza distinguerle per tipo o dettagli implementativi. In questo modo non è necessario modificare il collegamento alla risorsa, quando la sua rappresentazione cambia. REST infatti utilizza un identificativo univoco per indicare una particolare risorsa coinvolta in un'interazione fra i componenti. Ad esempio, se devo recuperare la lista dei pazienti di nefrologia del policlinico di Bari suddivisi per trattamento nel biennio 2012-2014, l'identificativo della risorsa sarà un URI nella forma:

```
http://{{BASE_URI}}/rest/analytics/pat_vs_tratment/12-14/POL_BA
```

Al variare della lista di pazienti, non cambierà l'identificativo. I connettori forniscono un'interfaccia per accedere ai valori di una risorsa e manipolarli, incuranti dell'implementazione software che gestisce la richiesta. Il componente incaricato di decidere la corrispondenza identificativo-risorsa, nel nostro caso l'implementazione del backend, è l'unico responsabile del mantenimento della validità semantica del mapping nel tempo.

2.3.2 L'architettura EJB

Gli **Enterprise JavaBean (EJB)** sono componenti software che, all'interno della piattaforma Java EE, implementano la logica di business di un'applicazione web lato server per fornire servizi lato client. Rappresentano dunque uno strato software residente su un application server all'interno di un'architettura multi-tier. Le specifiche per gli EJB definiscono diverse proprietà fra le quali la persistenza, il supporto alle transazioni, la gestione della concorrenza e della sicurezza, e l'integrazione con altre tecnologie.

I *beans* vengono istanziati in un container EJB, tipicamente un application server. Le specifiche descrivono come un EJB interagisce con il server e come il client interagisce con la combinazione container/EJB. I client non istanziano i beans direttamente tramite l'operatore Java *new*, bensì ottengono un collegamento tramite il container EJB. Solitamente questo collega-

mento non è fatto verso l'implementazione del bean, bensì ad un proxy che implementa l'interfaccia locale o remota del servizio richiesto dal client per facilitare la gestione di transazioni, sicurezza e altre proprietà.

Esistono tre tipi di EJB:

- **EJB di entità (*Entity EJB*)**: inglobano gli oggetti che memorizzano i dati. La persistenza può essere gestita dal container, e in questo caso è un altro modulo software che si occupa della persistenza, oppure può essere implementata all'interno del bean.
- **EJB di sessione (*Session EJB*)**: gestiscono l'elaborazione delle informazioni sul server, e generalmente si pongono come interfaccia tra i client e i servizi offerti lato server. Possono essere stateful, e in questo caso l'accesso alla memoria è limitato ad un unico client, o stateless, e in questo caso viene permesso l'accesso concorrente alle funzionalità offerte dal bean.
- **EJB guidati da messaggi (*Message driven EJBs*)**: bean con funzionamento asincrono, non richiedono di essere istanziati da parte dei client.

2.4 Tecnologie utilizzate

La scelta delle componenti tecnologiche è ricaduta sulle seguenti soluzioni open source:

- **Pentaho CE (Community Edition)**, per implementare la logica di business intelligence. In particolare sono stati utilizzati:
 - **Kettle** e **Spoon** come tool di definizione del processo di ETL
 - **Mondrian** come OLAP server per l'analisi multidimensionale
 - **Schema Workbench** per la creazione dei cubi OLAP
- **JBoss Application Server** per il deploy dell'applicazione web
- **database Oracle** per il data warehouse
- **HTML5, JQuery, Bootstrap e Angular JS** per lo sviluppo dell'interfaccia applicativa

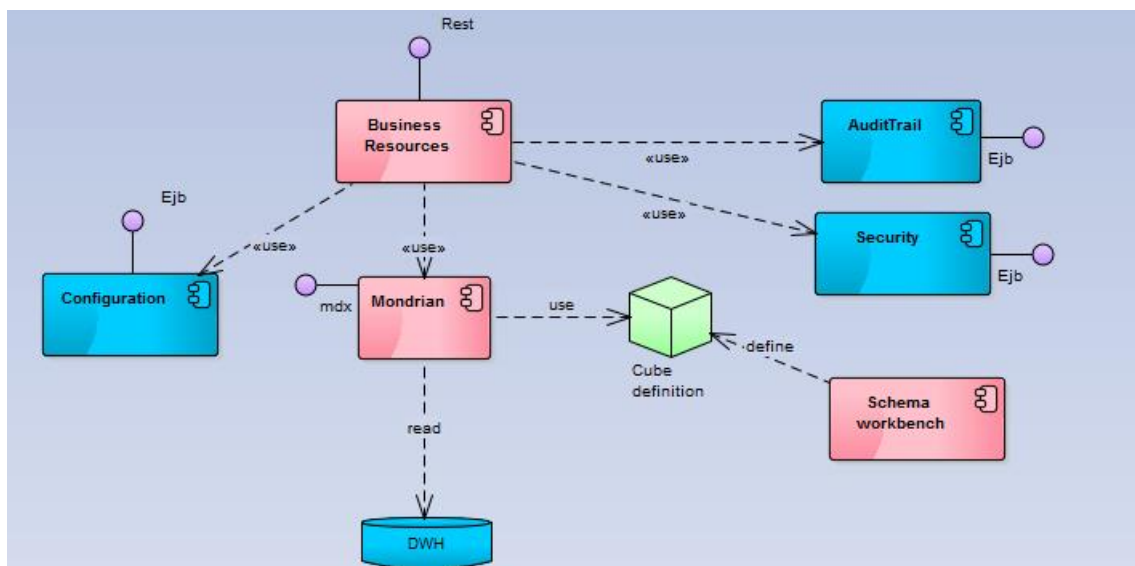


Figura 2.2: Architettura di *Viewpoint* e relative tecnologie utilizzate

ETL

Il processo di ETL (*Extraction Transformation Load*) è necessario per trasferire i dati provenienti da sorgenti esterne ed eterogenee (csv, fogli di calcolo Excel, documentale, etc.) verso il DWH centrale a supporto delle analisi di business intelligence. Tale processo prevede:

- una fase di estrazione dei dati sulla base di criteri ben definiti e necessari al contesto applicativo
- una fase di trasformazione e normalizzazione dei dati al fine di eliminare duplicati, scartare elementi non validi (es. esami che non presentano un risultato numerico, quali ad esempio quelli conclusi con una nota *campione non valido*), omogenizzare informazioni concettualmente equivalenti codificate in modi diversi nei vari sistemi, o trasformare i valori secondo le stesse unità di misura
- una fase di caricamento nelle tabelle del DWH, seguendo un approccio di tipo incrementale: il sistema si aggiorna aggiungendo solo dati o eventi nuovi rispetto alla data dell'ultimo caricamento avvenuto, non alterando mai in alcun modo i dati già presenti sul DWH

Per effettuare tali operazioni è stato utilizzato **Kettle**, il componente di *data integration* distribuito da Pentaho. Questa soluzione mette a disposizione le specifiche API per la gestione ottimizzata del processo di ETL, permettendo agevolmente l'estrazioni dei dati da sorgenti di origine differenti. È stato inoltre utilizzato **Spoon**, un ambiente grafico per il design del workflow di estrazione, trasformazione e caricamento sul DWH, che semplifica l'implementazione degli step necessari sia attraverso una vasta libreria di trasformazioni pre-impostate, sia fornendo la possibilità di definire operazioni *ad-hoc* sui dati.

Data warehouse

I dati estratti ed elaborati dal modulo di *data integration* verranno successivamente caricati sul data warehouse (DWH) di riferimento. Il DWH può essere impostato secondo tre diversi approcci:

- **MOLAP** (Multidimensional On-Line Analytical Processing): i dati vengono memorizzati su strutture dati multidimensionali (detti cubi OLAP)
- **ROLAP** (Relational On-Line Analytical Processing): utilizza un database relazionale per la rappresentazione dei dati multidimensionali
- **HOLAP** (Hybrid On-Line Analytical Processing): approccio ibrido, in cui i dati di dettaglio risiedono sul database relazionale mentre quelli aggregati vengono mantenuti sull'OLAP server.

L'approccio adottato per *Viewpoint* è di tipo HOLAP poichè la struttura del database relazionale su cui si basa il DWH viene gestita da Pentaho in parte in memoria attraverso meccanismi di caching, raggiungendo prestazioni simili a quelle ottenibili attraverso i sistemi MOLAP.

Seguendo tale impostazione, il DWH si basa su un database relazionale opportunamente modellato per rendere efficiente l'accesso al contenuto informativo, secondo uno schema a stella (v. Fig. 2.3) strutturato secondo tabelle *dei fatti*, contenenti una serie di valori misurabili (ad esempio il risultato numerico di un esame di laboratorio) collegati ai singoli eventi (quali ad esempio gli episodi clinici oppure i risultati di laboratorio), e tabelle contenenti le *dimensioni*, che rappresentano invece gli attributi a cui sono correlati i singoli fatti (ad esempio la dimensione temporale colloca il fatto in un determinato giorno, mese e anno, la dimensione del genere distingue il sesso dei pazienti, etc.). I fatti rappresentano il centro della stella, mentre le punte sono le dimensioni. Ogni fatto contiene i puntatori alle tabelle di dimensione e valori relativi al singolo fatto, denominate *misure*. Più tabelle dei fatti condividono le tabelle delle dimensioni.

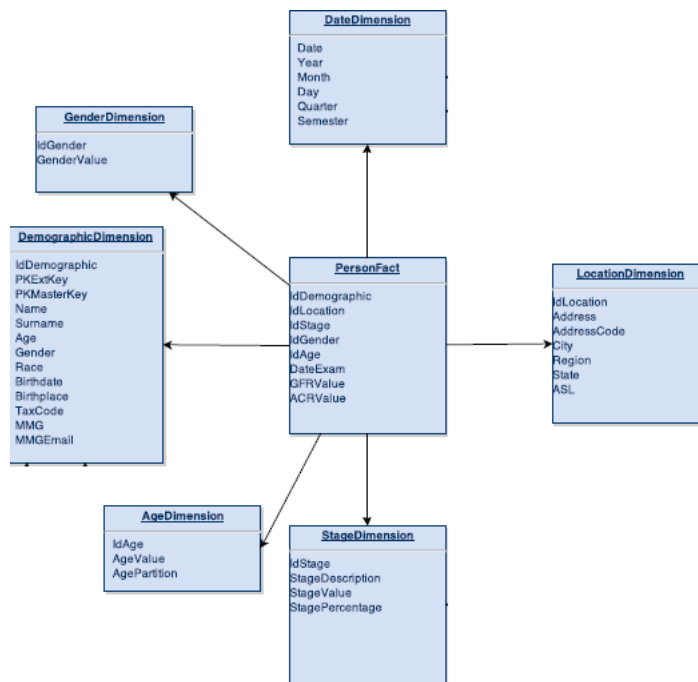


Figura 2.3: Diagramma a stella che rappresenta il data warehouse utilizzato

Il data warehouse di *Viewpoint* per il contesto di nefrologia, gestisce due tabelle dei fatti:

- **BI_ORDER**, che definisce gli ordini inviati al laboratorio, con i dati generici del paziente, età, sesso e cap di residenza e informazioni relative al reparto di provenienza
- **BI_OBSERVATION**, i cui eventi sono identificati dai risultati degli esami di laboratorio (a fronte degli ordini inviati e memorizzati nella **BI_ORDER**). In questa tabella (e nella sua estesa **BI_OBSERVATION_EXTENDED**) sono caricate le informazioni relative al calcolo del eGFR secondo l'algoritmo CKD EPI, lo stadio del paziente, etc.

Le dimensioni sono identificate dalle seguenti tabelle:

- **BI_DIM_OU**: dimensione dei reparti, strutturati su una gerarchia che passa dall'ospedale di appartenenza fino ad arrivare alla città e alla regione

- `BI_DIM.LOC`: dimensione della posizione geografica relativa al paziente (CAP, città, provincia e regione)
- `BI_DIM.TIME`: dimensione temporale, con gerarchia data, mese, semestre, anno.

In aggiunta a queste, vi sono dimensioni degeneri, ovvero dimensioni che categorizzano un evento in base ad una proprietà priva di gerarchia, identificate cioè soltanto da uno dei possibili valori che può assumere; esempio di dimensione degenera può essere il sesso del paziente, che può essere maschio, femmina o non definito.

Per ottimizzare le performance del DWH, le colonne sono state impostate a NOT NULL dove possibile, sono state abilitate le foreign key, e sono stati aggiunti gli indici corrispondenti alle foreign key sulla tabella dei fatti. Inoltre, sono state minimizzate le operazioni di join fra le tabelle, ricorrendo piuttosto alla creazione di viste materializzate, aggiornate al termine di ogni processo di ETL.

OLAP server Mondrian

Al fine di ottimizzare i tempi di analisi su notevoli quantità di dati, è stato introdotto nello schema architetturale **Mondrian**, componente della soluzione **Pentaho CE**, un OLAP server open source scritto interamente in Java che gestisce il database sottostante in modalità HOLAP. Esso permette di effettuare interrogazioni al DWH implementando le fondamentali operazioni di aggregazione (slice, dice, drill down e drill up). Mondrian si interfaccia con il database relazionale attraverso schemi logici che mappano lo strato fisico del database; tali schemi definiscono i cubi OLAP e sono rappresentati attraverso dei semplici file XML.

L'OLAP server Mondrian permette di effettuare analisi attraverso la definizione di una gerarchia di concetti associati alle informazioni, che collegano concetti di livello inferiore a concetti di un livello superiore. In questo modo, ad esempio, è possibile associare un singolo fatto al giorno esatto del suo

accadimento, per poi poterne dedurre la correlazione con il mese in cui si è verificato, oppure col semestre, o con l'anno. Allo stesso modo un risultato di laboratorio, che in *Viewpoint* rappresenta un evento, è legato ad una persona associata al concetto di anziano, maturo o giovane, piuttosto che al concetto sottostante di fascia di età (0-15, 16-25, 26-35, etc.), fino al livello di dettaglio che corrisponde alla sua esatta età anagrafica.

Attraverso anche queste gerarchie sono possibili le seguenti operazioni sui cubi messe a disposizione da Mondrian:

- Drill-up: corrisponde ad una operazione di aggregazione dei dati dei cubi presenti nel DWH. Si può ottenere, ad esempio, risalendo la gerarchia lungo una dimensione, oppure è ottenibile attraverso l'eliminazione di una dimensione dall'interrogazione al cubo, aumentando il livello di aggregazione.
- Drill-down: operazione inversa rispetto al drill-up, mostra i dati ad un minor livello di aggregazione. Anche in questo caso può essere fatto percorrendo - verso il basso - i livelli di gerarchia di una dimensione, oppure aggiungendo dimensioni all'interrogazione che stiamo effettuando.
- Slice: operazione che seleziona una dimensione del cubo, fissandola ad un singolo valore.
- Dice: focalizza l'analisi verso un sottoinsieme di dati, prendendo in considerazione un sottocubo rispetto al cubo originale.

Per effettuare le interrogazioni al server ed effettuare le suddette operazioni, Mondrian accetta chiamate in linguaggio **MDX (MultiDimensional eXpressions)**, standard per le query multidimensionali introdotto da Microsoft, che vengono successivamente tradotte in query SQL. *Viewpoint* crea dinamicamente query MDX per recuperare, ad esempio, i dati aggregati che definiscono la distribuzione degli stadi di nefrologia sulla popolazione analizzata, piuttosto che la distribuzione fra pazienti in regime ambulatoriale o di

ricovero. I risultati delle query vengono poi utilizzati dal frontend sia per popolare le tabelle con i dati aggregati, sia per riportare i risultati graficamente attraverso istogrammi, grafici a torta, etc.

Application server JBoss 6.1

Per lo sviluppo delle logiche applicative, il sistema si avvale del supporto di **JBoss AS 6.1**. JBoss è un server applicativo prodotto da Red Hat open source e scritto in Java. Implementa le specifiche Java EE ed è compatibile con molteplici piattaforme. Supporta l'architettura EJB e l'integrazione con Mondrian, caratteristiche che lo rendono adatto ad essere utilizzato come application server per *Viewpoint*.

Su questo server sono installati:

- i servizi di accesso a Mondrian per le interrogazioni di business intelligence
- i servizi trasversali della Foundation, quali *Security*, *Audit*, *Configuration and Preferences* e *Notification System*
- la web application
- opzionalmente l'OLAP server Mondrian, che può essere eseguito sia su un application server a parte, sia integrato nello stesso server applicativo

2.5 Interfaccia utente

Per massimizzare la fruibilità e la distribuzione dell'applicazione senza bisogno di ricorrere a onerose installazioni su ogni singola macchina, la scelta dello strato di presentazione è ricaduta su una applicazione di tipo web. Per lo sviluppo del frontend, perciò, sono state scelte tecnologie quali **HTML5** e **jQuery**, ampiamente utilizzate in ambito web, poichè permettono di implementare interfacce dinamiche di ottimo livello. Per migliorare l'usabilità e l'aspetto grafico dell'applicazione, è stato utilizzato **Bootstrap**, framework CSS che gestisce automaticamente il design responsive adatto a dispositivi con risoluzione arbitraria, si basa unicamente su Javascript e CSS, pertanto non richiede l'installazione di componenti aggiuntive, permette di rendere l'interfaccia accattivante ma comunque personalizzabile ed è molto semplice da utilizzare.

In aggiunta alla definizione di questi strumenti per lo sviluppo e lo UI, è stato introdotto l'utilizzo di **Angular JS**, un framework Javascript open source prodotto e distribuito da Google, necessario per l'organizzazione del codice e l'ottimizzazione dello sviluppo. Angular fornisce il supporto al data binding, ovvero collega in maniera bidirezionale elementi del DOM con valori definiti nel modello Javascript, agevolando così lo sviluppo dichiarativo del codice. Inoltre facilita l'utilizzo del pattern MVC, distinguendo il modello (il contenitore con le proprietà da visualizzare), dalla view (ovvero lo strato di presentazione verso l'utente) e dal controller, il componente permette di collegare modello e view. Angular supporta la dependency injection delle proprietà, ovvero non è necessario scrivere codice per recuperare le proprietà degli oggetti, ma queste vengono direttamente inserite dal controller attraverso una specifica variabile di comunicazione denominata *scope*. Questo aspetto è particolarmente importante anche per lo sviluppo dei test di unità sul codice sviluppato. Infine, il framework fornisce una serie di direttive che permettono di sviluppare l'applicazione in modo efficiente e in tempi rapidi.

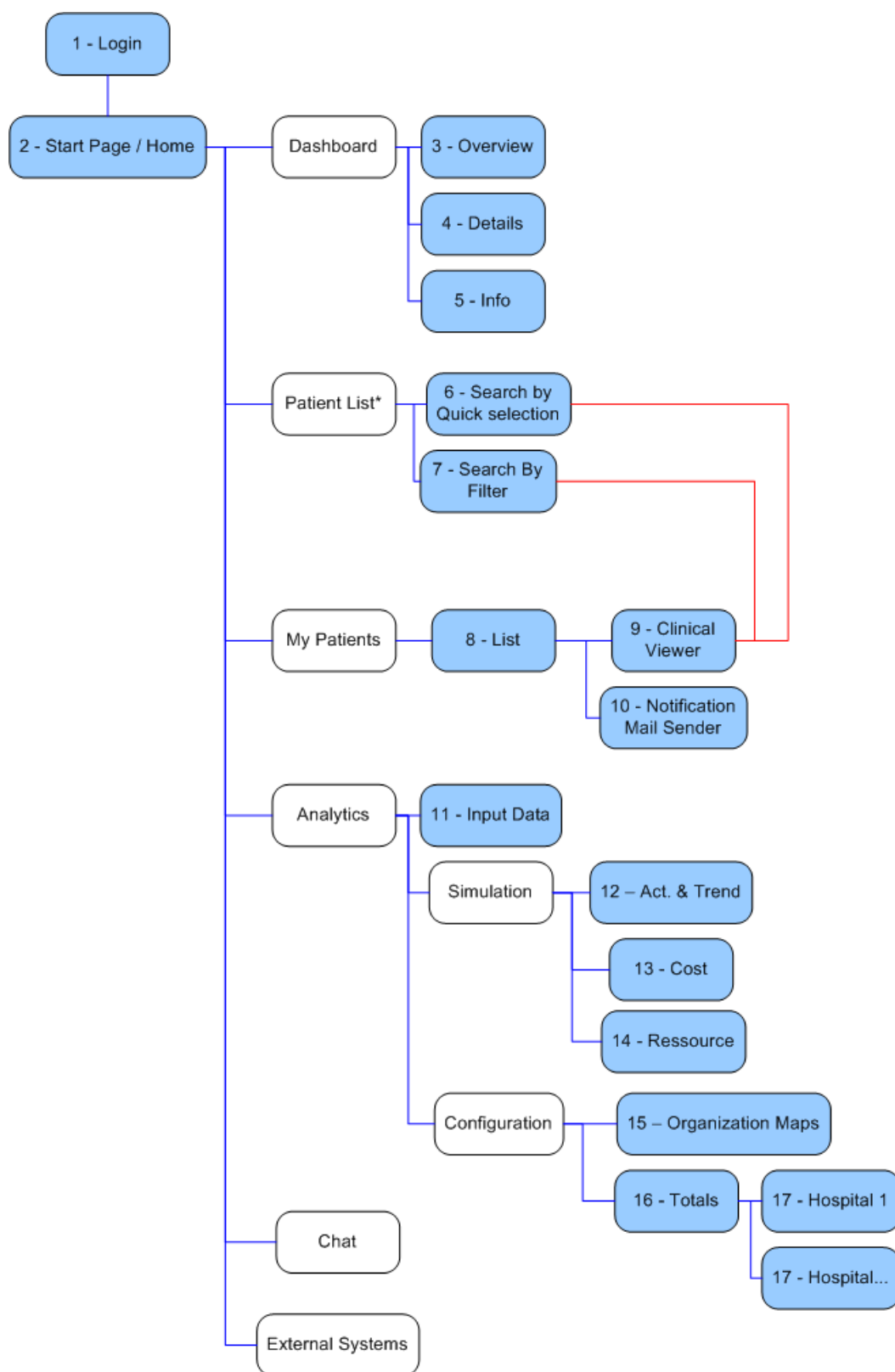


Figura 2.4: Organizzazione delle schermate utente dell'applicazione

Capitolo 3

La sezione *Analytics*

La sezione di *Analytics* in *Viewpoint* implementa le funzionalità che consentono di visualizzare tramite grafici e tabelle dati aggregati sui pazienti del reparto di nefrologia e di effettuare simulazioni sui pazienti futuri. Si compone principalmente di due parti:

- la sezione *Input data*, in cui vengono visualizzati informazioni sui pazienti degli anni passati
- la sezione *Simulation*, dalla quale è possibile effettuare simulazioni sui pazienti negli anni a venire

Dalla sezione *Configuration*, invece, è possibile selezionare una struttura clinica ben precisa (un ospedale, una ASL, un'intera regione, etc.) e impostare una serie di parametri numerici per quella determinata struttura.

Questa sezione basa tutte le statistiche e le simulazioni effettuate partendo dal dato relativo al trend di crescita dei pazienti classificati attraverso la formula CKD EPI, con particolare attenzione ai pazienti negli stadi 3/4 o 5. Le simulazioni riguardano l'andamento di crescita del numero dei pazienti, e i costi e le risorse sanitarie che sarà necessario investire per il reparto di nefrologia a fronte di un aumento piuttosto che una diminuzione del numero di individui malati. Questa sezione è pensata per fornire ai medici uno stru-

mento che li guidi in valutazioni di tipo clinico sui pazienti, e che mostri dati relativi all'utilizzo di risorse ospedaliere.

Lato backend, la sezione *analytics* si compone di:

- una classe *AnalyticsResource* che implementa i servizi REST, e quindi riceve le richieste e reinoltra le risposte al client
- un'interfaccia *AnalyticsServiceLocal* e la sua relativa implementazione, che si occupa di recuperare dati dal database applicativo e svolgere i calcoli sui dati necessari
- vari *objects* a supporto della comunicazione fra frontend e backend, in grado di incapsulare richieste e/o risposte in strutture serializzabili

Lato frontend, invece, sono state utilizzate:

- un *service* con il ruolo di modello all'interno del pattern MVC: è quel modulo che si occupa di inoltrare le richieste REST al backend e ricevere le risposte
- due *views* (*inputData* e *simulation*), che implementano la vista in formato HTML
- due *controllers* (*inputDataCtrl* e *simulationCtrl*), con il ruolo di collegare le viste al modello
- varie direttive utilizzate perlopiù per il popolamento di elementi grafici quali tabelle, istogrammi o menu, che consentono uno sviluppo più modulare e il riuso del codice comune alle varie sezioni.

3.1 La configurazione dei parametri

La sezione *Analytics* deve presentare all'utente la possibilità di impostare una serie di parametri tipici della propria struttura e necessari per andare poi ad effettuare simulazioni e drill down per singola unità indagata. Inizialmente viene mostrata una *organization map*, ovvero una rappresentazione grafica delle unità sanitarie che è possibile selezionare, organizzate secondo una struttura gerarchica ad albero. Il sistema concepito è multi-livello, in cui è possibile indagare dalla singola struttura di nefrologia fino ai livelli organizzativi più alti, andando a gestire l'aggregazione di dati provenienti da differenti laboratori e che servono diverse strutture di nefrologia. Il mapping è mostrato in Tab. 3.1.

Livello	Entità coinvolta	Visualizzazione
1	Regione	Rappresenta il sistema sanitario regionale e ne comprende tutte le ASL. Con un doppio click sul riquadro corrispondente si accede ai dati aggregati delle singole entità, valorizzate ai livelli inferiori
2	ASL	Le ASL costituiscono il secondo livello. Per ogni ASL, la mappa descrive le sotto-strutture disponibili (ospedali, centri dialisi associati, etc.)
3	Hospital	A questo livello sono presentati i dati di dettaglio che riguardano risorse, costi e personale per ogni struttura ospedaliera in merito al relativo reparto di nefrologia e dialisi.

Tabella 3.1: Mapping livelli - unità organizzative afferenti al SSN

Questa schermata rappresenta l'intera organizzazione su una serie di livelli, ognuno dei quali possiede informazioni puntuali in caso di selezione di una singola foglia, o aggregate (somme o valori medi) se si sale ai livelli più alti. Questa configurazione permette la gestione sia di casi semplici, in cui si

considera la singola struttura con un solo database da analizzare, che di casi complessi, in cui è necessaria la mappatura ed aggregazione dei dati a livello regionale da differenti database su unico data warehouse. Nel database e a livello applicativo, la differenza fra unità semplici e complesse si realizza denominando *organization unit* le strutture semplici, *configuration unit* le strutture i cui dati sono il risultato di un'aggregazione.

Tramite la sezione di configurazione di *Viewpoint* l'utente seleziona un nodo sulla mappa organizzativa, e le simulazioni effettuate nelle sezioni successive saranno legate all'unità scelta. In questo modo si permette di avere visibilità e capacità di simulazione dalla singola foglia fino ad arrivare ai nodi principali.

In Tab. 3.2 viene mostrato l'insieme delle regole disponibili, suddivise in singole sezioni di interesse. Anche in questa schermata viene gestita la gerarchia multi livello, e in ogni momento è possibile passare ai parametri di configurazione di altre unità organizzative. La tab *Totals* riporta i totali cumulativi per le strutture mappate.

Label	Semantica	Aggregazione
Sezione <i>Staff ospedaliero</i>		
HD Doctor	Numero di medici in servizio dedicati alla gestione dei pazienti in emodialisi ospedaliera	Somma di valori di ogni singola foglia
PD Doctor	Numero di medici in servizio dedicati alla gestione dei pazienti in dialisi peritoneale	Somma
TR Doctor	Numero di medici in servizio dedicati alla gestione dei pazienti trapiantati	Somma
HD Nurse	Numero di infermieri in servizio dedicati alla gestione dei pazienti in emodialisi ospedaliera	Somma
PD Nurse	Numero di infermieri in servizio dedicati alla gestione dei pazienti in dialisi peritoneale	Somma
TR Nurse	Numero di infermieri in servizio dedicati alla gestione dei pazienti trapiantati	Somma

Sezione <i>Organizzazione ospedaliera</i>		
HD Bed	Numero di letti/postazioni dedicate all'emo-dialisi ospedaliera	Somma di valori di ogni singola foglia
PD Capabilities	Numero massimo di pazienti in dialisi peritoneale gestibili con le risorse a disposizione	Somma
TR Capabilities	Numero massimo di pazienti trapiantati gestibili con le risorse a disposizione	Somma
HD Home Capabilities	Numero massimo di pazienti in emodia-lisi domiciliare gestibili con le risorse a disposizione	Somma
HD Session/day	Numero di turni giornalieri previsti per le sessioni di dialisi. Normalmente 2 (mattina e pomeriggio), in casi particolari si aggiunge il turno serale. Certi centri effettuano soltanto il turno al mattino.	Visibile solo se tutte le foglie gestiscono lo stesso numero di turni, altrimenti il campo viene rimosso
Sezione <i>Costi medi per trattamento</i>		
Bic HD	Costo medio del trattamento di bicarbonato- emodialisi	Media fra i valori di ogni singola foglia
AFB HD	Costo medio del trattamento di <i>acetate-free biofiltration</i>	Media
HDF HD	Costo medio del trattamento di emodiafiltra-zione	Media
CAPD	Costo medio del trattamento di dialisi peritoneale continua	Media
APD	Costo medio del trattamento di dialisi peritoneale automatizzata	Media
Home HD	Costo medio del trattamento di emodialisi domiciliare	Media

HD Session/day	Numero di turni giornalieri previsti per le sessioni di dialisi. Normalmente 2 (mattina e pomeriggio), in casi particolari si aggiunge il turno serale. Certi centri effettuano soltanto il turno al mattino.	Visibile solo se tutte le foglie gestiscono lo stesso numero di turni, altrimenti il campo viene rimosso
Sezione <i>Regole</i>		
1 HD doctor each	Numero di pazienti in emodialisi ospedaliera gestiti con 1 medico	Valore visualizzato solo nelle unità foglia
1 PD doctor each	Numero di pazienti in dialisi peritoneale gestiti con 1 medico	Solo se foglia
1 TR doctor each	Numero di pazienti trapiantati gestiti con 1 medico	Solo se foglia
1 HD nurse each	Numero di pazienti in emodialisi ospedaliera gestiti con 1 infermiere	Solo se foglia
1 PD nurse each	Numero di pazienti in dialisi peritoneale gestiti con 1 infermiere	Solo se foglia
1 TR nurse each	Numero di pazienti trapiantati gestiti con 1 infermiere	Solo se foglia

Tabella 3.2: Sintassi e semantica delle regole di configurazione che è possibile impostare dalla sezione *Configuration*. La colonna *Aggregazione* indica con quale operazione vengono aggregati i dati delle unità di livello superiore rispetto ai dati impostati nelle foglie dell'albero.

Le regole possono essere impostate e salvate solo da un utente con ruolo di amministratore: ogni centro dialisi avrà il proprio *superUser* che sarà in grado di effettuare modifiche legate al proprio centro dialisi. I parametri impostati saranno invece visualizzabili in sola lettura da tutti gli utenti, e i report e le simulazioni generate saranno sempre collegate all'ultima versione delle regole impostate. Nel caso in cui una delle regole non sia compilata, risulterà impossibile procedere alle simulazioni economiche e di risorse legate a quello specifico parametro.

3.2 La sezione *Input data*

La sezione *Input data* calcola il trend sui dati reali secondo la logica multi-livello, seguendo quindi le seguenti regole:

- Tab *Totali*, livello regionale: rappresenta la rete regionale delle cure o comunque il livello più alto. Il risultato aggrega dati dalle varie ASL regionali.
- Tab *ASL*: rappresenta una struttura formata da differenti ospedali dislocati su un territorio.
- Scelta del singolo ospedale: rappresenta una nodo semplice della struttura.



Figura 3.1: Screenshot della sezione *Input data*

Nella parte in alto a destra della schermata è posizionato il calendario, grazie al quale è possibile impostare il periodo di calcolo del trend. L'utente può scegliere date puntuali di inizio e fine, oppure un lasso di tempo

preselezionato (ultimi due anni, ultimo anno, etc.). Il calendario si aggiorna autonomamente sulla data di inizio andando a considerare come anno 0 il primo anno in cui i dati da elaborare sono disponibili. Il sistema basa i propri calcoli relativi al trend di crescita partendo da un database che rende disponibile almeno tre anni di dati. Un valore di anni disponibili inferiore a 3 non garantisce una sufficiente efficacia nel calcolo dell'andamento.

In primis viene mostrato il grafico con il trend dei pazienti in stadio 5, che quindi rappresentano i dializzati o comunque gli individui sottoposti a trattamento in nefrologia. Il trend di crescita tiene conto dei valori e dei dati rilevati dal laboratorio di analisi negli ultimi anni, con un minimo di tre anni di dati retrospettivi. Il valore percentuale esprime il tasso di crescita: rappresenta la crescita reale media dei pazienti anno dopo anno nell'unità precedentemente selezionata. Il tasso di crescita è calcolato come:

$$g_t = \frac{\text{pazienti}_t}{\text{pazienti}_{t-1}} - 1$$

dove t rappresenta l'anno corrente esaminato. Il dato visualizzato è la media delle crescite nell'intervallo di anni impostati nel calendario:

$$\bar{g} = \frac{1}{n} \sum g_t$$

dove n rappresenta in numero di anni presi in considerazione.

Nella seconda parte della schermata, viene invece mostrato il grafico con il trend dei pazienti in stadio 3 e 4, ovvero la somma dei pazienti in stadio 3A,3B e 4. La tabella adiacente riporta i valori mostrati nel grafico, e il valore percentuale rappresenta la crescita media percentuale per questa tipologia di pazienti, calcolata come sopra.

Lato frontend, la tabella e il grafico vengono popolati rispettivamente dalle direttive *nl-table* e *nl-chart-line*. Questi componenti Javascript si occupano di inviare una richiesta GET al backend, passando gli opportuni parametri (intervallo di tempo e codice dell'unità organizzativa di cui vogliamo conoscere le informazioni).

```
1 <div nl-chart-line class="chart" id="
    year_vs_people_past_line_5" url-service="/rest/analytics/
```

```

    past_trend" url-parameters="/5/5/{{yearRange}}/{{ou.code
    }}/{{ouLevel2.code}}/{{ouLevel3.code}}"></div>
2 ...
3 <div nl-table table-id="table_past_trend_5" url-service="/
    rest/analytics/past_trend" url-parameters="/5/5/{{
    yearRange}}/{{ou.code}}/{{ouLevel2.code}}/{{ouLevel3.code
    }}"></div>

```

Il tasso di crescita medio, invece, viene recuperato dal controller JavaScript *inputDataCtrl*, che invia una richiesta GET al backend e, una volta ricevuta la risposta, imposta una variabile opportunamente.

```

1     var req = $http.get(urlBackendBase + "/rest/analytics/
    past_avg_gr/5/5/" + $scope.yearRange);
2     req.success(function(result) {
3         $scope.avg5 = result.cellset[1][1].value;
4     });

```

Lato backend, le richieste vengono raccolte da *AnalyticsResource*, che imposta opportunamente i parametri ed esegue query MDX utilizzando la funzione *executeNamedOlapQuery*.

```

1     @GET
2     @Path("/past_trend/{stage1}/{stage2}/{yearRange}")
3     @Produces("application/json")
4     @RolesAllowed(value={SecurityRoles.
        NEPHROLOGY_ANALYTICS_READ})
5     public QueryCellSet executeQueryPastTrend(
6         @PathParam("stage1") String stage1,
7         @PathParam("stage2") String stage2,
8         @PathParam("yearRange") String yearRange,
9         @MatrixParam("id") String id) throws ViewpointException
10    {
11        StringBuffer sb = new StringBuffer();
12        String[] years = yearRange.split("-");
13        sb.append(stage1).append(",").append(stage2).append(",20"
        ).append(years[0]).append(",20").append(years[1]);
14        return super.executeNamedOlapQuery("past_trend", sb.
        toString(), id);

```



```
14     }
15
16     @GET
17     @Path("/past_avg_gr/{stage1}/{stage2}/{yearRange}")
18     @Produces("application/json")
19     @RolesAllowed(value={SecurityRoles.
20         NEPHROLOGY_ANALYTICS_READ})
21     public QueryCellSet executeQueryPastAvgGrowth(
22         @PathParam("stage1") String stage1,
23         @PathParam("stage2") String stage2,
24         @PathParam("yearRange") String yearRange,
25         @MatrixParam("id") String id) throws ViewpointException
26     {
27         StringBuffer sb = new StringBuffer();
28         String[] years = yearRange.split("-");
29         Integer nextYr= Integer.parseInt(years[0]) + 1;
30         sb.append(stage1).append(",").append(stage2).append(",20"
31             ).append(years[0]).append(",20").append(nextYr).append("
32             ,20").append(years[1]);
33         return super.executeNamedOlapQuery("past_avg_gr", sb.
34             toString(), id);
35     }
```

3.3 La sezione *Simulation*

La sezione del sistema dedicata alla simulazione si compone di tre viste:

- *Actuals and trend*, che permette di visualizzare il trend futuro e affinare alcuni dati per le simulazioni
- *Costs*, con l'obiettivo di mostrare all'utente l'impatto economico dovuto al coinvolgimento di un numero di pazienti arbitrario definito dall'utente
- *Resource*, strettamente correlata alla precedente, prende in considerazione le necessità di personale e risorse ospedaliere da prevedere a seconda del numero di pazienti per specifico trattamento.

3.3.1 *Actuals and trend*

La vista *Actuals and trend* si compone di tre parti.

La sezione superiore della schermata (Fig. 3.2) rappresenta una stima dei costi sostenuti per l'anno corrente, calcolato come:

$$\text{Costo totale} = \sum_{i \in \text{Trattamenti}} \# \text{pazienti}_i \cdot \text{costo}_i$$

e la spartizione dei pazienti rilevati nell'ultimo anno per tipologia di trattamento. I valori numerici, che sono pre-valorizzati in base ai dati reperiti dal DWH utilizzando gli algoritmi visti in precedenza, sono modificabili manualmente. Dal momento che alcuni risultati potrebbero essere imprecisi, i box possono essere raffinati da parte dell'utente sulla base delle proprie conoscenze per proporre simulazioni il più possibile verosimili. I valori e i grafici calcolati nelle altre sezioni si aggiornano ogni volta che l'utente appone modifiche ai dati.

La seconda sezione (Fig. 3.3) riporta il trend di crescita dei pazienti in stadio 5 previsto per i prossimi 3 anni futuri. Il sistema propone il tasso medio di crescita rilevato applicando un semplice algoritmo di regressione

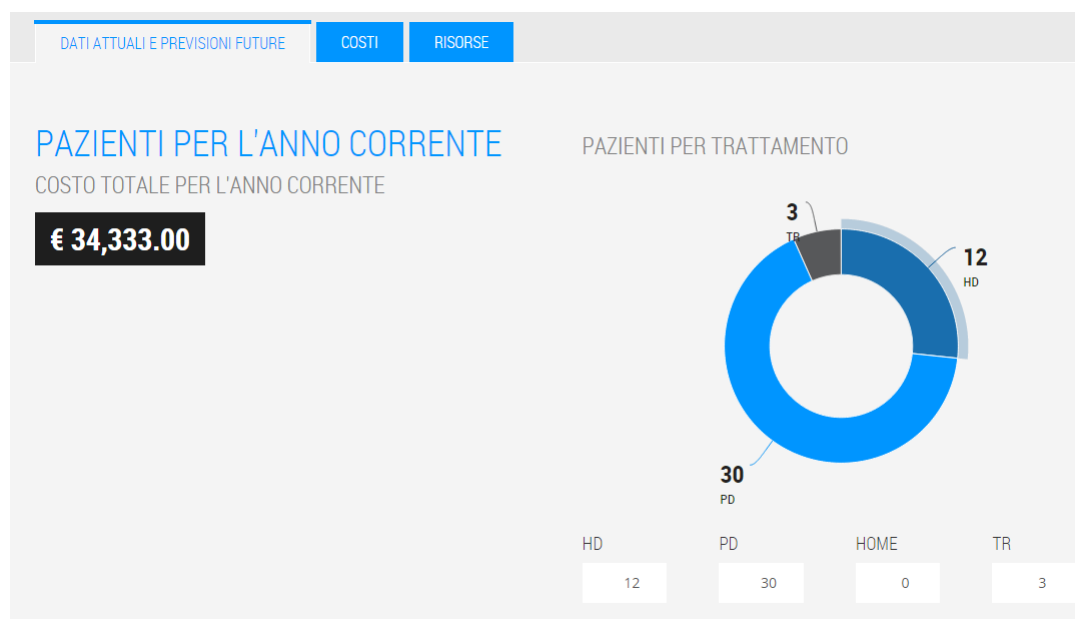
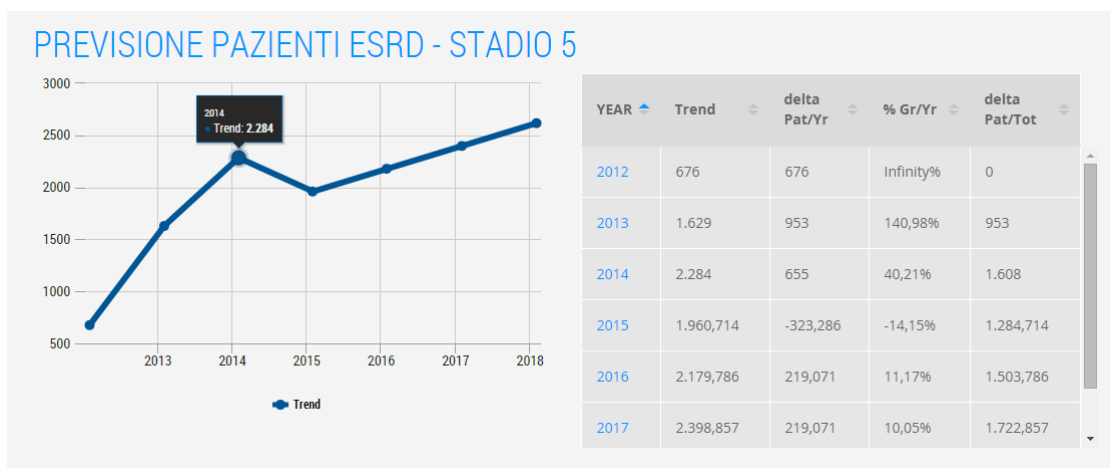
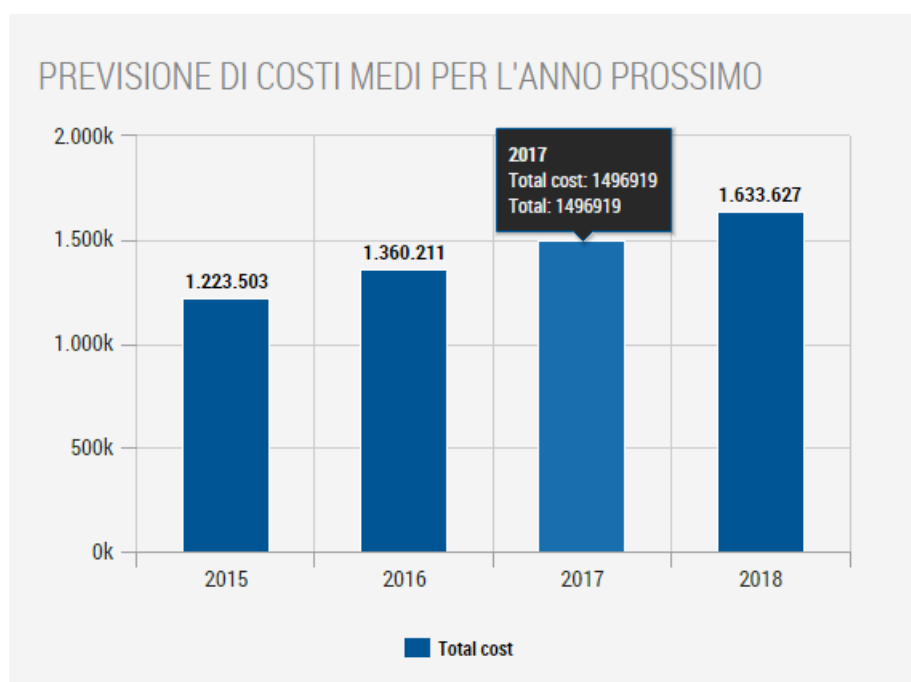


Figura 3.2: Screenshot della sezione *Simulation*

lineare sulla crescita registrata negli ultimi anni. La simulazione effettuata può riferirsi ad una singola foglia nella mappa organizzativa ad albero vista in precedenza, ad una ASL o ad un'intera regione. Nella tabella vengono indicati in forma numerica:

- il numero dei pazienti previsti per ogni anno
- il delta dei pazienti per ogni anno preso in considerazione, espresso come pazienti in aggiunta rispetto all'anno precedente
- il delta dei pazienti per ogni anno preso in considerazione, espresso come calcolo percentuale in aggiunta rispetto all'anno precedente
- il delta totale dei pazienti accumulati rispetto al primo anno preso in considerazione

Infine, nella terza parte della schermata (Fig. 3.4) viene proposta un'analisi di costo medio previsto per gli anni successivi, tenendo in considerazione

Figura 3.3: Screenshot della sezione *Simulation*Figura 3.4: Screenshot della sezione *Simulation*

il numero di pazienti per trattamento proiettati con una proporzione sul numero di pazienti calcolato dall'algoritmo di regressione lineare.

Lato frontend, le tabelle e i grafici vengono popolati grazie a direttive apposite, che inviano richieste ad *AnalyticsResource* presente nel backend, come visto per la sezione *Input data*. Lato backend, la classe *AnalyticsResource* a sua volta si avvale dei servizi forniti da *AnalyticsService*, con la funzione di calcolare i dati necessari. Ad esempio, la funzione che calcola il costo totale:

```

1  @Override
2  public Long getTotalCost(CellSet patPerTreatment, String
    ouCode) throws ViewpointException {
3
4      Long tot = (long) 0;
5
6      Map<String,Integer> mapCosts = new HashMap<String,Integer>
    >();
7
8      mapCosts = getCostsPerTreatment(ouCode);
9
10     List<CellSetAxis> cellSetAxes = patPerTreatment.getAxes()
    ;
11     CellSetAxis rowsAxis = cellSetAxes.get(Axis.ROWS.
    axisOrdinal());
12     int cellOrdinal = 0;
13     int np;
14     Cell cell;
15
16     for (Position rowPosition : rowsAxis.getPositions()) {
17
18         for (Member member : rowPosition.getMembers()) {
19
20             if(!member.getName().equals("#null")){
21                 cell = patPerTreatment.getCell(cellOrdinal);
22                 if (cell.getFormattedValue() != null){
23                     np = Integer.parseInt((cell.getFormattedValue()).
    replaceAll("[^\\d]", ""));
24                     Logger.getLogger().info("# pazienti "+ np);

```

```
25         if (mapCosts.get(member.getName()) != null)
26             tot = (long) (tot + np * mapCosts.get(member.
getName()));
27     }
28 }
29 ++cellOrdinal;
30 }
31 }
32
33 return tot;
34 }
```

si avvale sia dei pazienti suddivisi per trattamento in *patPerTreatment* recuperati via query MDX, sia dei parametri di costo recuperati dal database attraverso la funzione *getCostsPerTreatment*.

3.3.2 *Costs*

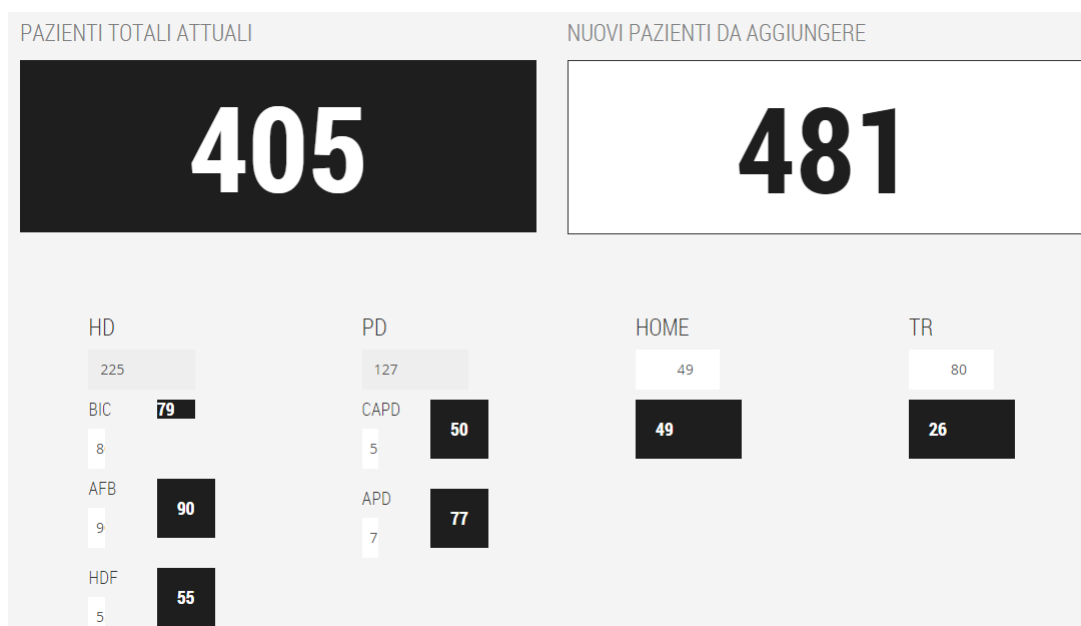
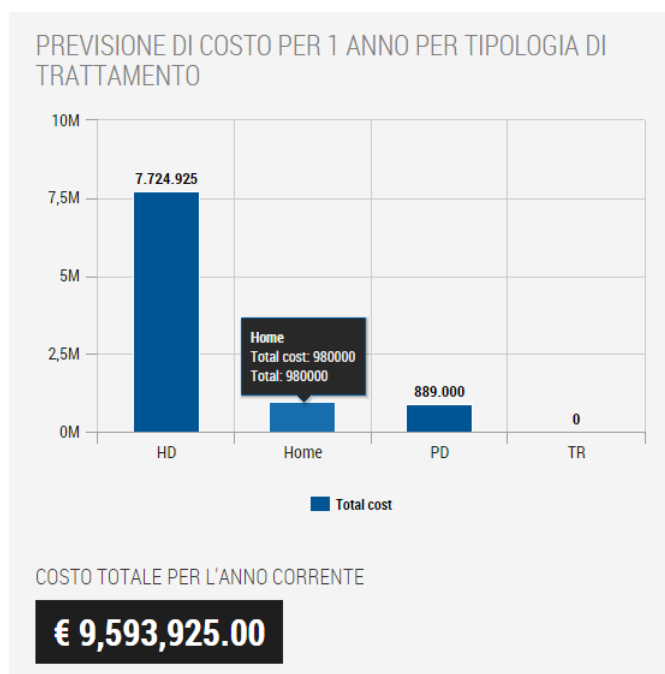
La sezione dedicata alla simulazione dei costi ha l'obiettivo di mostrare l'impatto economico dovuto all'inserimento di un numero di pazienti arbitrario definito dall'utente.

L'utente ha il compito di inserire manualmente i pazienti suddivisi per singoli trattamenti. Per ogni trattamento, vengono visualizzati il numero di nuovi pazienti inseriti (box bianchi), e la differenza fra i pazienti nuovi e quelli attuali, recuperati da query sul DB (box neri).

I contatori mostrati nella parte in alto rappresentano:

- **Pazienti totali attuali**, il numero di pazienti attualmente in trattamento nello stadio 5 relativi all'anno corrente
- **Nuovi pazienti da aggiungere**, la somma di tutti i nuovi pazienti inseriti dall'utente per effettuare la simulazione

In questo grafico (Fig. 3.6) viene proposta un'analisi di costo previsto per l'anno successivo che tenga conto dei pazienti suddivisi per trattamento impostati dall'utente. I dati, presentati attraverso un istogramma per avere

Figura 3.5: Screenshot della sezione *Costs*Figura 3.6: Screenshot della sezione *Costs*

immediatamente visualizzato il budget necessario, vengono calcolati dalla funzione *getCostTrendPerTreatment* presente all'interno di *AnalyticsService*:

```
1  @Override
2  public Map<String,Long> getCostTrendPerTreatment(
    PatientPerTreat patPerTr, String ouCode) throws
    ViewpointException {
3
4      //retrieve costs from DB
5      Map<String,Integer> mapCosts = new HashMap<String,Integer>
6      >();
7
8      mapCosts = getCostsPerTreatment(ouCode);
9
10     for(String key : mapCosts.keySet()){
11         Logger.getLogger().info("cost MAP "+key+" - "+
12         mapCosts.get(key));
13     }
14
15     //fill map patients with patients per treatment
16     Map<String,Integer> patients = new HashMap<String,Integer>
17     >();
18
19     patients.put("HD", patPerTr.getBic()+patPerTr.getAfb()+
20     patPerTr.getHdf());
21     patients.put("PD", patPerTr.getCapd() + patPerTr.getApd()
22     );
23     patients.put("Home", patPerTr.getHome());
24     patients.put("TR", patPerTr.getTr());
25
26     Map<String,Long> costTrendMap = new HashMap<String,Long>
27     >();
28
29     for(String key : patients.keySet()){
30         Long value = (long)mapCosts.get(key) * patients.get(
31         key);
32         costTrendMap.put(key, value);
33     }
34 }
```



```

27     return costTrendMap;
28 }

```

3.3.3 *Resource*

HOSPITAL STAFF				HOSPITAL ORGANIZATION			
	ACTUALS	ESTIMATE	TOTALS		ACTUALS	ESTIMATE	TOTALS
HD DOCTOR	8	0	-8	PD CAPABILITIES	46	127	81
PD DOCTOR	4	0	-4	HD HOME CAPABILITIES	24	49	25
TR DOCTOR	4	0	-4	TR CAPABILITIES	12	134	122
HD NURSE	20	0	-20	HD BED	30	113	83
PD NURSE	8	0	-8				

Figura 3.7: Screenshot della sezione *Resource*

Questa schermata (Fig. 3.7) è strettamente correlata alla precedente, dal momento che ne riporta la parte superiore, riguardante l'impostazione dei pazienti suddivisi per trattamento. Nella parte inferiore, vengono invece mostrate le necessità di staff e risorse ospedaliere da prevedere a seconda dell'assegnazione dei pazienti agli specifici trattamenti dialitici.

Per quanto riguarda lo staff ospedaliero, le risorse sono calcolate come segue:

- la colonna **Actuals** riporta il valore recuperato dalla sezione di configurazione per l'unità selezionata nella mappa organizzativa che stiamo prendendo in considerazione

- la colonna ***Estimate*** calcola le risorse stimate come

$$\text{Estimate} = \frac{\text{pazienti attuali} + \text{nuovi pazienti}}{\text{regola doctor/nurse each}}$$

- la colonna ***Totals*** costituisce la differenza fra i valori stimati e quelli attuali

$$\text{Totals} = \text{Estimate} - \text{Actuals}$$

Per quanto riguarda le risorse ospedaliere generiche, invece, il calcolo della colonna *Estimate* è una semplice somma fra il valore recuperato dalla sezione *Configuration*, che rappresenta il numero massimo di risorse attualmente disponibili in quell'unità organizzativa per tipologia di trattamento (espresso in numero di pazienti gestibili), e il numero dei nuovi pazienti da gestire. Fa eccezione il numero dei letti destinati all'emodialisi ospedaliera (HD BED), che dipende da un ulteriore parametro, ovvero il numero di sessioni giornaliere previste, ed è calcolato come:

$$\text{HD BED} = \frac{\text{pazienti attuali HD} + \text{nuovi pazienti HD}}{\text{HD Session/day} \cdot 2}$$

dove 2 rappresenta il numero di pazienti a settimana per letto, considerato che ogni paziente in emodialisi deve fare 3 dialisi ogni settimana (in 6 giorni). Ad esempio, con 2 sessioni al giorno e 3 letti, riesco a fornire dialisi settimanali a 12 persone.

Lato backend, le risorse vengono calcolate in *AnalyticsService* e restituite al frontend grazie a due oggetti serializzabili di supporto:

- *SimulationResources*, che contiene una lista di *ResourceValues* e una stringa che identifica il nome della sezione

```

1 public class SimulationResources implements Serializable
2     {
3         List<ResourceValues> values;
4
5         String section;
6     }

```

```

7    ...
8 }

```

- *ResourceValues*, che rappresenta una riga nella tabella visualizzata nel frontend:

```

1 public class ResourceValues implements Serializable {
2
3     String name;
4
5     Long actuals;
6
7     Long estimated;
8
9     Long totals;
10
11     ...
12 }

```

Lato frontend, ogni box risorsa viene visualizzato iterando sulla variabile *ResourceBoxes*, che corrisponde a *SimulationResources*:

```

1 <div class="col-lg-6" ng-repeat="box in resourceBoxes">
2   <h3>{{box.section}}</h3>
3   <div class="row form-group">
4     <div class="col-sm-4 col-md-3 col-sm-offset-2 col-md-
      offset-3">
5       <h5 class="text-center">ACTUALS</h5>
6     </div>
7     <div class="col-sm-4 col-md-3">
8       <h5 class="text-center">ESTIMATE</h5>
9     </div>
10    <div class="col-sm-2 col-md-3">
11      <h5 class="text-center">TOTALS</h5>
12    </div>
13  </div>
14
15 <div class="row form-group" ng-repeat="val in box.values">

```

```
16     <div class="col-sm-2 col-md-3">
17         <h4 class="text-right">{{val.name}}</h4>
18     </div>
19     <div class="col-sm-4 col-md-3">
20         <div class="mobile-spinner">
21             <input type="number" class="form-control" id="hd2"
22 value="{{val.actuals}}">
23         </div>
24     </div>
25     <div class="col-sm-4 col-md-3">
26         <div class="mobile-spinner">
27             <input type="number" class="form-control" id="hd2"
28 value="{{val.estimated}}">
29         </div>
30     </div>
31     <div class="col-sm-2 col-md-3">
32         <div id="AFB-box" class="h4 box-block">{{val.totals}}</div>
33     </div>
```


Conclusioni

Il sistema *Viewpoint* persegue lo scopo di ampliare gli orizzonti medici attraverso l'ausilio di strumenti informatici, i quali sono in grado di indagare il contenuto informativo dei grandi archivi dei laboratori di analisi disponibili ed estrapolare informazioni utili a fini clinici. La finalità è quella di creare un sistema di riconoscimento precoce di potenziali patologie in stadio non avanzato, in modo da poter intervenire sui pazienti, sia attraverso l'utilizzo di farmaci specifici, sia semplicemente indirizzandoli verso uno stile di vita mirato alla prevenzione delle nefropatie.

Durante lo sviluppo dell'applicativo, abbiamo affrontato difficoltà di varia natura. Le più significative hanno riguardato la consistenza dei dati, che molto spesso risultano incompleti o errati. In particolare è stato difficile definire in collaborazione coi medici del policlinico di Bari degli algoritmi che, a partire dai dati a disposizione, classificassero i pazienti di nefrologia per trattamento. Nelle basi di dati a disposizione, infatti, non era presente il dato, ma è stato necessario desumerlo dalla tipologia di esami del sangue effettuati dal paziente e dalla frequenza degli stessi esami. I risultati prodotti da questi algoritmi non sono sempre esatti, ma perlopiù probabilistici. Un'altra difficoltà è stata causata dall'assenza dell'informazione sull'eventuale decesso dei pazienti, che rende le stime numeriche imprecise.

Le simulazioni prodotte mostrano un trend di pazienti in stadi nefropatici avanzati in costante aumento. È quindi importante continuare a studiare le azioni possibili per prevenire o ritardare il più possibile le cure dialitiche, che risultano invasive per il paziente e dispendiose in termini di risorse pubbliche

dispensate dal SSN. In secondo luogo, *Viewpoint* si pone come obiettivo il miglioramento del percorso di cura del paziente attraverso un percorso di notifica più mirato e preciso, che coinvolga non solo il medico di base ma anche il medico specialista in nefrologia. La sperimentazione del sistema avverrà presso il policlinico di Bari: sono stati avviati gli iter burocratici necessari, con particolare attenzione verso gli aspetti legali legati alla privacy. È infatti necessario modificare le informative rivolte ai pazienti per permettere l'adozione di un nuovo modello operativo che preveda la possibilità di notificare direttamente il paziente sulla base dei suoi dati storici, informandolo ed invitandolo ad un controllo mirato per una migliore e tempestiva azione di cura.

In futuro verranno sviluppate ulteriori funzionalità in *Viewpoint*, ad esempio verrà istituito un servizio di chat che sarà utilizzabile tra il medico di base e il medico specialista, con l'obiettivo di instaurare una comunicazione fra le due figure professionali che valuteranno insieme il percorso di cura più adatto. Si tratta di uno strumento pensato per promuovere una maggiore collaborazione tra territorio e ospedale. Inoltre, l'applicazione verrà estesa per poter gestire contesti clinici ulteriori, ma sempre legati a patologie croniche e asintomatiche, quali l'obesità, il diabete e le cardiopatie. Il carattere di generalità e il di codice modulare e riutilizzabile sono state le principali linee guida dello sviluppo dell'applicativo, proprio per poter integrare facilmente altri contesti medici.

Bibliografia

- [1] National Kidney Foundation Guidelines
[<https://www.kidney.org/kidneydisease>]
- [2] De Nicola L, Donfrancesco C, Minutolo R, et al. Epidemiologia della malattia renale cronica in Italia: stato dell'arte e contributo dello studio CHARES. G Ital Nefrol 2011; 28 (4): 401-7.
- [3] Go AS, Chertow GM, Fan D, McCulloch CE, Hsu CY. Chronic kidney disease and the risks of death, cardiovascular events, and hospitalization. N Engl J Med 2004; 351: 1296-305.
- [4] D.A. Procaccini et al Qualità della Vita nei nefropatici. G Ital Nefrol, 2008; 25 (6): 694-701
- [5] Censis [<http://www.censis.it>]
- [6] Martello M, Di Luca M. Acetate-free biofiltration. G Ital Nefrol 2012; 29 (S55): S62-S71.
- [7] ASNET (Associazione Sarda Nefropatici e Dializzati).
[<http://www.asnetsardegna.com/>]
- [8] Nephromeet. [<http://www.nephromeet.com/>]
- [9] Fielding RT, Taylor RN. Principled design of the modern web architecture. ACM Trans. Inter. Tech., 2(2):115-150, 2002

- [10] Fielding RT. Architectural Styles and the Design of Network-based Software Architectures, Doctoral dissertation, University of California, Irvine, 2000.
- [11] Pautasso C, Zimmermann O, Leymann F. RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision, 17th International World Wide Web Conference (WWW2008), Beijing, China, 2008-04
- [12] Johnson R. J2EE Design and Development. Wrox Press Ltd, 2002.
- [13] Principe P. Java 7 Guida completa. Apogeo, 2011.
- [14] Marrs T, Davis S. JBoss At Work: A Practical Guide. O'Reilly, 2009.
- [15] Maurizzi M, Argentiero L, Billi C. Progettazione e sviluppo del layer BPM. Documento NoemaLife SpA, 2014
- [16] Camilli G, Bevilacqua L. Smart Health 2.0, Capitolato tecnico.
- [17] Galileo, the Clinical Community Solution. [<http://www.noemalife.com/>]
- [18] DNLab, la soluzione per la diagnostica di laboratorio. [<http://www.noemalife.com/>]

Ringraziamenti

In prima istanza, ringrazio il mio responsabile Cesare e tutto lo staff *NoemaLife*, in particolare Marco Giglione e il team di Bari, che mi hanno seguito nella mia prima esperienza aziendale. Grazie a loro ho imparato tanto in pochissimo tempo.

Ringrazio poi la mia famiglia, senza la quale nulla sarebbe stato possibile, le mie amiche, in particolare Marianna e Michela, con le quali ho condiviso tanto, e Natasja, prima confidente poi coinquilina. Ringrazio tutti i colleghi ed ex-colleghi di università: Luca e Rita, Andre, Pasquale, Laura, Saro, Gianni, Ivan, Massimo, Marco, Massimiliano, Pillo, Stefano, Roberto, Vincenzo e tutti gli altri. Un grazie va anche alle segretarie Barbara, Irene e Camilla, che mi hanno seguito nel mio percorso da tutor didattico in questo ultimo anno accademico, e a tutto lo staff del dipartimento CS. Ringrazio Enrico, che in questi due anni è stato fidanzato, amico, confidente e beta tester, e che, nonostante tutto, mi ha sempre consigliato, consolato e incoraggiato.

Infine un ringraziamento speciale va al mio relatore, prof. Alessandro Amoroso, che oltre ad essere un ottimo insegnante, mi ha seguito nella stesura della tesi, sapendomi guidare e consigliare.