POLITECNICO DI TORINO

Collegio di Ingegneria Biomedica

Corso di Laurea Magistrale
in Ingegneria Biomedica

Tesi di Laurea Magistrale

Sviluppo di un'applicazione e studio delle tecniche di Natural Language Processing per l'estrazione di dati dalle cartelle cliniche

Relatrici
Prof.ssa Gabriella Balestra
Prof.ssa Samanta Rosati

Candidata
Giuliana Fiorella

Anno accademico 2020-2021

Sommario

I Percorsi Diagnostico Terapeutici Assistenziali (PDTA) costituiscono dei modelli strutturati di fondamentale importanza per le aziende sanitarie, poichè consentono di individuare i percorsi di cura e le pratiche cliniche migliori nell'ambito della gestione di uno specifico problema di salute.

L'obiettivo di questa tesi è quello di porre le basi necessarie relative all'organizzazione dei dati per le operazioni successive di definizione del PDTA per pazienti fragili chirurgici, per i quali vi è l'intenzione di costruire il modello.

Per raccogliere i dati strutturati estraibili dalle cartelle cliniche, è stato costruito un database relazionale ed è stata progettata e programmata un'applicazione dedicata alla procedura di caricamento dei dati, in modo da renderla standardizzata e funzionale.

Al fine di integrare tra le fonti anche i dati contenuti nei testi clinici scritti in linguaggio naturale, è stato condotto uno studio delle tecniche di Natural Language Processing (NLP) e dei principali tool open source disponibili per la lingua italiana. Tramite un'analisi delle performance, sono state selezionate due librerie di Python da poter utilizzare in modo combinato per la realizzazione di un sistema di NLP in grado di estrarre dati ed informazioni da testi clinici.

Indice

Introduzione

Il contesto

Il principale effetto di una medicina sempre più efficace e mirata nel trattare le malattie è quello dell'invecchiamento della popolazione, fenomeno che ormai può essere considerato globale [1].

Da uno studio dell'ISTAT del 2019 è emerso che già solo in Italia l'11,7% della popolazione è costituita da anziani dai 75 anni in sù [2]. A livello mondiale già nel 2015, in occasione della Giornata Internazionale degli Anziani, il *World Report on Ageing and Health* pubblicato dalla *World Health Organization* prevedeva che entro il 2050 la popolazione anziana con età superiore a 60 anni sarebbe raddoppiata [3].

Con l'invecchiamento della popolazione non aumenta solo il numero di soggetti con età superiore ai 65 anni, che possiamo definire "anziani", ma anche i "grandi anziani", aventi un'età maggiore di 85 anni [1]. Soprattutto per quest'ultima categoria di soggetti, l'aumento dell'aspettativa di vita comporta una serie di problematiche che coinvolgono sia le caratteristiche fisiologiche e psicologiche del paziente sia quelle socio-economiche, ambientali e culturali dello stesso, tale da renderlo un paziente "complesso" [4].

Il fenotipo "complesso" non definisce solo una somma delle patologie a cui il paziente è soggetto, nè la sua età, ma rappresenta una condizione con specifiche caratteristiche relative all'eziopatogenesi[1], alle necessità terapeutiche e alla prognosi [5].

Ad aggravare la complessità di un paziente, vi è molto spesso la coesistenza di vari disturbi, condizione definita dal Ministero della Salute come *multimorbilità* [5].

Secondo l'*Agency for Healthcare Reasearch and Quality*, un paziente complesso soggetto a patologie croniche multiple è dunque una persona avente due o più malattie croniche, quindi caratterizzato da una condizione di multimorbilità, e ognuna di queste malattie può influenzare l'esito dei trattamenti delle altre patologie concomitanti [1].

In generale, la multimorbilità risulta essere oggi una delle più grandi sfide socio-sanitarie da fronteggiare, poichè già solo in Italia interessa circa un terzo della popolazione

[1]Eziopatogenesi: l'analisi del processo di insorgenza di una patologia e del suo sviluppo (patogenesi), con particolare attenzione alle sue cause (eziologia). Risorsa online: https://it.wikipedia.org/wiki/Eziopatogenesi (Ultimo accesso: 12 gennaio 2021).

adulta [1]. Non è dunque una condizione che riguarda unicamente gli anziani, ma con l'aumentare dell'età aumenta la sua prevalenza e sugli anziani e i grandi anziani produce un grande impatto sulla qualità della vita [1] [4].

Un paziente anziano complesso, caratterizzato da una condizione di multimorbilità, di conseguente politerapia, molto spesso di ridotta autosufficienza e a volte oggetto di problematiche sociali e familiari viene definito come paziente "fragile" [6]. Nello specifico [7]:

> la fragilità [...] comporta una limitazione delle attività quotidiane dovuta alla presenza di pluripatologie e un deterioramento della salute e dello stato funzionale, che predispone a esiti negativi. In particolare si tratta di soggetti anziani con comorbilità[2] e instabilità clinica, disabilità e rischio di eventi avversi, con elevata incidenza di ospedalizzazione e/o morte.[...] Più recentemente la fragilità è stata considerata un'entità multidimensionale, definita da fattori fisici, psicologici, sociali e ambientali[...].

Si individua dunque l'interesse per un approccio clinico mirato, non dedicato esclusivamente alla cura della singola malattia, ma strategicamente pensato per garantire al paziente fragile un servizio e un'assistenza specifici per le sue esigenze sanitarie e socio-sanitarie, tramite anche l'individuazione di percorsi diagnostico terapeutici adeguati [7].

Si inserisce in questo panorama la volontà da parte dell'ospedale Humanitas Gradenigo di Torino di realizzare un reparto *ad hoc* per il paziente fragile grande anziano che deve essere sottoposto ad un intervento chirurgico, in cui il lavoro sinergico di un team interdisciplinare di specialisti consenta un percorso di cura mirato e specifico per questa categoria di pazienti.

Soprattutto nella situazione di grave crisi sanitaria che tutto il mondo vive ormai dai primi mesi del 2020 e che ha interessato in prima linea la popolazione anziana, perchè prima vittima del virus Sars-CoV-2, nonostante la pandemia abbia anche causato uno sconvolgimento nell'organizzazione delle strutture sanitarie, la tutela della salute dei pazienti anziani e l'aumento della qualità del servizio sanitario offertogli rimangono un aspetto di fondamentale importanza da attenzionare e curare.

Il progetto

L'obiettivo del progetto riguarda lo sviluppo di un Percorso Diagnostico Terapeutico Assistenziale (PDTA) per pazienti fragili chirurgici, da adottare all'interno del reparto che

[2]Comorbilità: presenza di ogni altra patologia distinta preesistente o coesistente rispetto alla malattia "indice", ovvero alla malattia che determina un peggioramento dello stato di salute in un individuo, e/o l'evento acuto o la malattia che ne condiziona maggiormente la prognosi [7].

l'ospedale Humanitas Gradenigo di Torino vuole sviluppare per questa categoria di pazienti. Si tratta nello specifico di pazienti geriatrici caratterizzati da una condizione di multimorbilità e aventi un'età maggiore di 85 anni che vengono ricoverati per subire almeno un intervento chirurgico durante la degenza.

I PDTA sono degli schemi clinico-assistenziali-organizzativi in cui vengono delineate tutte le attività e le risorse che fanno parte dei processi che si svolgono all'interno dell'organizzazione sanitaria e che riguardano una specifica categoria di pazienti [8]. L'adozione di un modello di questo tipo all'interno di una struttura ospedaliera serve a garantire un'elevata qualità dell'assistenza effettivamente erogata e di quella percepita dal paziente, di migliorare i risultati che si ottengono nel percorso di cura e di promuovere la sicurezza del paziente, tramite l'individuazione di tutte le risorse necessarie [9].

Questi percorsi, che normalmente vengono rappresentati tramite dei diagrammi di flusso, possono essere realizzati usufruendo di diverse tecniche. In questo lavoro vi è l'interesse nell'utilizzare metodi automatici per l'ottenimento del modello. In particolare, si vuole utilizzare la tecnica del process mining, già ampiamente utilizzata da anni nell'ambito della gestione dei processi aziendali.

Il process mining è una tecnica di gestione dei processi che ne consente l'analisi, la modellizzazione e lo sviluppo [10]. Si tratta di un insieme di algoritmi di data mining[3] e computational intelligence[4] che vengono applicati sui dati al fine di ottenere una rappresentazione descrittiva del processo di interesse [11].

I dati strutturati su cui saranno applicati gli algoritmi verranno opportunamente estratti dalle cartelle cliniche appartenenti alla categoria di pazienti descritta sopra e rese disponibili dall'ospedale Humanitas Gradenigo di Torino e verranno riorganizzati in modo da ricostruire la sequenza temporale delle attività.

Gli algoritmi molto spesso forniscono dei modelli quasi ideali, che nella maggior parte delle volte risultano impraticabili così per come sono, soprattutto nell'ambito di un contesto multiforme e dinamico come quello sanitario. A tal proposito, in seguito alla realizzazione del primo modello di PDTA, sarà necessario rivalutare manualmente tutto il percorso, cercando di calarlo il più possibile nel contesto di impiego.

Maggiori dettagli in merito al percorso di tesi verranno presentati nel paragrafo successivo, mentre tutti gli approfondimenti relativi agli strumenti e alle tecniche a cui si è fatto cenno sopra verranno forniti nel capitolo 1.

[3] Il data mining è l'insieme di tecniche e metodologie che hanno per oggetto l'estrazione di informazioni utili da grandi quantità di dati, attraverso metodi automatici o semi-automatici e l'utilizzo scientifico, aziendale, industriale e operativo delle stesse. Risorsa online: https://it.wikipedia.org/wiki/Data_mining (Ultimo accesso: 12 gennaio 2021).

[4] L'espressione *Computational Intelligence* (CI) si riferisce ad un set di metodologie computazionali ispirate alla natura e alla biologia utilizzate per affrontare complessi problemi del mondo reale. Risorsa online: https://en.wikipedia.org/wiki/Computational_intelligence (Ultimo accesso: 12 gennaio 2021).

Obiettivo della tesi

Nel paragrafo precedente è stata fornita una descrizione essenziale dei tratti salienti del progetto relativo alla definizione di un Percorso Diagnostico Terapeutico Assistenziale per pazienti fragili chirurgici, pensato in collaborazione con l'ospedale Humanitas Gradenigo di Torino.

Il lavoro da portare avanti, essendo dedicato ad una specifica realtà ospedaliera, necessita dell'impiego di diverse tipologie di conoscenze, sia provenienti da fonti materiali come le cartelle cliniche, sia di natura specialistica e professionale.

A causa dei problemi legati alla pandemia da COVID-19, che hanno determinato forti rallentamenti in tutti gli altri ambiti della sanità e non solo, non è stato possibile avvalersi del supporto di esperti, nè della disponibilità dei dati necessari su cui potersi basare per lo sviluppo del modello.

Date tali motivazioni, all'interno di questo panorama progettuale, la presente tesi si inserisce nella fase preliminare all'applicazione degli algoritmi di process mining e si prepone l'obiettivo di curare gli step di progettazione e costruzione di una base dati in cui verranno raccolti tutti i dati di interesse estratti dalle cartelle cliniche.

Prendendo come riferimento lo studio prodotto dalla Dott.ssa Giulia Parternostro nella sua tesi "Analisi di cartelle cliniche di pazienti geriatrici sottoposti ad intervento chirurgico", è stata progettata e programmata un'applicazione per PC tale da rendere più semplice e comoda la fase di inserimento dei dati nel database.

L'applicazione è stata progettata nei particolari tramite l'uso dei diagrammi resi disponibili da UML[5] e si è preso spunto dal lavoro della collega per curare i contenuti delle interfacce, realizzate appositamente per rispecchiare la posizione degli oggetti nelle schede reali delle cartelle. Tramite l'uso del toolbox App Designer di Matlab, l'applicazione è stata sia programmata a livello di codice sia costruita in termini di interfacce grafiche.

Per valutare il corretto funzionamento dell'interfacciamento del programma con il database MySQL precedentemente progettato e strutturato, è stata eseguita un'operazione di testing, facendo uso delle sei cartelle cliniche, opportunamente anonimizzate, che l'ospedale Humanitas Gradenigo ha messo a disposizione.

Il database è stato costruito e pensato per contenere dati di tipo strutturato, ma le cartelle cliniche contengono anche documentazione di altra natura, come i referti degli esami e delle visite specialistiche e i diari clinici, che possono essere fonte di ulteriori informazioni utili per la realizzazione del PDTA.

La seconda parte del lavoro di questa tesi dunque si inserisce nell'ambito dell'elaborazione dei testi scritti in linguaggio naturale.

[5]Lo *Unified Modeling Language* (UML) è un linguaggio che permette, tramite l'utilizzo di modelli visuali, di analizzare, descrivere, specificare e documentare un sistema software anche complesso. Risorsa online: https://www.html.it/guide/guida-uml/(Ultimo accesso: 12 gennaio 2021).

Il Natural Language Processing (NLP) è una disciplina molto vasta che si occupa dell'analisi e dell'elaborazione del linguaggio umano al fine di rendere i computer capaci di interagire con l'uomo. In questo frangente si fa riferimento all'applicazione delle tecniche di NLP per l'analisi dei testi clinici scritti in lingua italiana. L'obiettivo finale per gli scopi del progetto sarebbe quello di sviluppare un sistema in grado di estrarre dalla documentazione non strutturata, contenuta nelle cartelle cliniche, dati strutturati, al fine di inserirli nel database e utilizzarli nella fase successiva di applicazione degli algoritmi di process mining.

Si è svolto a tal proposito uno studio approfondito dei principali tool open source di NLP adatti alla compresione della lingua italiana, tramite un processo di valutazione delle performance nell'analisi di testi standard non clinici, che ha portato alla selezione di due librerie di Python. La scelta è stata effettuata sia sulla base dei risultati ottenuti durante la valutazione sia perchè l'ambiente Python fornisce la possibilità di usarli in sinergia e sfruttare le loro potenzialità.

Utilizzando alcuni referti estratti dalle sei cartelle messe a disposizione, si è proseguito con la scrittura di un semplice programma, ottenuto in linguaggio Python tramite l'uso dei tool selezionati, per la valutazione delle performance nell'analisi di testi clinici e per stimare la capacità di un sistema così sviluppato di estrarre informazioni da testi di questa natura.

La fase di verifica portata avanti per i testi clinici ha dimostrato che, grazie ai diversi metodi impiegati dal NLP e alla moltitudine di possibilità che questi offrono, è possibile ideare vari approcci tramite i quali estrarre informazioni. L'approccio che si sceglie di adottare dipende dalla quantità e dalla tipologia di dati che si vogliono ottenere, insieme alla collezione di documenti che si hanno a disposizione per eventuali procedure di apprendimento degli algoritmi. Il lavoro termina dunque con una analisi dedicata a presentare le diverse opzioni e possibilità che il NLP open source offre per la lingua italiana per l'elaborazione dei testi clinici e per la progettazione di un sistema pensato per le necessità di questo progetto.

Struttura della tesi

L'organizzazione del documento ricalca perfettamente il flusso delle attività che è stato seguito in questo lavoro, pertanto la struttura è la seguente:

1. Capitolo 1: Come costruire il PDTA.

   In questa sezione vengono delineati e descritti nel dettaglio cosa sono i PDTA e i metodi e gli strumenti che verranno utilizzati in questo progetto per realizzare un PDTA per pazienti fragili chirurgici.

2. Capitolo 2: Sviluppo del database e dell'applicazione a supporto.

   Si descrivono tutte le fasi di progettazione, costruzione e testing dell'applicazione e del database, compresi i diagrammi realizzati per la modellizzazione.

3. Capitolo 3: Natural Language Processing.

   Questo capitolo contiene un *excursus* descrittivo dedicato a spiegare cosa è il NLP in tutte le sue sfaccettature e come si inserisce in un contesto clinico.

4. Capitolo 4: NLP per i testi clinici in italiano.

   Si racchiude in quest'ultimo capitolo lo studio che è stato condotto sulle performance dei tool di NLP per la lingua italiana e sulle possibilità che un sistema sviluppato *ad hoc* possa fornire per l'estrazione di specifiche informazioni dai testi clinici.

Capitolo 1

Come costruire il PDTA

1.1 I Percorsi Diagnostico Terapeutici Assistenziali

L'allungamento della vita media e soprattutto l'aumento dell'incidenza di malattie croniche e di condizioni di multimorbilità hanno determinato negli ultimi decenni un conseguente aumento della complessità degli interventi terapeutici e riabilitativi e del sistema di cure complessivo.

Le strutture ospedaliere dunque si rivelano essere delle realtà multidisciplinari e interdisciplinari, che coinvolgono diverse figure professionali per il trattamento dei problemi di salute dei pazienti [12]. Meccanismi di questo tipo possono indurre a creare involontariamente situazioni di variabilità e di disequità nell'erogazione delle cure, facilitando la possibilità che insorgano errori [12].

Nasce allora l'esigenza da parte delle strutture sanitarie di possedere degli strumenti ben definiti e delineati in grado di migliorare l'operatività dell'organizzazione e di fornire servizi sanitari di elevata qualità.

L'obiettivo dei Percorsi Diagnostico Terapeutici Assistenziali (PDTA) è quello di fornire dei modelli strutturati, ma non eccessivamente rigidi, dedicati a specifiche categorie di pazienti e costruiti sulla base delle migliori pratiche cliniche, delle Linee Guida fornite relativamente ad una patologia o ad una problematica clinica e delle risorse che si hanno a disposizione [12]. Si tratta dunque di [12]:

> [...] modelli locali che, sulla base delle linee guida ed in relazione alle risorse disponibili, consentono un'analisi degli scostamenti tra la situazione attesa e quella osservata in funzione del miglioramento della qualità. I PDTA sono, in pratica, strumenti che permettono all'azienda sanitaria di delineare, rispetto ad una patologia o un problema clinico, il miglior percorso praticabile all'interno della propria organizzazione.

Secondo la definizione che è stata accordata durante il consensus meeting internazionale del 2015 in Slovenia, i PDTA rappresentano «una metodologia mirata alla condivisione dei processi decisionali e dell'organizzazione dell'assistenza per un gruppo specifico di pazienti durante un periodo di tempo ben definito» [9]. Nel 2015 tale definizione è stata ufficialmente accettata dall'European Pathway Association (E-P-A). Negli anni successivi, l'E-P-A, in seguito a vari dibattiti internazionali e a studi approfonditi, ha ritenuto più pertinente considerare i PDTA come dei veri e propri "interventi complessi", piuttosto che appellarli genericamente come una "metodologia" [9].

In effetti i PDTA rappresentano degli «schemi clinico-assistenziali-organizzativi ad alta complessità che utilizzano la logica di gestione per processi per ricostruire l'iter assistenziale, visto come insieme di processi, sotto processi, attività, attori, confini (input e output) e responsabilità» [8].

Secondo la nuova definizione dell'E-P-A, gli aspetti basilari di un PDTA includono [9]:

1. Una chiara esplicitazione degli obiettivi e degli elementi chiave dell'assistenza basata su evidenze scientifiche, best practice, aspettative dei pazienti e loro caratteristiche;
2. La facilitazione delle comunicazioni tra i membri del team e i pazienti e le loro famiglie;
3. Il coordinamento del processo di assistenza tramite il coordinamento dei ruoli e l'attuazione consequenziale delle attività dei team multidisciplinari di assistenza, dei pazienti e delle loro famiglie;
4. La documentazione, il monitoraggio e la valutazione delle varianze e degli outcome;
5. L'identificazione delle risorse appropriate.

Un PDTA riguarda dunque la rappresentazione di un processo o di più processi in termini di sequenza temporale e spaziale delle attività che lo caratterizzano, mettendo in luce gli outcome, i ruoli di ciascuna risorsa, sia interna sia esterna all'organizzazione e come esse devono collaborare [12].

Tramite l'implementazione di un PDTA si realizza un'unica visione sistemica della struttura ospedaliera o addirittura dell'intera organizzazione sanitaria territoriale relativamente ad uno specifico problema di salute, con l'obiettivo di garantire continuità degli interventi, integrazione tra le unità organizzative e operative e omogenità nell'erogazione dei servizi e delle prestazioni sanitarie [8].

1.1.1 Le fasi di sviluppo di un PDTA

Un PDTA deve essere progettato tenendo sempre conto dei bisogni del paziente "tipo", di quelli degli operatori e dell'intera organizzazione. E' stato definito come un modello

locale, proprio perchè è necessario calarlo all'interno del contesto specifico in cui verrà impiegato [12]. A tal proposito, è fondamentale che nella fase precedente alla progettazione venga dichiarato il suo ambito di estensione, definendolo come PDTA ospedaliero o PDTA territoriale o come Profilo Integrato di Cura (PIC) se deve rappresentare entrambe le entità [12].

Al fine di garantire sempre e comunque un'efficiente applicabilità del modello, l'attività di implementazione di un PDTA è essa stessa un processo continuo e sempre soggetto a fasi di monitoraggio e miglioramento.

Sebbene il processo di realizzazione del percorso sia piuttosto complesso e composto da un numero molto elevato di step, esso può essere riassunto nelle seguenti fasi [12]:

1. Scelta del problema di salute
2. Ricognizione dell'esistente
3. Costruzione del percorso ideale
4. Costruzione del percorso di riferimento
5. Fase pilota
6. Attuazione del PDTA all'interno dell'azienda.

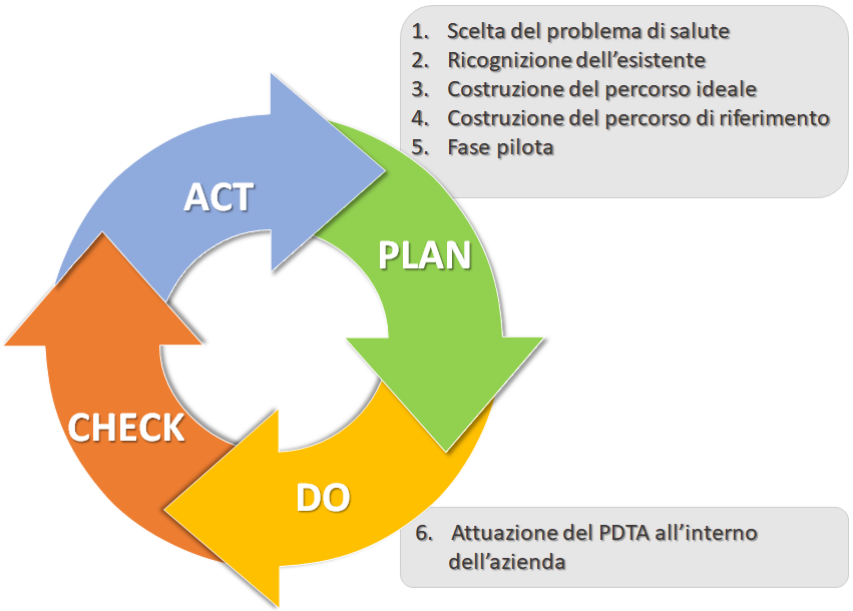


Figura 1.1: Sviluppo di un PDTA rappresentato in un ciclo PDCA. Immagine realizzata a partire da: Berti, E., La Porta, P., Serra, V. et al. *Guida per i valutatori alla verifica dei Percorsi Diagnostico Terapeutici Assistenziali (PDTA) nell'ambito delle visite di accreditamento.* Regione Emilia-Romagna. Servizio Sanitario Regionale Emilia-Romagna. 2013.

Lo sviluppo di un PDTA può essere perfettamente rappresentato in un ciclo PDCA (Plan-Do-Check-Act) [8], come mostrato in figura 1.1.

Gli step di progettazione (1-5) si inseriscono nella fase di pianificazione (Plan), mentre l'attuazione del PDTA all'interno dell'azienda si colloca nella fase di implementazione (Do) [8]. Durante tutto l'arco di tempo in cui il modello viene utilizzato dall'organizzazione vengono effettuati degli Audit[6] periodici, stabiliti durante la fase di pianificazione, tramite i quali si esegue una revisione sistematica e continuativa delle attività (Check), al fine di individuare aspetti critici e di attuare opportuni miglioramenti (Act) [8] [12].

Di seguito si descrivono più nel dettaglio le fasi del processo di realizzazione di un PDTA.

Scelta del problema di salute La scelta del problema di salute deriva da un'analisi dei bisogni che deve essere contestualizzata in ogni realtà organizzativa [12]. Esistono dei criteri sulla base dei quali si sceglie la patologia o la problematica clinica su cui costruire il PDTA. Tra i criteri di priorità che possono incidere sulla scelta è possibile indicare [12]:

- impatto sulla salute del cittadino (prevalenza, incidenza e mortalità della patologia);
- impatto sulla salute della comunità;
- impatto sulla rete familiare;
- presenza di linee-guida specifiche;
- variabilità e disomogeneità delle prestazioni;
- precisa definizione della patologia in esame;
- semplicità clinica/assistenziale;
- impatto economico.

Ricognizione dell'esistente In questa fase l'obiettivo è raccogliere tutte le informazioni relative alla gestione del problema individuato al punto precedente.

Si comincia dalla raccolta della documentazione ufficiale contenuta negli archivi o nei database della struttura, a seconda che essa sia cartacea o informatizzata [12]. Di norma si tratta delle cartelle cliniche dei pazienti, che per ovvi motivi verranno utilizzate opportunamente anonimizzate. A questi documenti si associano interviste mirate, ottenute presso il personale addetto e direttamente coinvolto nel percorso da definire, e i risultati degli eventuali focus group [12].

I focus group sono sostanzialemte gruppi di persone selezionate appositamente per una discussione focalizzata su uno specifico argomento. Da un incontro di questo tipo

[6] In ambito sanitario, l'Audit è un'analisi critica e sistematica della qualità dell'assistenza medica (o sanitaria) che valuta le procedure clinico/organizzative utilizzate per la diagnosi e il trattamento, l'uso delle risorse, gli outcome risultanti e la qualità di vita per i pazienti [12].

emergono opinioni, problemi ed idee progettuali che possono essere davvero utili per la costruzione del percorso [12].

Costruzione del percorso ideale Il primo percorso che viene elaborato è un percorso "ideale", in quanto definisce le procedure che idealmente dovrebbero essere eseguite. Questo modello è frutto del risultato di quanto emerge dalla letteratura e dalle Linee Guida in merito al problema di salute scelto al primo punto e verrà utilizzato come confronto con il percorso "di riferimento", al fine di identificare problemi e scostamenti [12].

Costruzione del percorso di riferimento Una volta individuato il percorso ideale, il gruppo di lavoro si focalizza sugli aspetti pratici del processo, quindi sugli obiettivi attesi, sulle singole attività e sugli attori e ricostruisce il percorso di riferimento, definito come «la migliore sequenza temporale e spaziale possibile delle attività da svolgere nel contesto di una determinata situazione organizzativa e di risorse» [12].

Si tratta dunque del percorso reale che verrà adottato all'interno della struttura e che, in seguito a continui confronti effettuati con le Linee Guida e con il percorso ideale, sarà soggetto a monitoraggio e a variazioni.

Fase pilota Nella fase pilota si valuta l'applicazione del percorso di riferimento all'interno dell'organizzazione, con l'obiettivo di individuare punti critici da riprogettare e attività da aggiungere o rimuovere, nell'ottica di soddisfare gli obiettivi individuati alla fase precedente [12].

Attuazione del PDTA all'interno dell'azienda In quest'ultimo step si integra il percorso di riferimento sviluppato e revisionato all'interno dell'azienda, pianificando comunque delle procedure di monitoraggio periodiche [12].

1.1.2 Come rappresentare un PDTA

In seguito a diversi studi, il diagramma di flusso è risultato essere lo strumento più immediato ed esplicativo per rappresentare un PDTA [8] [12]. Tramite l'utilizzo di un diagramma di flusso è possibile rappresentare schematicamente tutte le diverse attività che costituiscono un processo, mettendone in risalto le responsabilità, le risorse, gli snodi decisionali e le interfacce tra le diverse strutture [8].

L'obiettivo definitivo, come già citato in precedenza, riguarda la realizzazione della «migliore sequenza temporale e spaziale possibile delle attività da svolgere» [12].

In figura 1.2 si riportano i principali simboli utilizzati all'interno di un diagramma di flusso e il loro significato.

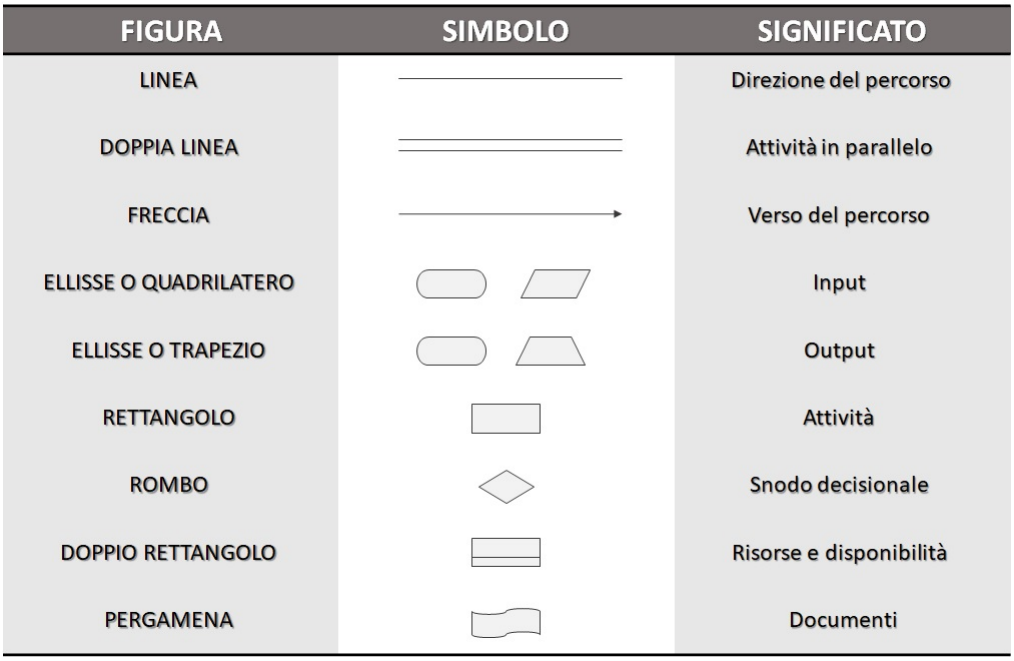


Figura 1.2: Simboli del diagramma di flusso. Immagine realizzata a partire da: Martelli, S., Laura, B. et al. *Raccomandazioni per la costruzione di Percorsi Diagnostico Terapeutici Assistenziali (PDTA) e Profili Integrati di Cura (PIC) nelle Aziende Sanitarie della Regione Piemonte*. Regione Piemonte, Agenzia Regionale per i Servizi Sanitari (Aress). 2007.

In figura 1.3 si rappresenta inoltre un esempio di diagramma di flusso realizzato per il seguente percorso [8]:

> ... La paziente accede all'ambulatorio di Oncologia Medica per essere sottoposta a visita ambulatoriale per l'esecuzione di eventuale triplo test. In questa fase, una volta verificata l'esistenza del sospetto diagnostico, l'ICM attiva il percorso ambulatoriale ASI e inserisce la paziente nel percorso. Nel caso in cui il sospetto diagnostico non venga confermato la paziente seguirà altri percorsi assistenziali. Successivamente viene eseguita una valutazione congiunta multidisciplinare per la valutazione e la scelta del programma terapeutico ...

L'esempio di percorso e la sua rappresentazione in figura 1.3 sono stati tratti dalla "Guida per i valutatori alla verifica dei Percorsi Diagnostico Terapeutici Assistenziali (PDTA) nell'ambito delle visite di accreditamento" realizzata dalla Regione Emilia-Romagna.

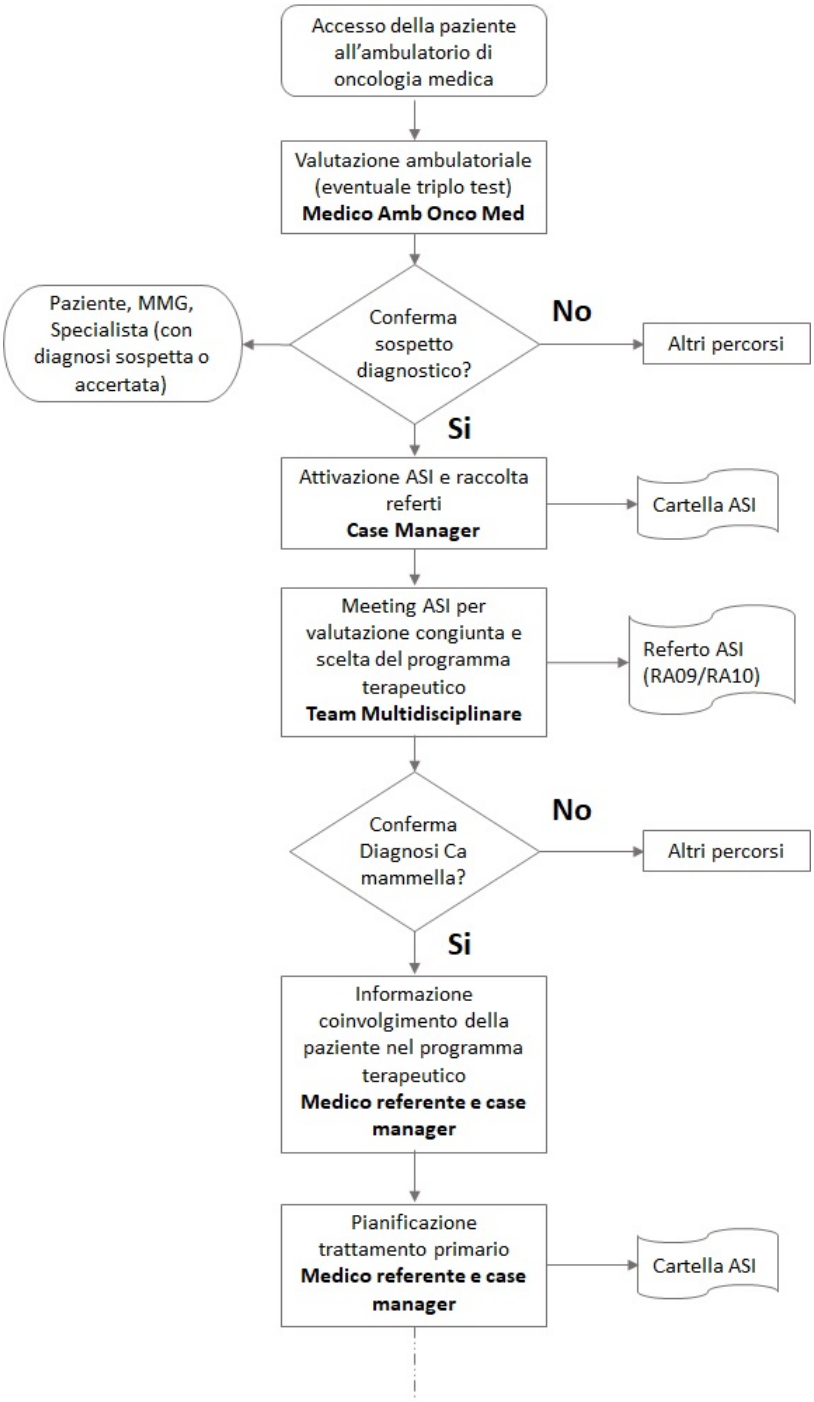


Figura 1.3: Esempio di un diagramma di flusso per PDTA. Immagine tratta da: Berti, E., La Porta, P., Serra, V. et al. *Guida per i valutatori alla verifica dei Percorsi Diagnostico Terapeutici Assistenziali (PDTA) nell'ambito delle visite di accreditamento.* Regione Emilia-Romagna. Servizio Sanitario Regionale Emilia-Romagna. 2013.

1.2 Process mining

I processi sanitari descritti e modellizzati nei PDTA possono essere considerati come dei processi aziendali, poichè anch'essi si focalizzano su come tutte le figure e i dipartimenti della stessa organizzazione debbano lavorare insieme per produrre un outcome [13]. Le

tecniche di analisi, progettazione e gestione dei processi aziendali sono dunque applicabili anche nel caso dei processi sanitari.

Il *Business Process Analysis* (BPA) tradizionale è il metodo principale sfruttato dalle aziende di tutto il mondo per capire un processo e migliorarne la sua efficienza [14]. Secondo l'analisi prevista dal BPA, i modelli dei processi vengono realizzati a partire dalle descrizioni fornite dalle risorse dell'organizzazione in merito alle attività che caratterizzano il processo e a come vengono svolte [13]. E' necessario dunque assumere che quanto riportato sia valido, non ambiguo e rappresenti effettivamente la realtà [13]. Nonostante il BPA sia il metodo tradizionale per l'analisi dei processi aziendali, oltre ad essere un procedimento lungo che richiede tempo, è influenzato dalle discrepanze che invece molto spesso sussitono tra gli effettivi processi, per come realmente si svolgono all'interno dell'azienda, e la percezione che ne hanno le risorse coinvolte [13].

Il process mining offre un nuovo approccio utile per oltrepassare questi problemi [13], combinando tecniche di data mining e computational intelligence all'analisi, alla modellizzazione e allo sviluppo dei processi [10] [15].

Il data mining viene normalmente implementato all'interno del processo di estrazione di conoscenza dai dati, meglio conosciuto come *Knowledge Discovery in Databases* (KDD) [16], con lo scopo di riconoscere relazioni, associazioni, anomalie e pattern tra grandi quantità di dati e di riassumerli per eventuali elaborazioni successive [15] [16].

Il process mining è una branca altamente specialistica del data mining, a sua volta coadiuvata dall'utilizzo di strumenti tipici della computational intelligence (es. reti neurali, logica fuzzy, ecc.) e calata nel contesto del *Business Process Management* (BPM)[7] [13] [17].

Il concetto di process mining può essere agevolmente riassunto dal grafico riportato i figura 1.4.

Ogni organizzazione, come ad esempio un'azienda sanitaria, ha a disposizione diversi sistemi informativi tramite i quali viene memorizzarata un'elevata mole di dati [13]. In ambito sanitario basti pensare alle cartelle cliniche, che contengono tutte le informazioni diagnostico-terapeutiche relative ad un paziente [18]. L'analisi del processo viene effettuata proprio a partire dai dati raccolti, riducendo i tempi di elaborazione ed affidandosi *in primis* a quanto viene archiviato nei database [13]. La strutturazione schematica di un database consente di avere a disposizione tutte le informazioni relative alle attività del processo in modo organizzato e specifico e fornisce la possibilità di operare un'estrazione automatica dei dati.

Tuttavia, gli algoritmi di process mining non vengono applicati sui dati grezzi contenuti all'interno dei database, ma più nello specifico su quelli che vengono definiti *event log*,

[7] *Business Process Management*: disciplina che combina la conoscenza della tecnologia dell'informazione e delle scienze gestionali e le applica ai processi aziendali operativi [15].

ovvero registri di eventi.

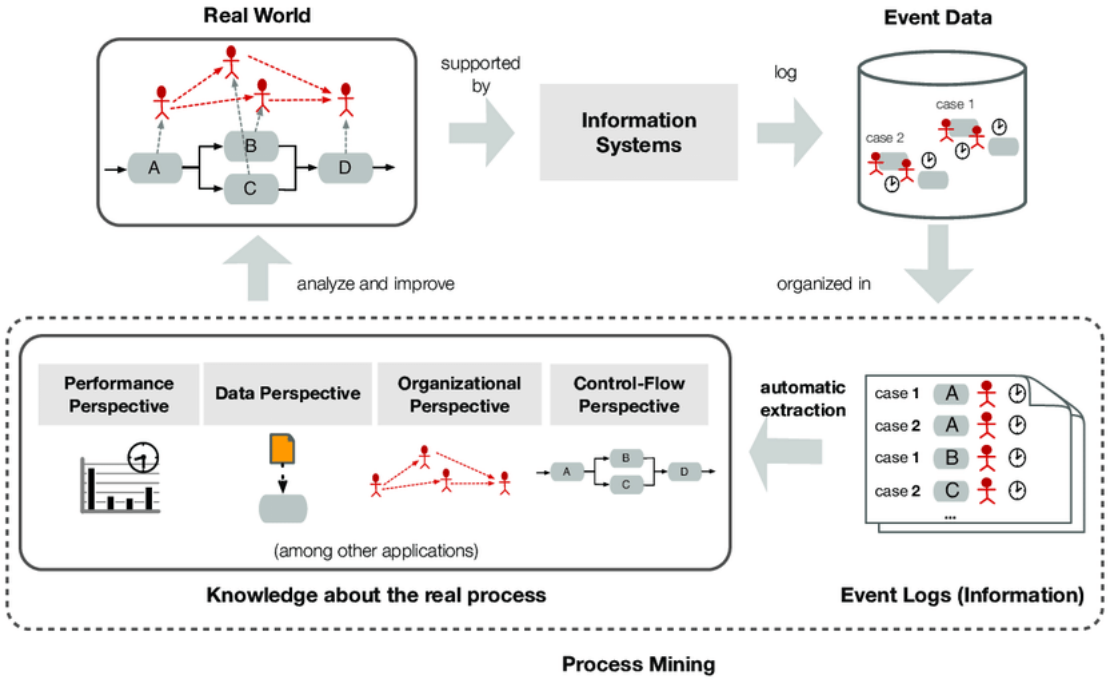


Figura 1.4: Il concetto di Process Mining. Immagine tratta da: Silva Rebuge, A. J. da. «Business Process Analysis in Healthcare Environments». Dissertation for the degree of Master of Science. Technical University of Lisbon, 2012.

Negli *event log* i dati vengono organizzati in modo da riprodurre la sequenza temporale e spaziale degli eventi, tramite l'identificazione del tipo di attività, di chi l'ha svolta e quando e delle risorse che ne sono rimaste coinvolte, con l'opportunità di inserire eventuali dati aggiuntivi considerati necessari [19], come mostrato nell'esempio in figura 1.5.

patient	activity	timestamp	doctor	age	cost
5781	make X-ray	23-1-2014@10.30	Dr. Jones	45	70.00
5541	blood test	23-1-2014@10.18	Dr. Scott	61	40.00
5833	blood test	23-1-2014@10.27	Dr. Scott	24	40.00
5781	blood test	23-1-2014@10.49	Dr. Scott	45	40.00
5781	CT scan	23-1-2014@11.10	Dr. Fox	45	1200.00
5833	surgery	23-1-2014@12.34	Dr. Scott	24	2300.00
5781	handle payment	23-1-2014@12.41	Carol Hope	45	0.00
5541	radiation therapy	23-1-2014@13.57	Dr. Jones	61	140.00
5541	radiation therapy	23-1-2014@13.08	Dr. Jones	61	140.00
...	...	...	...	...	...

case id | activity name | timestamp | resource | other data

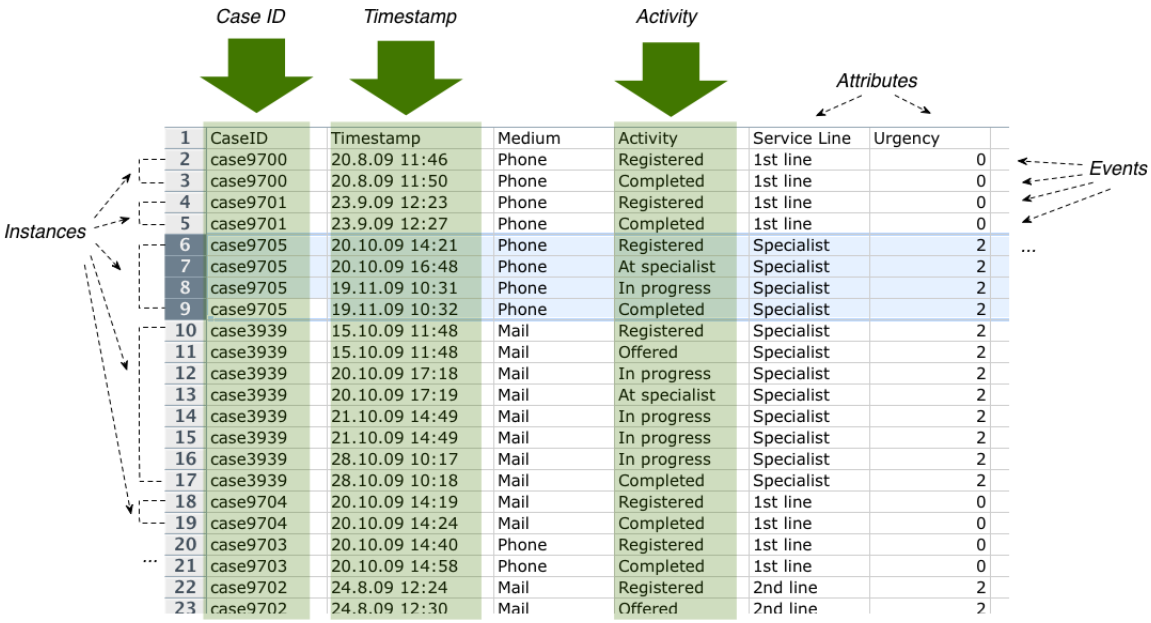
Figura 1.5: Esempio di Event Log. Immagine tratta da: Rudnitckaia, J. «Process Mining. Data science in action». In: University of Technology, Faculty of Information Technology (2015), pp. 1–11.

Tre sono le informazioni minimali che vanno indicate in un *event log* relativamente ad un evento [19]:

- *Case ID*: indice identificativo dell'evento, necessario per distinguere diverse istanze dello stesso processo;
- *Activity*: specifica azione compiuta nel processo;
- *Timestamp*: dettagli temporali (data e orario) per identificare la corretta sequenza degli eventi.

In aggiunta, per ottenere una ricostruzione fedele degli eventi è necessario inserire anche le risorse, ovvero indicare gli attori che compiono le attività del processo [15] [19].

Ogni riga del registro di eventi si riferisce ad un evento, mentre tutti gli eventi relativi ad un unico *Case ID* costituiscono un'istanza del processo [19], così come è possibile osservare in figura 1.6.



	CaseID	Timestamp	Medium	Activity	Service Line	Urgency
2	case9700	20.8.09 11:46	Phone	Registered	1st line	0
3	case9700	20.8.09 11:50	Phone	Completed	1st line	0
4	case9701	23.9.09 12:23	Phone	Registered	1st line	0
5	case9701	23.9.09 12:27	Phone	Completed	1st line	0
6	case9705	20.10.09 14:21	Phone	Registered	Specialist	2
7	case9705	20.10.09 16:48	Phone	At specialist	Specialist	2
8	case9705	19.11.09 10:31	Phone	In progress	Specialist	2
9	case9705	19.11.09 10:32	Phone	Completed	Specialist	2
10	case3939	15.10.09 11:48	Mail	Registered	Specialist	2
11	case3939	15.10.09 11:48	Mail	Offered	Specialist	2
12	case3939	20.10.09 17:18	Mail	In progress	Specialist	2
13	case3939	20.10.09 17:19	Mail	At specialist	Specialist	2
14	case3939	21.10.09 14:49	Mail	In progress	Specialist	2
15	case3939	21.10.09 14:49	Mail	In progress	Specialist	2
16	case3939	28.10.09 10:17	Mail	In progress	Specialist	2
17	case3939	28.10.09 10:18	Mail	Completed	Specialist	2
18	case9704	20.10.09 14:19	Mail	Registered	1st line	0
19	case9704	20.10.09 14:24	Mail	Completed	1st line	0
20	case9703	20.10.09 14:40	Phone	Registered	1st line	0
21	case9703	20.10.09 14:58	Phone	Completed	1st line	0
22	case9702	24.8.09 12:24	Mail	Registered	2nd line	2
23	case9702	24.8.09 12:30	Mail	Offered	2nd line	2

Figura 1.6: Esempio di Event Log. Immagine tratta da: https://fluxicon.com/blog/2012/02/data-requirements-for-process-mining/(Ultimo accesso: 14 gennaio 2021)

Gli eventi che si riferiscono ad un *Case ID* sono inseriti in modo ordinato all'interno del registro, seguendo le indicazioni temporali, e ognuno di essi può essere caratterizzato anche da altri attributi, scelti in base al contesto [19].

A partire dunque da processi reali, gli algoritmi di process mining vengono applicati agli *event log* con l'obiettivo di estrarre informazioni in modo automatico. La conoscenza può essere estratta nell'ottica di quattro diverse prospettive [11] [15] [19]:

- *Control-flow perspective*: si focalizza sul creare un ordine temporale delle attività, tramite l'uso delle reti di Petri o di altre notazioni come UML;

- *Organization perspective*: si pone l'obiettivo di strutturare l'organizzazione tramite l'identicazione dei ruoli di ciascuna risorsa e unità organizzativa;

- *Case perspective o Data perspective*: si focalizza sulle proprietà dei *case*, prendendo in considerazione i valori dei dati contenuti nell'*event log*;

- *Time perspective o Performance perspective*: si concentra sugli aspetti temporali degli eventi e della frequenza con cui si presentano.

Come mostrato in figura 1.7, all'interno del flusso di applicabilità del process mining, se ne possono individuare tre diverse tipologie [11] [15] [19]:

- *Discovery*: a partire dai registri di eventi, senza il supporto di un modello già esistente *a priori*, si realizza un modello di processo, rappresentato con diverse possibili tecniche di raffigurazione, come le reti di Petri;

- *Conformance*: i nuovi registri di eventi vengono messi a confronto con un modello di processo già realizzato in una fase precedente, con l'obiettivo di procedere ad una valutazione e ad una convalida dello stesso, così da assicurarsi che rispecchi sempre lo svolgimento dei processi reali all'interno dell'organizzazione;

- *Enhancement*: il nuovo *event log* viene messo a confronto con il modello già esistente, con lo scopo di arricchirlo e migliorarlo con nuovi aspetti e prospettive.

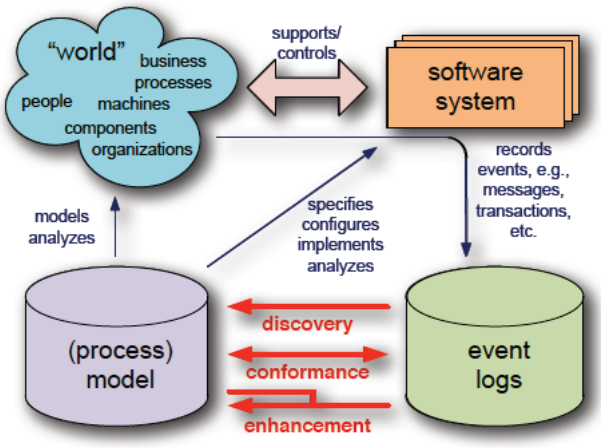


Figura 1.7: Il flusso di applicabilità del process mining. Immagine tratta da: Aalst et al. «Process Mining Manifesto». In: vol. 99. Ago. 2011, pp. 169–194. isbn: 978-3-642-28107-5. doi: 10.1007/978-3-642-28108-2_19.

In figura 1.8 si riporta una rappresentazione schematica delle tre tecniche in termini di input e output prodotti.

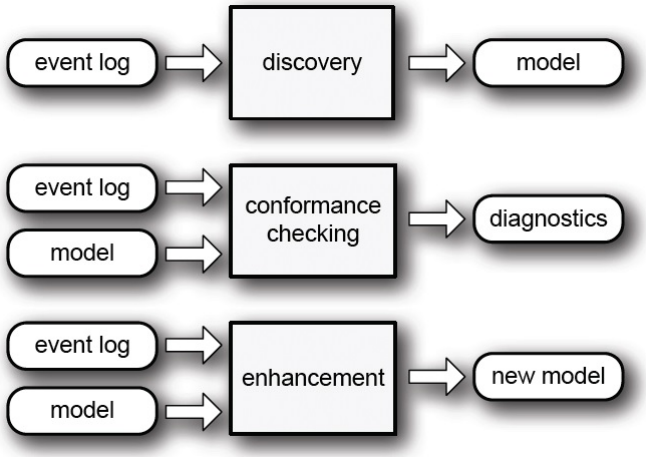


Figura 1.8: Input e output delle tre tecniche di process mining. Immagine tratta da: Aalst et al. «Process Mining Manifesto». In: vol. 99. Ago. 2011, pp. 169–194. isbn: 978-3-642-28107-5. doi: 10.1007/978-3-642-28108-2_19.

Sulla base delle prospettive di interesse e del tipo di tecnica che si sceglie di adottare, è possibile sfruttare il process mining per diverse applicazioni.

Nel caso specifico del progetto in questione, si fa riferimento alla sviluppo di un modello *ex novo* tramite l'implementazione della tecnica del *Discovering*. Il risultato atteso si identifica con un PDTA, che verrà rappresentato in forma di diagramma di flusso e ottenuto a partire dall'elaborazione dei dati estratti dalle cartelle cliniche. Come verrà spiegato meglio nel paragrafo successivo, prima della costruzione dei registri di eventi, in questo frangente è necessario inserire una fase preliminare di costruzione e riempimento di un database, a causa della mancanza di informatizzazione delle cartelle cliniche di interesse. L'obiettivo complessivo del presente lavoro di tesi è dunque quello di realizzare strutturalmente un database in grado di contenere tutti i dati necessari per l'elaborazione successiva degli *event log*.

1.3 La cartella clinica

La principale fonte di dati e di informazioni per lo sviluppo del PDTA obiettivo di questo progetto è la cartella clinica.

La cartella clinica è lo strumento per eccellenza usato in ambito sanitario per la gestione della salute di un paziente, in quanto documenta tutte le informazioni diagnostico-terapeutiche dello stesso [18].

Le definizioni in letteratura sono molteplici. Tra le tante si riporta quella del prof. Renzo Celesti, professore dell'Università degli Studi di Genova presso il Dipartimento di Scienze della Salute, secondo il quale la cartella clinica è «il complesso ordinato e scritto dei dati clinici (anamnestici, obiettivi, specialistici, strumentali e documentali) raccolti dai sanitari sulla persona del malato nel corso della degenza ospedaliera» [20].

La tutela della salute del paziente non è però l'unica finalità del documento, ma è altresì un atto pubblico, in quanto redatta da un pubblico ufficiale esercente le proprie funzioni, di fede privilegiata (Art. 2699 e seg c.c.) e quanto riportato in essa fa fede fino a querela di falso [20]. Ha dunque «efficacia probatoria, ha valore storico documentale e attesta il consenso informato» [20].

Il documento deve contenere tutte le informazioni relative all'anagrafica, alle pratiche cliniche di diagnostica e di terapia che sono state effettuate durante tutto l'arco di tempo in cui il paziente è stato preso in carico dalla struttura sanitaria, quindi dal momento dell'accettazione fino alla fase di dimissione.

Nonostante ogni organizzazione possieda delle proprie schede realizzate in modo specifico per i propri meccanismi di gestione, ogni cartella clinica deve comunque contenere le seguenti informazioni [20]:

- Generalità del paziente
- Data e ora di ricovero
- Diagnosi all'ingresso
- Anamnesi patologica remota e prossima
- Esame obiettivo al ricovero
- Diario clinico
- Indagini diagnostiche, esami di laboratorio, accertamenti specialistici
- Terapie praticate durante la degenza
- Diagnosi alla dimissione
- Data e ora della dimissione
- Parere dei sanitari alla dimissione
- Eventuale trasferimento ad altro ospedale con o senza ambulanza
- Dichiarazioni esplicite relative al consenso del paziente.

La circolare del Ministero della Sanità (n.900 2/AG454/260) del 19/12/1986 attesta che «le cartelle cliniche, unitamente ai relativi referti, vanno conservate illimitatamente, poiché rappresentano un atto ufficiale indispensabile a garantire la certezza del diritto, oltre a costituire preziosa fonte documentaria per le ricerche di carattere storico-sanitario» [21]. Le cartelle cliniche forniscono dunque un'elevatissima quantità di dati ed informazioni, che possono essere utilizzati anche a scopo di ricerca.

Uno dei problemi principali legati all'uso delle cartelle cliniche è che non tutte le strutture hanno ancora adottato le cartelle cliniche elettroniche[8] per cui la fruizione della documentazione cartacea presenta una serie di svantaggi, tra cui [22]:

- Non facile e veloce consultazione (complessità)
- Non presenza di modalità di numerazione delle pagine/moduli
- Frammentazione delle infomazioni
- Eccessiva lunghezza, ridondanza
- Difficoltà nel trovare le informazioni di interesse
- Leggibilità (problemi legati alla grafia).

1.3.1 La cartella clinica di Humanitas Gradeningo

L'ospedale Humanitas Gradenigo di Torino si trova nella fase iniziale del processo di adozione della cartella clinica elettronica e negli ultimi mesi tale processo è stato interessato da un cospicuo rallentamento dovuto alla condizione pandemica da COVID-19 che ha coinvolto il mondo da marzo 2020 a questa parte. Per tali motivi, ai fini di questo lavoro sono state rese disponibili solo sei cartelle cliniche scansionate dal formato cartaceo. Si auspica che nelle fasi successive di questo progetto ci si troverà già in una fase avanzata di "sanità elettronica" [22], per cui il lavoro di estrazione di dati dalle cartelle risulterà maggiormente agevolato.

I pazienti "tipo" per i quali si vuole sviluppare un Percorso Diagnostico Terapeutico Assistenziale in questo progetto sono i pazienti fragili chirurgici, ovvero quei soggetti di età superiore agli 85 anni, definiti nell'introduzione grandi anziani, interessati da una condizione pluripatologica e che durante il ricovero debbano subire degli interventi chirurgici.

E' dunque questa la categoria di pazienti le cui cartelle cliniche devono essere analizzate e da cui è necessario estrarre i dati fondamentali per la progettazione del modello.

Questa tipologia di cartelle è già stata analizzata in un lavoro precedente dalla Dott.ssa Giulia Paternostro, nel suo progetto di tesi "Analisi di cartelle cliniche di pazienti geriatrici sottoposti ad intervento chirurgico", in cui, dalla documentazione fornita da Humanitas Gradeningo, sono state selezionate alcune specifiche tipologie di schede, perchè contenenti dati considerati fondamentali per lo sviluppo del percorso.

[8] La cartella clinica elettronica (*Electronic Patient Record*) è assimibilabile come contenuto alla cartella clinica di ricovero ospedaliero o a quella ambulatoriale specialistica. Si distingue dal Fascicolo Sanitario Elettronico (*Electronic Health Record*) che è invece un fascicolo di cartelle cliniche, indagini diagnostiche preventive e tutto quanto riguarda la salute presente e trascorsa della persona [22].

Di seguito vengono elencate le schede scelte [23]:

- Anamnesi, obiettività, piano di cura
- Attività fisioterapica
- Cartella anestesiologica ed intra-operatoria
- Esami di laboratorio
- Pianificazione giornaliera
- Scheda allergie/anamnesi farmacologica
- Scheda di terapia unificata
- Scheda dispositivi
- Scheda monitoraggio e medicazione lesioni
- Scheda monitoraggio dolore
- Scheda monitoraggio accessi venosi media-lunga permanenza
- Scheda parametri
- Scheda di dimissione ospedaliera (SDO)
- Scheda accettazione ingresso
- Tracciabilità trasfusioni ed emocomponenti
- Esami strumentali
- Valutazione rischio cadute
- Verbale di pronto soccorso

Soprattutto a causa della natura cartacea della documentazione a disposizione, si è rivelato necessario realizzare una base dati strutturata in cui memorizzare i dati e progettare e sviluppare un'applicazione tramite la quale interfacciarsi con la base dati stessa e così consentire un processo di inserimento dei dati più intuitivo ed agevole.

1.4 I database

Le basi di dati offrono un organizzato meccanismo di memorizzazione, gestione e reperimento delle informazioni in modo indipendente rispetto all'applicazione con cui si interfacciano [24]. Si distinguono da altre entità di conservazione dei dati come i file, in quanto in essi il formato dei dati è dipendente dal tipo di applicazione tramite i quali vengono gestiti [24].

Un database dunque è caratterizzato da un insieme di dati strutturati memorizzati elettronicamente all'interno di un sistema informatico [25] e organizzati in una collezione di file [24]. Il file è suddiviso a sua volta in record logici, ognuno dei quali rappresenta un insieme di dati [24].

Rispetto ad altre tipologie di sistemi di memorizzazione, i database sono in grado di contenere grandi quantità di dati, visualizzabili e gestibili da utenti diversi contemporaneamente, tramite opportuni metodi di amministrazione dei privilegi, e consentono un interfacciamento con i dati di tipo diverso a seconda delle applicazioni.

Per la creazione di un database è necessario affrontare tre diverse fasi di progettazione:

1. Progettazione concettuale: si distacca completamente dall'implementazione e dal tipo di database che si sceglie di usare. Il modello concettuale prodotto ha l'unico scopo di descrivere *cosa* deve essere rappresentato, dunque quali dati ed entità e le relazioni tra loro [26]. Questo step prescinde dal tipo di struttura del database e dai software di gestione che si adotteranno;

2. Progettazione logica: si scende nel particolare e si realizza un modello logico dedicato a descrivere *come* sono organizzati i dati nello specifico [26]. Questa modellizzazione è legata alla tipologia di struttura di database che si vuole implementare e alla natura effettiva dei dati, ma necessita di un modello concettuale di partenza per poter essere realizzata correttamente. Il modello logico che viene prodotto al termine di questo step di progettazione può essere realizzato con diagrammi di tipo diverso, a seconda del database che si sceglie.

3. Progettazione fisica: costruzione fisica delle entità del database e delle relazioni tra esse.

1.4.1 Il diagramma Entità-Relazione

Il modello concettuale viene descritto tramite un diagramma specifico che mette in risalto le entità e le relazioni che sussistono tra esse, chiamato diagramma Entità - Relazione (*Entity-Relationship diagram* in inglese), o diagramma E-R.

In figura 1.9 si riportano i componenti principali per la costruzione di un diagramma E-R. L'entità rappresenta uno specifico concetto appartenente al campo d'interesse e possiede diversi attributi che identificano le proprietà del concetto. Ogni entità possiede

un attributo, definito *chiave primaria*, che consente di distinguere un record da un altro e tramite il quale è possibile instaurare una relazione tra entità diverse [26].

Le relazioni che possono instaurarsi tra le entità sono di tre tipi:

- Relazione 1:1 (uno a uno)
- Relazione 1:N (uno a molti)
- Relazione N:N (molti a molti)

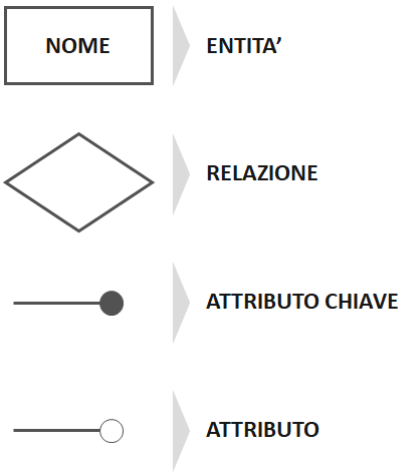


Figura 1.9: Elementi principali di un diagramma E-R. Immagine realizzata a partire da: Balestra, G. *Construction process*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

Un esempio di associazione tra due entità è rappresentato in figura 1.10.

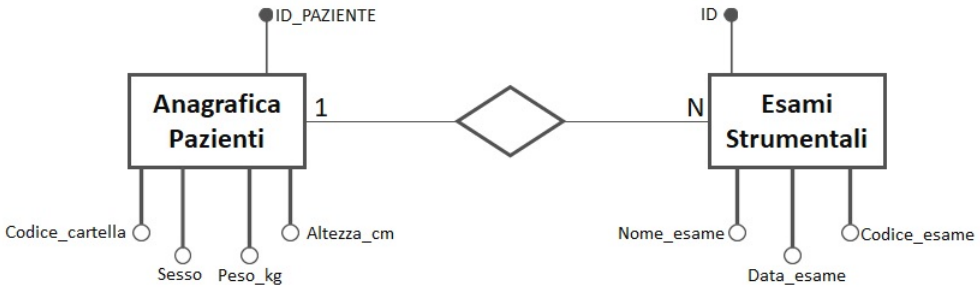


Figura 1.10: Esempio di relazione tra due entità rappresentata tramite il diagramma E-R

In questo semplice diagramma *Anagrafica Pazienti* e *Esami Strumentali* costituiscono due entità e tutte le proprietà indicate identificano i loro attributi. Le due chiavi primarie sono rispettivamente *ID_PAZIENTE* e *ID*, che oltre a determinare univocamente il record, in questo caso uno specifico paziente e uno specifico esame, consentono di instaurare una relazione uno a molti. In questo esempio la natura dell'associazione è dovuta al fatto che ad un paziente è possibile associare diversi esami strumentali, ma uno stesso esame non può essere stato eseguito con le stesse proprietà su pazienti diversi.

1.4.2 Principali tipologie di database

Esistono diversi tipi di database e ognuno di essi è adatto ad una specifica organizzazione dei dati. Tre sono i principali sistemi che si sono susseguiti dagli anni '60 fino agli anni '90 [25]:

- Database gerarchico
- Database reticolare
- Database relazionale

Negli anni '90, in seguito alla nascita della programmazione ad oggetti, sono nati i database orientati agli oggetti, in cui i dati vengono organizzati proprio come oggetti [25]. Nell'ultimo ventennio si è stati spettatori della creazione di tante tipologie diverse di database, caratterizzati da una struttura via via più complessa e specifica per determinati campi d'impiego, in modo tale da rispondere alle esigenze sempre più pressanti dello sviluppo delle applicazioni Web e, più di recente, del massiccio utilizzo del cloud[9] [25].

Degni di nota, soprattutto per l'interessante utilizzo in ambito sanitario, sono i data warehouse. Si tratta di repository centralizzati per grandi quantità di dati storici provenienti da diverse sorgenti, in cui i dati vengono memorizzati e archiviati allo scopo di essere analizzati [25]. Sono dei database a tutti gli effetti, dunque vengono costruiti ed interrogati con le stesse modalità, ma di norma contengono una mole di dati di gran lunga superiore, poichè vengono costruiti a partire dalle informazioni contenute in diversi database. I data warehouse oggi sono progettati per supportare gli utilizzatori nell'analisi retrospettivia dei dati a lungo termine e nei processi di *Data Quality Improvement* [24].

A livello sanitario si sta sviluppando sempre di più la tendenza delle strutture a creare dei data warehouse in cui archiviare la documentazione dei pazienti che hanno terminato la degenza. I dati così raccolti rimangono memorizzati e disponibili per essere consultati per studi successivi o *trial* clinici, in piena ottemperanza della *General Data Protection Regulation*[10] (GDPR).

Di seguito si riporta una descrizione più dettagliata delle prime tre tipologie di database, poichè essi hanno dato il via alla macro-evoluzione che ha interessato il mondo dei database dalla fine del secolo scorso fino ad oggi. Particolare importanza riveste l'analisi

[9] Il cloud è una rete globale di server, ognuno con una funzione univoca. Il cloud non è un'entità fisica, ma è una vasta rete di server remoti ubicati in tutto il mondo, collegati tra loro e che operano come un unico ecosistema. Risorsa online: https://azure.microsoft.com/it-it/overview/what-is-the-cloud/ (Ultimo accesso: 15 gennaio 2021).

[10] Il Regolamento Generale sulla Protezione dei Dati è il regolamento ufficiale n.2016/679 dell'Unione Europea in materia di trattamento dei dati personali e di privacy, adottato il 27 aprile 2016. Risorsa online: https://it.wikipedia.org/wiki/Regolamento_generale_sulla_protezione_dei_dati (Ultimo accesso: 17 gennio 2021).

del database relazionale, in quanto oggi ancora ampiamente utilizzato perchè è il «più efficiente e flessibile per accedere alle informazioni strutturate» [25].

Database gerarchico I dati sono organizzati in un modello ad albero, in cui i collegamenti sono di natura gerarchica [26], come si può apprezzare in figura 1.11. Ogni albero è costituito da un unico record radice detto "padre" (A) e da un insieme di rami definiti "figli" (B,C,D,E) [26]. Il numero di rami può variare, ma a livello più alto può esistere un unico "padre" [26]. Nonostante sia una struttura che consente facilmente la navigazione tra i dati, mantenendo comunque un livello di sicurezza di accesso elevato, essa consente esclusivamente una relazione uno-a-molti tra le entità che contengono i dati e per tale motivo, a causa della necessità di definire delle relazioni più complesse, è stato sviluppato il modello reticolare [26].

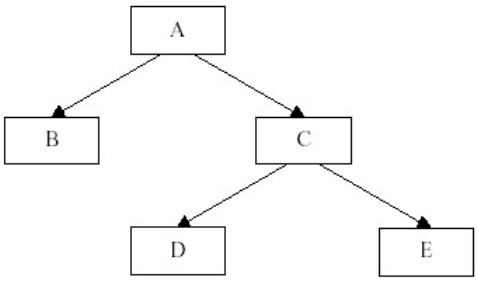


Figura 1.11: Struttura di un database gerarchico. Immagine tratta da: http://ilprofdinformatica.altervista.org/classe5/db_organizzazione.htm (Ultimo accesso: 15 gennaio 2021)

Database reticolare Questo tipo di organizzazione sfrutta una struttura a grafo, tale da consentire molteplici tipologie di relazioni tra i dati (figura 1.12) [26]. Un database reticolare risulta essere più flessibile di quello gerarchico, ma più difficile da gestire e da implementare e per tale motivo si è passati alla diffusione del database relazionale [26].

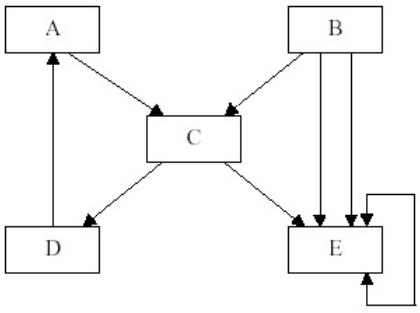


Figura 1.12: Struttura di un database reticolare. Immagine tratta da: http://ilprofdinformatica.altervista.org/classe5/db_organizzazione.htm (Ultimo accesso: 15 gennaio 2021)

Database relazionale L'approccio di tipo relazionale è quello più utilizzato, in quanto organizza i dati in tabelle, ognuna delle quali formata da righe e colonne. Una rappresentazione di questo tipo risulta particolarmente versatile e intuitiva da costruire e gestire [26]. Ogni tabella viene definita *entità*, le righe della tabella fanno riferimento alle *istanze* o *record*, mentre le colonne costituiscono gli *attributi* dell'entità, ovvero le sue proprietà [26].

Il modello logico di un database relazionale viene molto spesso rappresentato tramite un *Enhanced Entity-Relationship diagram* (Diagramma EER). Si tratta letteralmente di diagrammi E-R *estesi*, che consentono di ottenere una raffigurazione più dettagliata e specifica della struttura del database.

In riferimento all'esempio di modello concettuale mostrato in figura 1.10, in figura 1.13 si riporta il corrispettivo modello logico. Inoltre, le tabelle mostrate in figura 1.14 consentono di comprendere in modo più intuitivo e immediato lo schema logico di un database relazionale.

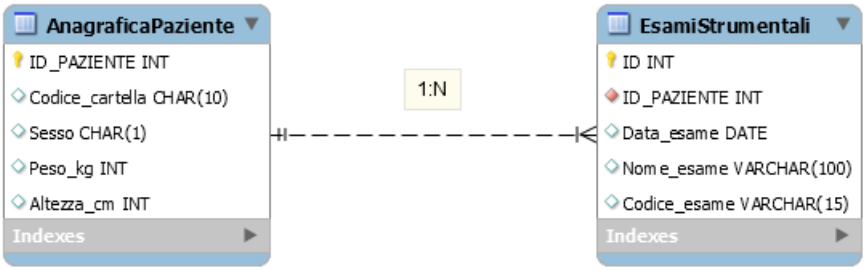


Figura 1.13: Esempio di relazione tra due entità rappresentata nel modello logico

ANAGRAFICA PAZIENTI

ID_PAZIENTE	Codice_cartella	Sesso	Peso_Kg	Altezza_cm
101	202000001	M	85	180
102	202005002	F	50	160
103	202040003	F	56	158
104	202050104	F	64	165

ESAMI STRUMENTALI

ID	ID_PAZIENTE	Nome_esame	Data_esame	Codice_esame
1	101	TC addome	15/01/2020	202000012873
2	101	RX femore	15/01/2020	202000052674
3	103	RM total body	15/01/2020	202000079913
4	101	RX femore	25/01/2020	202000085631

Figura 1.14: Esempio di relazione tra due entità rappresentata tramite le tabelle

Oltre all'inserimento del formato dei dati, un aspetto molto importante che va sottolineato è la presenza di una *chiave esterna*. La chiave esterna, che nell'esempio è

ID_PAZIENTE inserito nella tabella *Esami Strumentali*, non è altro che la chiave primaria di una tabella aggiunta come attributo ad un'altra tabella con la quale è in relazione. Nel caso specifico dell'esempio, la chiave *ID_PAZIENTE* ci consente di identificare in modo univoco qual è il paziente che ha effettuato l'esame.

1.4.3 Il Database Management System

Le basi di dati vengono gestite tramite un *Database Management System* (DBMS), ovvero un software di gestione del database. Esempi di DBMS possono essere MySQL, Microsoft Access, Microsoft SQL Server e Oracle Database [24].

Il DBMS è fondamentale poichè fa da interfaccia tra le applicazioni che accedono alla base di dati e la base di dati stessa e consentono di definire i privilegi con cui gli utenti possono accedere e modificare i dati.

I principali vantaggi di un DBMS sono i seguenti [24]:

- Indipendenza dei dati: i dati e i programmi che vi accedono sono indipendenti l'uno dall'altro;
- Integrità e sicurezza dei dati: la gestione dei privilegi protegge i dati sia dal danneggiamento sia dalla duplicazione;
- Flessibilità: grazie all'indipendenza rispetto alle applicazioni, qualsiasi cambiamento o nel database o nel programma non genera conseguenze;
- Velocità con cui si ottengono le informazioni richieste;
- Controllo della ridondanza dei dati;
- Eliminazione delle inconsistenze: tramite il controllo sulla duplicazione dei dati si eliminano i problemi di inconsistenza.

Il DBMS dunque è un software utilizzato per definire, creare e mantenere i database [24]. Esso consente di mantenere nascosto agli utilizzatori come sono memorizzati e organizzati i dati al suo interno, si assicura della consistenza degli stessi, controlla chi può accedervi e gestisce il loro recupero e il backup [24].

Quando si costruisce un database, bisogna fare riferimento all'architettura a tre livelli per DBMS, definita anche architettura ANSI/SPARC[11](figura 1.15) la quale divide il database in tre diversi livelli [24] [27]:

[11]Il nome ANSI/SPARC deriva dalla proposta che è stata fatta nel 1975 in merito all'architettura a tre livelli del DBMS da parte del comitato *Standard Planning and Requirements Committee* (SPARC) dell'ANSI (*American National Standards Institute*) [27].

- Livello Interno o Fisico: descrive la struttura a basso livello della base dati, dunque l'effettiva localizzazione dei dati contenuti al suo interno. Tale livello dipende dallo specifico DBMS che si utilizza;
- Livello Logico o Concettuale: descrive la struttura dell'intero database, quindi come sono organizzate le tabelle e le relazioni che sono instaurate tra di esse. La struttura in questo caso dipende dal modello di database che si sceglie (es. gerarchico, relazionale, reticolare, ecc.);
- Livello Esterno o Utente: rappresenta le viste specifiche per i diversi gruppi di utilizzatori, ognuno dei quali ha bisogno di una porzione diversa dei dati.

E' proprio grazie a questa astrazione dei dati su tre livelli che il DBMS è in grado di mantenere l'indipendenza logica (indipendenza dei vari livelli utente rispetto ai cambiamenti che avvengono a livello logico) e fisica (indipendenza del livello logico rispetto ai cambiamenti che avvengono a livello fisico) [27]. Un ulteriore vantaggio riguarda proprio la possibilità di avere un livello utente specifico per ogni tipo di user o gruppi di user, fornendogli esclusivamente la vista del database che gli interessa e in questo modo semplificarne l'utilizzo e ridurre l'insorgenza di errori [27].

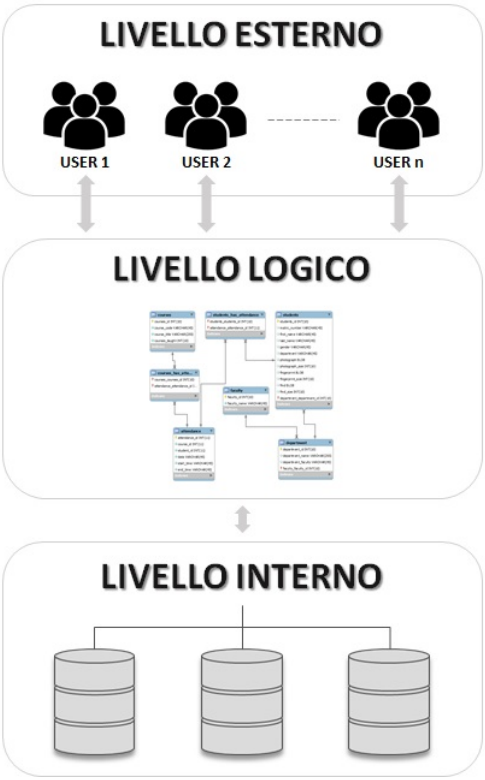


Figura 1.15: Architettura a 3 livelli

1.4.4 Lo Structured Query Language

Per comunicare con le basi di dati esistono dei linguaggi sviluppati appositamente per [24]:

- descrivere i dati, la loro struttura e i livelli interno ed esterno (*Data Definition Language*);
- manipolare i dati (*Data Manipulation Language*);
- controllare gli accessi al database e ai dati stessi (*Data Control Language*).

La maggior parte dei database relazionali utilizza un linguaggio di programmazione specifico, definito *Structured Query Language* (SQL) e accettato come linguaggio standard per i database relazionali dall'ANSI (*American National Standard Institute*) e dall'ISO (*Organization for Standardization*) [24].

SQL è un unico linguaggio che consente di effettuare tutte le operazioni inserite nell'elenco riportato sopra, per cui racchiude in un unico sistema i tre linguaggi per la definizione, la manipolazione e il controllo dei dati. In particolare, SQL consente di [24]:

- eseguire le *query*, dunque di interrogare il database per il reperimento dei dati;
- manipolare i record delle tabelle tramite le operazioni di inserimento, aggiornamento e cancellazione;
- manipolare il database stesso con operazioni di creazione, cancellazione, alterazione e sostituzione delle tabelle;
- controllare gli accessi degli utenti e i privilegi;
- garantire la consistenza del database e la sua integrità.

Prendendo come riferimento l'esempio mostrato in figura 1.13 e in figura 1.14, se si volessero estrarre dalla tabella *Esami Strumentali* le informazioni degli esami relativi al paziente *101* basterebbe scrivere la seguente *query* in linguaggio SQL:

```
SELECT EsamiStrumentali.Data_esame, EsamiStrumentali.Nome_esame,
EsamiStrumentali.Codice_esame
FROM EsamiStrumentali
WHERE ID_PAZIENTE = 101
```

Una volta eseguita la *query* il risultato ottenuto è mostrato in figura 1.16.

Per il lavoro svolto in questa tesi si è scelto di lavorare con un database relazionale MySQL e di sviluppare un'applicazione con il toolbox App Designer di Matlab per agevolare l'inserimento dei dati.

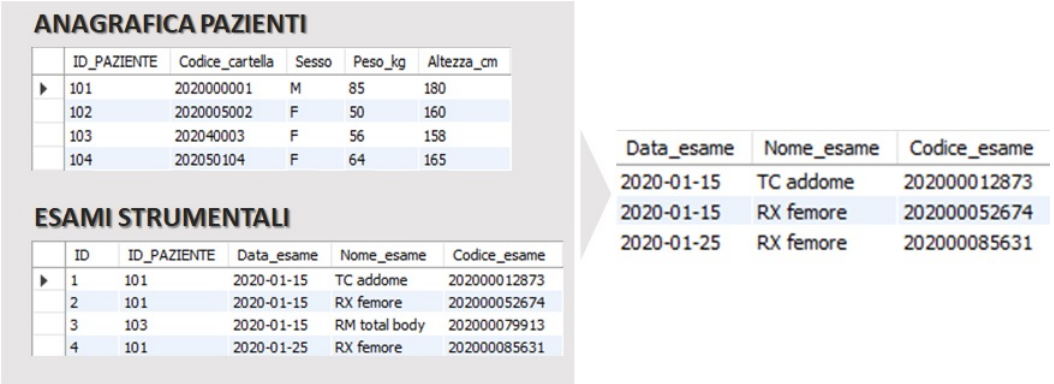


Figura 1.16: Esempio di risultato dell'esecuzione di una *query*

Nel capitolo successivo si affronteranno tutti i passi di progettazione e di implementazione che sono stati seguiti sia per la costruzione della base dati sia per la creazione dell'interfaccia utente.

Capitolo 2

Sviluppo del database e dell'applicazione a supporto

Il PDTA per pazienti fragili chirurgici che si vuole realizzare in questo progetto prevede, in uno dei primi step di sviluppo, la fase di ricognizione dell'esistente, come è stata definita nel primo capitolo. Raccogliere e memorizzare tutti i dati che servono per la costruzione del modello è fondamentale per l'elaborazione successiva tramite gli algoritmi di process mining.

Per potere essere applicate, le tecniche di process mining necessitano della creazione di registri di eventi, i quali devono contenere tutti gli eventi raccolti e ordinati che riguardano il processo in esame. Negli *event log* è importante inserire informazioni relative all'attività che viene svolta, a chi la svolge e quando, insieme ad altre informazioni aggiuntive ritenute essenziali [19].

Per realizzare i registri è necessario disporre di una base dati strutturata in cui raccogliere i dati da utilizzare per la ricostruzione dei processi.

La fonte principale di informazioni si identifica in questo caso con le cartelle cliniche dei pazienti "tipo". Si tratta di pazienti geriatrici interessati da una condizione di fragilità che sono stati ricoverati presso l'ospedale Humanitas Gradenigo di Torino e che nel corso della degenza ospedaliera hanno subito almeno un intervento chirurgico. E' importante che la documentazione utilizzata sia strettamente legata alle attività di Gradenigo, poichè l'obiettivo del progetto riguarda la realizzazione di un PDTA ospedaliero espressamente dedicato ai processi che interessano questa struttura. Nonostante la cartella clinica sia un documento che caratterizza la storia clinica di qualsiasi paziente, indipendentemente dall'ospedale in cui viene ricoverato, ogni organizzazione sanitaria possiede delle schede costruite appositamente per i propri meccanismi di gestione. Per tale motivo, per consentire un processo di inserimento dei dati immediato, in modo anche da abbassare la probabilità di insorgenza di errori, è stata sviluppata un'applicazione *ad hoc*.

Lo sviluppo di un sistema software è una procedura altamente complessa, in cui è

possibile distinguere una serie di fasi specifiche [28], come mostrato in figura 2.1.

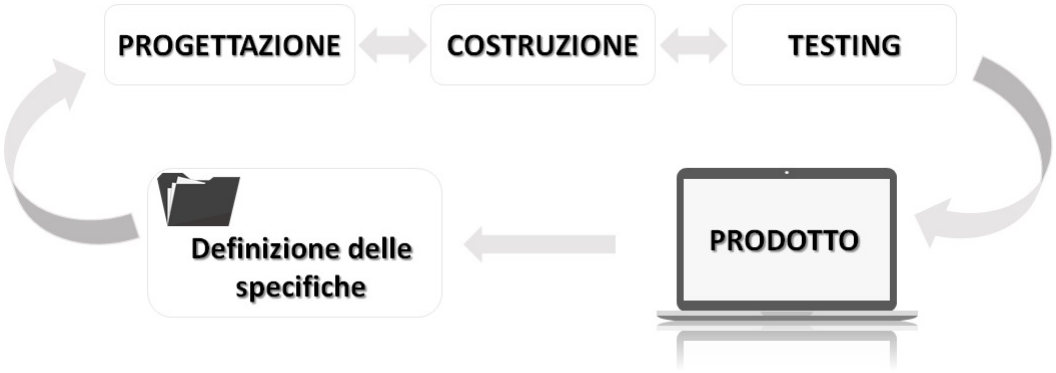


Figura 2.1: Flusso delle fasi di sviluppo di un software. Immagine realizzata a partire da: Balestra, G. *Requirements analysis*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

In una prima parte si fa riferimento alla fase di definizione delle specifiche in cui si descrivono i processi che vedono coinvolto il sistema, così da consentire l'individuazione delle funzionalità nell'operazione successiva di analisi. L'analisi delle specifiche e dunque la loro modellizzazione sono delle attività che fanno parte dello step di progettazione e risultano completamente indipendenti dagli strumenti che si utilizzeranno nella fase di costruzione per sviluppare fisicamente il software [28]. Tra le attività progettuali si inserisce anche la fase di progettazione concettuale del database. Una volta ottenuto un progetto strutturato, si scelgono i linguaggi e gli strumenti per la costruzione del sistema, sia per la scrittura del codice del programma sia per la creazione dello schema logico e fisico della base dati. Infine, si esegue una fase di testing, dedicata alla valutazione del corretto funzionamento del sistema, per identificare errori e modifiche da apportare.

Questo capitolo ha lo scopo di illustrare e documentare tutti i passaggi di progettazione e sviluppo del database e dell'applicazione che è stata realizzata a supporto, insieme ad una breve fase di testing finale, resa possibile grazie alla disponibilità di sei cartelle cliniche messe a disposizione dall'ospedale Humanitas Gradenigo di Torino.

Nonostante la scelta degli strumenti per la costruzione del sistema venga di norma fatta in una parte successiva del percorso, si è scelto di presentare nel paragrafo che segue tutti i software e i linguaggi che sono stati usati per questo lavoro, sia per la fase di progettazione sia per quella di costruzione.

2.1 Software e linguaggi usati

Il database che si è scelto di adottare in questo lavoro è un database relazionale, gestito dal DBMS MySQL. La scelta è stata effettuata sulla base delle necessità progettuali, in quanto serviva una tipologia di database facile da costruire e progettare e che potesse essere interrogato tramite SQL.

In particolare, per seguire fedelemente le tre fasi di progettazione sono stati utilizzati due diversi software:

- *RISE Editor* (versione 4.5.0.15) per la creazione del modello concettuale.
- *MySQL Workbench* (versione 8.0) per la progettazione logica e fisica. Tramite l'utilizzo del tool per la costruzione del diagramma EER, è stato realizzato il modello logico base e grazie alla possibilità di applicare il *forward engeneering*, a partire dal diagramma, è stato creato automaticamente lo scheletro fisico del database.

A supporto del database, per consentire una procedura di caricamento dei dati standardizzata e funzionale, è stato necessario sviluppare un'applicazione. Essendo un linguaggio abbastanza semplice da gestire e avendo a disposizione un tool per la costruzione di GUI (*Graphical User Interface*), è stato scelto Matlab come ambiente di sviluppo. Sebbene il programma creato risulti piuttosto semplice se paragonato alla grande moltitudine di software e app che esistono oggi, si è rivelato particolarmente utile e comodo seguire le principali tappe di progettazione che di norma si affrontano nell'ambito dello sviluppo dei software. La documentazione prodotta è fondamentale per essere sempre in grado di tener traccia delle modifiche effettuate e per consentire di seguire un lavoro modulare e monitorabile, anche per eventuali studi successivi e prospettive future nell'ambito di questo progetto.

A tale scopo, sono stati utilizzati i seguenti strumenti:

- *AstahUML* (versione 8.1.0) come UML tool per l'analisi e la modellizzazione delle specifiche. UML è stato usato per raccogliere ed organizzare le funzionalità (casi d'uso o *use case*) del programma, tramite l'adozione dello *Use case diagram*. Per rappresentare poi il flusso delle attività relative ai vari casi d'uso si è impiegato l'*Activity Diagram*.
- *MATLAB App Designer* (versione R2019b) per la costruzione fisica delle interfacce utente e la scrittura del codice. Grazie alle opzioni di condivisione, è stato possibile creare una *Standalone Desktop App*, gestibile al di fuori dell'ambiente Matlab e direttamente localizzata sul desktop. Inoltre, l'esistenza del toolbox *Database Explorer* ha consentito la creazione del collegamento con il database MySQL e l'opportunità di costruire le *query* necessarie per il caricamento dei dati.

2.2 Progettazione dell'applicazione

Il principale scopo dello sviluppo di questa applicazione è quello di consentire il caricamento all'interno del database dei dati estratti dalle cartelle cliniche. Nell'ottica di ottenere alla fine del progetto un PDTA, non tutti i dati contenuti nelle schede delle cartelle sono utili, ma sono state selezionate solo quelle voci considerate fondamentali per la

ricostruzione delle attività e delle risorse nell'ambito della gestione sanitaria dei pazienti fragili chirurgici. Il sistema che ci si è apprestati a costruire ha dunque come unica macro-funzionalità quella relativa alle operazioni di inserimento di dati strutturati. Non è stata dunque pensata per la modifica, la cancellazione o la visualizzazione di quanto già memorizzato, ma esclusivamente per il caricamento. Grazie poi alla semplicità di utilizzo del database adottato, è possibile effettuare modifiche e variazioni direttamente lavorando sulle tabelle in ambiente MySQL.

Inoltre, questa applicazione è stata pensata per essere utilizzata da utenti generici, in particolare da chiunque abbia l'obiettivo di inserire dati clinici nel database. Non è dunque un software medicale nè è stata concepita per essere utilizzata in ambito clinico, ma unicamente per scopi didattici.

Nel seguito di questo documento si farà riferimento all'applicazione anche con l'appellativo *Clinical records GUI* o semplicemente *GUI*.

2.2.1 Definizione delle specifiche

La fase preliminare alla progettazione è relativa all'individuazione dei processi che coinvolgono l'applicazione, al fine di definire il flusso delle attività di base.

L'utilizzatore, che d'ora in avanti in questo documento verrà chiamato *Utente*, farà sempre uso dell'applicazione con l'obiettivo di compilare le varie schede della cartella clinica e dunque caricare i dati nel database. Prima di effettuare questa operazione sarà sempre necessario o inserire il paziente in questione, se esso non è stato ancora caricato, o cercarlo nel database se era già stato inserito ma non erano state precedentemente compilate tutte le sue schede. In maniera schematica è possibile individuare tre macro-processi che coinvolgono l'utilizzo dell'applicazione:

- Inserimento del paziente
- Ricerca del paziente
- Compilazione della scheda

Per modellizzare i processi si è deciso di procedere con i *Synopsis diagram* e con i *Workflow diagram*. Il synopsis diagram è stato scelto perchè in modo sintentico ed immediato consente di inviduare gli aspetti di confine di un processo, ovvero gli attori, i dati di input e quelli di output, l'evento trigger del processo e lo stato del sistema alla fine dello stesso [29]. Per mettere in evidenza il flusso delle attività invece si è scelto di rappresentare il processo tramite i workflow diagram. Di seguito si riporta nel dettaglio la descrizione dei tre processi insieme ai modelli costruiti.

Inserimento del paziente & Ricerca del paziente Nel processo di inserimento del paziente l'evento trigger è proprio l'attività di inserimento della cartella clinica di quel

paziente nel database. Questa procedura si verifica nel momento in cui l'*Utente* non ha mai compilato l'anagrafica del paziente e non è mai stato assegnato l'*ID_PAZIENTE*. Il dato che in questo documento viene chiamato *ID_PAZIENTE* non è altro che il codice identificativo del paziente, necessario per l'identificazione del soggetto. Questo codice viene creato sempre anche in ambito sanitario, ma in questo caso è fondamentale soprattutto perchè i dati sono stati utilizzati in forma anonima, in ottemperanza al GDPR. In questo lavoro dunque è risultato un dato di grande importanza sia per la creazione delle relazioni tra le tabelle, come si vedrà nel paragrafo 2.3, sia per la gestione del salvataggio dei record a livello dell'applicazione.

Per potere avere la possibilità di inserire nuovi dati è necessario che l'*Utente* effettui l'accesso al server tramite le proprie credenziali. Questa funzionalità è fondamentale per mantenere la sicurezza del database ed evitare l'inserimento di dati errati o non autorizzati.

Anche il DBMS è stato considerato un attore del processo, in quanto gestisce l'esecuzione delle *query* ed effettua la memorizzazione dei record.

L'operazione di valutazione della correttezza del formato dei dati di input inseriti, secondo il flusso del percorso ideato, viene effettuato direttamente dalla *GUI*, in modo da gestire parte degli errori e dei problemi tramite la scrittura del codice relativo all'applicazione.

In figura 2.2 e 2.3 vengono riportati rispettivamente il synopsis diagram e il workflow diagram relativi al processo *Inserimento del paziente*.

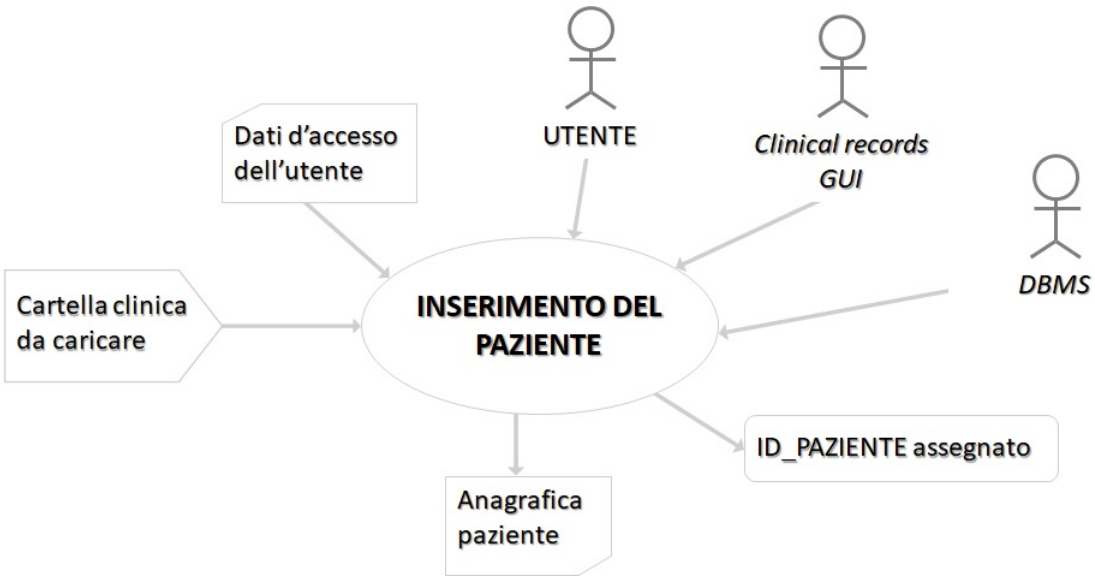


Figura 2.2: Synopsis diagram relativo al processo *Inserimento del paziente*

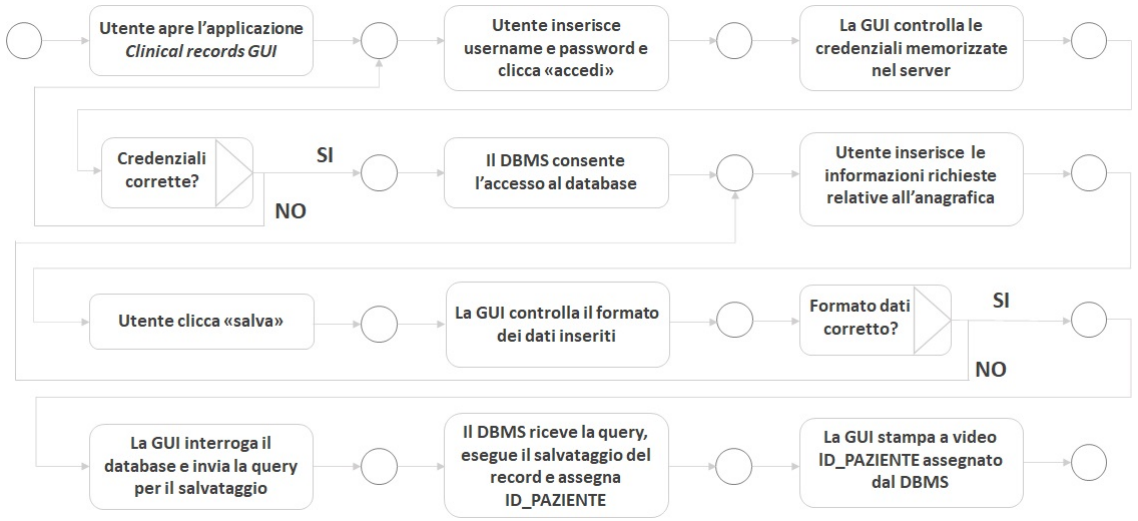


Figura 2.3: Workflow diagram relativo al processo *Inserimento del paziente*

Il processo *Ricerca del paziente* presenta sostanzialmente le stesse peculiarità, come è possibile apprezzare dai diagrammi 2.4 e 2.5, ma è più semplice poichè l'*Utente* invece di compilare i campi relativi all'anagrafica, inserisce il codice della cartella clinica e, se corretto, il sistema stampa a video l'*ID_PAZIENTE* che gli era già stato assegnato.

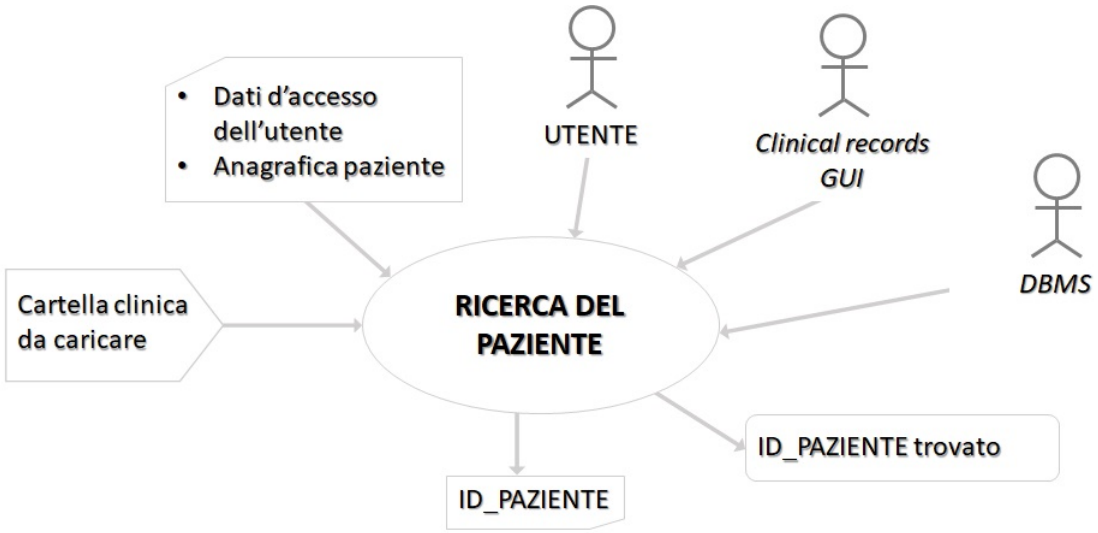


Figura 2.4: Synopsis diagram relativo al processo *Ricerca del paziente*

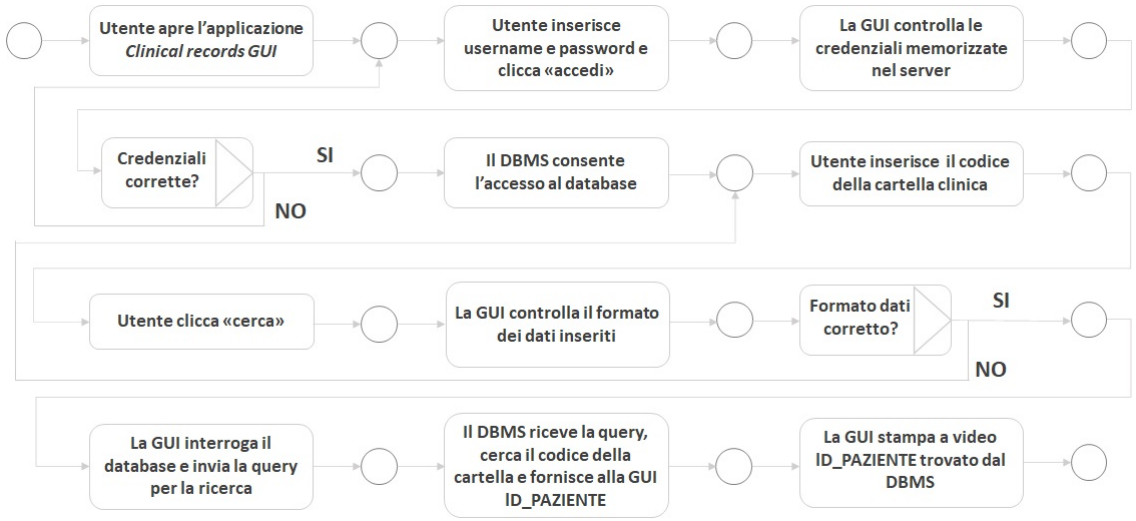


Figura 2.5: Workflow diagram relativo al processo *Ricerca del paziente*

Compilazione scheda Il processo relativo alla compilazione delle schede della cartella clinica è stato considerato come univoco, ignorando le distinzioni rispetto alle varie tipologie di schede. Si è giunti a questa conclusione in quanto, al di là del contenuto e dunque dei diversi campi da compilare, la sequenza delle attività che si eseguono durante questo processo è la stessa ed è rappresentata in figura 2.7.

Nel synopsis diagram in figura 2.6, si può notare come il dato di input fondamentale per consentire la compilazione e il salvataggio della scheda nel database sia l'*ID_PAZIENTE*, come già anticipato nella descrizione del processo precedente.

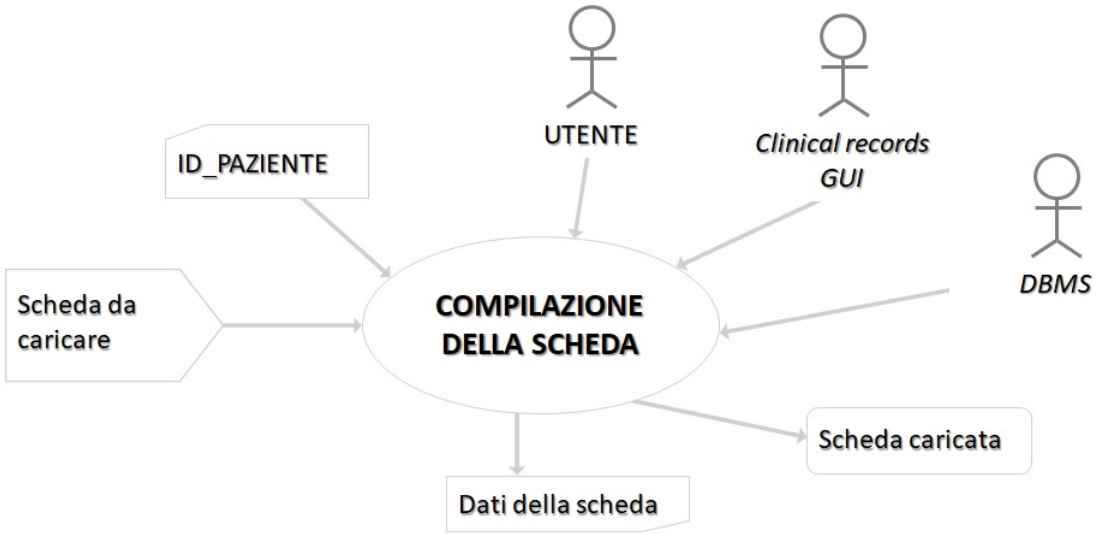


Figura 2.6: Synopsis diagram relativo al processo *Compilazione scheda*

Figura 2.7: Workflow diagram relativo al processo *Compilazione scheda*

2.2.2 Modellizzazione delle specifiche

La modellizzazione delle specifiche è una fase dell'attività di progettazione che in generale si prepone l'obiettivo di individuare i casi d'uso, le interfacce utente e tutti gli oggetti gestiti dal sistema a partire dalla descrizione processuale eseguita in precedenza.

Tramite l'analisi dei processi descritti e rappresentati nella fase di definizione delle specifiche, sono stati dunque individuati tutti i casi d'uso dell'applicazione ed è stato costruito lo use case diagram, un diagramma messo a disposizione da UML in grado di riassumere tutte le funzionalità del sistema e gli utenti che interagiscono con esse.

Il passo di progettazione successivo che è stato seguito riguarda la strutturazione delle interfacce. Le interfacce utente costituiscono lo strumento di comunicazione principale con l'utilizzatore. Devono essere sempre progettate in maniera semplice, devono essere esteticamente gradevoli, facili da usare e consentire flessibilità nella navigazione tra le pagine [28]. Nonostante la *Clinical records GUI* sviluppata in questo lavoro sia dedicata unicamente al riempimento di un database e all'utilizzo da parte di utenti generici, gli aspetti relativi all'usabilità delle interfacce sono stati curati, nell'ottica di ridurre al minimo gli errori di caricamento dei dati.

A tal proposito, il contenuto delle interfacce è stato realizzato prendendo spunto dal lavoro di tesi "Analisi di cartelle cliniche di pazienti geriatrici sottoposti ad intervento chirurgico", svolto dalla Dott.ssa Giulia Paternostro, in cui era stata fatta un'analisi dettagliata delle cartelle cliniche di Humanitas ed erano state selezionate le voci delle schede considerate fondamentali per gli scopi del progetto finale. Tuttavia in questa continuazione del lavoro gli aspetti grafici ed estetici e tutte le funzionalità dell'applicazione sono state nuovamente studiate e realizzate nel dettaglio.

Una volta disegnate le interfacce è stato necessario definire il flusso di tutte le operazioni che è possibile svolgere nella fruizione della *GUI* per ogni singolo caso d'uso. Utilizzando

dunque l'activity diagram di UML è stato possibile costruire tutti i percorsi, sia principali, sia alternativi, relativi alle attività ad alto livello che in sequenza possono essere compiute relativamente ad uno specifico *use case* [28].

La fase di modellizzazione delle specifiche prevede anche la costruzione degli oggetti entità, ovvero gli oggetti che rappresentano i dati gestiti dall'applicazione. Questo passaggio è stato volontariamente bypassato, poichè la procedura di individuazione e modellizzazione di questi dati è stata eseguita direttamente durante le operazioni di progettazione logica e fisica del database, tramite la costruzione delle entità e delle tabelle.

Nelle sezioni successive si descrive nel dettaglio la costruzione dello use case diagram e si riportano le interfacce e gli activity diagram relativi ai casi d'uso *Anagrafica*, *Inserimento nuovo paziente* e *Cerca paziente* e ad un esempio di scheda della cartella clinica.

Use case diagram Lo use case diagram realizzato per questa applicazione è rappresentato in figura 2.8.

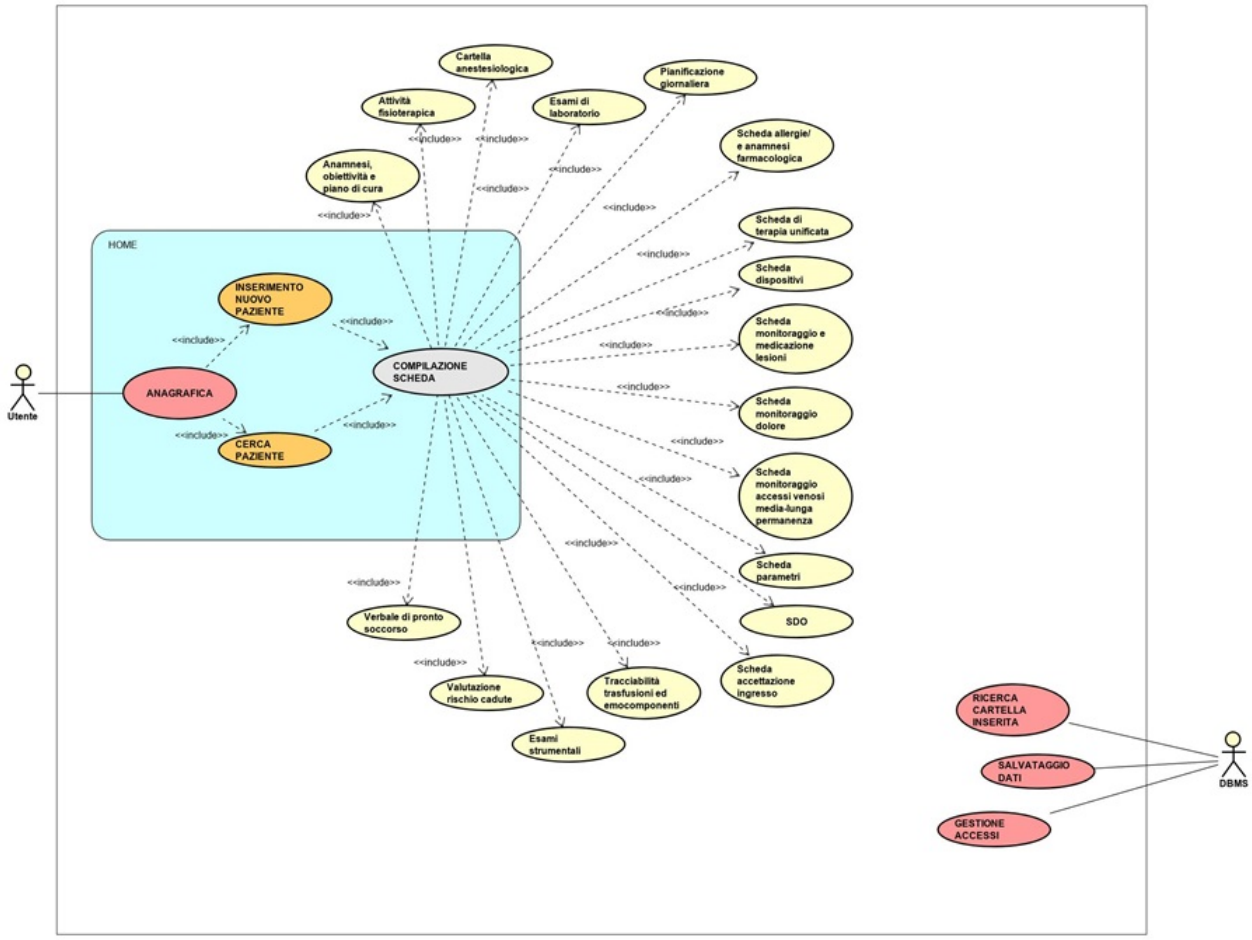


Figura 2.8: Use case diagram relativo alla *Clinical record GUI*

Gli *use case* racchiusi nella *Home* sono stati direttamente estrapolati dai processi. In particolare il sistema consente all'*Utente* di accedere alla pagina che viene chiamata *Anagrafica*, dalla quale è possibile, una volta effettuato l'accesso, procedere con le operazioni di inserimento o di ricerca di un paziente (casi d'uso *Inserimento nuovo paziente* e *Cerca paziente*). Una volta individuato il paziente, in entrambi i casi, le attività svolte riguardano la ricerca di una scheda e la sua compilazione (caso d'uso *Compilazione scheda*). A questo livello è stato necessario distinguere le varie schede della cartella clinica, in quanto, sebbene le operazioni previste siano le stesse, le interfacce utente saranno completamente diverse e questo impone la creazione di un caso d'uso per ogni scheda.

Come già spiegato nel dettaglio nel primo capitolo, in merito alla cartella clinica di Humanitas, le schede scelte e dunque i casi d'uso individuati sono i seguenti:

- Anamnesi, obiettività, piano di cura
- Attività fisioterapica
- Cartella anestesiologica ed intra-operatoria
- Esami di laboratorio
- Pianificazione giornaliera
- Scheda allergie/anamnesi farmacologica
- Scheda di terapia unificata
- Scheda dispositivi
- Scheda monitoraggio e medicazione lesioni
- Scheda monitoraggio accessi venosi media-lunga permanenza
- Scheda monitoraggio dolore
- Scheda parametri
- Scheda di dimissione ospedaliera (SDO)
- Scheda accettazione ingresso
- Tracciabilità trasfusioni ed emocomponenti
- Esami strumentali
- Valutazione rischio cadute
- Verbale di pronto soccorso

Viene considerato come utente anche il DBMS , in quanto interagirà con l'applicazione al fine di cercare e salvare dati e gestire gli accessi (casi d'uso rispettivamente *Ricerca cartella inserita*, *Salvataggio dati* e *Gestione accessi*).

Interfacce e Activity diagram Le interfacce sono state realizzate graficamente con lo strumento MATLAB App Designer, in modo da velocizzare le operazioni successive di costruzione e scrittura del codice.

In figura 2.9 è rappresentata l'interfaccia *Anagrafica*, ovvero la pagina nella quale si svolgono i casi d'uso racchiusi nella *Home* dello use case diagram.

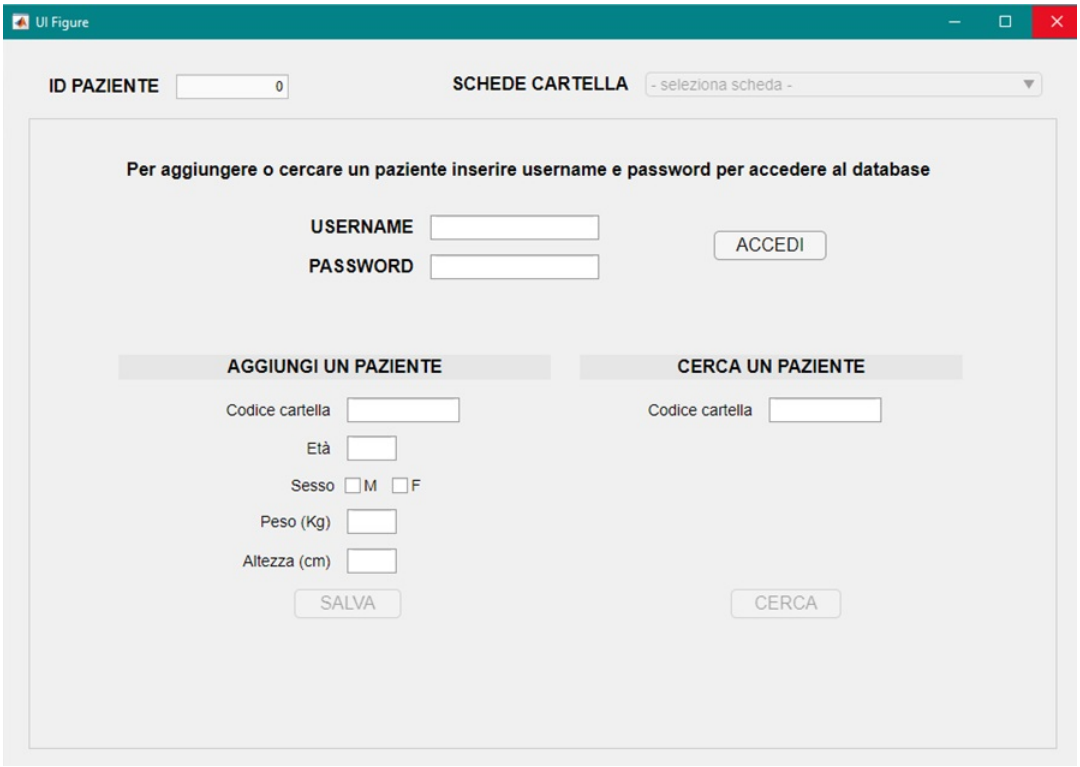


Figura 2.9: Interfaccia *Anagrafica*

La gestione dei possibili percorsi è stata progettata già direttamente a livello di interfaccia. Nello specifico, come si può apprezzare dalla figura 2.9, nessun tasto risulta cliccabile prima di avere eseguito le operazioni di autenticazione. Una volta effettuato l'accesso esistono due percorsi alternativi: aggiungere un paziente o cercarlo. Il flusso di questo percorso è descritto nell'activity diagram in figura 2.10, in cui si definiscono in sequenza tutte le possibili attività relative all'utilizzo dell'interfaccia *Anagrafica*.

Le altre due funzionalità, in particolare quelle che possono essere eseguite dopo avere effettuato l'accesso (figura 2.10), ovvero *Inserimento nuovo paziente* e *Cerca paziente*, sono modellizzate rispettivamente in figura 2.11 e 2.12.

Il percorso dell'inserimento di un nuovo paziente è caratterizzato da un flusso molto semplice, in cui l'*Utente*, una volta inseriti i dati dell'anagrafica del paziente, clicca il tasto *Salva*. A livello di software, leggermente più complesso è il caso della ricerca di un paziente già inserito, per cui il sistema oltre a cercare il codice della cartella, nel caso di codice assente, deve presentare una schermata di errore. Al termine di entrambi i percorsi, a meno che l'*Utente* non decida di chiudere l'applicazione, si passa al caso d'uso

della *Compilazione scheda*, come si può osservare nelle figure 2.11 e 2.12 .

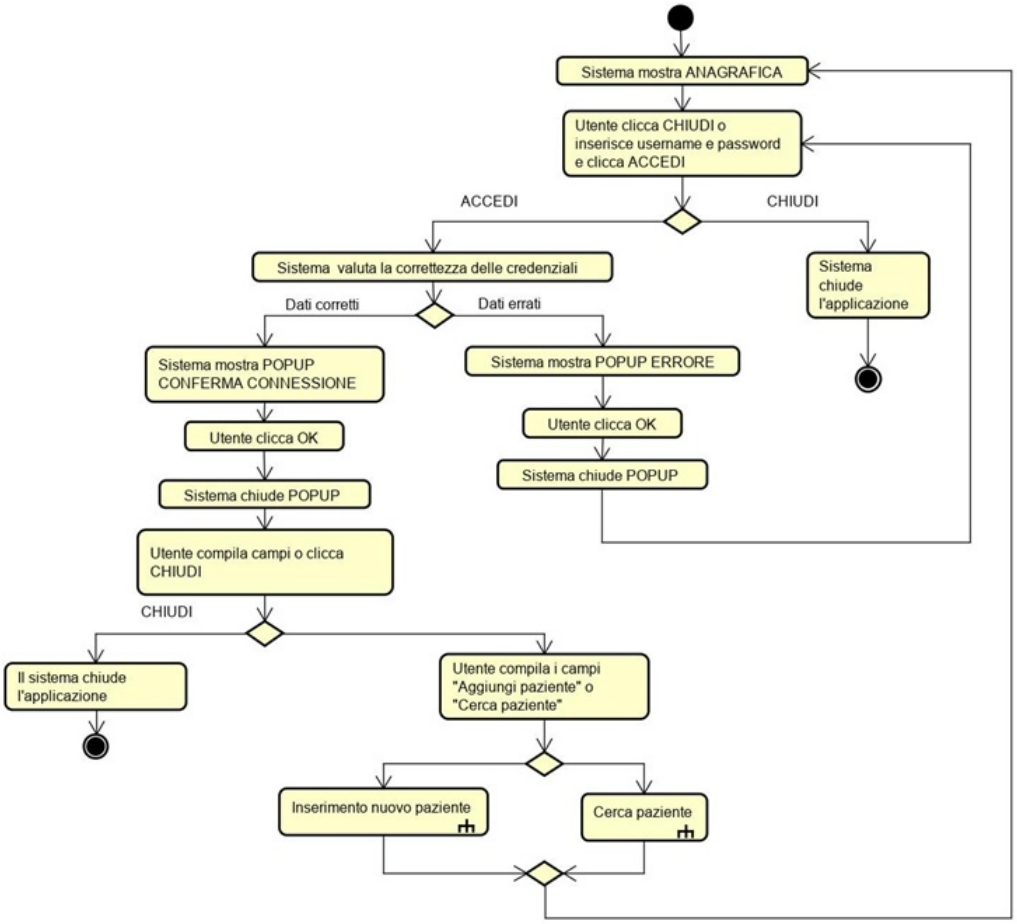


Figura 2.10: Activity diagram *Anagrafica*

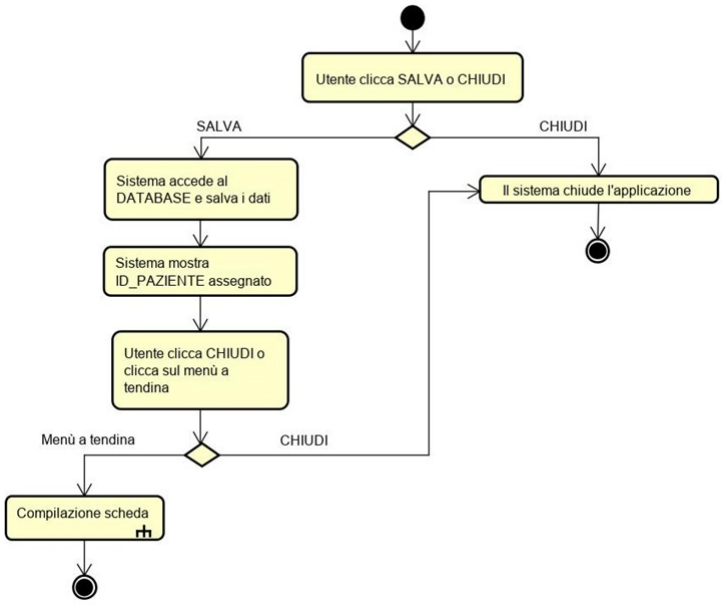


Figura 2.11: Activity Diagram *Inserimento nuovo paziente*

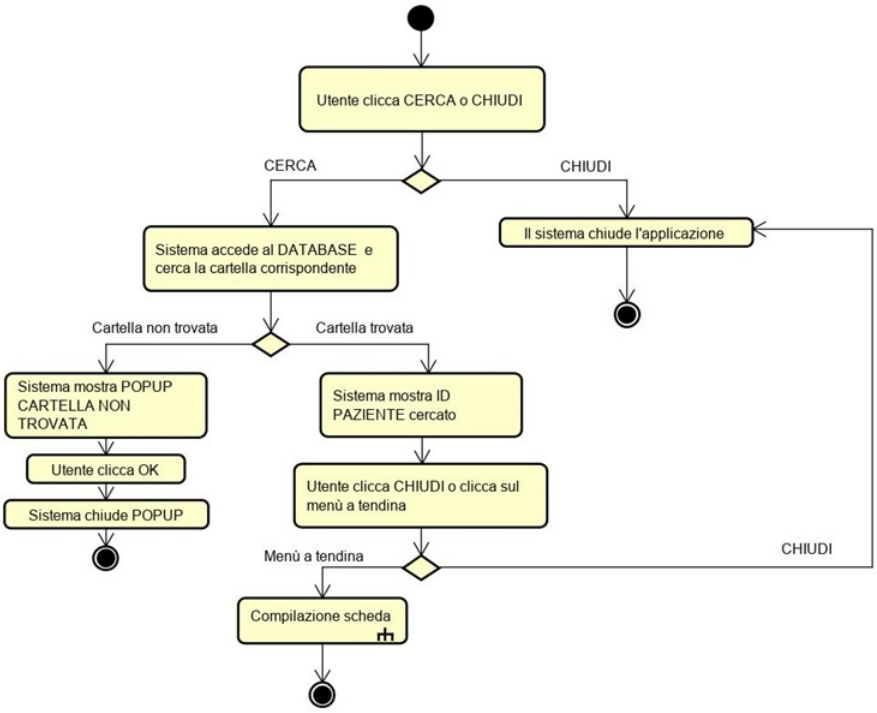


Figura 2.12: Activity diagram *Cerca paziente*

In ogni interfaccia utente dell'applicazione si è deciso di mantenere il menù a tendina in alto a destra che consente una facile e veloce navigabilità tra tutte le schede della cartella, garantendo un elevato grado di flessibilità (figura 2.13). Fondamentale risulta inoltre il campo in alto a sinistra relativo all'*ID_PAZIENTE* (figura 2.13), in modo da tenere sempre traccia del paziente a cui fanno riferimento i dati che si stanno inserendo.

L'obiettivo dell'utilizzo delle *GUI* è sempre quello di caricare nel database le informazioni delle schede della cartella clinica del paziente, dunque il passo seguente è quello della selezione di una scheda di interesse dal menù in alto a destra (caso d'uso *Compilazione scheda*) (figura 2.13) e della sua compilazione.

L'activity diagram relativo al caso d'uso *Compilazione scheda* è molto semplice dal punto di vista teorico, poichè riguarda esclusivamente l'apertura del menù a tendina e la selezione di una delle schede, ma non non viene riportato poichè, a causa dell'elevato numero dei casi d'uso richiamati, risulta di difficile lettura.

Al fine di mostrare un esempio di scheda, si sceglie il percorso relativo alla scheda *Anamnesi, obiettività e piano di cura* (caso d'uso *Anamnesi, obiettività e piano di cura*), la cui interfacce sono riportate in figura 2.14, 2.15 e 2.16.

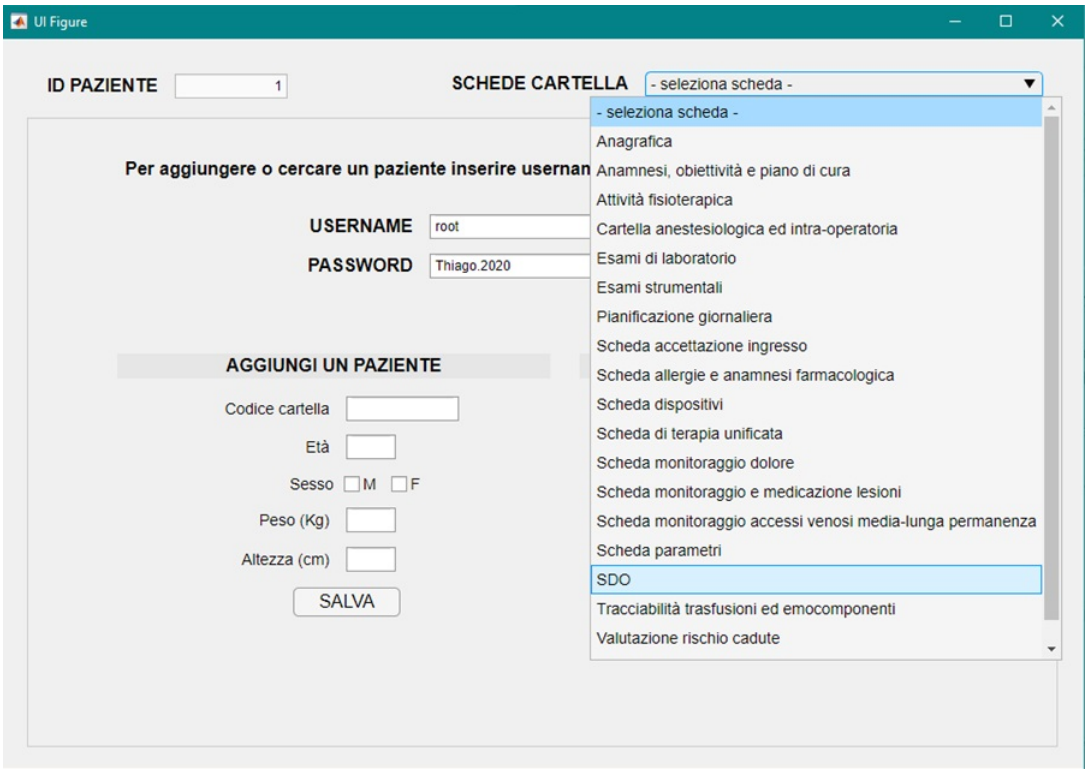


Figura 2.13: *Interfaccia Anagrafica* con menù a tendina

Figura 2.14: Scheda *Anamnesi, obiettività e piano di cura*: interfaccia *Anamnesi*

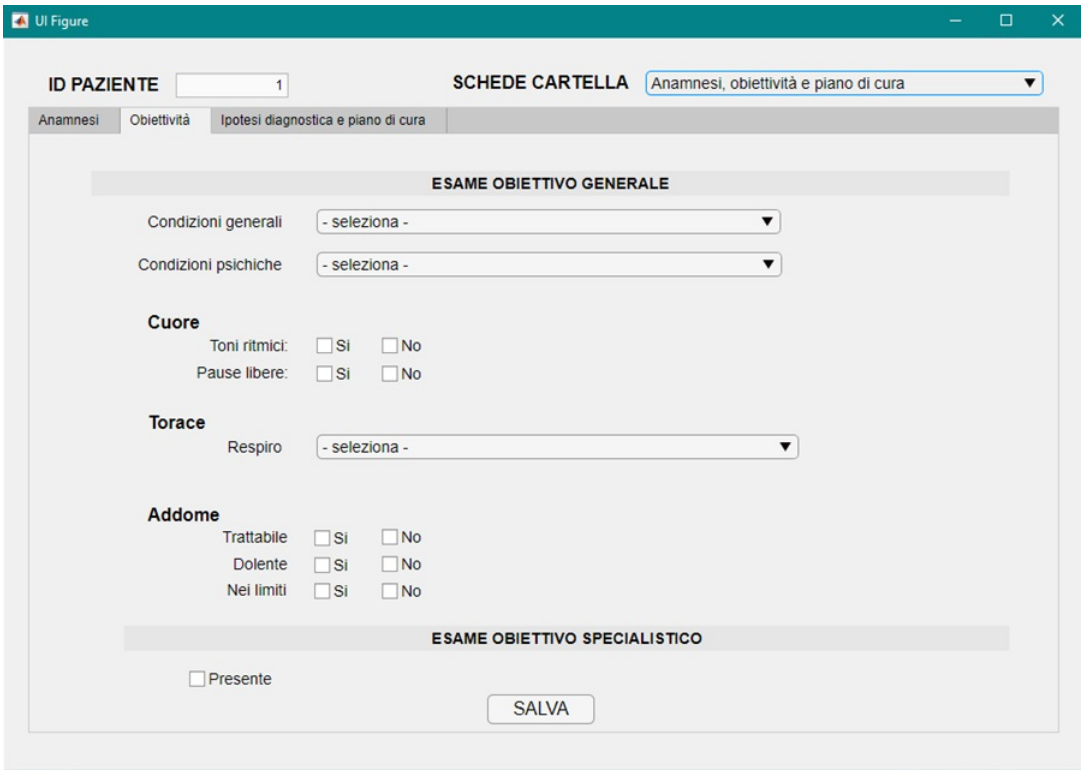


Figura 2.15: Scheda *Anamnesi, obiettività e piano di cura*: interfaccia *Obiettività*

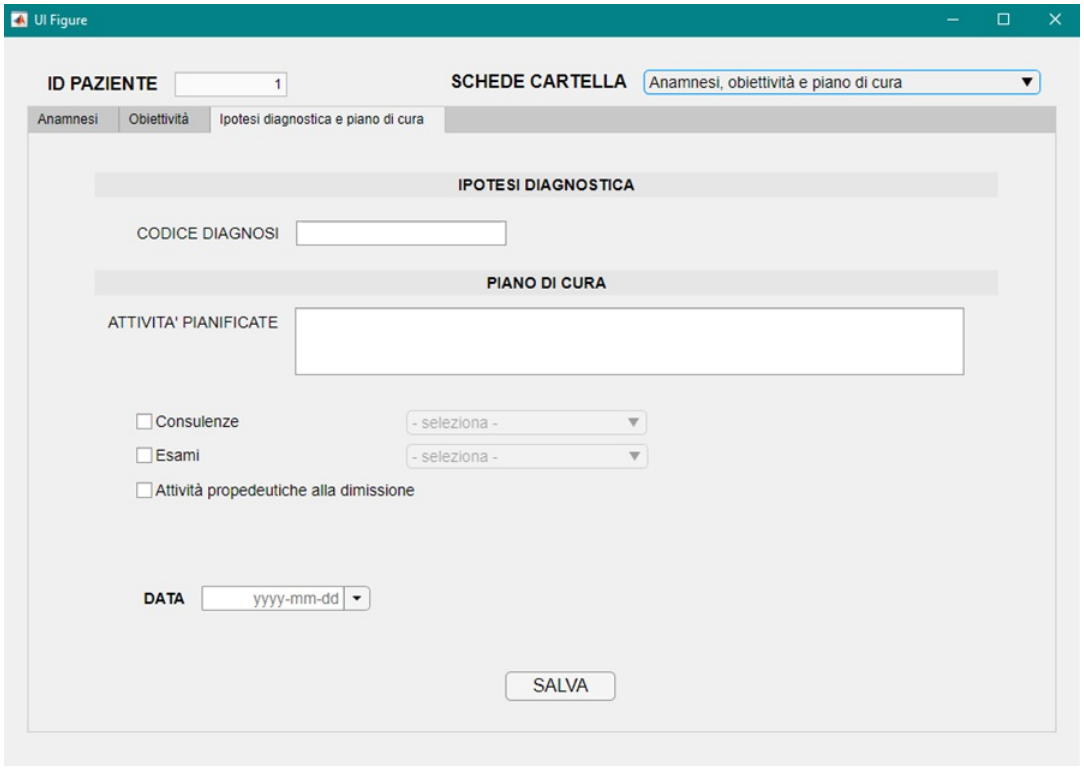


Figura 2.16: Scheda *Anamnesi, obiettività e piano di cura*: interfaccia *Piano di cura*

E' possibile navigare tra le pagine della scheda e, una volta compilate, salvare i dati inseriti, come descritto nel flusso riportato in figura 2.17.

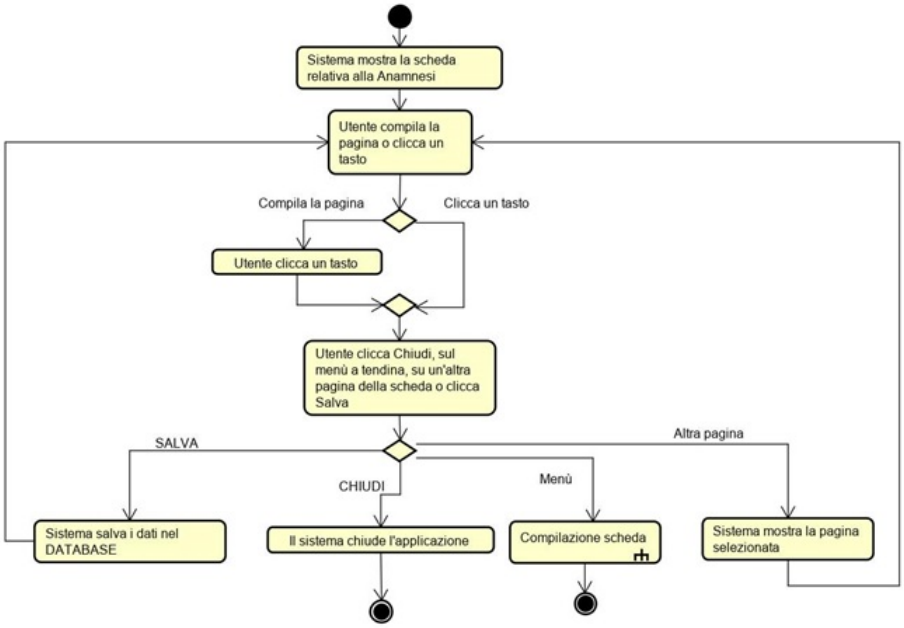


Figura 2.17: Activity diagram *Anamnesi, obiettività e piano di cura*

Tale diagramma mostra anche la flessibilità resa possibile dall'utilizzo di un menù fisso all'interno di tutte le interfacce, garantendo all'*Utente* una facile navigazione tra le schede.

In generale, per quanto riguarda gli activity diagram relativi agli altri casi d'uso, essi risultano sostanzialmente assimilabili a quello riportato in figura 2.17 per la scheda *Anamnesi, obiettività e piano di cura*, con l'unica variante relativa al nome della scheda.

Lo stesso non vale per le interfacce, in quanto ogni scheda è stata costruita in modo diverso, al fine di riprodurre e rispecchiare il più possibile il posizionamento delle stesse voci nella documentazione reale di Humanitas Gradenigo.

A tal proposito, per maggiori dettagli è possibile trovare le figure relative a tutte le interfacce dell'applicazione nell'Appendice A di questo documento.

2.3 Progettazione e costruzione del database

Tra le tante tipologie di base dati, si è scelto di adottare la struttura relazionale, perchè facile da gestire e particolarmente flessibile per le operazioni di modifica delle tabelle, soprattutto dal momento che la *GUI* è stata programmata unicamente per l'inserimento dei dati.

Come già preannunciato nel primo capitolo, quando si sviluppa una base dati è importante affrontare tre fasi di progettazione:

1. Progettazione concettuale
2. Progettazione logica
3. Progettazione fisica

Sebbene in questo documento la sezione dedicata allo sviluppo della base dati sia stata inserita dopo quella relativa alla *GUI*, in realtà tutti gli aspetti che riguardano la progettazione e la costruzione sono stati affrontati in unica visione, quella dello sviluppo di un sistema univoco. In particolare, lo schema concettuale del database è stato realizzato dopo l'identificazione delle funzionalità dell'applicazione. Una volta individuate le entità generali e le relazioni tra esse, al fine di disegnare nel dettaglio le interfacce e decidere il formato dei dati, ci si è dedicati alla progettazione logica e fisica.

2.3.1 Progettazione concettuale

Il modello concettuale di una base dati viene costruito mediante il diagramma E-R, poichè considerato il metodo più immediato per focalizzarsi subito sui dati ad alto livello e sulle relazioni di base. In questo lavoro, il diagramma non è stato costruito manualmente, ma si è fatto uso del programma RISE Editor. I simboli adottati da questo tool sono leggermente diversi da quelli riportati nel primo capitolo per il diagramma E-R tradizionale, ma risultano comunque facilmente comprensibili.

Nonostante sia di difficile lettura, a causa dell'elevata quantità di entità, in figura 2.19 si riporta lo schema finale.

A livello generale, si è scelto di inserire come attributi solo le chiavi primarie, in modo da rendere il modello più semplice da leggere e da usare. L'individuazione di tutti gli attributi verrà eseguita durante la fase di costruzione fisica del database.

Dalla rappresentazione fornita, si può subito notare come tutte le entità siano collegate tramite una relazione uno-a-molti ad una entità centrale, quella dell'*ANAGRAFICA*, in modo da rendere sempre chiaro a quale soggetto appartiene ogni singolo record.

Le schede della cartella clinica di Humanitas che sono state selezionate non presentano tutte quante la stessa struttura. Alcune di esse sono unicamente delle schede in cui si tiene traccia della singola prestazione, come la *Scheda accettazione ingresso*, mentre altre sono composte da diverse pagine e sezioni, come ad esempio la *Scheda di terapia unificata* [23].

Di seguito si riporta la suddivisione delle sezioni delle varie schede [23]:

- Anamnesi, obiettività, piano di cura
  - Anamnesi
  - Obiettività
  - Ipotesi diagnostica e piano di cura
- Attività fisioterapica

  - Mobilità-deambulazione
  - Indicazioni riabilitative
  - Scheda trattamento riabilitativo

- Esami di laboratorio
  - Esami ematochimici
  - Esami delle urine
  - Esami batteriologici
  - Emogasanalisi venosa
  - Emogasanalisi arteriosa

- Scheda allergie/anamnesi farmacologica
  - Allergie e intolleranze
  - Anamnesi farmacologica e terapie non convenzionali

- Scheda di terapia unificata
  - Terapia orale
  - Terapia sottocutanea
  - Terapia endovenosa
  - Terapia endovenosa in continuo

- Scheda dispositivi
  - Catetere venoso periferico
  - Catetere vescicale
  - Catetere peridurale
  - Sondino nasogastrico

Alcune di queste sezioni sono caratterizzate a loro volta da sotto-sezioni di monitoraggio e gestione, considerate fondamentali ai fini del progetto, poichè tramite esse è possibile ricostruire la sequenza temporale delle attività [23]. Si è deciso quindi di separare alcune schede in sezioni principali e sotto-sezioni [23]. Nell'ambito della progettazione concettuale le sezioni verranno definite d'ora in poi come entità principali e sotto-entità, poichè è necessario progettare un modello che sia indipendete dal tipo di database che si realizzerà.

Le entità uniche sono le seguenti [23]:

- Anamnesi
- Obiettività
- Ipotesi diagnostica e piano di cura
- Mobilità-deambulazione
- Indicazioni riabilitative
- Scheda trattamento riabilitativo
- Esami ematochimici
- Esami delle urine
- Esami batteriologici
- Emogasanalisi venosa
- Emogasanalisi arteriosa
- Allergie e intolleranze
- Anamnesi farmacologica e terapie non convenzionali
- Terapia orale
- Terapia sottocutanea
- Terapia endovenosa
- Terapia endovenosa in continuo
- Catetere vescicale
- Cartella anestesiologica ed intra-operatoria
- Pianificazione giornaliera
- Scheda monitoraggio e medicazione lesioni
- Scheda monitoraggio dolore
- Scheda accettazione ingresso
- Tracciabilità trasfusioni ed emocomponenti
- Esami strumentali

La divisione in entità e sotto-entità invece riguarda le seguenti schede [23]:

- Scheda monitoraggio accessi venosi media-lunga permanenza
  - Gestione catetere venoso centrale
  - Monitoraggio catetere venoso centrale
- Catetere venoso periferico
  - Gestione catetere venoso centrale
  - Monitoraggio catetere venoso centrale
- Catetere peridurale
  - Gestione catetere peridurale
  - Monitoraggio catetere peridurale
- Sondino nasogastrico
  - Gestione sondino nasogastrico
- Scheda parametri giornalieri
  - Parametri
  - HGT
- Scheda di dimissione ospedaliera (SDO)
  - Tabella diagnosi SDO
  - Tabella interventi SDO
- Valutazione rischio cadute
  - Tabella interventi rischio cadute
- Verbale di pronto soccorso
  - Interventi di pronto soccorso

Nel diagramma E-R le entità principali sono collegate tramite una relazione uno-a-molti all'entità *ANAGRAFICA*, mentre le sotto-entità sono a loro volta in relazione uno-a-molti anche con la propria entità principale.

Al fine di rendere più chiara l'organizzazione del modello concettuale, si riporta in figura 2.18 lo schema relativo alla *Scheda dispositivi*.

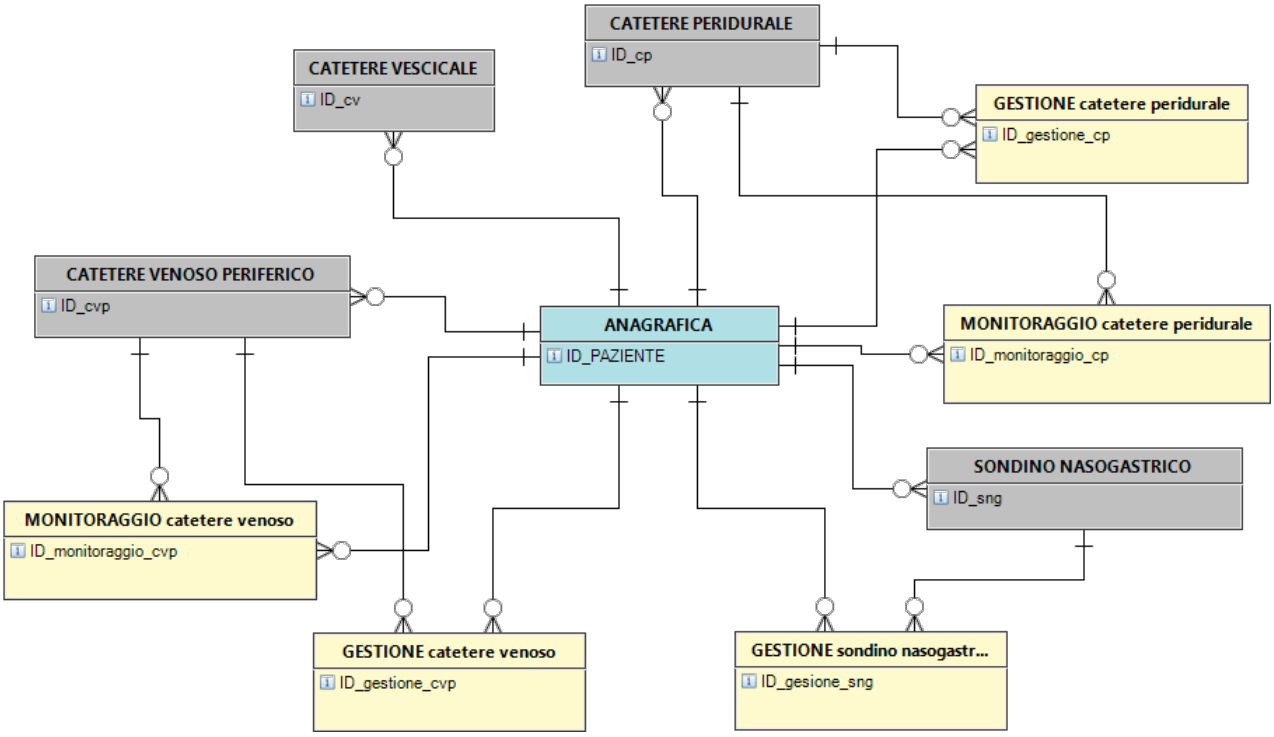


Figura 2.18: Modello concettuale per la *Scheda dispositivi*

La natura delle relazioni segue sostanzialmente lo stesso principio per tutte le altre schede.

Facendo riferimento a questo esempio si può analizzare il perchè del tipo di relazioni instaurate, considerando che ogni dispositivo può essere utilizzato su un unico paziente, ma uno stesso paziente può avere associati diversi dispositivi. Inoltre, ogni attività di gestione e monitoraggio, relativa a determinate indicazioni di data e orario, come si vedrà meglio in dettaglio durante l'analisi della modello logico e fisico, può essere riferita unicamente ad uno solo dei dispositivi memorizzati nell'entità, a sua volta associato ad un soggetto specifico.

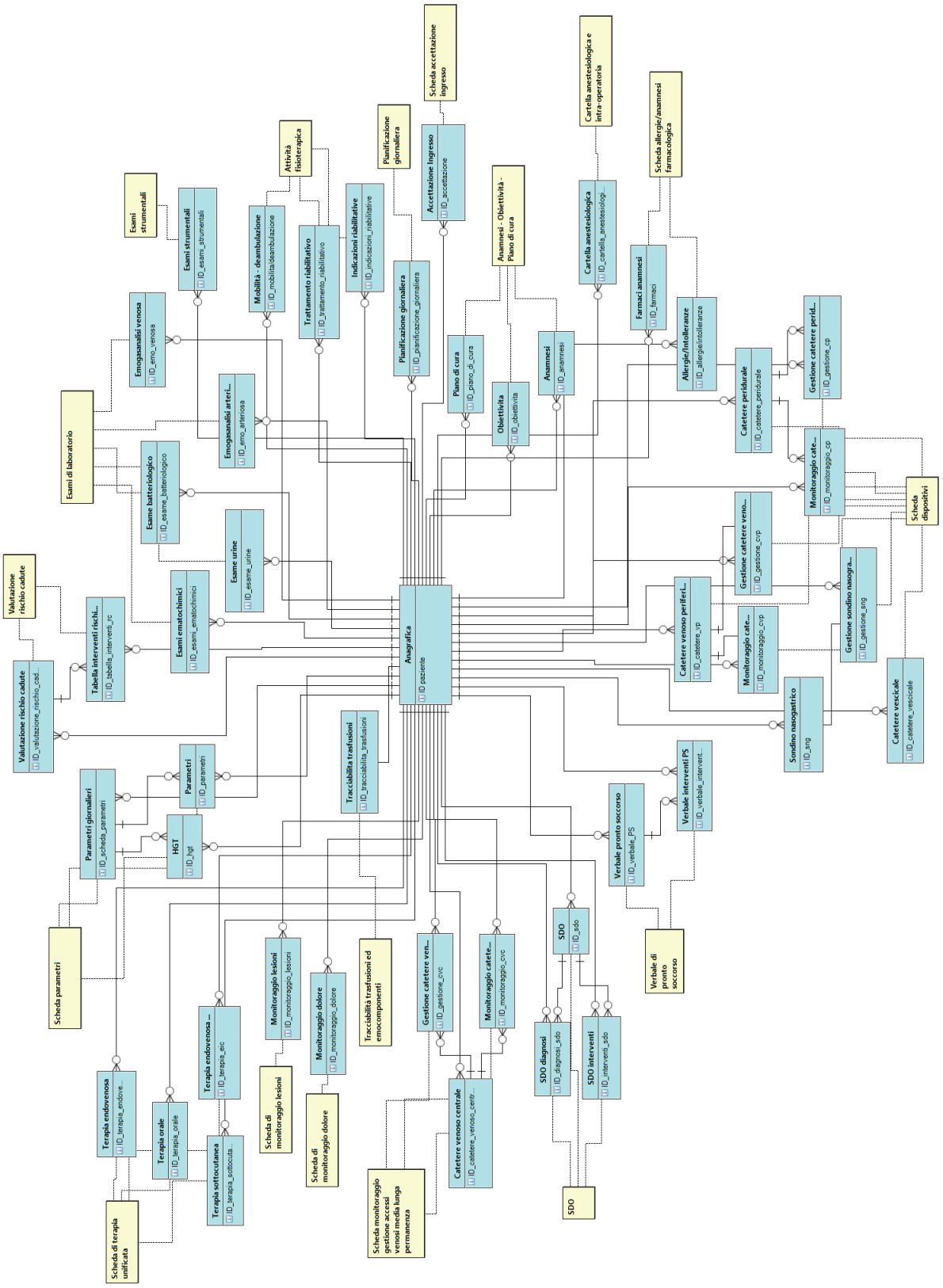


Figura 2.19: Modello concettuale del database complessivo

2.3.2 Progettazione logica e fisica

Lo schema logico che è stato realizzato in questo lavoro si può considerare come una trasposizione del modello concettuale all'interno dell'ambiete MySQL.

MySQL Workbench mette a disposizione un tool appositamente creato per costruire i diagrammi EER, al fine poi di ottenere automaticamente lo scheletro dei database. E' uno strumento particolarmente versatile che consente di effettuare operazioni sia di *forward engeneering* (dal modello EER al database) sia di *reverse engeneering* (dal database al modello EER).

Durante la fase di progettazione logica, il diagramma EER è stato disegnato a partire dal diagramma E-R sviluppato con RISE Editor. Poichè si è scelto di fare uso del database relazionale, da questa fase in avanti le entità principali e le sotto-entità verranno definite tabelle e sotto-tabelle. Sempre al fine di possedere uno strumento chiaro e facile da comprendere, nel nuovo diagramma, mostrato in figura 2.20, all'interno delle entità sono stati inseriti unicamente gli attributi chiave, ma stavolta insieme alle chiavi primarie sono state specificate anche tutte le chiavi esterne, fondamentali per instaurare le relazioni e le dipendenze tra le tabelle. Esattamente come il modello concettuale di figura 2.19, anche il modello relazionale qui presentato risulta di difficile lettura, a causa dell'elevato numero di entità.

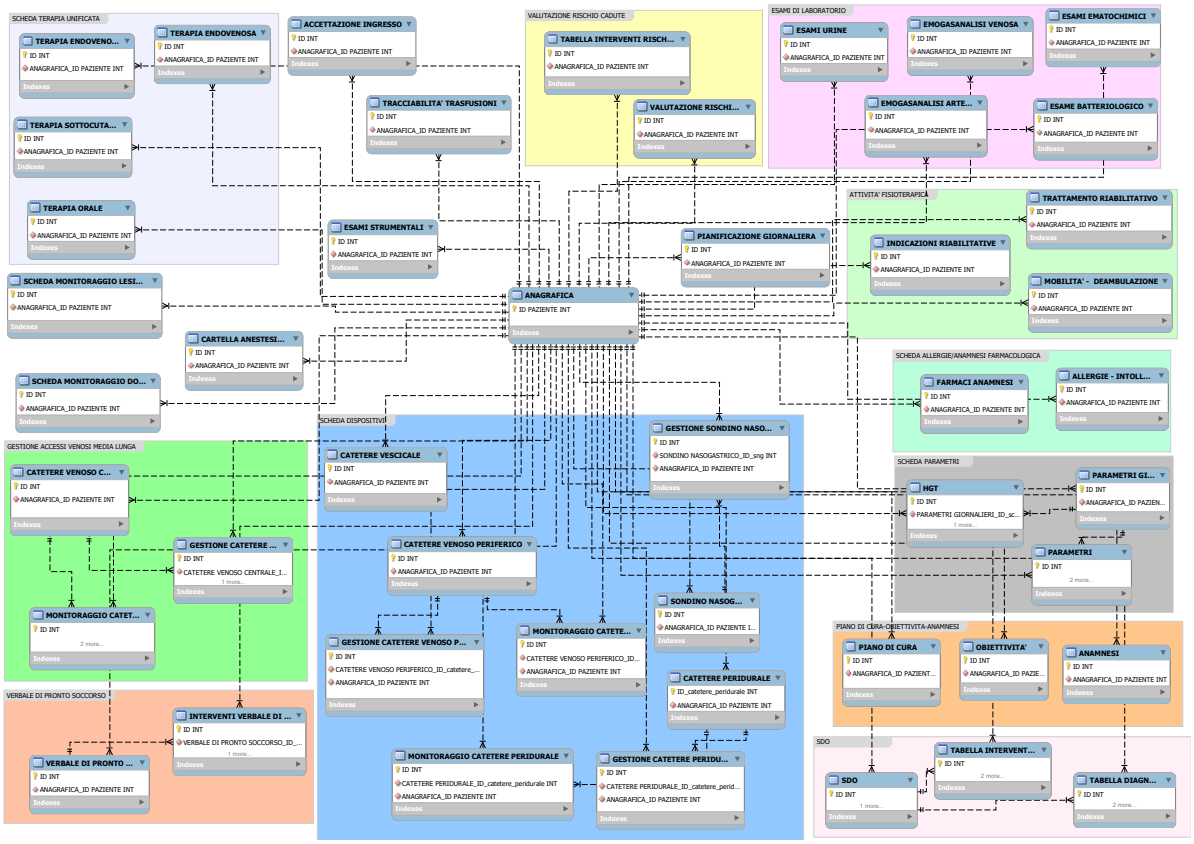


Figura 2.20: Modello relazionale del database complessivo

Riprendendo l'esempio della *Scheda dispositivi*, si riporta in figura 2.21 il corrispettivo modello relazionale.

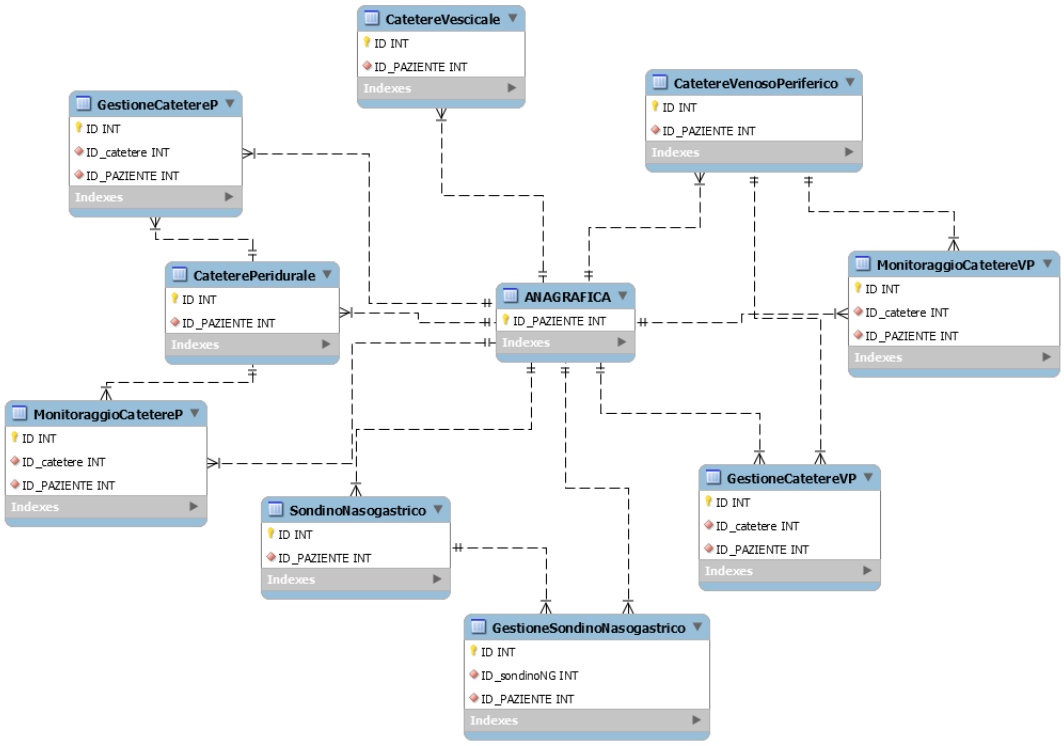


Figura 2.21: Modello relazionale per la *Scheda dispositivi*

Soprattutto dalle sotto-tabelle di gestione e monitoraggio, risulta chiaro l'utilizzo delle chiavi esterne, che consentono la relazione non solo con l'entità *ANAGRAFICA* ma in particolare con le proprie tabelle principali.

Tramite l'opzione di *forward engeneering* si è poi costruito automaticamente lo scheletro del database, costituito da tutte le tabelle principali e le sotto-tabelle studiate nella sezione precedente.

Si è giunti dunque all'ultima fase di progettazione del database, quella fisica. E' importante sottolineare come effettivamente questo step di sviluppo non sia stato seguito concentrandosi unicamente sulla base dati, ma contemporaneamente, dal punto di vista dei contenuti, sono state disegnate le interfacce della *Clinical records GUI*.

La cartella clinica è un documento che in generale racchiude dati di diverso tipo, in quanto contiene testi, immagini, numeri e altro [23]. In questa prima parte della tesi ci si è concentrati sui dati di tipo strutturato, ovvero dati codificati sotto forma di [23]:

- stringhe di testo
- numeri interi e decimali
- variabili booleane

Utilizzare dati strutturati è uno degli aspetti fondamentali da tenere in considerazione per l'applicazione successiva delle tecniche di process mining.

I testi clinici scritti in linguaggio naturale racchiudono in sè ovviamente informazioni importanti per la ricostruzione temporale e spaziale degli eventi, ma poichè nascono come dati non strutturati, prima dell'utilizzo, devono essere necessariamente elaborati. Maggiori approfondimenti in merito ai testi scritti in linguaggio naturale e alle tecniche di elaborazione si trovano nei successivi due capitoli della tesi, mentre in questo frangente ci si è occupati esclusivamente di tutti quei dati già direttamente inseribili nella base dati.

Lavorando contemporaneamnete con le interfacce, sono state individuate delle strategie per la gestione dei dati e del loro formato. In particolare, per tutti i dati numerici si è deciso di mantenere invariata la voce all'interno delle interfacce e dunque anche il formato del dato nelle tabelle [23] (figura 2.22).

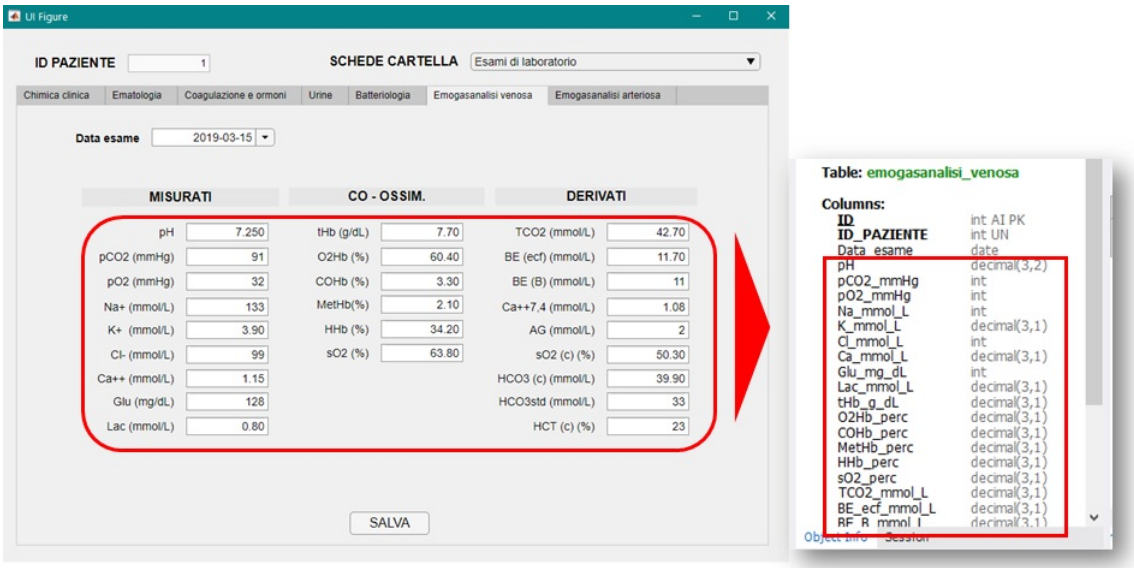


Figura 2.22: Esempio di dato numerico

Lo stesso vale per tutti i dati di tipo booleano, per i quali sono stati realizzati dei *checkbox*, in modo da indicare semplicemente se quella voce è presente o assente [23](figura 2.23).

Per le stringhe di testo non codificabili nè prevedibili è stata resa disponibile una voce compilabile (figura 2.24), mentre in tutti quei casi in cui la stringa rappresenta una sola tra un numero limitato di scelte possibili si è deciso di utilizzare un menù a tendina, contenente tutte le opzioni selezionabili [23] (figura 2.25). In quest'ultimo caso sono state aggiunte all'interno della base dati delle nuove tabelle contenenti le voci elencate nei menù a tendina [23]. In tal modo tramite l'utilizzo di un semplice indice identificativo è possibile associare all'interno delle tabelle il numero selezionato con la stringa corrispondente [23].

Nonostante i controlli sul formato dei dati di input vengano effettuati già all'interno della GUI, è chiaro che per ogni singolo attributo di ogni singola tabella del database sono

stati dichiarati non soltanto il formato dei dati, ma anche il numero massimo di cifre o di caratteri inseribili. A tal proposito un ulteriore controllo prima del salvataggio dei dati viene effettuato direttamente dal DBMS.

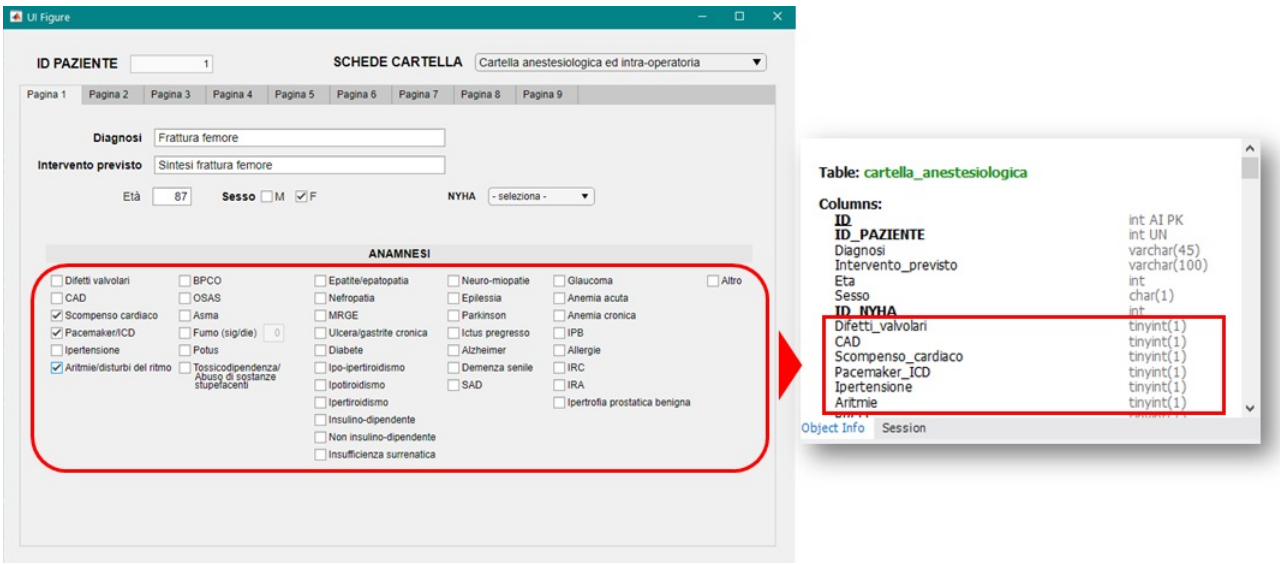


Figura 2.23: Esempio di dato di tipo booleano

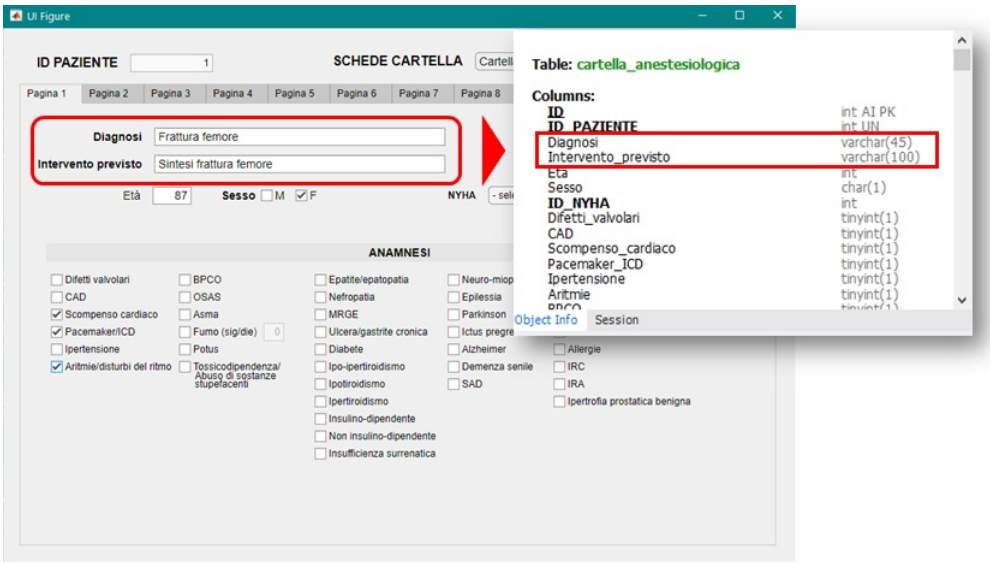


Figura 2.24: Esempio di stringa di testo

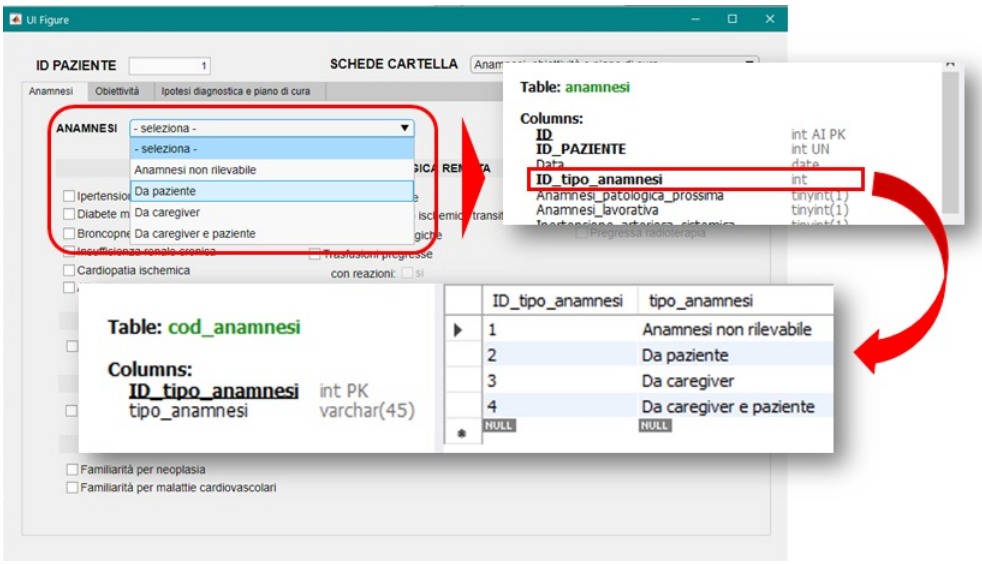


Figura 2.25: Esempio di stringa codificata mediante menù a tendina

2.4 Costruzione dell'applicazione

Durante la fase di progettazione dell'applicazione, in parallelo alla realizzazione delle interfacce, è stata eseguita la costruzione fisica del database, consentendo così la definizione di una strategia per la gestione del formato dei dati.

In seguito all'elaborazione delle interfacce grafiche sono stati costruiti tutti i possibili percorsi funzionali dell'applicazione tramite gli activity diagram. La modellizzazione del flusso delle attività relative alle funzionalità è stata di fondamentale importanza per la scrittura successiva del codice.

Matlab App Designer, oltre a rendere disponibile un ambiente *Design View* per la costruzione delle interfacce, componente per componente, possiede una sezione *Code View* per la programmazione del comportamento della GUI, ovviamente in linguaggio Matlab.

Dopo avere dunque definito il codice di base per il comportamento della *Clinical records GUI*, è stato creato un collegamento con il database tramite il toolbox Database Explorer, e sono state scritte direttamente in ambiente Matlab, all'interno delle specifiche sezioni del codice, tutte le *query* per l'interrogazione della base dati e per il caricamento nelle tabelle dei dati inseriti tramite la *GUI*.

Un esempio di funzionamento del sistema è riportato in figura 2.26, in cui, una volta inseriti i dati richiesti in merito all'anagrafica del paziente, cliccando il tasto *Salva*, il record è stato automaticamente inserito nella tabella *anagrafica* del database e l'*ID_PAZIENTE* assegnato è stato stampato a video nell'apposito campo.

Poichè l'applicazione è stata pensata nell'ottica di potere essere utilizzata anche da altri utenti, oltre che dalla sottoscritta, per la continuazione del progetto, si è deciso di sfruttare le opzioni di sharing messe a disposizione da Matlab e di condividerla come *Standalone Desktop App*. In questo modo, chiunque voglia utilizzare la app non la

eseguirà dall'ambiente App Designer e non avrà bisogno di saper usare Matlab, basterà semplicemente scaricare il compilatore e il Run Time di Matlab ed aprire l'applicazione direttamente dal desktop del proprio PC.

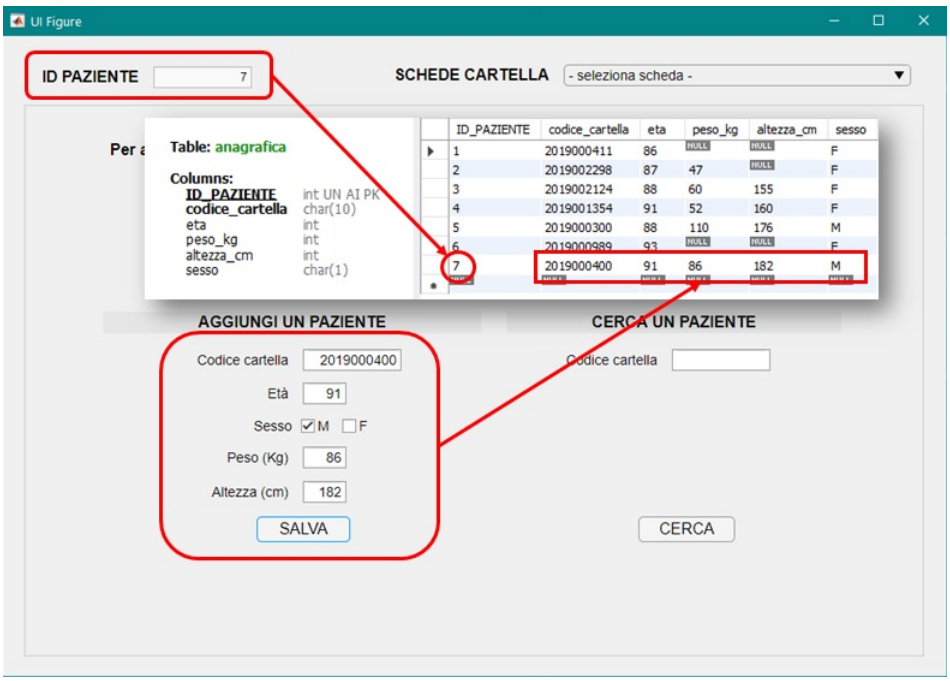


Figura 2.26: Esempio di inserimento dei dati dalla *GUI* al database

2.5 Testing

Per valutare l'efficienza delle singole funzionalità dell'applicazione e successivamente anche del sistema complessivo caratterizzato dalla *Clinical records GUI* e dal database, grazie anche alla possibilità di aver potuto usufruire di sei cartelle cliniche rese disponibili dall'ospedale Humanitas Gradenigo di Torino, sono state eseguite delle operazioni di testing.

In termini teorici, la fase di testing è fondamentale per la valutazione del sistema software realizzato e prevede due diverse tappe [30]:

- *Verifica* per valutare se il sistema risponde in maniera corretta alle specifiche che ci si era posti all'inizio del progetto;
- *Validazione* per testare se il sistema soddisfa le aspettative e le necessità dell'utente.

Lo scopo della verifica è quello di individuare gli errori del sistema e correggerli, iniziando con delle operazioni di *Unit testing*, dedicate a testare le singole routine del codice e di *Integration testing*, tramite le quali si valuta l'integrazione tra le varie funzionalità [30]. L'operazione di verifica finale riguarda il *System testing*, eseguito per mettere alla prova il sistema complessivo [30].

Nonostante la fase di verifica sia piuttosto lunga da progettare e da eseguire, la validazione è più complessa dal punto di vista dell'organizzazione, poichè deve essere condotta coinvolgendo in modo attivo dei potenziali utenti [30].

Di norma, chi sviluppa il sistema esegue anche le prime due operazioni di verifica, costruendo tutte le prove relative ai vari percorsi, sia principali sia alternativi, contemplati negli activity diagram, e componendo una checklist, tramite la quale tenere traccia dei risultati delle prove, degli strumenti e delle modalità in cui esse sono state condotte [30].

E' ovvio che per un sistema semplice come quello elaborato in questo lavoro e per le finalità e la categoria di utenti per cui è stato sviluppato, ai fini di analizzarne il funzionamento, non è stata progettata una fase di testing strutturata come viene effettuata solitamente, ma ci si è semplicementi dedicati a condurre delle prove relative ai percorsi degli activity diagram.

Per ogni possibile percorso ideato nei diagrammi deve essere eseguita un prova, mirata a verificare che tutto funzioni per come ci si aspetterebbe.

Se ad esempio, facendo riferimento al diagramma riportato in figura 2.10, si volesse valutare il percorso relativo all'inserimento delle credenziali di accesso errate, la sequenza delle attività da verificare sarebbe la seguente:

1. Deve essere visualizzata l'interfaccia *Anagrafica*;
2. L'utente deve inserire username e/o password errati e cliccare *Accedi*;
3. Il sistema non deve considerare i dati inseriti come corretti e deve mostrare la popup *ERRORE*;
4. L'utente deve cliccare *Ok*;
5. Il sistema deve chiudere la popup *ERRORE*.

Il comportamento corretto della *GUI* rispetto all'esecuzione di queste azioni è riportato nelle figure da 2.27 a 2.31.

Questa prova riportata a titolo esemplificatico è molto semplice ed è formata da poche attività, ma consente di mostrare in modo chiaro le modalità di creazione ed esecuzione della verifica, in particolare del *System testing*.

Nonostante questo processo di verifica non sia stato elaborato in maniera particolarmente strutturata, tuttavia è stato seguito il flusso delle attività previste, poichè si è partiti con l'obiettivo di individuare tutti i possibili mal funzionamenti del codice per ogni singola funzione della *GUI*. Ogni volta che è stato programmato il codice per la gestione anche solo di un unico tasto, ne è stato verificato il funzionamento. Completato il programma relativo ad ogni scheda, si è proceduto il lavoro testando l'integrazione di tutte le sue funzionalità, fino ad approdare alla valutazione del sistema complessivo, effettuata con l'esecuzione di prove simili a quella riportata sopra.

Figura 2.27: Prova *Inserimento credenziali errate*: step 1

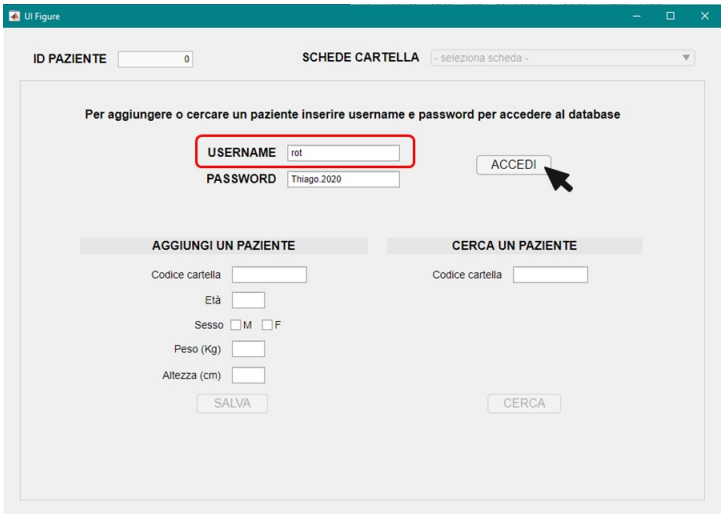


Figura 2.28: Prova *Inserimento credenziali errate*: step 2

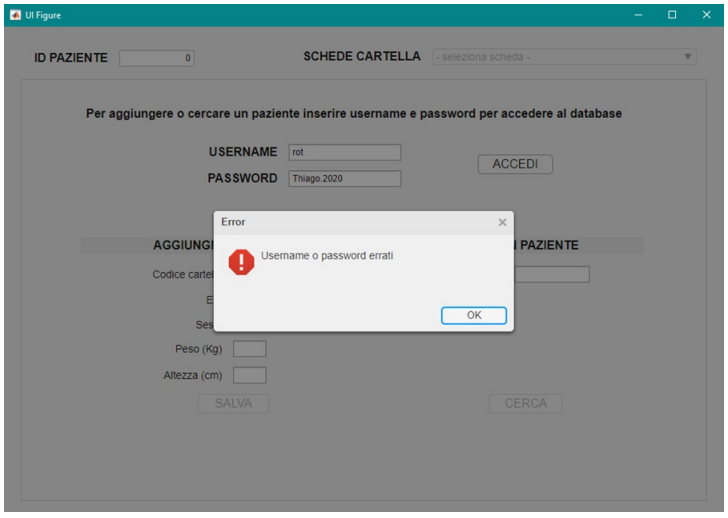


Figura 2.29: Prova *Inserimento credenziali errate*: step 3

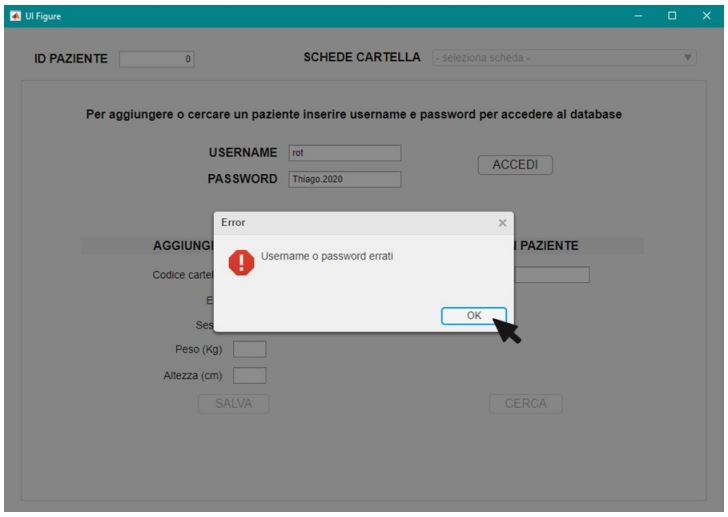


Figura 2.30: Prova *Inserimento credenziali errate*: step 4

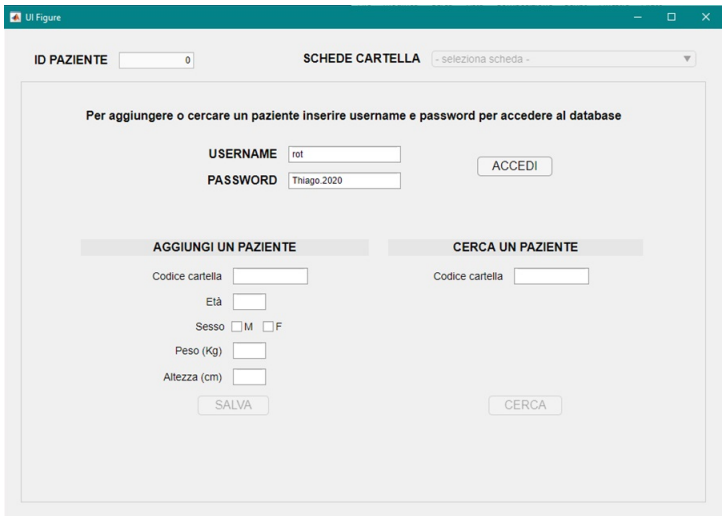


Figura 2.31: Prova *Inserimento credenziali errate*: step 5

Il testing dunque non è stato eseguito alla fine della procedura di costruzione, come si potrebbe erroneamente pensare, ma ogni volta che è stato predisposto un percorso funzionale, sono state condotte delle prove di verifica, testando anche le operazioni di interfacciamento con la base dati.

Per effettuare le operazioni di *Unit testing* e di *Integration testing*, al fine di controllare unicamente che le singole routine funzionassero e che si integrassero correttamente, sono stati utilizzati dei dati plausibili in termini di formato e di ordini di grandezza, soprattutto per i valori numerici, ma non veritieri.

Questa primo livello di valutazione è stato portato avanti per verificare che effettivamente l'applicazione funzionasse e una volta appurato ciò tutti i dati inseriti sono stati cancellati dal database.

Successivamente, grazie alle sei cartelle cliniche che sono pervenute, la fase di *System testing* è stata condotta su dati reali, consentendo allora di apportare modifiche e variazioni necessarie in riferimento proprio all'interfacciamento con il database. Si è dovuti tornare spesso alla fase di costruzione della base dati, per correggere gli ordini di grandezza e le cifre di alcuni attributi, soprattutto quelli relativi ai valori degli esami di laboratorio.

La procedura di testing dell'intero sistema è stata distinta in due momenti. Tre delle cartelle cliniche fornite sono state caricate eseguendo l'applicazione dall'ambiente Matlab App Designer, mentre i restanti documenti sono stati utilizzati direttamente sulla versione finale della *Clinical records GUI* da desktop, in modo da accertarsi che il funzionamento non fosse in qualche modo compromesso.

L'obiettivo di questa prima parte del lavoro era quello di costruire un database per l'inserimento di dati ed informazioni estraibili dalle cartelle cliniche, con la prospettiva di applicare su di essi le tecniche di process mining per la modellizzazione di un PDTA per pazienti fragili chirurgici. Per rendere più funzionali, veloci e sicure le operazioni di caricamento dei dati, è stata realizzata un'applicazione dedicata.

Fino a questo momento ci si è concentrati sui dati strutturati e codificati, direttamente inseribili nella base dati ed utilizzabili per l'applicazione degli algortmi. I testi clinici contenuti nelle cartelle, come i referti e le consulenze specialistiche, posso costituire però un'ulteriore ed importante fonte di informazioni, che se presa in considerazione potrebbe rendere ancora più ricca la documentazione raccolta. Per tale motivo, la seconda parte del lavoro di tesi, che si concentrerà nei successivi due capitoli, ha riguardato lo studio delle tecniche di Natural Language Processing per l'elaborazione dei testi scritti in linguaggio naturale e l'analisi delle opportunità che questa disciplina potrebbe offrire per l'estrazione di informazioni da testi clinici scritti in lingua italiana.

Capitolo 3

Natural Language Processing

L'aspirazione di rendere i computer capaci di interagire con l'essere umano e di comprendere il suo linguaggio è da sempre uno dei *main focus* dell'informatica e della tecnologia in generale. Sebbene sembri un'idea utopica, l'innovazione tecnologica che ha interessato il mondo già a partire dagli anni 50 del secolo scorso ci ha consentito oggi di potere usufruire nella nostra quotidianità di assistenti virtuali e case "intelligenti". Questi ultimi sono solo alcuni dei tanti esempi che si potrebbero portare in relazione alle tecniche e agli algoritmi in grado di elaborare il linguaggio umano in tutte le sue forme e che fanno capo ad una disciplina ormai consolidata da anni, il Natural Language Processing (NLP).

Il Natural Language Processing è un insieme di tecniche computazionali provenienti dall'ambito della linguistica, della computer science e dell'intelligenza artificiale utilizzate per analizzare e rappresentare in modo automatico il linguaggio umano, al fine di rendere i computer capaci di relazionarsi con l'uomo e con i suoi strumenti [31] [32].

Le origini del Natural Language Processing risalgono alla prima metà del novecento, grazie agli studi del matematico inglese Alan Touring [31]. Le prime applicazioni riguardarono la traduzione automatica dei testi, perchè di grande importanza politica già ai tempi della seconda guerra mondiale [31]. Nel periodo tra il 1950 e il 1990 i sistemi di NLP erano basati su complessi set di regole scritte a mano e il compito del computer era quello di applicare tali regole sui testi al fine di comprenderli ed elaborarli in base al tipo di applicazione [31].

A partire dal 1980 ci fu una vera rivoluzione dovuta all'introduzione degli algoritmi di machine learning, che, insieme alla grande disponibilità di dati provenienti dal web, dagli anni 2000 fino ad oggi hanno comportato un aumento esponenziale di applicazioni e di campi di applicabilità e un grande miglioramento delle prestazioni [31].

Grazie agli studi condotti in quest'arco temporale, si è pervenuti oggi all'individuazione di due specifiche categorie di NLP [33] [34]:

- *Natural Language Generation* (NLG): a partire da informazioni di tipo strutturato si compongono dei testi scritti in linguaggio umano;

- *Natural Language Understanding* (NLU): testi scritti in linguaggio umano vengono manipolati ed elaborati al fine di ottenere dei dati strutturati che possono essere memorizzati nei database e sfruttati dai software.

Per molte applicazioni è necessario attingere a tecniche appartenenti ad entrambe le categorie, in modo da affiancare all'analisi dei testi anche la generazione di nuovi documenti, realizzati a partire dalle informazioni che si è riusciti ad estrarre.

Proprio l'*Information Extraction*, ovvero l'attività relativa all'estrazione di informazioni a partire da dati non strutturati, è il principale motivo dell'interesse mostrato in questo lavoro verso il NLP.

Per la realizzazione di un Percorso Diagnostico Terapeutico Assistenziale le cartelle cliniche costituiscono la fonte primaria di informazioni, ma i dati contenuti al loro interno hanno diversa natura. La gestione dei dati strutturati è già stata trattata nel capitolo precedente, in cui sono stati messi da parte i documenti caratterizzati da testi scritti in linguaggio naturale, come i referti degli esami e delle visite specialistiche o i diari clinici, nonostante essi potessero effettivamente custodire ulteriori dati utili e necessari per la ricostruzione degli eventi. Si tratta in questo caso di dati non strutturati, non direttamente inseribili in un database, ma che dunque necessitano di diverse operazioni di elaborazione per consentirci in modo automatico di individuare le entità di interesse. Le attività svolte in merito saranno descritte con dovizia di particolari nel capitolo 4, mentre il capitolo che segue ha lo scopo di illustrare i punti nevralgici della disciplina e come essi si possono applicare in campo clinico.

3.1 Principali applicazioni

Le applicazioni di una disciplina come il NLP sono innumerevoli ed è impossibile elencarle tutte, tuttavia è possibile riportare le più utilizzate nel mondo di oggi [35]:

- *Sentiment Analysis*
- *Chatbot & Virtual Assistant*
- *Text Classification*
- *Text Extraction*
- *Machine Translation*
- *Text Summarization*
- *Auto-Correct*
- *Speech Recognition*

La branca del Natural Language Understanding costituisce la base fondamentale per tutti gli ambiti in cui si sfrutta l'analisi del linguaggio naturale. Ognuna delle applicazioni presenti nell'elenco sopra necessita di una fase di compresione e analisi del testo per potere produrre un risultato accettabile e pertinente.

A tal proposito un qualsiasi sistema di NLP generalmente effettua una serie di operazioni preliminari di analisi grammaticale e sintattica, come verrà spiegato in dettaglio nel paragrafo 3.2, mentre di seguito si riporta una descrizione più approfondita delle applicazioni citate sopra.

Sentiment Analysis Tramite il Sentiment Analysis è possibile estrarre dai testi le emozioni, le opinioni e la loro polarità (positiva o negativa) [35]. E' un ambito di grande interesse per il mondo del marketing e dei social media, poichè consente l'individuazione delle reazioni dei consumatori in merito ad uno specifico prodotto o ad una campagna pubblicitaria e il monitoraggio dei commenti e delle discussioni nei social network [35]. Soprattutto all'interno di un panorama aziendale, la percezione del gradimento dei clienti è fondamentale per la rivalutazione del prodotto o delle modalità plubblicitarie adottate.

Chatbot & Virtual Assistant Questo tipo di applicazione è anche conosciuta come *Question Answering* [35]. Le Chatbot ma soprattutto gli assistenti virtuali, che stanno spopolando negli ultimi tempi, come Siri, Alexa e Google Assistant, sono dei sistemi altamente complessi ed intelligenti, in quanto sfruttano sia le tecniche del NLU per comprendere il linguaggio umano e dunque le domande che gli vengono poste sia quelle del NLG per produrre delle risposte e consentire la possibilità di un dialogo [35].

Vengono definite "macchine intelligenti" proprio perchè, tramite l'interazione continua con l'uomo, sono in grado di migliorare ed apprendere sempre di più [35].

Text Classification Classificare i testi in modo automatico è un'operazione molto utile, soprattutto per le piattaforme di web marketing, ma non solo, infatti una classica funzione di Text Classification è il filtro spam della posta elettronica [36]. Effettivamente è una abilità particolarmente versatile, in quanto è possibile fornire al sistema delle proprie categorie e dei propri criteri secondo i quali effettuare la classificazione [35].

Questo è un tipo di task che può essere usato in combinazione con altri, come ad esempio il Sentiment Analysis ed è in grado di comprendere, processare e classificare un testo non strutturato scritto in linguaggio umano [35].

Text Extraction L'ambito della Text Extraction è meglio conosciuto come *Information Extraction* o *Named Entity Recognition* ed è una delle prime applicazioni più complesse che sono state sviluppate. Si tratta della capacità del sistema di estrarre da un testo parole chiave afferenti a specifiche categorie predefinite, come luoghi, nomi di persona e

compagnie [35]. Tramite l'apprendimento è possibile individuare altre classi di parole, come ad esempio i farmaci o le patologie riscontrabili nei testi clinici.

Le informazioni estratte da un testo possono servire per essere memorizzate all'interno di un database, classico esempio di estrazione di dati strutturati a partire dai testi, oppure possono essere usate per interpretare il contenuto di un testo e dunque agevolare l'evoluzione di altre applicazioni, come la Text Summarization o la Text Classification [35].

Machine Translation La traduzione automatica è una delle applicazioni più utilizzate già dalla seconda metà del novecento ed è soggetta ad un continuo miglioramento. Basti pensare a Google Translate, che negli ultimi anni ha subito un grande perfezionamento [35]. La maggior parte degli algoritmi di Machine Translation si basa sulle reti neurali [12]. Proprio grazie alle caratteristiche di apprendimento di questi modelli computazionali, il grande aumento della disponibiità dei dati online ha reso possibile un livello di traduzione molto più approfondito e preciso [35]. Il problema nevralgico, che ha interessato i sistemi di traduzione automatica sin dalle origini e che concerne la qualità delle traduzioni prodotte, non riguarda però esclusivamente il numero di parole e dati che il sistema apprende, ma anche e soprattutto la sua capacità di capire il significato delle frasi e dunque di effettuare l'analisi morfologica, sintattica e semantica [36].

Text Summarization La capacità di sintetizzare i testi in modo automatico è una delle peculiarità più versatili del NLP. Il processo di sintesi è fondamentale per le operazioni di revisione e valutazione, soprattutto in ambito scientifico, ma è importante anche per eventuali operazioni di elaborazione successive, come il Sentiment Analysis [36].

Esistono due diversi modi in cui si utilizza la capacità del NLP di riassumere i testi [35]:

- *Extraction-based summarization*: tramite l'impiego dei processi di Information Extraction, i testi vengono sintetizzati estraendo le parole chiave;
- *Abstraction-based summarization*: parafrasando il testo originale si creano delle nuove frasi riassuntive.

Auto-Correct Il motore di ricerca di Google è un classico esempio di sistema di autocorrezione e autocompletamento delle frasi. In realtà anche tutti i software che implementano i controlli di ortografia e grammatica, come Word, o che effettuano operazioni di autocompletamento delle parole nelle chat sono degli esempi tipici di questa applicazione e proprio perchè sono degli strumenti che vengono usati quotidianamente se ne può chiaramente comprendere l'importanza e la versatilità.

[12] Le reti neurali artificiali sono dei modelli computazionali, la cui architettura è ispirata alla rete neuronale biologica tipica dell'essere umano. Risorsa online: https://it.wikipedia.org/wiki/Rete_neurale_artificiale (Ultimo accesso: 1 febbraio 2021).

Speech Recognition Lo Speech Recognition è una delle principali caratteristiche degli assistenti virtuali come Siri, Alexa e Google Assistant, poichè riguarda la capacità di comprendere il linguaggio umano e trasformarlo in linguaggio informatico, quindi leggibile dalle macchine [35]. Il *Text-to-Speech* è un'altra delle peculiarità di questi sistemi e procede proprio nel senso inverso rispetto allo Speech Recognition, poichè a partire da testi scritti in linguaggio informatico produce un testo in linguaggio naturale, che nel caso degli assistenti virtuali ad esempio può essere riprodotto sottoforma di messaggio vocale.

3.2 I task del NLP e la classica pipeline

Ogni attività di processamento dei testi scritti in linguaggio naturale è caratterizzata da quattro livelli di analisi, eseguiti in progressione [33] [34] [37]:

- Analisi lessicale e morfologica: studio della forma delle parole nella frase;
- Analisi sintattica: studio della funzione sintattica delle parole nella proposizione e delle proposizioni nella frase complessa;
- Analisi semantica: studio del significato delle singole parole e della loro combinazione per apprendere il significato della frase intera;
- Analisi pragmatica: studio del contesto in relazione al significato delle frasi.

Il percorso di elaborazione di un testo si sviluppa dunque a partire dal focus sulle singole parole per approdare poi alla comprensione semantica delle frasi e di come esse si collegano per dare significato al testo complessivo.

Una "classica" pipeline [13] di NLP è illustrata in figura 3.1.

Gli step principali che descrivono il flusso ad alto livello sono quelli raffigurati in figura 3.1 a sinistra e, come si può notare, corrispondono esattamente ai livelli di analisi spiegati prima, con l'aggiunta di un'operazione preliminare di *Tokenization*.

La Tokenization è una fase di preprocessing di fondamentale importanza per lo svolgimento delle successive elaborazioni. Essa riguarda la scomposizione del testo in unità strutturali più piccole, definite *token* [33]. Un token può essere sia una parola sia una singola frase, in quanto di norma viene considerata come parte integrante del processo anche la *Sentence Segmentation*, ovvero la divisione del documento in frasi [33].

Un esempio di divisione in token è riportato in figura 3.2.

[13] *Pipeline* è un termine inglese che in italiano può essere tradotto con *tubatura*. In informatica questo termine viene usato per indicare un insieme di componenti software collegati tra loro in cascata in modo che l'output di uno costituisca l'input del successivo. Risorsa online: https://it.wikipedia.org/wiki/Pipeline_software (Ultimo accesso: 1 febbraio 2021).

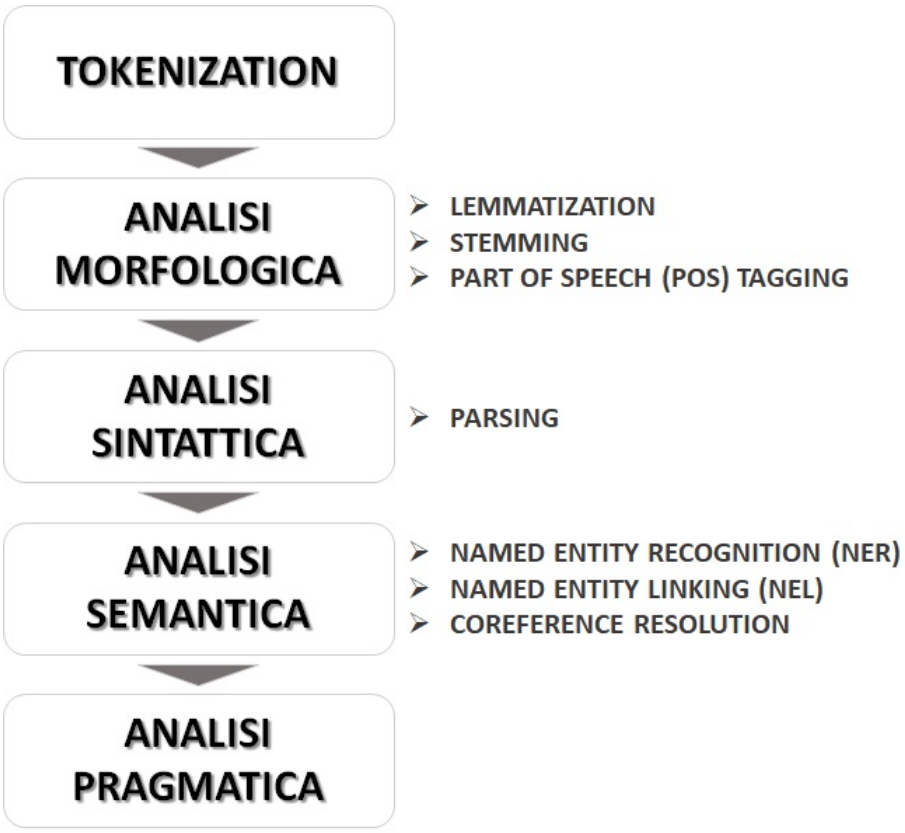


Figura 3.1: La "classica" pipeline. Immagine realizzata a partire da: Guts, Y. *Natural Language Processing*. NLP Morning@Lokiha, 2016

Figura 3.2: Esempio di Tokenization

Ognuna di queste quattro macro-fasi prevede l'esecuzione di specifiche operazioni algoritmiche. In figura 3.1 sono state riportate le principali in merito all'analisi lessicale e morfologica, sintattica e semantica e verrano di seguito spiegate nel dettaglio. L'analisi pragmatica risulta invece completamente dipendente dal tipo di applicazione [37], per cui non è opportuno definire dei task principali. Quelli elencati in figura 3.1 costituiscono i task più comunemente usati in quasi tutti i sistemi di NLP, in quanto esistono molti altri tipi di algoritmi e procedure, alcuni più specifici di altri e utilizzati solo per particolari applicazioni. Si rimanda dunque ad una documentazione più ricca ed esaustiva per apprezzare tutte le ulteriori funzionalità e possibilità della disciplina.

Lemmatization & Stemming I processi di Lemmatization e di Stemming sono abbastanza simili in riferimento al risultato prodotto, perchè entrambi hanno l'obiettivo di risalire al lemma[14] della parola. A partire da forme flesse o derivate, si effettua un'elaborazione della parola al fine di ottenere la sua forma base [38]. Si tratta di una procedura di analisi lessicale per entrambi i metodi, tuttavia sussistono delle differenze di approccio.

Lo Stemming consiste nell'utilizzo di un metodo euristico abbastanza rude che elimina la parte finale delle parole e a volte anche i suffissi e i prefissi, indipendentemente dal significato della parola stessa [38].

La Lemmatization invece fa uso del vocabolario e dell'analisi morfologica, rimuovendo solo le desinenze flessive delle parole e fornendo la forma base contenuta nel dizionario [38].

In entrambi i casi, il risultato del processo conduce al riconoscimento del lemma della parola, come si può osservare nell'esempio di figura 3.3.

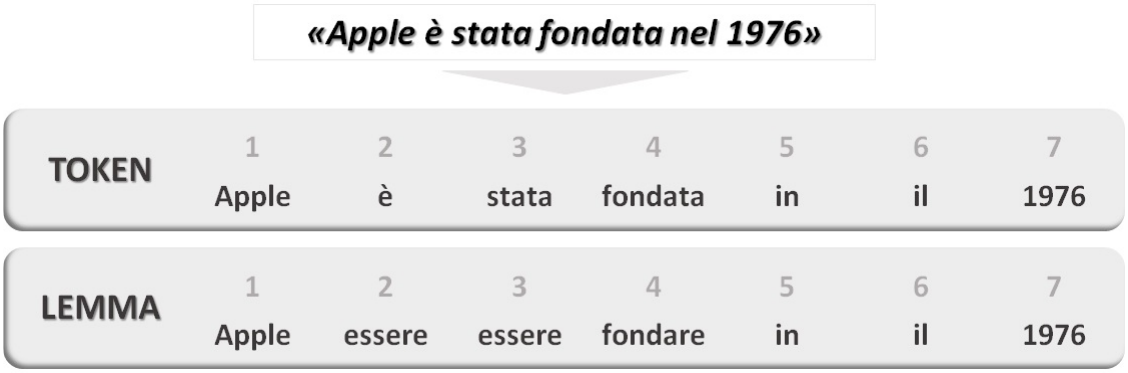


Figura 3.3: Esempio di Lemmatization/Stemming

L'aspetto maggiormente vantaggioso della Lemmatization rispetto allo Stemming è l'accuratezza, ma una procedura più accurata comporta un maggiore tempo computazionale, per cui la scelta nell'utilizzo di uno dei due approcci va fatta in riferimento all'applicazione per cui si vuole sviluppare il sistema.

Part of speech (POS) tagging L'analisi morfologica del testo fornisce le informazioni relative al ruolo della parola nella frase (es. nome, aggettivo, verbo, ecc.) e alla loro flessione (es. genere, persona, numero, ecc.). Nel processo di POS tagging ad ogni parola vengono assegnate una o più etichette, a seconda della natura dettagliata del software che si utilizza, che identificano letteralmente quale "parte del discorso" esse rappresentano [33]. In figura 3.4 si riportano le parti del discorso per la frase degli esempi 3.2 e 3.3.

I tag esistenti sono molti di più, soprattutto per la lingua italiana, ma per fornire un esempio concreto si descrivono di seguito quelli usati in figura 3.4:

[14]In lessicografia, il lemma è la forma di citazione di una parola in un dizionario. Risorsa online: https://it.wikipedia.org/wiki/Lemma_(linguistica) (Ultimo accesso: 28 gennaio 2021).

- PROPN (*Proper noun*): nome proprio;
- AUX (*Ausiliary*): verbo ausiliario;
- VERB (*Verb*): verbo;
- ADP (*Adposition*): preposizione;
- DET (*Determiner*): aggettivo determinativo;
- NUM (*Numeral*): numero o aggettivo numerale.

Figura 3.4: Esempio di POS tagging

Parsing Il parsing è quel processo che consente di identificare la struttura grammaticale della frase, individuando le relazioni sintattiche tra i diversi componenti e rappresentandola tramite una grafo ad albero, o *parse tree* [34] [39].

A seconda dei formalismi grammaticali adottati, questi componenti possono essere sia singole parole sia gruppi di parole, molto spesso definiti *chunck* [39]. Sulla base di questa distinzione, è possibile parlare di due diverse categorie di parsing: *Dependency Parsing* e *Costituency Parsing*.

Il Costituency Parsing realizza un grafo in cui la frase viene divisa in chunk che appartengono ad una specifica categoria grammaticale [39]. Nell'esempio riportato in figura 3.5 le categorie grammaticali riguardano le frasi nominali (NP o *Noun Phrase*), i predicati verbali (VP o *Verb Phrase*) caratterizzati da un verbo e da una frase nominale, e infine dalla dichiarativa semplice (S o *Simple declarative clause*), formata da una NP e da una VP [34].

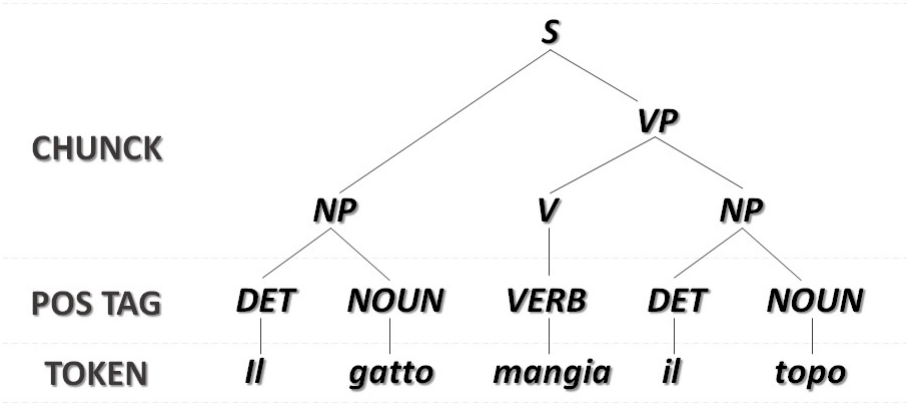


Figura 3.5: Esempio di Costituency Parsing. Esempio tratto da: Basili, R., Pazienza, M.T. e Zanzotto, F.M. «Evaluating a robust parser for Italian». In: Carroll, Basili, et al (1998).

Il Dependency Parsing invece rappresenta la dipendenza sintattica tra le singole parole della frase [39]. Nell'esempio in figura 3.6 il significato delle dipendenze è il seguente:

- det (*determiners*): aggettivo determinativo;
- nsubj (*nominal subject*): soggetto;
- obj (*object*): complemento oggetto.

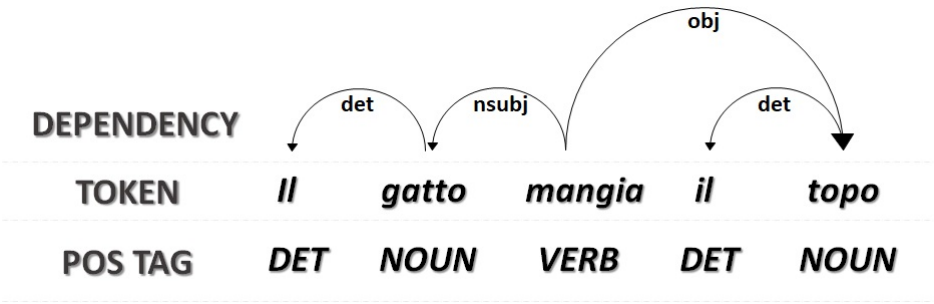


Figura 3.6: Esempio di Dependency Parsing. Esempio tratto da: Basili, R., Pazienza, M.T. e Zanzotto, F.M. «Evaluating a robust parser for Italian». In: Carroll, Basili, et al (1998).

Come si può notare dagli esempi forniti, le informazioni contenute nei parse tree sono differenti e lo è anche la rappresentazione stessa [39]. Non esiste un metodo migliore dell'altro, anche in questo caso infatti è necessario scegliere un tipo di analisi rispetto all'altra solo in dipendenza dell'applicazione finale.

Named Entity Recognition (NER) L'individuazione di entità specifiche all'interno di un testo rappresenta la procedura fondamentale per un'attività di Information Extraction.

Il processo di Named Entity Recognition può essere diviso in due step [34]:

1. identificazione dei confini dell'entità
2. assegnazione dell'etichetta

La maggior parte delle librerie e dei tool di Natural Language Processing possiedono già delle categorie predefinite di entità in quanto considerate le più comuni. Si parla in questo caso di *Named Entity Generic* e le categorie che ne fanno parte sono i nomi di persona (PERSON - PER), le organizzazioni (ORGANIZATION - ORG), e i luoghi (LOCATION - LOC) [40]. Un'esempio di riconoscimento NER è mostrato in figura 3.7.

Figura 3.7: Esempio di NER

Di entità generiche ne esistono altre, ma quali e quante dipende dal tipo di software che si utilizza e ovviamente dalla lingua.

E' anche possibile sviluppare un *Domain-specific Named Entity*, ovvero un sistema di riconoscimento di entità specifiche, scelte sulla base del contesto di analisi [40].

I metodi tramite i quali si costruisce un sistema NER sono sostanzialmente quattro [40]:

- *Rule-based*: si definisce un set di regole scritte manualmente da esperti, sulla base dei quali si riescono a riconoscere specifici pattern all'interno del testo.
- *Supervised Learning*: si usano algoritmi di machine learning per individuare regole di NER a partire da un set di documenti (*corpus* [15]) manualmente annotati da un esperto, al fine di identificare le entità e le informazioni che si vogliono riconoscere e dunque estrarre.
- *Unsupervised Learning*: gli algoritmi di machine learning adottati imparano a sviluppare le regole man mano che gli vengono forniti degli esempi di testo non annotati. Il percorso di apprendimento termina nel momento in cui le regole trovate non cambiano più.
- *NE Extraction* (NEX): l'approccio è simile a quello non supervisionato, ma in questo ambito l'obiettivo non è quello di sviluppare delle regole, piuttosto di generare una *look-up list*, ovvero un vocabolario ricco di esempi di entità.

Le tecniche sopra elencate non costituiscono esclusivamente dei metodi impiegati per le fasi di NER, ma più in generale possono essere considerate come degli approcci di base per lo sviluppo di un intero sistema di NLP, come verrà definito maggiormente nel dettaglio nel prossimo paragrafo.

[15] Il termine *corpus* deriva dal latino e in linguistica indica un grande e strutturato set di testi. Quando ci si riferisce a più collezioni di testi, si usa il termine *corpora*, ovvero il plurale di *corpus*. Risorsa online: https://en.wikipedia.org/wiki/Corpus (Ultimo accesso: 27 gennaio 2021).

Named Entity Linking (NEL) Il processo di Named Entity Linking è conosciuto anche come *Named Entity Disambiguation* (NED) o *Named Entity Normalization* (NEN) e si esplica nell'assegnazione di una identità univoca alle entità riconosciute nel testo [41] [42]. Alla base dell'algoritmo esiste un *Knowledge Base* (KB), ovvero una banca dati specifica, che in questo ambito è Wikipedia, la quale contiene le entità univoche a cui vanno collegate le entità rinvenute nel testo [42].

Le tecniche di *entity linking* possono essere suddivise in due diverse tipologie [42]:

- *Text-based*: l'associazione tra l'entità rinvenuta nel testo e quella univoca della KB viene effettuata a partire dall'analisi di apposite *feature* estratte dai *corpora*;
- *Graph-based*: per rappresentare il contesto e le relazioni tra le entità, non ci si basa unicamente sulle informazioni rilevate dai *corpora*, ma si usano i *knowledge graph* [16] creati proprio a partire dai KB.

Di norma il NEL viene preceduto da una fase di NER, tramite la quale si riconoscono le entità da collegare a quelle della KB [42], come rappresentato nell'esempio in figura 3.8.

Figura 3.8: Esempio di NEL. Immagine realizzata a partire da: https://en.wikipedia.org/wiki/Entity_linking (Ultimo accesso: 27 gennaio 2021).

Coreference Resolution La Coreference Resolution costituisce quel task tramite il quale si individuano quali nomi comuni, nomi propri e pronomi si riferiscono alle stesse entità nel testo [41].

Se ad esempio si considera la frase:

> Giulia mi ha ridato le chiavi. Lei le aveva prese ieri per sbaglio dalla mia scrivania

le parole *Giulia* e *Lei* appartengono alla stessa entità, così come *mi* e *mia*.

[16] Un *Knowledge graph* o grafico della conoscenza è un modello di organizzazione grafica di un insieme di dati, i quali vengono rappresentati come una raccolta di descrizioni interconnesse di entità: oggetti, eventi o concetti. Risorsa online: https://en.wikipedia.org/wiki/Knowledge_graph (Ultimo accesso: 29 gennaio 2021).

Questo tipo di attività è fondamentale per alcune tipologie di applicazioni come la Text Summarization, l'Information Extraction e altre [41].

3.3 I metodi

L'evoluzione storica raccontata in breve nell'introduzione di questo capitolo ha segnato lo sviluppo di tre diversi metodi tramite i quali realizzare sistemi di Natural Language Processing [31] [33]:

- *Symbolic method* o *knowledge-based method* (1950 - 1980)
- *Statistical method* (1990 - 2010)
- *Machine Learning method* (2010 - present)

Oggi i Symbolic method sono stati quasi totalmente soppiantati per dare spazio ai metodi statistici e all'apprendimento automatico, in quanto molto meno complessi a livello di preparazione dei dati di input e maggiormente performanti. Nonostante in effetti quello basato sul machine learning sia il metodo più utilizzato, in molti casi si mantiene ancora l'uso dei metodi statistici, soprattutto quando usati come componenti di sistemi ampi, caratterizzati da diversi task [31].

Symbolic method Data una lingua, il sistema riceve un set di complesse regole grammaticali e sintattiche, realizzate da esperti, sulla base dei quali diventa in grado di emulare la capacità dell'uomo di comprendere il linguaggio naturale e di analizzare il contenuto di un testo [33]. Più propriamente si parla in questo caso di *Rule-based techinque* [33].

In diverse applicazioni però l'utilizzo di regole non è sufficiente affichè il sistema sia capace di estrarre specifiche informazioni o entità dal documento e per tale motivo nell'ambito dei Symbolic method sono state ideate le *Dictionary look-up* o *Look-up list* [33]. In alcuni software si utilizza anche la definizione di *Gazetteer* [34]. Si costruiscono dunque una o più liste di parole, suddivise per categoria [34], dove ognuna di esse è arricchita da informazioni lessicali e morfologiche, quali [43]:

- il lemma
- il POS tag
- le caratteristiche flessive

Tramite la conoscenza insita nei dizionari e le regole scritte a mano con accuratezza dagli esperti, l'algoritmo è in grado di effettuare l'analisi lessicale, morfologica e sintattica della frase, ma anche di identificare le categorie semantiche sulla base del riconoscimento delle parole.

Statistical method Dal 1990 i modelli statistici hanno consentito di oltrepassare il problema della redazione manuale delle regole, proponendo dei sistemi capaci di costruire le proprie regole a partire direttamente dall'analisi di un grande corpus [33].

L'algoritmo non impara regole già predefinite, ma prende delle decisioni probabilistiche sulla base dei dati di input che gli vengono forniti [31]. In questo modo l'analisi che verrà effettuata sarà quella "più probabile" e dunque ampiamente affidabile, soprattutto poichè vengono fornite diverse risposte possibli, piuttosto che una sola [31] .

Machine Learning method Grazie agli studi più approfonditi che sono stati fatti nel campo del machine learning, a partire dal 2010 si è provato ad impiegare algoritmi di apprendimento automatico nell'elaborazione dei testi. Nell'ambito del machine learning, i metodi di apprendimento automatico possono essere distinti in metodi *supervisionati* e *non supervisionati*.

Quando si parla di apprendimento automatico di tipo supervisionato si fa riferimento alla peculiarità di fornire al sistema una conoscenza di base, a partire dalla quale l'algoritmo sarà in grado di imparare quanto fornitogli in input. Spesso questo metodo è anche conosciuto come *Connectionist method* [17] in quanto è proprio dalle reti neurali artificiali che si è partiti per l'applicazione del machine learning al NLP.

In figura 3.9 viene riportato un flusso concettuale base per l'utilizzo di questi metodi.

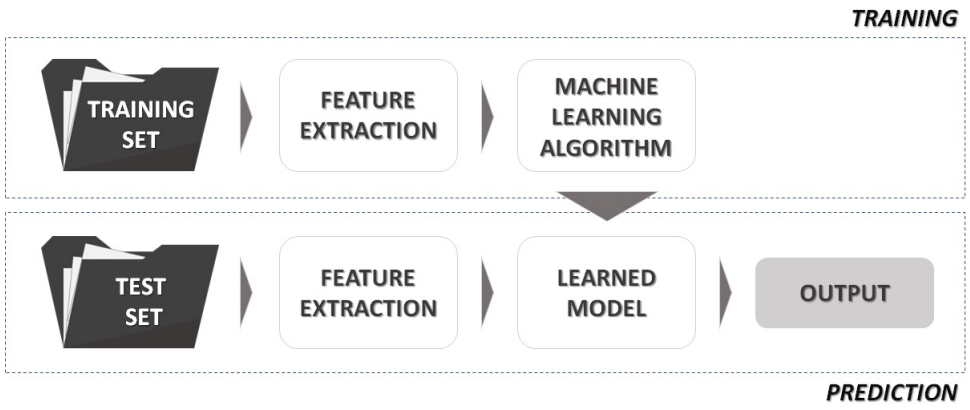


Figura 3.9: Supervised Machine Learning method. Immagine realizzata a partire da: Rosati, S. *Natural Language Processing.* Politecnico di Torino.

Si costruisce un corpus annotato da esperti che viene etichettato come *training set*. Da questo set di dati vengono estratte delle *feature*, tramite opportuni metodi di Feature Extraction, che verranno successivamente date in input al sistema. A partire dalle *feature*

[17] Il connessionismo è un approccio dell'intelligenza artificiale che riguarda la creazione di algoritmi computazionali ispirati al comportamento della mente umana, conosciuti come reti neurali artificiali. Risorsa online: https://it.wikipedia.org/wiki/Connessionismo (Ultimo accesso: 31 gennaio 2021).

l'algoritmo effettua una procedura di apprendimento automatico, le cui modalità dipendono dal tipo di approccio che si sceglie di utilizzare (es. Reti neurali, Support Vector Machines (SVM), ecc.).

Una volta ottenuto un modello definitivo, si estrae dai corpora selezionati un corpus non annotato che verrà utilizzato come *test set*. Si effettua nuovamente l'operazione di estrazione delle *feature* e si danno in input al modello, il quale produrrà l'output richesto.

Per gli scopi del NLP possono essere utilizzati anche gli approcci non supervisionati, che risultano più comodi dal punto di vista della preparazione del *training set*. I dati di input infatti non necessitano di annotazioni, ma il modello, grazie all'esperienza, è in grado esso stesso di estrarre le similarità tra i dati e trovare le classi a cui appartengono. E' ovvio che sistemi di questo tipo abbiano bisogno di una mole di informazioni di input più elevata per consentirgli di individuare i pattern tra i dati e dunque di classificare correttamente quelli successivi.

Feature Extraction per NLP Nell'ambito del NLP, le tecniche di Feature Extraction che di norma precedono l'applicazione degli algoritmi di machine learning sono due: il *Bag-of-Words* (BoW) e il *Word Embedding* [44].

Il BoW è un modello di rappresentazione dei dati di un documento, che semplicemente conta quante volte una parola compare nel documento stesso [44] e lo trasforma in un vettore di numeri, dove ogni numero rappresenta l'occorrenza della parola o la sua attivazione (figura 3.10).

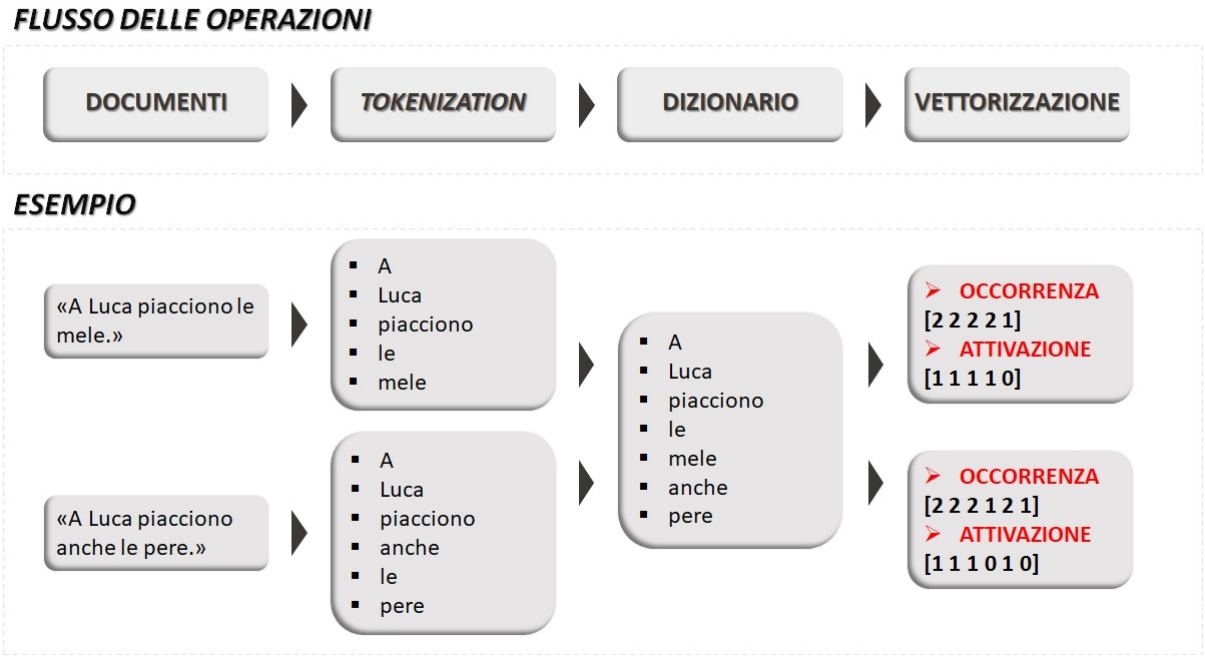


Figura 3.10: Esempio di un flusso di trasformazione Bag-of-Words

In molti casi risulta più utile indicare quanto una parola sia importante per il documento di un corpus. A tal proposito si fa riferimento alla tecnica del *Term Frequency/Inverse*

Document Frequency (TF-IDF), che associata al BoW, fornisce dei vettori contententi un valore statistico che indica l'importanza della parola [33].

Lo score in questione viene calcolato come prodotto tra il *Term Frequency* (TF) e l'*Inverse Document Frequency* (IDF) (eq. (3.1)) [33].

$$TF - IDF = TF * IDF \tag{3.1}$$

Il TF (eq. (3.2)) rappresenta l'occorrenza della parola w nel documento, mentre l'IDF (eq. (3.3)) quanto la parola w sia rara tra i documenti del corpus [33].

$$TF = \frac{numero \quad di \quad occorrenze \quad di \quad w \quad nel \quad documento}{numero \quad totale \quad di \quad parole \quad nel \quad documento} \tag{3.2}$$

$$TF = \log \frac{numero \quad di \quad documenti \quad nel \quad corpus}{numero \quad di \quad documenti \quad contenenti \quad w} \tag{3.3}$$

Il Word Embedding fornisce nuovamente una rappresentazione dei dati nello spazio dei vettori, ma lo score calcolato tiene conto del contesto e delle relazioni tra le parole, fornendo un'individuazione più accurata delle parole simili [44]. Grazie all'utilizzo delle reti neurali, il sistema è in grado di catturare le proprietà semantiche e le relazioni linguistiche tra le parole [45].

Le applicazioni per le quali sta aumentando l'utilizzo del Word Embedding sono molteplici, come Information Extraction, Information Retrieval, Sentitment Analysis, Question aswering e Text summarization [45]. Anche in ambito biomedicale è particolarmente utilizzato per Named Entity Recognition, Relation Extraction e per altre operazioni di riconoscimento di abbreviazioni e di sinonimi di termini medici [45].

Le implementazioni del Word Embedding sono diverse (word2vec, GloVe, FastText ecc.), ma quella in assoluto più popolare è il *word2vec* [44].

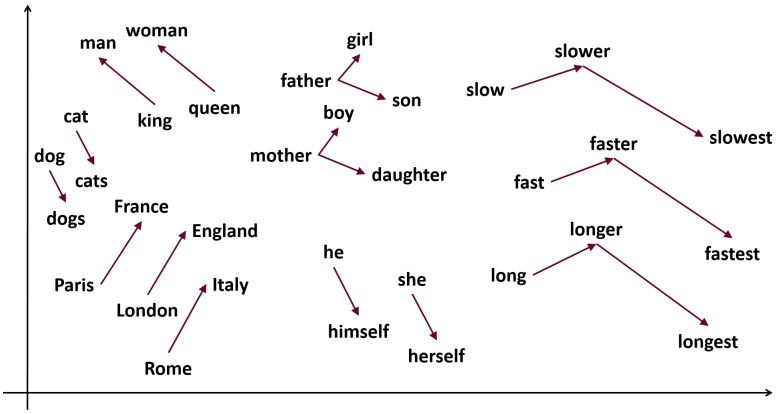


Figura 3.11: Esempio di spazio vettoriale prodotto da un algoritmo word2vec. Immagine tratta da: https://medium.com/analytics-vidhya/implementing-word2vec-in-tensorflow-44f93cf2665f (Ultimo accesso: 3 febbraio 2021).

Il modello word2vec è in generale una rete neurale a due layer, che preso in ingresso un corpus restituisce un insieme di vettori che rappresentano la distribuzione semantica delle parole nel testo [46]. Un vettore corrisponde ad una parola e nello spazio dei vettori le parole più saranno vicine più saranno semanticamente simili [46] (figura 3.11).

L'architettura computazionale che sta dietro al word2vec è molto più complessa rispetto a come è stata descritta in questo paragrafo, ma poichè il fine di questa breve dissertazione è unicamente quello di fornire al lettore gli strumenti per comprendere la trattazione successiva, anche in questo caso, per maggiori approfondimenti, si rimanda ad una documentazione più completa.

3.4 L'ambiguità del linguaggio naturale

La capacità dell'essere umano di comunicare deriva da un background e da una conoscenza approfondita della lingua che ognuno di noi ha sviluppato sin da bambino. I sistemi di NLP consentono di effettuare operazioni complesse sul testo molto più velocemente di noi e probabilmente in maniera molto più dettagliata, ma ciò che gli manca è proprio la conoscenza *a priori* dei contesti semantici. Questo aspetto rende spesso difficoltosa la comprensione corretta dei testi, generando errori e fraintendimenti.

L'ambiguità è dunque la principale problematica relativa al Natural Language Understanding e può essere individuata a diversi livelli [47]:

- Ambiguità lessicale: una parola può avere diversi significati [33].

  Si osservi il seguente esempio, in cui la parola *porto*, oltre a costituire una parte del discorso diversa, nome in un caso, verbo in un altro, ha due significati completamente differenti:

  > Oggi questa casa è un **porto** di mare
  > Oggi ti **porto** al mare

- Ambiguità sintattica: la sintassi di una frase può essere interpretata in diversi modi [47]. Ad esempio, nella frase [47]:

  > Chiara ha visto Luca in giardino con il cannocchiale

  a livello sintattico, non si capisce se Luca ha il cannocchiale o se Chiara ha visto Luca in giardino grazie all'uso del cannocchiale. In questo ambito è possibile inserire anche il problema legato all'interpretazione dei pronomi, per cui nella domanda:

  > E' **tuo** questo regalo?

il pronome possessivo *tuo* è ambiguo, poichè chi fa la domanda può voler chiedere se il regalo è *per te*, ma potrebbe anche voler sapere se è effettivamente *tuo*, ovvero se lo hai comprato *tu*.

- Ambiguità semantica: il significato di una parola o di una frase può essere interpretato diversamente a seconda del contesto [33]. Nella frase:

  La carta **brucia**

  il termine *brucia* non è ambiguo dal punto di vista lessicale, nè la frase può essere mal interpretabile a livello sintattico, poichè contiene esclusivamente un soggetto e un verbo, ma a livello semantico può voler dire che vi è l'evidenza che un pezzo di carta stia bruciando oppure che la carta, come materiale, può essere bruciata. In questo specifico esempio, l'ambiguità è unicamente legata al senso della frase, che potrebbe essere letta con diverse accezioni a seconda del contesto in cui è inserita, ma in generale può dipendere anche dai primi due livelli di ambiguità.

Oltre alle criticità legate all'ambiguità del linguaggio umano, ci sono altre problematiche che rendono l'analisi dei testi piuttosto complessa. Basti pensare all'uso di acronimi o di abbreviazioni, ad esempio in ambito medico, oppure di tipologie di linguaggio specifico, come quello giuridico, in cui si usano parole insolite e strutture sintattiche più complesse.

Anche le modalità in cui si esprimono i concetti possono indurre confusione ed errori, poichè una stessa idea può essere esposta in diverse forme [33].

Ogni lingua ha le proprie peculiarità, alcune sono più complesse dal punto di vista sintattico, come l'italiano, mentre altre, ad esempio come l'inglese, costituiscono una maggiore fonte di ambiguità. A tal proposito, è necessario che, quando si sviluppa un sistema di NLP, le lingue supportate debbano essere gestite in tutte le loro sfaccettature.

Per gestire al meglio tutte le complicazioni che derivano letteralmente dall'*ambiguità* del linguaggio naturale, qualunque esso sia, oltre all'uso di metodi e tecniche *ad hoc*, è molto importante che vengano forniti al sistema corpora ricchi di esempi provenienti da diversi ambienti, così da fornigli la possibilità di apprendere e la capacità di interpretare correttamente i testi.

3.5 Il NLP in ambito clinico

Le fonti da cui potere trarre i documenti contenenti informazioni biomedicali sono diverse e spaziano dalla letteratura scientifica e dai dati ottenibili dai social media fino ad arrivare ovviamente alle cartelle e ai fascicoli clinici elettronici, che racchiudono la storia clinica dei pazienti [48]. Quando si vuole sviluppare un sistema di NLP clinico si fa riferimento a quelli che in inglese vengono definiti *Electronic Health Record* (EHR), che in italiano possono essere identificati con i fascicoli sanitari elettronici. E' stato proprio l'imponente

processo di informatizzazione a livello sanitario che ha consentito oggi di avere a disposizione una grande mole di dati con la quale potere raccogliere i corpora che servono per realizzare un sistema di NLP clinico.

In base alla tipologia di documentazione che si ha a disposizione, allo specifico ambito clinico e soprattutto al caso d'uso, è possibile individuare tre diverse applicazioni [33] [48]:

- *Information Extraction* per identificare specifiche entità semantiche e spesso anche le relazioni tra esse (*Relation Extraction*), insieme al riconoscimento di definizioni temporali, negazioni e frasi ipotetiche che aiutano a comprendere il contesto relativo alle informazioni estratte;
- *Text Classification* per classificare interi documenti o piccole parti di essi in un set di categorie predefinite;
- *Text Summarization* per estrarre i dati salienti dei documenti, soprattutto di un insieme di documenti provenienti da diversi pazienti, in modo da rendere le attività di revisione e di *decision making* più agevoli e veloci.

In figura 3.12 si riporta una descrizione schematica delle fasi di realizzazione di un sistema di NLP clinico.

Figura 3.12: Flusso delle attività per lo sviluppo di un sistema di NLP clinico. Immagine realizzata a partire da: Viani N. & Velupillai, S. *Natural language processing methods for clinical text.* NHS - National Institute for Health Research.

Al di là della scelta dello *use case* e dunque dell'obiettivo primario, la prima vera attività operativa riguarda la creazione di un corpus. La scelta dei documenti da utilizzare è una delle fasi fondamentali e per tale motivo, per focalizzarsi sui dati che interessano ai fini del progetto, è consigliabile effettuare questo lavoro insieme ad un team di esperti [48].

Le stesse figure professionali interpellate nelle operazioni di raccolta della documentazione saranno quelle che si occuperanno di annotare manualmente i testi. L'annotazione è una procedura piuttosto lunga che va effettuata con cura nell'ottica di individuare nei testi le entità e le informazioni che si vogliono estrarre o riconoscere [48]. Definire dunque delle linee guida per l'annotazione è importante, anche per fare in modo che gli esperti seguano la stessa linea e gli stessi principi [48]. E' consigliabile impiegare almeno due figure indipendenti in questo frangente, in modo da rendere il corpus standard di riferimento di elevata qualità e più robusto [48]. Da questo corpus non si estrarrà unicamente il *training set*, ma anche il set di documenti che si sfrutterà per la valutazione finale (*test set*) [48].

Indipendentemente dall'applicazione e dal metodo che si sceglierà di utilizzare, ogni documento deve affrontare una fase di preprocessing caratterizzata dalle seguenti operazioni [48]:

- *Tokenization* e *Sentence Segmentation*;
- *Lemmatization* e *POS tagging* per l'analisi morfologica;
- *Parsing* per l'analisi sintattica.

E' importante scegliere un software in grado di eseguire correttamente queste attività, poichè i task successivi dipendono spesso dalla qualità del risultato ottenuto alla fine del preprocessing [48].

I metodi che è possibile usare sono già stati presentati nel paragrafo 3.3 ed in generale è preferibile usare rule-based method e look-up list per Information Extraction e machine learning method nell'ambito della classificazione [48]. Al fine di estrarre informazioni, se l'intento è quello di individuare sintomi, diagnosi o nomi di farmaci è molto comodo fare uso dei vocabolari già disponibili, come UMLS (*Unified Medical Language System*) o SNOMED CT (*Systematized Nomenclature of Medicine Clinical Terms)* ad esempio [48]. E' altresì possibile creare delle proprie liste, sulla base delle categorie semantiche d'interesse [48], ovviamente usando una elevata mole di documenti, che sarà necessario elaborare ulteriormente, con metodi di rimozione delle *stopword* [18] e operazioni di *lowercasing*, che convertiranno tutti i caratteri in minuscolo.

Agire invece con il *pattern matching* insieme all'utilizzo di regole grammaticali e sintattiche appositamente redatte da esperti è utile nel momento in cui si vogliono estrarre sia definizioni standard sia informazioni strutturalmente più complesse [48].

Oltre che per classficare i testi, in realtà, i metodi di apprendimento supervisionato si usano parecchio anche nell'ambito del Named Entity Recognition, per il riconoscimento

[18]Le *stopword* sono tutte le parole di una specifica lingua che, data la loro elevata frequenza, sono considerate poco significative, come articoli, preposizioni, congiunzioni ecc. Risorsa online: http://opac.pisa.sbn.it/opaclib/opac/helpopacpisa/sito-HelpOpacPisa/MenuPrincipale_HelpOpac/CaratteristicheRicerca/StopWord/index.html (Ultimo accesso: 29 gennaio 2021).

di specifici token [48]. Inoltre, quando l'applicazione lo consente, l'utilizzo di reti neurali non supervisionate è utile soprattutto per bypassare le lunghe operazioni di annotazione manuale, garantendo comunque risultati soddisfacenti [48].

Come già accenato, parte del corpus verrà utilizzato per valutare le performance del sistema sviluppato. Le percentuali di divisione del corpus tra *training set* e *test set* si scelgono in maniera specifica caso per caso, sulla base anche delle diverse tipologie di testi che si hanno a disposizione. Per effettuare la valutazione è opportuno calcolare delle metriche quantitative, che di norma, nell'ambito di un sistema di NLP, sono *recall*, *precision* e *F1-score* [48] [49].

In particolare, nell'ambito del Named Entity Recognition e dunque dell'estrazione di entità dal testo si fa riferimento ad un problema di classificazione binaria [49]. Nel testo, infatti, per ogni token bisogna valutare se esso verrà riconosciuto o meno [49]. Ci si focalizza dunque sugli esempi positivi, che saranno sicuramente in minor numero rispetto ai negativi e per tale motivo si scelgono le metriche recall e precision, le quali andranno a valutare principalmente gli aspetti quantitativi del riconoscimento ma anche quelli qualitativi [49] [50].

		Classe reale	
		N	**P**
Classe predetta	**N**	Veri Negativi (VN)	Falsi Negativi (FN)
	P	Falsi Positivi (FP)	Veri Positivi (VP)

Tabella 3.1: Confusion matrix per un classificatore binario

Prendendo come riferimento la confusion matrix riportata nella tabella 3.1, la recall, conosciuta anche come *sensitività*, viene calcolata come mostrato nell'equazione (3.4), dunque come rapporto tra i veri positivi (VP) e la somma di tutti i positivi che il sistema avrebbe dovuto predire (VP+FN) [49] [50].

$$recall = \frac{VP}{VP + FN} \tag{3.4}$$

Calando l'equazione nell'ambito di un sistema di NLP, la recall si calcola come rapporto tra tutti i corretti riconoscimenti che sono stati effettuati rispetto alle annotazioni di riferimento [48]. Se l'applicazione riguarda l'estrazione di entità, un elevato valore di sensitività indica che la maggior parte delle annotazioni degli esperti sono state correttamente riconosciute [48].

Sempre a partire dalla confusion matrix 3.1, il valore di precision si calcola come nell'equazione (3.5), in cui troviamo il rapporto tra i veri positivi (VP) e la somma di tutti i positivi riconosciuti dal sistema (VP+FP) [49] [50].

$$precision = \frac{VP}{VP + FP} \tag{3.5}$$

Come descrive correttamente l'espressione *valore predittivo positivo*, che è un altro nominativo tramite il quale si conosce la precision, un elevato valore di precision indica che la maggior parte delle entità estratte sono corrette [48].

Quale delle due metriche considerare maggiormente prioritaria è una decisione che va presa in riferimento ai propri obiettivi, poichè nella maggior parte dei casi risultano in contrapposizione, in quanto ad un elevato valore di sensitività corrisponde un valore di precision basso e viceversa [48]. Al fine di bilanciare le due misure ed ottenere un valore che fornisca un'indicazione complessiva sulle performance del sistema, si utilizza anche l' F1-score o *F-measure*, calcolata come media armonica tra la precision e la recall, come mostrato nell'equazione (3.6) [48] [49].

$$F1 - score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.6}$$

Quelle appena presentate sono le classiche operazioni che si svolgono nell'ambito dello sviluppo di un sistema di Natural Language Processing clinico, ma ovviamente è possibile creare una pipeline specifica per i propri scopi, aggiungendo ulteriori task.

Nel capitolo seguente ci si concentrerà sull'analisi dei principali tool open source disponibili per la lingua italiana e sulla valutazione delle opportunità che un sistema di NLP clinico può offrire ai fini del presente progetto.

Capitolo 4

NLP per i testi clinici in italiano

Il Natural Language Processing è oggi una delle discipline più sfruttate in assoluto per l'analisi automatica e semi-automatica del linguaggio umano. I campi d'impiego sono tanti e vanno dal marketing ai social network, dai processi business alla ricerca scientifica. Soprattutto per la lingua inglese, i task del NLP sono già ampiamente utilizzati anche in clinica, per la gestione e la manipolazione dei documenti clinici elettronici, al fine di agevolare e rendere più veloce il lavoro dei medici, degli operatori sanitari e dei ricercatori nelle operazioni di estrazione, classificazione ed aggregazione delle informazioni.

Avere la possibilità di analizzare i testi scritti in linguaggio naturale delle cartelle cliniche è importante anche ai fini di questo progetto, in cui, per ottenere un Piano Diagnostico Terapeutico Assistenziale il più dettagliato e fedele possibile in riferimento ai processi che si svolgono nell'organizzazione sanitaria, è di fondamentale importanza che vengano raccolti tutti i dati e le informazioni considerati necessari. Oltre ai dati codificabili e direttamente estraibili dalla documentazione, per i quali è già stato sviluppato un sistema utile per la raccolta ed il salvataggio, potrebbe rivelarsi necessario anche avere a disposizione le informazioni contenute all'interno dei documenti scritti in linguaggio naturale, come i referti degli esami e delle visite specialistiche e i diari clinici.

Nonostante nel corso degli ultimi anni siano stati studiati dei sistemi di NLP clinici per la lingua italiana, non esiste ancora oggi un tool o un software open source in grado di effettuare queste elaborazioni, nè possono essere usati i sistemi appositamente ideati per la lingua inglese.

A tal proposito, la seconda parte di questo lavoro riguarda lo studio delle possibilità che il NLP offre per l'estrazione di specifiche entità a partire dai documenti clinici dei pazienti geriatrici oggetto di questo lavoro. Lo scopo di questo studio non è tuttavia quello di sviluppare un sistema altamente complesso e generico, ma proprio porre i task del NLP al servizio degli obiettivi di questo progetto, per il quale sarà necessario individuare solo

alcune e specifiche informazioni, oltrettutto a partire da poche e ben definite categorie di testi.

Il capitolo in questione dunque tratterà proprio di quanto è emerso dalla ricerca e dall'analisi effettuata in proposito e porrà le basi per la definizione futura di un semplice programma *custom* in grado di elaborare le schede di interesse delle cartelle cliniche dei pazienti dell'ospedale Humanitas Gradenigo di Torino.

La prima sezione del capitolo è dedicata alla descrizione dei principali tool open source che sono stati considerati adatti per l'elaborazione della lingua italiana e alla valutazione delle performance degli stessi, tramite l'utilizzo di tre documenti selezionati appositamente dai corpora standard per stabilire quale o quali tool risultano più adatti ad essere utilizzati.

Nel seguito della trattazione, una volta selezionati gli strumenti da usare, si esegue una valutazione anche su testi clinici, che, come si vedrà, sono stati estratti dalle sei cartelle messe a disposizione per questo lavoro di tesi.

A tal proposito, è stata programmata una semplice pipeline per Information Extraction dedicata a dimostrare come, una volta definite le espressioni e i dati da rinvenire, è possibile estrarli dal testo, senza avere a disposizione strumenti eccessivamente complessi.

Lo scopo dello sviluppo di questo piccolo programma è dunque quello di verificare la predisposizione degli strumenti scelti per l'elaborazione dei testi clinici e di fornire un approccio di base per gli obiettivi finali di raccolta dei dati per la ricostruzione degli eventi che riguardano la gestione dei pazienti fragili chirurgici dell'ospedale Humanitas Gradenigo di Torino.

4.1 Principali tool open source per la lingua italiana

Nel mondo del NLP esistono tantissimi software sviluppati appositamente per ottenere performance di elevata qualità anche nell'ambito di applicazioni specifiche, come quelle cliniche. Persino Amazon ha sviluppato un servizio per l'estrazione di dati clinici da documentazione medica, conosciuto come *Amazon Comprehend Medical*. Sistemi come questo, altamente performanti e precisi, nella maggior parte dei casi però costituiscono dei servizi a pagamento o necessitano di specifiche licenze o autorizzazioni. Le risorse gratuite disponibili online esistono, anche dedicate principalmente all'analisi dei testi medici, ma esclusivamente per la lingua inglese e comunque non per l'italiano.

A questo punto si è condotta una ricerca dedicata ad individuare dei tool open source che supportino l'italiano e che consentano di ottenere un'elaborazione di base accettabile e di qualità sufficiente per i task specifici dell'Information Extraction. Sulla base degli studi condotti in merito e della documentazione rinvenuta, sono stati selezionati i seguenti tool:

- GATE
- LinguA

- TINT
- spaCy
- Stanza

Ognuno di questi strumenti presenta delle peculiarità e delle novità rispetto ai software che di norma vengono più usati, come NLTK (*Natural Language Toolkit*), Stanford CoreNLP o OpenNLP, che sono in assoluto le piattaforme più diffuse per NLP. Nonostante alcuni di quelli elencati siano dei tool nuovi, mentre altri poco usati, possiedono comunque tutti e cinque specifici modelli per la lingua italiana, alcuni per di più sono stati sviluppati appositamente per essa ed è principalmente questo il motivo per cui sono stati selezionati per l'analisi.

Nel paragrafo successivo si descriverà la fase di analisi e valutazione che è stata eseguita per la scelta del software, mentre di seguito si riporta una breve presentazione dei tool.

4.1.1 GATE

GATE (*General Architecture for Text Engineering*) è un software toolkit open source per *Language Engineering*, capace di elaborare i testi scritti in linguaggio naturale [51]. GATE in realtà costituisce una famiglia di tool che include un ambiente di sviluppo integrato (IDE) per gli sviluppatori, una applicazione web, una libreria Java, una architettura e un processo per la creazione di servizi [51].

In particolare, l'ambiente di sviluppo integrato, chiamato *GATE Developer*, mette a disposizione l'uso di diversi componenti per NLP insieme ad un sistema di Information Extraction molto diffuso, chiamato ANNIE (*a Nearly-New Information Extraction System*), e un sistema di ulteriori plugin da installare in aggiunta quando serve [34] [51].

In generale, GATE propone una serie di task relativi al preprocessing, per cui è possibile creare delle proprie pipeline usando questi strumenti oppure utilizzare direttamente ANNIE nell'ambito dell'Information Extraction, il quale mette a disposizione i seguenti moduli [34] [51]:

- *Document Reset* per resettare le annotazioni aggiunte al documento nel caso in cui esso dovesse essere elaborato diverse volte;
- *Unicode Tokeniser* per lingue diverse dall'inglese che oltre ad effettuare la Tokenization fornisce anche le caratteristiche dei token in merito al tipo, alla lunghezza in caratteri e al tipo di carattere
- *Sentence splitter*
- *Part-of-speech tagger*
- *Gazeetter*

- *Semantic tagger* o *NE Transducer* basato su linguaggio JAPE [19], per la realizzazione di regole precise al fine di produrre ulteriori annotazioni o modificare quelle già esistenti;
- *Orthographic Coreference* o *OrthoMarcher* che relaziona le entità trovate al punto precendere ad una specifica identità.

Tutti questi moduli sono utilizzabili in generale per qualsiasi lingua, ma già a partire dal POS tagging le performance relative all'analisi dei testi in lingua italiana lasciano a desiderare, come si evencerà dall'analisi che verrà descritta nel paragrafo 4.2. Tuttavia, aggiornando le liste del Gazeetter e imparando a gestire il linguaggio JAPE è possibile ottimizzare questi strumenti per il riconoscimento delle entità che interessano.

4.1.2 LinguA

LinguA è una pipeline di annotazioni linguistiche all'avanguardia che combina algoritmi di machine learning e tecniche basate su regole per analizzare i testi in lingua italiana o in lingua inglese [52]. Il sistema è stato sviluppato da ItalianNLP Lab (*Italian Natural Processing Laboratory*) presso l'Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC-CNR) di Pisa. E' dunque un prodotto completamente italiano e sviluppato *ad hoc* per le caratteristiche della lingua italiana, nonostante supporti anche la lingua inglese.

La pipeline messa a disposizione possiede i seguenti moduli [52]:

- *Tokenization*
- *Sentence splitting*
- *Lemmatization*
- *POS tagging*
- *Depedency parsing*

Per eseguire le operazioni di POS tagging e di parsing, LinguA utilizza dei tagset specifici, rispettivamente *ISST-TANL Tagsets* e *ISST-TANLS dependency tagset*, i cui riferimenti si trovano nella bibliografia di questo documento alle voci [53] [54].

Come appare evidente dall'elenco dei moduli disponibili, LinguA non contempla le attività relative all'Information Extraction. Nonostante ciò l'applicazione web che lo supporta consente di potere scaricare l'analisi in formato CoNLL (*Conference on Computational Natural Language Learning*) in cui ogni frase è separata dall'altra tramite uno spazio bianco ed ad ogni token corrisponde una nuova riga, nella quale vengono riportate tutte le

[19] JAPE (*Java Annotation Patterns Engine*) è un linguaggio *pattern-matching* tramite il quale è possibile riconoscere espressioni standard e regolari nelle annotazioni dei documenti grazie alla definizione di un set di regole che poi verranno tradotte in codice Java [34] [51].

sue informazioni linguistiche (lemma, pos tag, *feature* morfologiche, informazioni relative alla dipendenza sintattica) [52]. L'assenza dei moduli relativi all'Information Extraction non è dunque limitante, poichè è possibile utilizzare il testo scaricato già processato per ulteriori task di NER, tramite l'impiego di altri strumenti e altri tool che lo supportino.

4.1.3 TINT

TINT (*The Italian NLP Tool*) è una pipeline sviluppata in linguaggio Java per l'elaborazione dei testi scritti in lingua italiana [58]. E' un tool completamente gratuito e open source che può essere usato come tool stand-alone, come libreria di Java o come servizio API REST [20] ed è basato su Standford CoreNLP [58].

La pipeline realizzata effettua le seguenti operazioni [58]:

- *Tokenization*
- *Sentence splitting*
- *Lemmatization*
- *POS tagging*
- *Dependency parsing*
- *Named Entity Recognition*

Come LinguA anche TINT è stato sviluppato appositamente per la lingua italiana, ma in questo caso, oltre a supportare anche i moduli per Information Extraction, è basato su uno dei software più utilizzati per il NLP, che è Standford CoreNLP.

4.1.4 spaCy

spaCy è una libreria di Python per NLP gratuita e open source, utilizzata da diverse aziende e marchi conosciuti in tutto il mondo. E' un tool talmente potente che è impossibile elencare tutte le sue *feature* e le opportunità che offre. Supporta più di 63 lingue, tra cui l'italiano e, nonostante metta a disposizione dei moduli già sviluppati ed allenati, è possibile creare delle proprie pipeline, sulla base delle applicazioni di interesse [60].

Per la lingua italiana spaCy rende disponibili tre modelli di tipo *core*, dunque preallenati, *general-purpose* e statistici realizzati con le seguenti *feature* [60]:

- *Tokenization*

[20] Una API (*Application Programming Interface*) REST (*Representational State Transfer*) è un'interfaccia di programmazione conforme ai vincoli dell'architettura REST. Risorsa online: https://www.redhat.com/it/topics/api/what-is-a-rest-api (Ultimo accesso: 2 febbraio 2021).

- *Sentence segmentation*
- *Lemmatization*
- *POS tagging*
- *Dependency parsing*
- *Named Entity Recognition*

Anche in questo caso i task dei modelli esistenti consentono già un primo livello di Information Extraction, a cui è possibile aggiungere ulteriori operazioni. Inoltre, soprattutto con la nuova versione, che tuttavia non è stata utilizzata in questo lavoro, i modelli sono ricchi anche di altri task più specifici e complessi, adatti per renderli ancora più potenti e performanti.

I tre modelli si distinguono sostanzialmente per la grandezza del corpus che è stato usato per allenarli, ma la tipologia di testi di riferimento è sempre la stessa, ovvero quella delle *news*, come si spiegherà nel dettaglio nel paragrafo 4.2.

4.1.5 Stanza

Stanza è un pacchetto open source per NLP su Python per l'elaborazione del linguaggio umano ed è stato creato dallo Standford NLP Group [61]. Il toolkit supporta 66 lingue, tra cui l'italiano, tramite l'implementazione del machine learning [61].

Come spaCy anche Stanza è una libreria potente, soprattutto grazie all'utilizzo di una pipeline basata su una rete neurale supervisionata [61].

I modelli pre-allenati sono divisi in due categorie [61]:

- *Universal Dependencies* (UD) [21] *models*, che sono allenati sulle UD *treebanks* e che comprendono i tipici task di preprocessing;
- *NER models* che supportano il Named Entity Recognition.

Purtroppo per la lingua italiana è stato sviluppato unicamente un UD model, per cui Stanza rende disponibili le seguenti attività [61]:

- *Tokenization*
- *Sentence segmentation*
- *Lemmatization*

[21] *Universal Dependencies* è un progetto cooperativo internazionale dedicato a creare dei corpus in cui venga annotata la struttura sintattica e semantica delle frasi, letteralmente chiamati *treebank*. Risorse online: https://en.wikipedia.org/wiki/Universal_Dependencies, https://en.wikipedia.org/wiki/Treebank. (Ultimo accesso: 2 febbraio 2021).

- *POS tagging & Morphological feature*
- *Dependency parsing*

In aggiunta, Stanza mette a disposizione un ulteriore modulo, definito *Multi-Word Token* (MWT) *expansion*, che può espandere un singolo token in parole multiple, caratterizzate da un legame sintattico [61]. Questo task è particolamente utile per le lingue come l'italiano in cui spesso più parole insieme in alcuni casi in realtà costituiscono un'unica entità, per cui varia la dipendenza sintattica, come ad esempio accade per le preposizioni articolate (del, dello, nella, ecc.) o i verbi pronominali (esserci, muoversi, ecc.).

Nonostante Stanza non supporti le attività di Information Extraction, in realtà esiste un'ulteriore pacchetto, chiamato *spacy-stanza* che consente di usare i modelli di StandfordNLP direttamente su spaCy ed integrarli con le altre sue funzionalità.

4.2 Valutazione delle performance dei tool su testi stardard

Per selezionare un tool con il quale gestire l'elaborazione dei testi scritti in linguaggio naturale appartenenti alle cartelle cliniche, è stata portata avanti una procedura di valutazione delle performance. In una prima fase iniziale si è deciso di lavorare con testi standard di varia natura, poichè nessuno dei tool e dei modelli è stato allenato su testi biomedicali o clinici.

E' importante sottolineare come in realtà sia Stanza, sia spaCy, sia GATE presentino dei pacchetti sviluppati nello specifico per questa categoria di documenti, ma, come era già stato accennato, purtroppo la lingua italiana non si trova tra le lingue supportate, anzi nella maggior parte dei casi l'unica lingua disponibile è l'inglese. Per tale motivo, si è deciso di effettuare una valutazione dedicata a scegliere il tool sulla base delle performance ottenute nell'ambito di tipologie di testi standard, come articoli di giornale, testi giuridici o articoli di Wikipedia. In genere questi modelli sono stati allenati anche su testi derivanti dai social network, come twitter o facebook, ma che si è scelto di non considerare per l'analisi, in modo da concentrarsi su testi semanticamente e sintatticamente più complessi, come ad esempio quelli giuridici.

Per eseguire una valutazione il più possibile uniforme ed omogenea, sono stati testati i task di base, tipici delle fasi di preprocessing, quali:

- *Tokenization*
- *Sentence splitting*
- *Lemmatization*
- *POS tagging*

- *Dependency parsing*

In aggiunta, nel caso in cui fosse presente, è stato valutato anche il modulo del Named Entity Recognition.

Le specifiche dei tool, le condizioni e gli ambienti in cui è stata condotta l'operazione di valutazione sono riassunte nella tabella 4.1.

Tool	Versione	Ambiente	Modello pre-allenato
GATE	8.6.1	GATE Developer	ANNIE
LinguA	/	Applicazione web	/
TINT	0.1	Demo online	/
spaCy	2.3.5	Jupyter Nootebook	it_core_news_lg
Stanza	1.0.0	Jupyter Nootebook	it 1.0.0

Tabella 4.1: Specifiche dei tool ed ambienti di elaborazione

A differenza di LinguA, per il quale si può unicamente utilizzare l'applicazione web, dalla quale poi è possibile scaricare il risultato, per quanto riguarda TINT invece, a causa di un problema riscontrato per l'installazione, è stato utilizzato il demo online, che comunque risulta essere particolamente performante e dettagliato, consentendo di valutare i risultati che si ottengono alla fine di ogni modulo della pipeline.

Nel paragrafo seguente ci si concentrerà sulla presentazione dei testi usati per l'analisi e sulle *feature* da valutare per i quali ognuno di essi è stato scelto.

4.2.1 Selezione dei testi dai corpora standard

Negli anni sono stati raccolti e creati parecchi corpora in lingua italiana, proprio al fine di utilizzarli per sviluppare i sistemi di NLP.

Ogni corpus è composto da diversi generi testuali e di norma, se scelto per la realizzazione di un sistema di NLP, viene diviso almeno in due set:

- *Training set*
- *Test set*

Molto spesso, soprattutto nell'ambito dei metodi di machine learning, si realizza anche un *Validation set*, a volte noto come *Development set* o *dev set*, utilizzato per valutare le caratteristiche architetturali del modello computazionale scelto e dunque per ottimizzarlo.

Per la valutazione si è deciso di servirsi di tre testi estratti dal *test set* del corpus UD Italian ISDT. Questo *treebank* fa parte di Universal Dependencies, per cui è stato

annotato secondo lo schema UD ed è stato ottenuto dalla conversione dell'ISDT (*Italian Standford Dependency Treebank*) [63].

La composizione del corpus è riportata nello schema in figura 4.1.

Original format	Source	Genre	Size in tokens	Size in sentences
TUT-CONLL	Evalita 2011 Dependency parsing	Legal texts, news articles, Wikipedia articles	101,309	3,842
ISST-TANL	Evalita 2011 Domain adaptation task	Newspaper articles	80,967	4,135
ISST-TANL	SPLeT 2012	Legal texts: European directives	6,166	260
MIDT	Several QA competitions	Questions	20,680	2,228
MIDT	Evalita 2014 Dependency parsing:test data set (partial)	News articles	7,618	304
TUT-CONLL	Parallel TUT (Italian part)	Various genres	55,942	2,131
UD	Due Parole	Simplified Italian news	24,977	1,421
UD2	New data	Various sentences	2,504	150

Figura 4.1: Composizione del corpus *UD Italian ISDT*. Immagine tratta da: https://github.com/UniversalDependencies/UD_Italian-ISDT/blob/master/README.md (Ultimo accesso: 3 febbraio 2021).

L'insieme dei testi è stato suddiviso dagli autori in *training set*, *dev set* e *test set*, dal quale sono stati estratti i tre documenti oggetto di questa valutazione. Questa scelta è stata fatta poichè il *test set* viene semplicemente utilizzato per testare il sistema, nè per allenarlo nè per ottimizzarlo. Poichè questo corpus è stato usato da alcuni dei tool selezionati per lo sviluppo dei modelli in lingua italiana, seguendo questa linea si valuta la loro capacità di generalizzare e dunque di elaborare dati non utilizzati durante l'apprendimento, evitando così di polarizzare l'analisi.

In generale, come viene riportato nella corrispondente scheda di Universal Dependencies [22], i generi dei testi contenuti nel corpus riguardano l'ambito giuridico, le news e gli articoli di Wikipedia. A tal proposito, partendo dalle singole frasi contenute nel corpus, si è deciso di estrarre un testo per ognuna di queste tre tipologie:

- Testo giuridico

> «1015. Abusi dell'usufruttuario. Se è perito un edificio e il proprietario intende di ricostruirlo con la somma conseguita come indennità, l'usufruttuario non può opporsi. Se la cosa è requisita o espropriata per pubblico interesse, l'usufrutto si trasferisce sull'indennità relativa (1000).»

[22] Disponibile: https://universaldependencies.org/#language (Ultimo accesso: 3 febbraio 2021).

- Articolo di Wikipedia

> «Nel 1467 a Perugia eseguì per conto delle suore terziarie del convento di Sant'Antonio un polittico, dove all'impostazione tardogotica voluta dalla committenza, si contrappone nella cimasa, un'annunciazione di chiaro stampo rinascimentale, che evidenzia il sapiente uso dell'arte prospettica nelle strutture architettoniche, palesando una conoscenza delle opere e dei postulati architettonici formulati qualche anno addietro da Filippo Brunelleschi e Leon Battista Alberti.»

- Articolo di giornale

> «ROMA - Nello Zaire, 34 anni fa (allora si chiamava Congo), furono trucidati 13 aviatori italiani a Kindu dov'erano stati inviati con aiuti e materiale su richiesta dell'Onu che aveva messo sotto controllo internazionale il Paese africano dopo aver ottenuto, il 30 giugno 1960, l'indipendenza dal Belgio.»

Ognuno di questi tre testi è stato scelto sulla base della complessità della struttura, della sintassi e della morfologia, nell'ottica di attenzionare le capacità dei tool nell'ambito di specifici task, come mostrato nella tabella 4.2.

Testo giuridico	Sentence splitting
	Lemmatization
	POS tagging
Articolo di Wikipedia	Lemmatization
	POS tagging
	Dependency parsing
	NER (Person, Location)
Articolo di giornale	Sentence splitting
	Dependency Parsing
	NER (Person, Location, Organization)

Tabella 4.2: Elenco dei task per cui ogni testo è stato selezionato

I risultati relativi alla Tokenization sono stati attenzionati allo stesso modo per tutti e tre i testi, poichè nessuno presenta un livello di difficoltà più elevato e poichè la qualità della procedura influenza le performance dei successivi task. Inoltre, poichè alcuni tool in esame possiedono l'attività di riconoscimento dei *multi-word token* già integrata nel

processo di Tokenization, nel caso di Stanza si è deciso di testare anche il modulo di MWT expansion e di considerarlo nella valutazione complessiva della divisione in token.

Le annotazioni effettuate dagli esperti sono state da loro eseguite manualmente e successivamente convertite in stile UD [63]. Esse riguardano le seguenti informazioni [63]:

- lemma
- UPOS (*Universal POS tags*)
- XPOS (*Treebank-specific POS tags*)
- Feature morfologiche
- Relazioni sintattiche

A titolo esemplificativo si riporta una frase del testo legale annotata in stile UD:

```
# text = 1015. Abusi dell'usufruttuario.
1 1015 1015 NUM N NumType=Card 3 nummod 3:nummod SpaceAfter=No
2 . . PUNCT FF _ 1 punct 1:punct _
3 Abusi abuso NOUN S Gender=Masc|Number=Plur 0 root 0:root _
4-5 dell' _ _ _ _ _ _ _ SpaceAfter=No
4 di di ADP E _ 6 case 6:case _
5 l' il DET RD Definite=Def|Number=Sing|PronType=Art 6 det 6:det _
6 usufruttuario usufruttuario NOUN S Gender=Masc|Number=Sing 3 nmod 3:nmod:di
SpaceAfter=No
7 . . PUNCT FS _ 3 punct 3:punct _
```

Nonostante i risultati prodotti dai cinque tool non siano tutti presentati in queste stesse modalità, la valutazione è stata effettuata confrontando l'elaborazione prodotta con le annotazioni scaricate direttamente dal corpus, la cui forma è identica a quella mostrata nell'esempio.

4.2.2 Risultato della valutazione e selezione di un tool

Tutte le risorse utilizzate ai fini della valutazione sono state scaricate e adoperate prima del 4 dicembre 2020. Il demo online di TINT e la versione di spaCy infatti sono stati aggiornati nel gennaio del 2021 e dunque sia i moduli disponibili sia le performance potrebbero essere cambiati.

Al fine di selezionare un tool, è stata condotta una valutazione sia qualitativa sia quantitiva rispetto ai modelli standard di riferimento annotati dagli esperti. In particolare, in figura 4.2 si riporta il risultato dell'analisi elaborata in merito ai task tipici delle fasi di preprocessing.

	GATE			spaCy			TINT			LinguA			STANZA		
	TESTO 1	TESTO 2	TESTO 3	TESTO 1	TESTO 2	TESTO 3	TESTO 1	TESTO 2	TESTO 3	TESTO 1	TESTO 2	TESTO 3	TESTO 1	TESTO 2	TESTO 3
TOKENIZATION	86%	75%	92%	91%	75%	90%	93%	73%	90%	93%	75%	90%	100%	100%	100%
SENTENCE SPLITTING	67%	100%	100%	100%	100%	100%	67%	100%	100%	67%	100%	100%	100%	100%	100%
LEMMATIZATION	/	/	/	73%	64%	80%	?	?	?	93%	74%	90%	98%	99%	100%
POS TAGGING	?	?	?	93%	94%	90%	86%	88%	88%	100%	85%	90%	100%	99%	100%
DEPENDENCY PARSING	/	/	/	93%	88%	70%	63%	74%	63%	?	?	?	100%	94%	95%

Risposte corrette ≥ 90% | Risposte corrette < 90% & ≥ 80 % | Risposte corrette < 80% | / Modulo non disponibile | ? Modulo non valutabile

Figura 4.2: Risultati della valutazione delle performance su testi standard

Nella tabella dei risultati in figura 4.2, la corrispondenza con il numero dei testi è la seguente:

- TESTO 1: testo giuridico
- TESTO 2: articolo di wikipedia
- TESTO 3: articolo di giornale

Il confronto è stato effettuato manualmente, calcolando il numero di risposte corrette fornite dai tool rispetto a quanto riportato nelle annotazioni, in modo da valutare anche quanto il risultato di uno step influenzi quello successivo. I risultati relativi a ciascun modulo testato indicano dunque la percentuale di risposte corrette che il sistema ha fornito considerando come riferimento l'annotazione degli esperti scaricata in stile UD. E' quindi importante fare notare come gli errori compiuti già sin dalla fase di Tokenization abbiano influenzato le performance dei successivi step. Questo dimostra l'importanza di scegliere uno strumento che garantisca una fase di preprocessing di elevata qualità, come già accennato nel capitolo precedente in merito all'uso di un sistema di NLP in ambito clinico.

Nel caso di TINT, sebbene la pipeline la comprendesse, l'operazione di Lemmatization non si è potuta valutare poichè il demo online non rendeva disponibile i lemmi riconosciuti. Inoltre, come già prennunciato durante la presentazione dei testi, non tutti i tool riportavano i risultati con le stesse modalità e gli stessi simbolismi di quelli standard, per cui la valutazione in alcuni casi è risultata lunga e complessa. A tal proposito per quanto concerne il POS tagging e il Dependency parsing rispettivamente per GATE e LinguA, si è arbitrariamente deciso di non effettuare alcun tipo di analisi, in quanto i tag utilizzati sono diversi rispetto a quelli di riferimento, per cui, per stimare correttamente le perfomance, sarebbe stato più opportuno consultare un esperto.

In generale, dopo aver testato i tool e sulla base dei risultati ottenuti, è possibile sicuramente affermare che Stanza fornisce le migliori performance in tutti i task esaminati. In realtà, la fase di elaborazione che si è rivelata fondamentale è stata proprio la MWT

expansion, ovvero la capacità del sistema di riconoscere i token composti, soprattutto i verbi pronominali e le preposizioni articolate. Gli errori compiuti da spaCy infatti nella fase di Tokenization derivano proprio dal mancato riconoscimento di quest'ultimi e ciò ha compromesso l'individuazione delle corrette relazioni sintattiche. Sempre per quanto riguarda spaCy, la divisione in token è anche in parte responsabile dell'errata assegnazione del lemma, ma non completamente, in quanto la Lemmatization risulta essere l'aspetto più critico del modello.

Lo stesso non si può affermare per TINT, che, sebbene abbia la capacità di individuare le parole composte, presenta dei moduli di POS tagging e di parsing poco soddisfacenti.

Le performance di LinguA risultano essere invece abbastanza convincenti in merito all'analisi morfologica, ma purtroppo non è possibile esprimere una valutazione oggettiva per l'analisi sintattica.

L'unico tool che è risultato veramente difficile da testare è stato GATE. Nonostante la presenza di una API possa essere comoda per la gestione del sistema, in questo caso è risultata di difficile fruizione e le modalità di presentazione dei risultati si sono rivelate difficili da confrontare con il modello standard. La documentazione esistente a riguardo è molto ricca, ma non fornisce esempi facilmente recuperabili su come usare il tool in maniera corretta. Per apprezzare tutte le potenzialità di GATE e così effettuare una valutazione esaustiva, sarebbe necessario uno studio approfondito, che nell'ambito di questo lavoro si è deciso di non condurre.

La scelta definitiva tuttavia non è stata effettuata unicamente sulla base delle considerazioni fatte fino ad ora, ma ad una analisi quantitativa si è deciso di associare anche una valutazione complessiva delle qualità dei singoli tool, contestualizzando gli errorri compiuti ed esaminando anche l'usabilità della risorsa. E' necessario allora contemplare anche la possibilità di implementare attività di Information Extraction per far fronte alle richieste e alle necessità del progetto ed effettuare una scelta anche sulla base della flessibilità degli strumenti e della disponibilità di ulteriori risorse. Si è deciso a questo proposito di analizzare, ove possibile, anche le performance relative al NER, nell'ambito delle entità comuni su cui i sistemi sono già stati allenati, quali nomi di persona, organizzazioni e luoghi.

Si riportano in figura 4.3 le metriche quantitive precision, recall ed F-measure calcolate relativamente alle entità riconosciute nei testi 2 e 3.

I testi standard di riferimento non presentavano annotazioni relative al NER, per cui è stato necessario decidere *a priori* quali dovessero essere le entità e a quale tipologia appartenessero. In particolare, i testi sono stati scelti appositamente anche per questa operazione, in modo da rendere semplice la scelta delle entità da riconoscere, anche senza l'intervento di un esperto. Le metriche sono comunque state calcolate considerando il numero di token individuati nel testo di riferimento, in modo da seguire gli stessi principi dell'analisi relativa ai task precedenti e così valutare quanto effettivamente la qualità del preprocessing possa influenzare le attività successive di Information Extraction.

	GATE		spaCy		TINT	
	TESTO 2	TESTO 3	TESTO 2	TESTO 3	TESTO 2	TESTO 3
PRECISION	0,22	0,25	1	1	1	0,75
RECALL	0,63	0,50	1	0,75	0,75	0,38
F-MEASURE	0,32	0,33	1	0,86	0,86	0,50

Figura 4.3: Risultati della valutazione delle performance del task di Named Entity Recognition su testi standard

Per costruire le confusion matrix, che è possibile ritrovare nell'appendice B e sulla base delle quali sono stati calcolati i parametri, sono state scelte le seguenti due classi:

- *Classe 0*: i token che non fanno parte di alcuna entità;
- *Classe 1*: i token che appartengono ad una delle tre entità comuni (PER, ORG, LOC).

La categoria semantica in questione può essere qualunque delle tre, poichè l'obiettivo è unicamente quello di valutare la capacità del sistema di riconoscere correttamente un'entità. I veri positivi dunque sono stati considerati come tutti quei token riconosciuti e associati all'etichetta corretta. I token individuati ma a cui è stata attribuita un'etichetta errata sono stati classificati come falsi positivi.

Dalla tabella di figura 4.3 emerge che spaCy e TINT forniscono risultati migliori rispetto a GATE, il cui problema è sostanzialmente dovuto all'elevato numero di falsi positivi che riconosce, come si può notare dai valori relativi alla precision. Le performance di spaCy si dimostrano tuttavia le migliori, sia relativamente alla sensitività sia al valore predittivo positivo, come rivelano anche i valori elevati dell'F-measure. spaCy manifesta buone capacità di riconoscimento delle entità nei testi e un buon bilanciamento tra falsi negativi e falsi positivi.

Con queste premesse dunque si è deciso di operare una ulteriore valutazione con i testi clinici, usufruendo di un semplice programma sviluppato per Information Extraction tramite l'utilizzo del connubbio *spacy-stanza*. In particolare l'ambiente spaCy fornisce la possibilità di scaricare un pacchetto integrato con Stanza e quindi di potere sfruttare i moduli di entrambi i tool per realizzare il proprio sistema. Ciò consente di utilizzare il modello di Stanza per la lingua italiana e ottenere così una fase di preprocessing di elevata qualità, ma al contempo di sfruttare i moduli di spaCy per l'estrazione delle entità di interesse dai testi clinici.

4.3 Analisi qualitativa delle performance su testi clinici

L'analisi quantitativa e qualitativa eseguita su testi standard ha determinato la scelta di due strumenti, ovvero spaCy e Stanza, utilizzabili tramite un pacchetto integrato spacy-stanza, gestibile in linguaggio Python. L'obiettivo è dunque quello di sfruttare l'elevata qualità del preprocessing fornito da Stanza ed integrare i moduli relativi all'Information Extraction che spaCy rende disponibili per la lingua italiana.

I testi standard tuttavia sono stati scelti appositamente sulla base della tipologia di documenti su cui i tool testati sono stati allenati e sviluppati. Non è detto dunque che le performance si mantengano soddisfacenti anche per i testi clinici, caratterizzati da un linguaggio specifico e spesso complesso. Si è deciso di condurre a tal propositvo un'ulteriore fase di valutazione tramite lo sviluppo di una pipeline pensata *ad hoc* per una determinata categoria di testi clinici. Non avendo a disposizione un buon corpus clinico, i testi per l'analisi sono stati estratti dalle cartelle messe a disposizione e sono stati convertiti in formato elettronico manualmente, poichè la documentazione fornita deriva da scansioni effettuate su materiale cartaceo.

Una volta selezionata la categoria di documenti da analizzare, sono state individuate le entità di interesse ed è stato sviluppato il codice di elaborazione in linguaggio Python.

Lo scopo di questa ulteriore analisi è, dunque, *in primis* quello di valutare le performance ottenute dalla combinazione dei due tool, spaCy e Stanza, nell'ambito dell'elaborazione dei testi clinici e, allo stesso tempo, di verificare se anche un sistema semplice basato su metodi rule-based possa fornire dei risultati accettabili e sulla base di ciò o operare un'ottimizzazione dello stesso o prendere in considerazione modelli di Information Extraction più complessi come quelli basati sul machine learning.

4.3.1 Selezione dei testi clinici

In ambito clinico, avere a disposizione un corpus in italiano non è facile per diversi motivi. La documentazione usata in altri lavori e in altre ricerce o è stata realizzata appositamente per gli scopi del progetto o è stata messa a disposizione da aziende sanitarie che hanno concesso l'autorizzazione all'utilizzo dei dati. Esistono delle fonti di testi in inglese, ma per ovvi motivi non possono essere utilizzate, poichè anche eventualmente optare per una traduzione richiederebbe l'ausilio di un esperto.

A causa della condizione pandemica da COVID-19 che ha interessato il mondo sanitario, non è stato possibile avere a disposizione ulteriori cartelle con le quali potere realizzare un corpus sulla base del quale portare avanti un'operazione di valutazione delle performance più dettagliata.

Per far fronte a queste difficoltà, si è deciso di selezionare una tipologia di referti presenti all'interno delle sei cartelle cliniche messe a disposizione dall'ospedale Humanitas

Gradenigo di Torino e di individuare specifiche entità comuni all'interno degli stessi. La scelta è stata fatta dopo un'attenta analisi di tutta la documentazione scritta in linguaggio naturale delle cartelle cliniche. I diari clinici e i referti delle consulenze specialistiche, sebbene contengano informazioni utili, non sono stati presi in considerazione in questo caso specifico perchè scritti a mano dai medici. L'interpretazione della scrittura e le operazioni di trascrizione hanno influenzato la scelta, poichè, oltre al lungo lavoro manuale che sarebbe stato necessario fare, le difficoltà relative alla comprensione del testo avrebbero potuto generare ambiguità ed errori e dunque un'analisi non realistica.

La tipologia di referto è stata invece scelta poichè in ogni cartella è stato individuato almeno un referto relativo all'*RX torace*, in alcuni casi anche più volte per uno stesso paziente. Da ciò si è dedotto che l'esame radiologico del torace costituisce una pratica comune per questa tipologia di pazienti e dunque un evento usuale nell'ambito della gestione organizzativa.

Dalle sei cartelle cliniche sono stati estratti otto referti, alcuni dei quali contengono anche valutazioni su altre tipologie di radiografie:

- Testo 1

  > «Indagine eseguita nel solo **radiogramma AP** consentito a **paziente supino**. Sfumato **addensamento parenchimale** sovradiaframmatico a destra, di possibile significato flogistico disventilativo, meritevole di correlazione clinico - laboratoristica. Minimo impegno del **seno costofrenico** omolaterale di significato versamentizio. **Ombra cardio-vasale** di dimensioni ai limiti superiori della normalità compatibilmente con il decubito; **aorto-sclerosi**. Ai radiogrammi che è stato possibile eseguire, corticale a livello del collo femorale, in sottocapitata, possibile espressione di frattura ingranata. Rapporti articolari conservati.»

- Testo 2

  > «Esame del torace eseguito a **paziente supino**, nella sola **proiezione antero-posteriore** e confrontato con il precedente del **28/11/2017**. Non **lesioni parenchimali** addensanti bilateralmente. Netto ampliamento dell'**ombra cardiaca** e divaricazione della carena tracheale. **Sclero ectasia aortica**. Frattura composta transtrocanterica; rapporti articolari conservati. Estese calcificazioni vascolari.»

- Testo 3

  > «L'esame è stato eseguito a **paziente seduta**. Non si osservano **lesioni parenchimali** acute a focolaio nei campi esplorabili. É presente **versamento pleurico** basale bilaterale. Si documenta la presenza di marcato ingrandimento delle camere cardiache. I restanti reperti sono sostanzialmente sovrapponibili al precedente controllo del **15.1.19**.»

- Testo 4

  «Esame eseguito in **AP**, **paziente supina**. Non sono presenti **lesioni pleuriche o parenchimali** addensanti. L'**ombra cardiaca** ha forma e dimensioni nei limiti di norma. Lieve accentuazione, vascolare, delle ombre ilari; **aorto-sclerosi**. I **seni costofrenici** sono liberi.»

- Testo 5

  «Esame eseguito al **letto del paziente**. Non lesioni acute a focolaio nel parenchima polmonare esplorabile. **Ombra cardiaca** mal valutabile.»

- Testo 6

  «Esame del torace eseguito a **paziente supina**, nella sola **proiezione antero-posteriore**. Diffusa velatura alveolo-interstiziale lobare superiore dx di verosimile natura infiammatoria. Accentuazione della trama interstiziale su entrambi gli ambiti. Netto sollevamento dell'emidiaframma di destra. **Sclero ectasia aortica**. Trachea a decorso scoliotico dx convesso. Frattura transtrocanterica scomposta con distacco del piccolo trocantere. Risalita del moncone distale. Estese calcificazioni vascolari. Osteoporosi diffusa. Depressione a lente biconcava del soma di D12 verosimilmente da frattura. Lieve deformazione della limitante somatica superiore di L2 e discopatie plurime.»

- Testo 7

  «Reperto scarsamente modificato rispetto al controllo TC del 12 us persistendo **addensamento parenchimale** lobare superiore dx, in prima ipotesi di significato fibrotico, e sfumati **addensamenti parenchimali** controlaterali di aspetto flogistico, più evidenti in sede perilare. **Ombra cardiaca** nei limiti. **Aorto - sclerosi**. Esame eseguito in **clinostatismo**.»

- Testo 8

  «**Addensamento parenchimale** al I o medio, di più verosimile significato flogistico. Utile ricontrollo dopo adeguata terapia antibiotica Non segni di significativo **versamento pleurico** in atto. **Ombra cardiaca** di dimensioni nei limiti della normalità; aorta toracica allungata e scoliotica con calcificazioni parietali.

  Coprostasi colica a destra e nello scavo pelvico, in presenza di catenella chirurgica in pregresso CA del colon operato. Sovradistensione gassosa di ans intestinali al fianco di sinistra. Nel radiogramma in ortostasi alcini livelli idroaerei a sinistra in assenza di falde di aria libera in addome.»

Le frasi dei testi che sono relative ad altri esami radiologici, come quello del femore o dell'addome, sono state mantenute, poichè ciò consente di valutare meglio la capacità del sistema di individuare solo le entità di interesse e poichè in una applicazione reale è importante ridurre al minimo le attività manuali dell'operatore, che in questo particolare caso dovrebbe selezionare esclusivamente le parti che fanno riferimento al referto radiologico del torace.

In ognuno di questi testi, i pattern evidenziati rappresentano le entità comuni riscontrabili più volte, che possono trovarsi nei vari referti anche scritte in modo leggermente diverso, ma che condividono la stessa tipologia di informazione, come la presenza delle date o la posizione del paziente durante l'esame. Non si tratta dunque prettamente di un'attività di Named Entity Recognition, ma, come si vedrà nel paragrafo successivo, di *pattern matching*.

Ciascun testo è stato utilizzato in forma anonima, in ottemperanza al GDPR, ed è stato inizialmente convertito in file di testo a partire dalle scansioni. La conversione è stata eseguita tramite l'utilizzo di un servizio gratuito web-based, chiamato *Online OCR* [23], che ha consentito di convertire file pdf scansionati in documenti di testo in formato txt.

Una volta convertiti i documenti, è stata eseguita una revisione manuale dei testi, in modo da assicurarsi della corretta corrispondenza con il file sorgente.

4.3.2 Sviluppo di una custom pipeline

Per la scrittura del codice e l'elaborazione dei testi selezionati, è stata utilizzata la piattaforma PyCharm Community Edition 2020.3, in cui sono state scaricate le stesse versioni dei pacchetti di spaCy e di Stanza utilizzate nella prima parte del processo di valutazione delle performance dei tool e la versione 0.2.4 di spacy-stanza. Il pacchetto integrato può essere usato dopo aver importato la libreria di spaCy ed è stato sfruttato per estrarre il modulo di Stanza per la lingua italiana.

In figura 4.4 si riporta il flusso delle operazioni scelte per l'elaborazione dei testi oggetto dell'analisi e per il riconoscimento delle entità individuate al paragrafo precedente.

La pipeline è stata realizzata con le tipiche fasi di preprocessing viste in precedenza, ad esclusione del parsing che non si è rivelato necessario per l'implementazione dei moduli successivi.

L'operazione di Sentence Segmentation è integrata nella fase di Tokenization di Stanza ed è stata aggiunto a questo livello anche il modulo di Multi-word token expansion, che si era rivelato fondamentale durante la prima valutazione del tool. I task di Lemmatization e POS tagging vengono elaborati dal pacchetto italiano di Stanza, mentre le successive operazioni relative proprio all'Information Extraction vengono rese disponibili da spaCy.

[23] Disponibile: https://www.onlineocr.net/it/service/about (Ultimo accesso: 8 febbraio 2021).

Figura 4.4: Flusso delle attività per lo sviluppo di un sistema di elaborazione dei testi clinici

In particolare, l'*Entity Matcher* è un modulo che è stato creato appositamente per questa pipeline e comprende l'uso del *Matcher* di spaCy.

Il Matcher è una classe che è in grado di individuare nei testi sequenze di token specifiche, basandosi appunto su un set di regole definite *a priori* [60]. La prima attività che dunque è stata svolta è relativa alla definizione dei pattern e alla creazione di una *pattern list*. I pattern vengono realizzati manualmente dall'operatore tramite la definizione di espressioni standard o regolari, caratterizzate non solo dal vocabolo che si vuole cercare ma anche dalle sue informazioni lessicali e morfolgiche.

Si riporta di seguito un esempio di dichiarazione del pattern dedicato all'individuazione nei referti delle espressioni relative alle lesioni:

```
[{"LEMMA": "lesione"}, {"POS": "ADJ", "OP": "?"}]
```

In questo esempio, il pattern indica che la parola *lesione* è il lemma, per cui in realtà può essere considerata anche al plurale, e che deve essere seguita al massimo da un solo aggettivo (simbolo "?"). In questo modo si sfrutta la versatilità del metodo del *pattern matching*, grazie al quale si possono individuare più entità con un unico pattern. Estendendo la ricerca è possibile infatti anche rintracciare più di un aggettivo che segue la parola "lesione", così da riconoscere anche espressioni più complete ed estrarre un contenuto informativo più ricco.

E' ovvio che, con queste premesse, il pattern può essere costruito in diversi modi, proprio sulla base di quello che si vuole riconoscere. In alcuni casi è importante dichiarare nello specifico quali vocaboli si cercano, in altri invece basta indicare il ruolo della parola.

L'Entity Matcher è stato realizzato grazie anche all'utilizzo del modulo di Named Entity Recognition di spaCy, che rende disponibile le funzionalità di ricerca delle entità

nei testi. Quando il modulo Entity Matcher viene richiamato nella pipeline, a partire dalla lista dei pattern viene attuata un'operazione di ricerca nel testo. Una volta trovato il matching, si realizza quello che nella documentazione di spaCy viene definito *span*. Lo span è una slice del documento, ovvero una frazione del testo caratterizzata da un token iniziale, uno finale e una label che gli viene associata per l'identificazione.

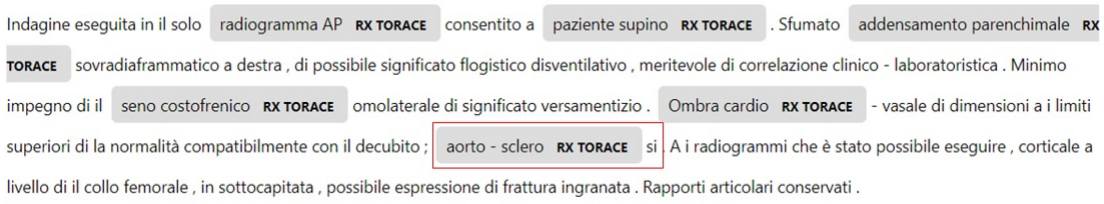
Il modulo NER è necessario poichè a questo punto inserisce lo span tra le sue entità e restituisce il documento etichettato, come si potrà apprezzare nei risultati riportati in figura 4.4 al paragrafo successivo.

4.3.3 Risultati dell'analisi

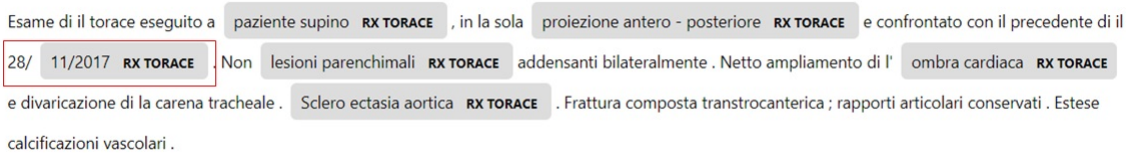
L'analisi che si è deciso di condurre ha lo scopo di valutare in modo qualitativo le performance ottenute, al fine di capire in generale la predisposizione dello strumento scelto per l'elaborazione dei testi clinici. Tale decisione è stata presa per diversi motivi. Innanzitutto i testi utilizzati non sono stati annotati da un esperto, per cui non esiste un testo standard di rifermento sulla base del quale calcolare le metriche, ma piuttosto esse sarebbero state calcolate rispetto ad un'analisi soggettiva. Inoltre, anche i pattern non sono stati costruiti da un esperto, per cui basterebbe variare un dettaglio rispetto anche solo ad una espressione e ciò potrebbe cambiare del tutto i risultati. Questo dimostra come soprattutto nell'ambito dei metodi basati su regole è fondamentale l'ausilio di un figura professionale specializzata.

Per queste motivazioni, l'analisi è dedicata a capire in generale la predisposizione di queste due librerie di Python per l'elaborazione dei testi clinici.

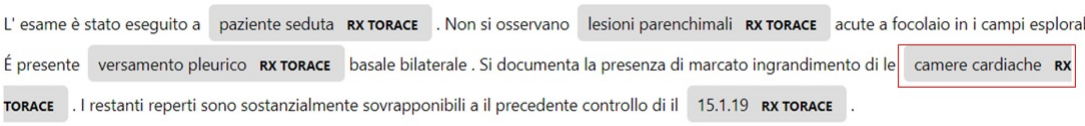
In figura 4.4 si riportano gli output ottenuti dalla pipeline per ogni testo dato in input.



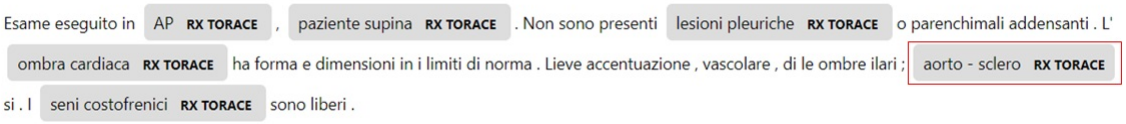
(a) *Testo 1*



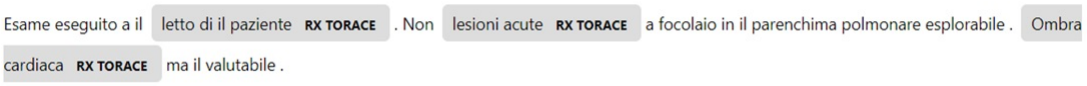
(b) *Testo 2*



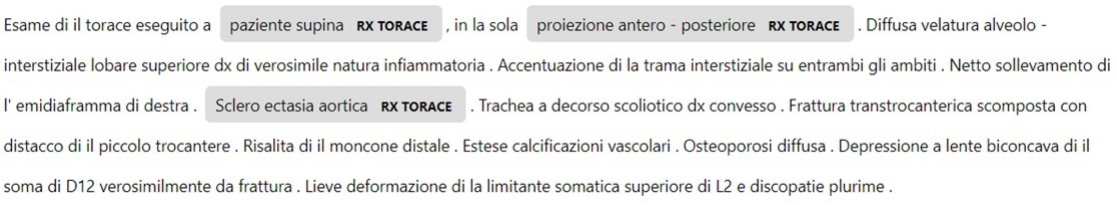
(c) *Testo 3*



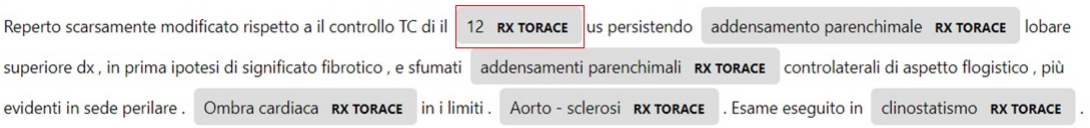
(d) *Testo 4*



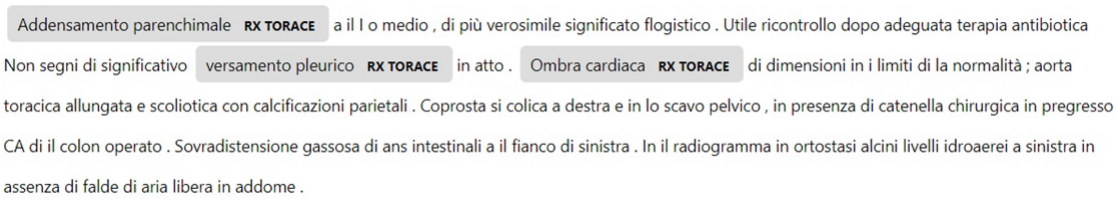
(e) *Testo 5*



(f) *Testo 6*



(g) *Testo 7*



(h) *Testo 8*

Figura 4.4: Risultati dell'elaborazione dei testi con la *custom* pipeline

Sempre per i motivi definiti sopra, non sono state scelte delle label specifiche per ogni tipologia di entità, ma tutte le espressioni trovate sono state etichettate come "RX torace", che è il nome che è stato affidato a questa determinata lista dei pattern.

Dai risultati ottenuti è possibile notare come la maggior parte delle entità vengano riconosciute correttamente. Questo è dovuto all'elevata qualità del preprocessing che identifica i lemmi e associa i POS tag corretti.

A causa della natura a volte schematica ed essenziale delle frasi, è ancora più importante avere a disposizione dei task di base altamente performanti e capaci di riconoscere soprattutto i token in modo corretto. Infatti, nei casi in cui le espressioni non vengano identificate nel modo giusto, l'errore è dovuto principalmente alla Tokenization, che separa ad esempio i diversi componenti di una data (Testo 2) o riconosce erroneamente un multi-word token (Testi 1 e 4).

Costruire i pattern, come già specificato prima, è un'operazione delicata, che deve essere fatta con precisione da un esperto o comunque con l'ausilio di uno studio approfondito, per evitare riconoscimenti mancati o errati (Testi 3 e 7).

E' importante sottolineare che i testi scelti costituiscono un livello di difficoltà piuttosto elevato rispetto alla documentazione su cui sono stati allenati i modelli utilizzati, sia per la ricercatezza del vocabolario, sia per l'utilizzo di abbreviazioni e sinonimi. La bontà dei risultati inoltre, sebbene metta in luce che il modello di Stanza per la lingua italiana possa essere tranquillamente utilizzato su testi clinici, è anche in parte dovuta all'uso di un rule-based method per l'attività di Information Extraction, tramite il quale si dichiara nello specifico cosa bisogna ricercare.

L'analisi dunque mostra che il modulo per la lingua italiana di Stanza può essere utilizzato per l'elaborazione dei testi clinici, poichè consente di ottenere un preprocessing accettabile e di qualità sufficiente per l'implementazione dei task successivi. Inoltre, poichè alla base dei moduli vi è una rete neurale, è possibile effettuare un training sui propri dati, in modo da migliorare ulteriormente le performance.

Per quanto riguarda spaCy, in questa valutazione si è fatto uso solo di due delle tantissime risorse che il tool mette a disposizione per le operazioni di Information Extraction, per cui, tramite uno studio maggiormente approfondito e facendo riferimento agli obiettivi di utilizzo, è possibile sviluppare un sistema allenato e specifico per i testi clinici, sfruttando anche moduli più complessi se è necessario o ottimizzando quanto già prodotto in questo lavoro, per ad esempio identificare anche le relazioni tra le entità rinvenute o delle stesse con altri elementi, come le negazioni o i riferimenti temporali.

4.4 Prospettive ed evoluzioni future

Lo studio condotto nell'ambito del NLP è nato con l'intento di realizzare un sistema che fosse in grado di analizzare i testi clinici scritti in linguaggio naturale contenuti all'interno delle cartelle cliniche dei pazienti dell'ospedale Humanitas Gradenigo di Torino. L'interesse principale era quello di estrarre dai documenti le informazioni utili per la ricostruzione degli eventi e delle attività che riguardano la gestione e la cura dei pazienti fragili chirurgici, così da consentire la realizzazione di un PDTA grazie all'implementazione delle tecniche di process mining sui dati raccolti.

Rimanendo dunque ancorati a questi obiettivi, partendo da quanto elaborato in questa tesi, il passo successivo sarebbe quello di progettare e programmare un sistema di NLP definitivo per Information Extraction.

E' chiaro che, nell'ottica di non volere sviluppare un codice generico, come ottimizzare la pipeline già costruita dipende esclusivamente dalle infomazioni che interessano e che si vogliono estrarre.

Sulla base di questi dati dunque, la scrittura delle regole potrebbe essere un metodo adeguato e si potrebbe affiancare anche l'utilizzo di una look-up list specifica. In tal caso sarebbe necessario avvalersi della consulenza di esperti per l'individuazione delle informazioni da estrarre e a quel punto costruire con maggior dettaglio i pattern per la ricerca di particolari espressioni, considerando anche eventualmente la presenza o meno di avverbi di negazione. Al contempo, l'utilizzo di un vocabolario potrebbe rendere più puntuale e sicura la ricerca dei termini specifici, come quelli relativi ai sintomi, alle diverse parti anatomiche o ai farmaci.

Per il vocabolario in genere si può anche fare uso delle versioni in italiano di SNOMED, di UMLS o di altre fonti disponibili, in dipendenza di quello che si vuole cercare nei testi. Per essere più specifici, avendo a disposizione un corpus di testi della stessa tipologia di quelli da analizzare, un vocabolario si potrebbe realizzare direttamente con Stanza e spaCy, elaborando ogni testo con i task base tipici del preprocessing, attuando operazioni di *lowercasing* e di rimozione delle stopword e della punteggiatura, mantenendo le *feature* relative al lemma, al POS tag e alle caratteristiche morfologiche per ciascun token.

Metodi più complessi e più potenti invece potrebbero rivelarsi necessari nel caso in cui non si vogliano esclusivamente estrarre delle specifiche entità definite *a priori*, ma piuttosto individuare dei concetti complessivi che riassumano l'esito del referto o della visita. In quest'ultimo caso si potrebbe allenare il modello statistico di spaCy relativo al NER con testi annotati, in modo da fornigli la capacità di riconoscere diverse categorie di entità, ed associargli una funzione in grado di identificare espressioni di negazione o di riferimenti temporali e di individuare il legame sintattico con le entità principali riconosciute, tramite l'utilizzo del Dependency parsing.

Le strade percorribili sono dunque varie e la scelta dipende dall'applicazione finale e dalla tipologia di dati e di informazioni che si vogliono ricavare dall'analisi dei testi clinici elaborati. Le potenzialità dei due tool scelti sono comunque tante e, sebbene il loro utilizzo richieda la capacità di sapere programmare in Python, la possibilità di sfruttare i moduli predefiniti integrandoli in un programma complessivo scritto appositamente per i propri scopi finali rappresenta il loro vero vantaggio.

Conclusioni

L'obiettivo di questa tesi era quello di fornire metodi e strumenti dedicati all'organizzazione e all'archiviazione dei dati necessari per la definizione di un PDTA per pazienti fragili chirurgici.

Raccogliere i dati utili per la ricostruzione dei processi e degli eventi che interessano l'azienda sanitaria, in questo caso l'ospedale Humanitas Gradenigo di Torino, nell'ambito della gestione di questo specifico problema di salute risultava essere un passo fondamentale per la successiva applicazione delle tecniche di process mining, tramite le quali, nel proseguo di questo lavoro, si costruirà il modello.

L'approccio che si è scelto di intraprendere ha visto concentrare questa ricerca in due diversi contesti:

1. sviluppo di un'applicazione a supporto della creazione di un database per la memorizzazione dei dati strutturati provenienti dalle cartelle cliniche;

2. studio delle tecniche di Natural Language Processing per l'estrazione di informazioni a partire dai dati non strutturati delle cartelle cliniche.

La prima parte del lavoro ha prodotto dunque come risultato la creazione di un'applicazione che consenta l'inserimento dei dati all'interno del database. Si è seguito a questo scopo una progettazione strutturata, in modo da rendere l'interfaccia grafica facilmente fruibile anche da altri utenti, che, nell'evoluzione di questo lavoro, si occuperanno delle operazioni di raccolta delle informazioni e di applicazione delle tecniche di process mining.

Al fine di avvalersi anche dei dati non strutturati contenuti nelle cartelle cliniche dei pazienti di interesse, si è proseguito con un'analisi dedicata a valutare le potenzialità delle tecniche di NLP per l'estrazione di ulteriori informazioni. Lo studio ha condotto alla scelta di due tool open source utilizzabili in modo integrato tramite linguaggio Python per la creazione di un sistema pensato per il riconoscimento di entità specifiche all'interno della documentazione medica scritta in lingua italiana. L'aver ideato inoltre una semplice pipeline per procedere ad una ulteriore verifica di queste due risorse ha dimostrato come, per gli scopi di questo progetto, non sia necessario ricorrere allo sviluppo di nuove e complesse soluzioni per ottenere risultati adeguati, ma studiando in maniera approfondita i moduli già disponibili per NLP ed integrandoli è possibile realizzare sistemi altamente performanti.

Il limite principale di entrambe le parti della tesi riguarda la mancanza di fonti e di informazioni sufficienti per l'ottimizzazione degli strumenti. Nel primo caso, tramite le sei cartelle cliniche messe a disposizione si è avuta la possibilità, anche se in maniera limitata, di testare e di conseguenza migliorare l'applicazione, ma non si è potuto procedere con la riorganizzazione delle attività e l'implementazione dei metodi per la costruzione del PDTA.

Per quanto riguarda il NLP, il programma realizzato non risulta specifico per gli scopi del progetto, per cui, nonostante sia stato dimostrato che, fornendogli delle regole scritte in modo corretto, sia in grado di individuare le espressioni e le entità ricercate, queste non corrispondono alle informazioni che sarebbe necessario estrarre al fine di ricostruire i processi.

L'evoluzione di questo studio dunque può essere identificato con la progettazione di un codice definitivo e specifico per la raccolta delle informazioni aggiuntive contenute nei testi clinici delle cartelle e con la creazione di un database complessivo, che racchiuda tutti i dati utili. Quali metodi utilizzare per la creazione del sistema di NLP clinico dipende dalla consistenza e dalla tipologia di dati che interessa estrarre.

Rispetto al progetto finale, l'applicazione delle tecniche di process mining e la revisione del modello costituiscono poi le prospettive future principali.

In aggiunta, soprattutto per consentire l'utilizzo anche ad utenti che non hanno conoscenza nell'ambito della programmazione o in generale della gestione di questi strumenti, ulteriori progressi potrebbero essere raggiunti progettando e costruendo un sistema unico, che non solo consenta il caricamento dei dati che servono, ma che sia in grado di elaborare i testi scritti in linguaggio naturale fornitigli in input, di estrarre le informazioni necessarie e di memorizzare tutto in un'unica base dati.

Adattando il tipo di informazioni e di dati da raccogliere, un approccio di questo tipo potrebbe rivelarsi utile in tutti quei casi in cui è necessario rielaborare la documentazione clinica per la realizzazione di un PDTA, approfittando soprattutto dell'impiego sempre più consistente che le aziende sanitarie stanno facendo della digitalizzazione delle cartelle e dei fascicoli clinici.

In conclusione, dunque, è importante spiegare che il percorso di tesi da affrontare in origine riguardava l'applicazione delle tecniche di process mining e la realizzazione di un PDTA, ma, a causa dei problemi dovuti alla pandemia da COVID-19, non è stato possibile avere a disposizione i dati necessari per portare avanti il lavoro. Nonostante siano sopraggiunte queste difficoltà, sono comunque stati ottenuti importanti risultati in merito alle modalità di estrazione e raccolta dei dati. Infatti, questa tesi vuole rappresentare un'analisi esaustiva relativamente ai metodi e agli strumenti che serviranno nelle fasi successive di aggregazione ed elaborazione delle informazioni, non solo a partire da dati strutturati, ma elaborando anche quelli non strutturati, garantendo così la definizione di fondamenta robuste e ricche per la creazione di un PDTA efficace.

Appendice A

Interfacce grafiche

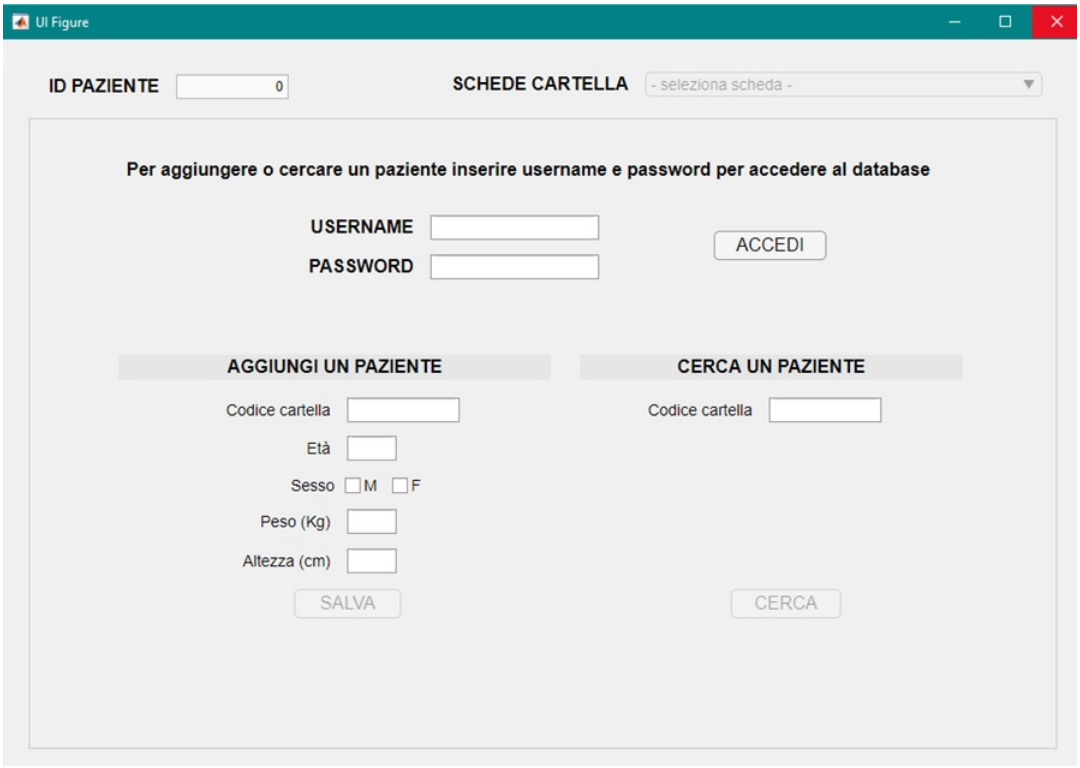


Figura 1: *Anagrafica*

Figura 2: *Anamnesi, obiettività e piano di cura: Anamnesi*

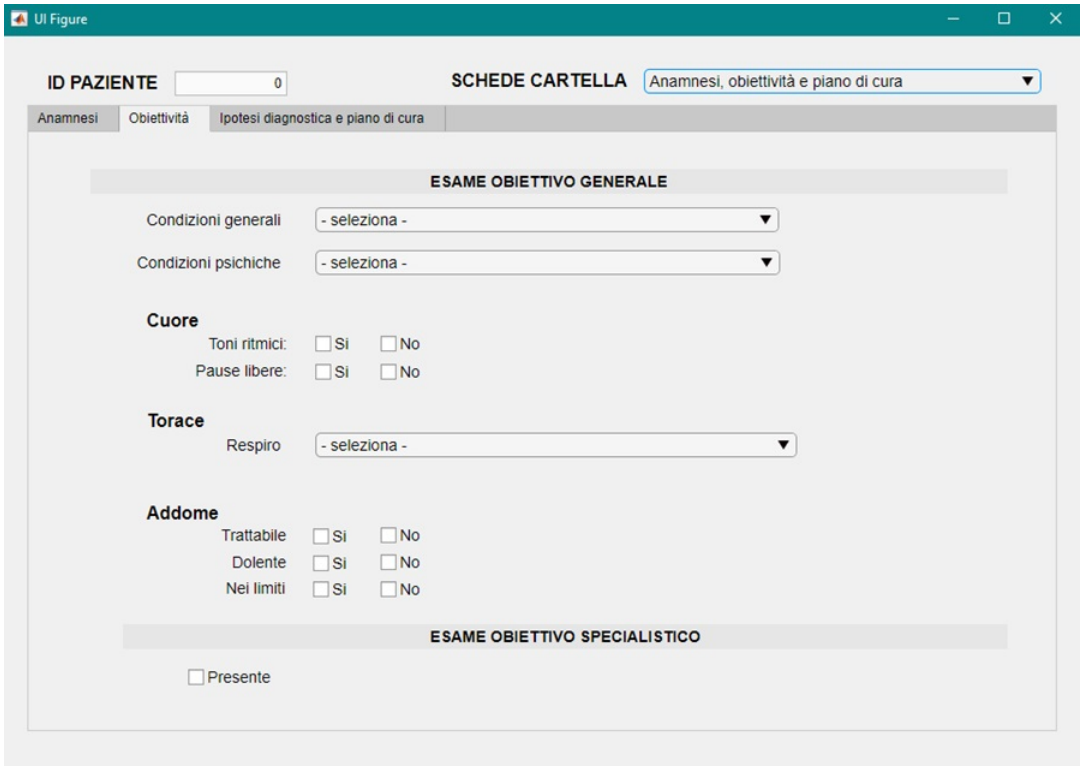


Figura 3: *Anamnesi, obiettività e piano di cura: Obiettività*

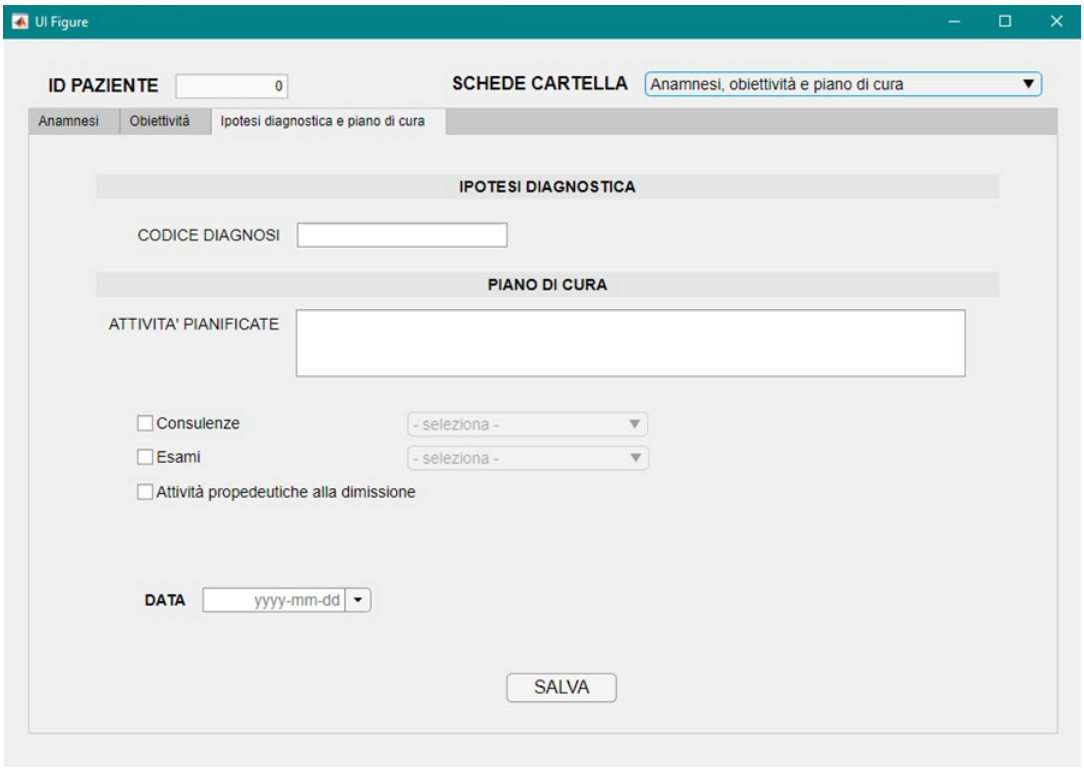


Figura 4: *Anamnesi, obiettività e piano di cura: Ipotesi diagnostica e piano di cura*

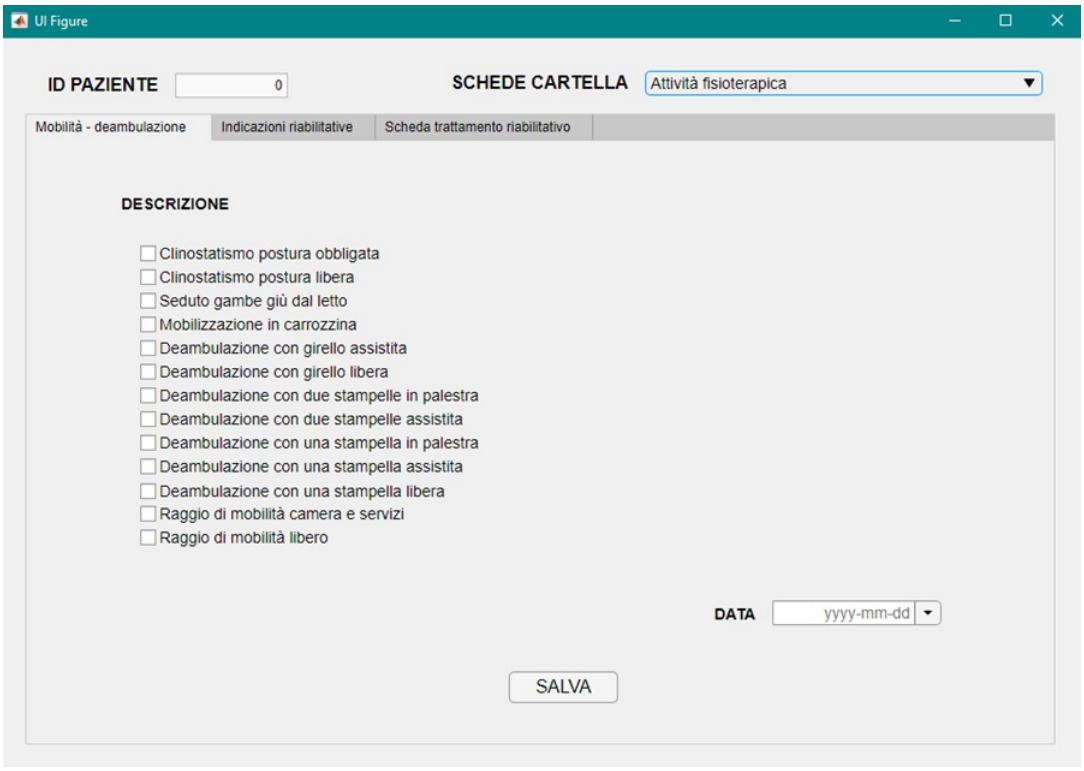


Figura 5: *Attività fisioterapica: Mobilità - deambulazione*

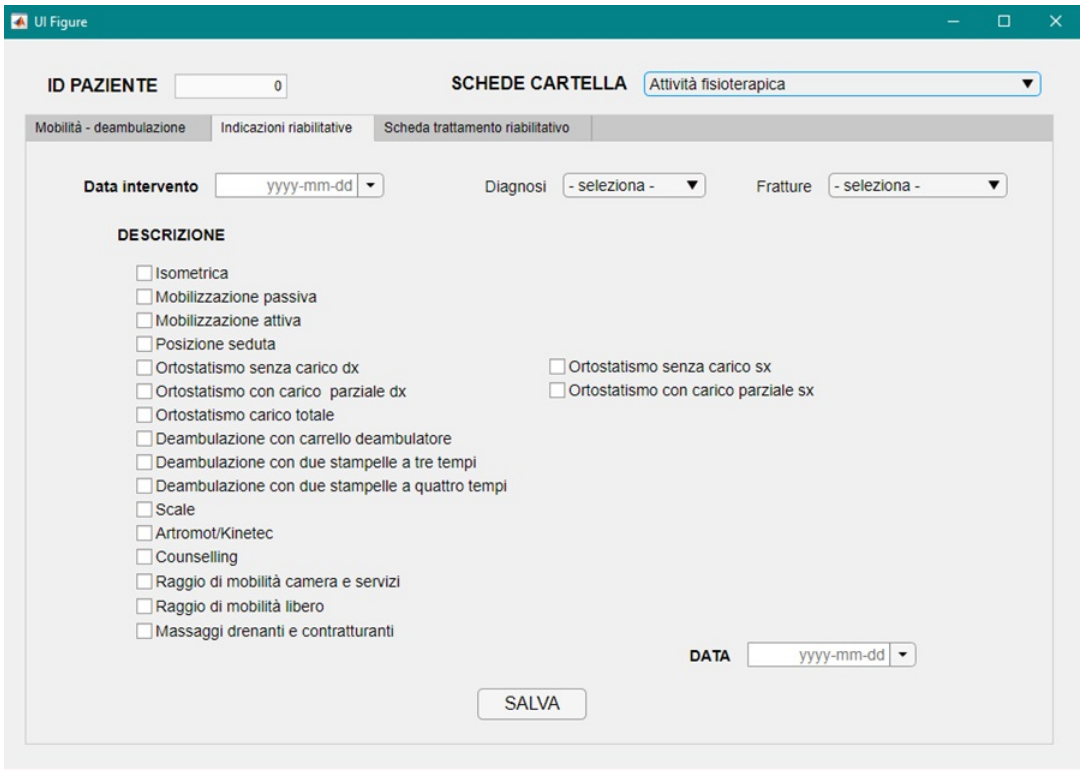


Figura 6: *Attività fisioterapica: Indicazioni riabilitative*

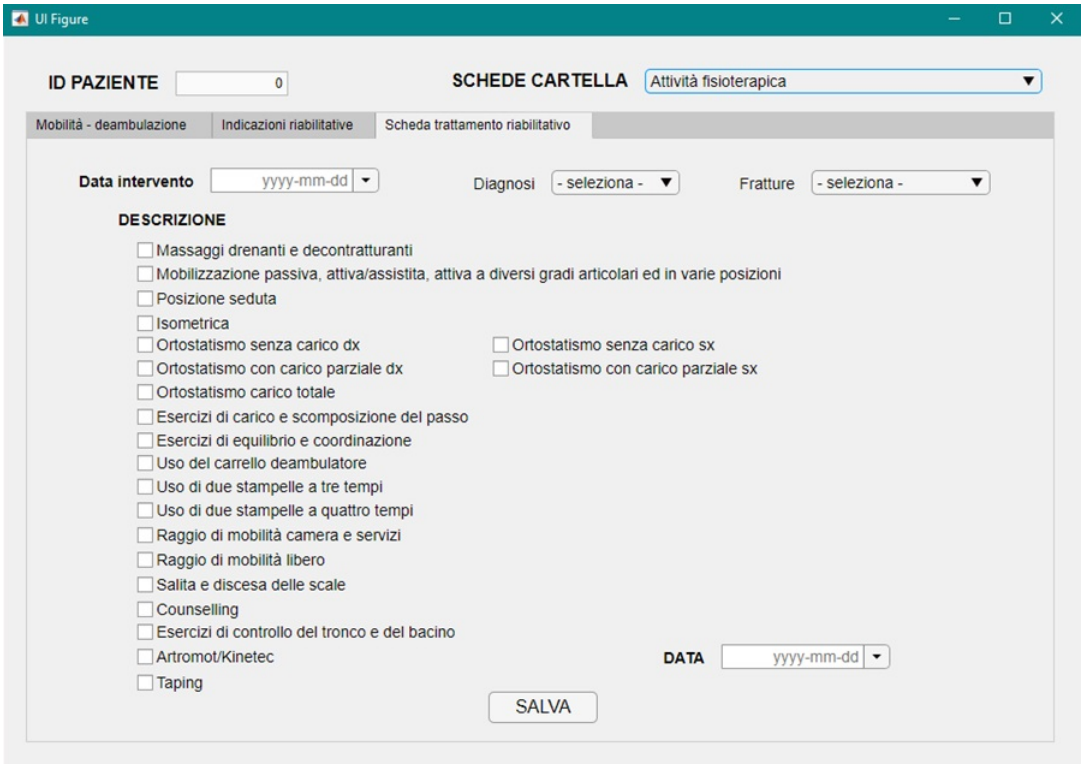


Figura 7: *Attività fisioterapica: Scheda trattamento riabilitativo*

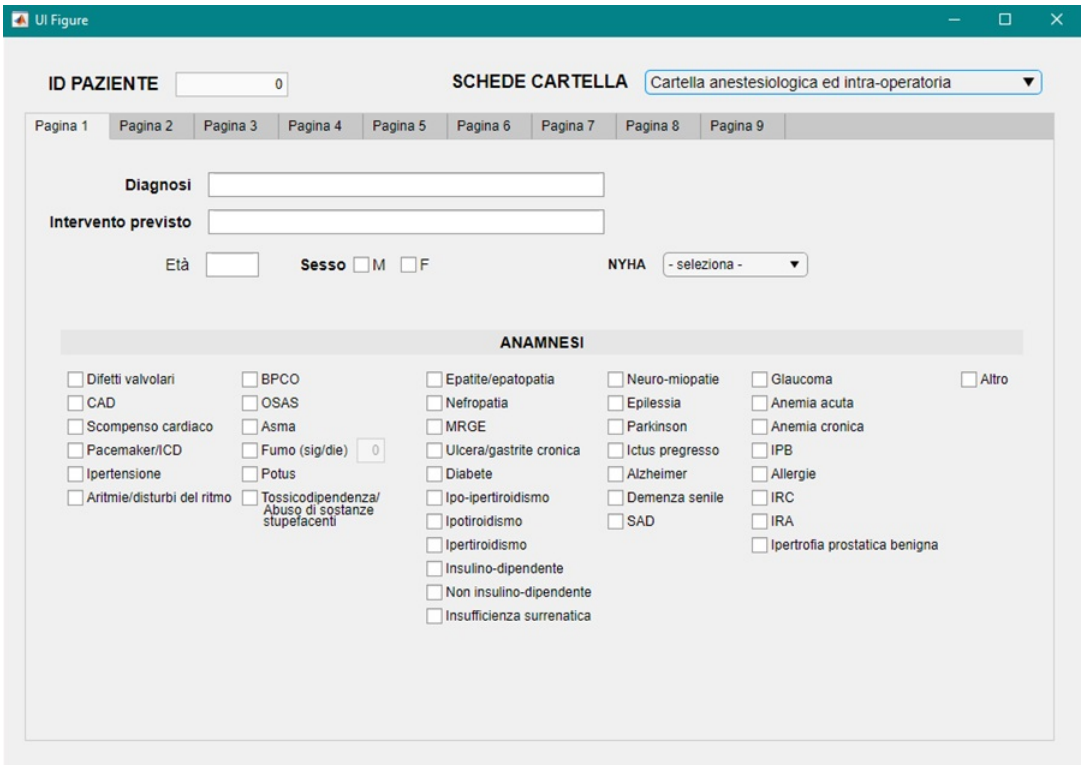


Figura 8: *Cartella anestesiologica ed intra-operatoria: Pagina 1*

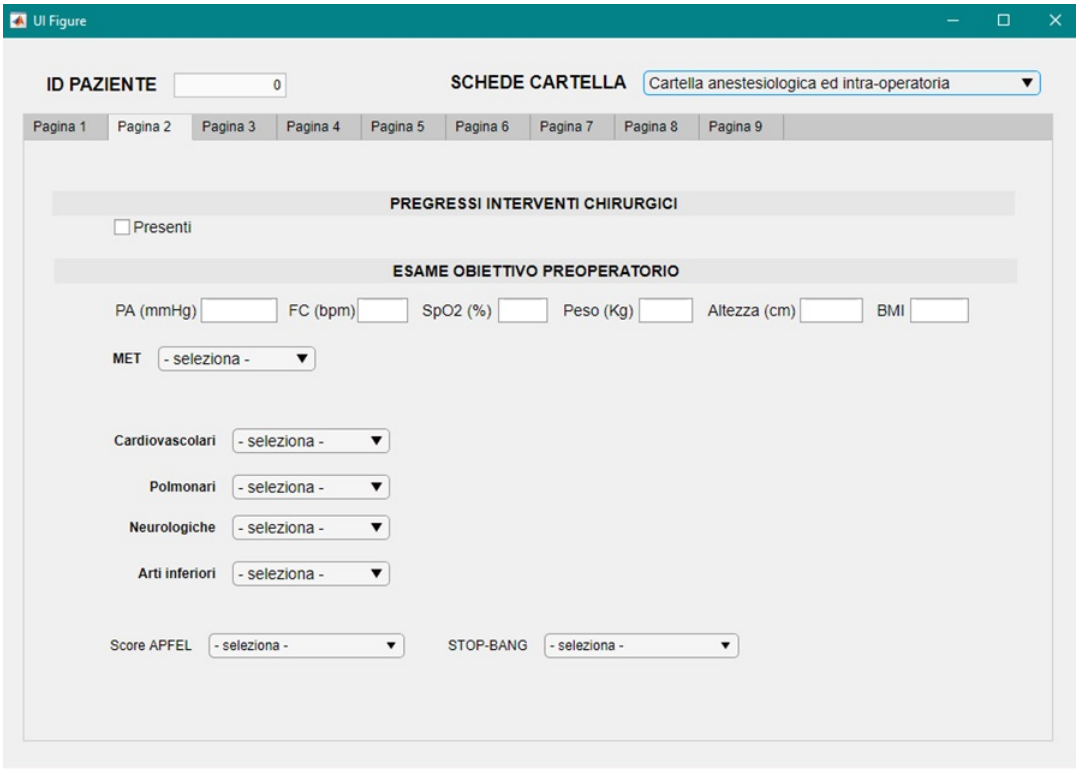


Figura 9: *Cartella anestesiologica ed intra-operatoria: Pagina 2*

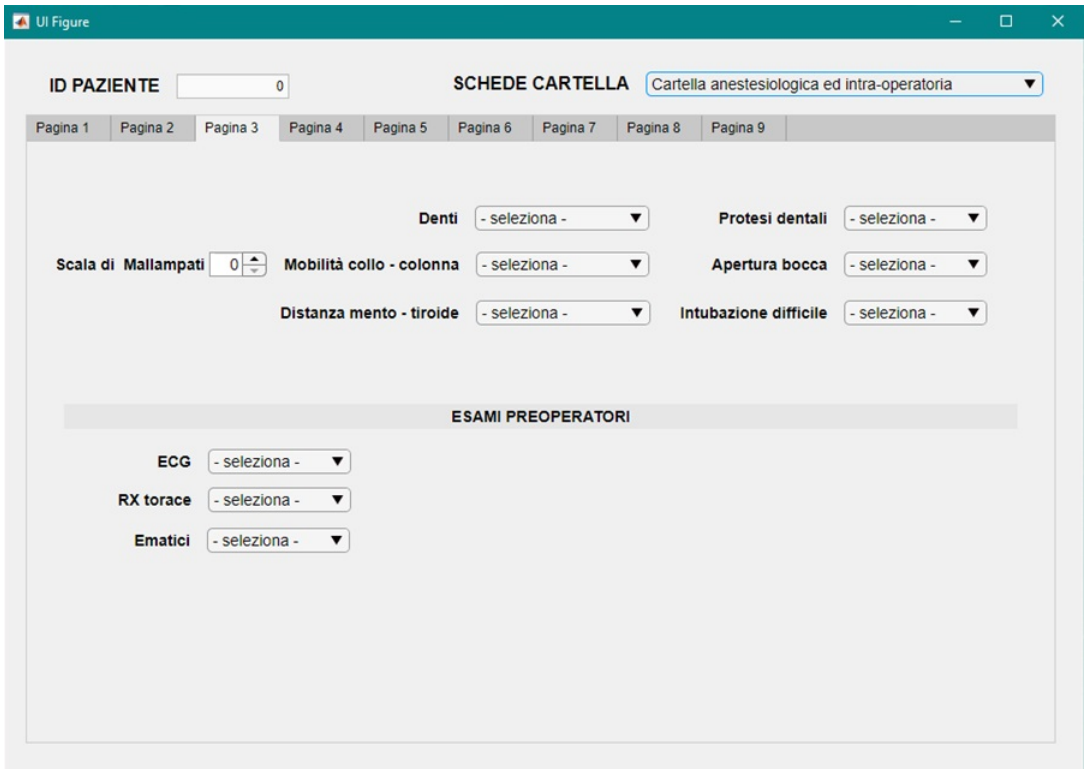


Figura 10: *Cartella anestesiologica ed intra-operatoria: Pagina 3*

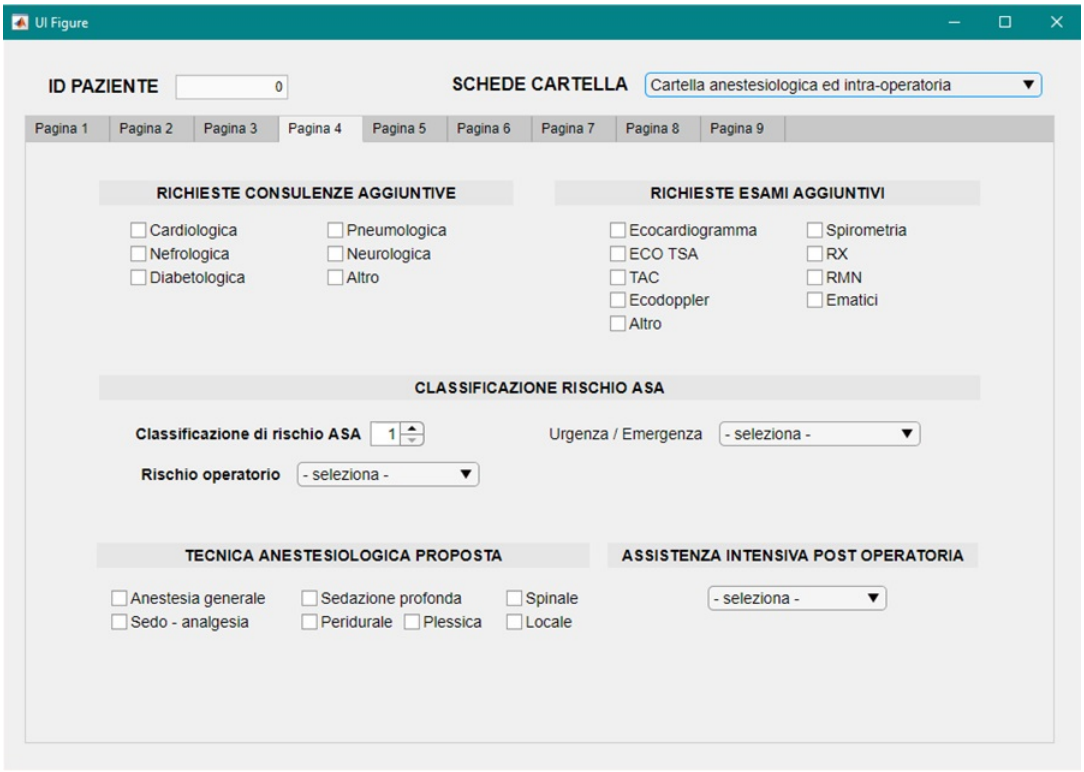


Figura 11: *Cartella anestesiologica ed intra-operatoria: Pagina 4*

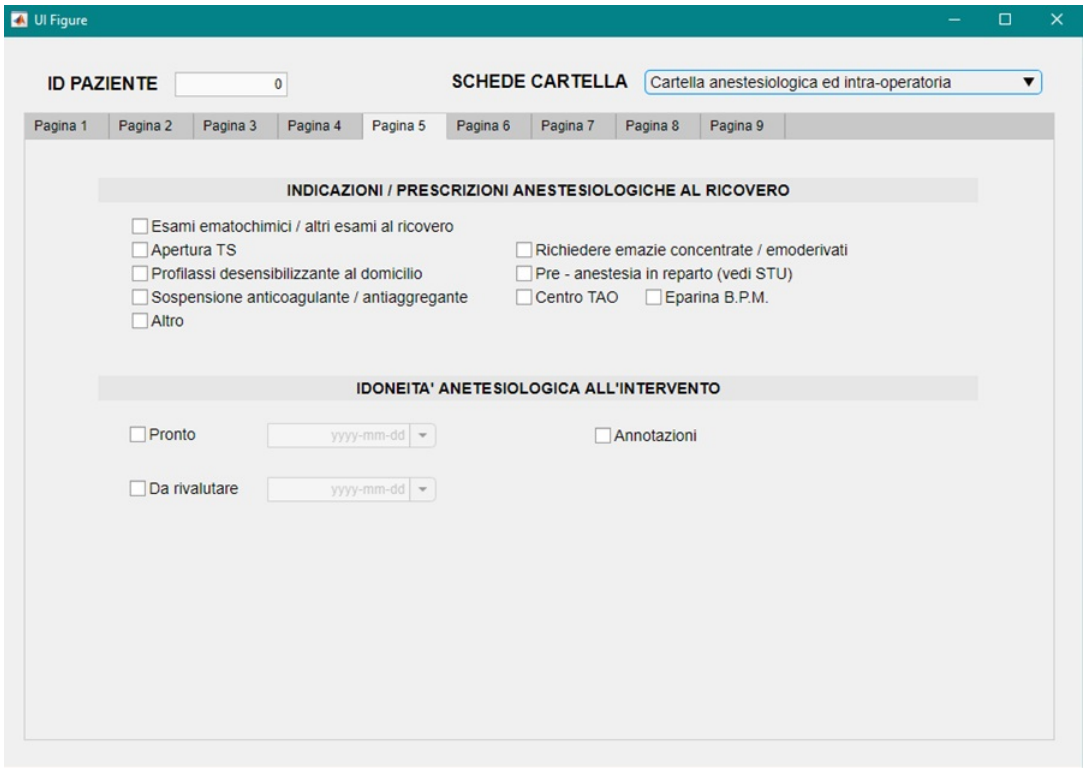


Figura 12: *Cartella anestesiologica ed intra-operatoria: Pagina 5*

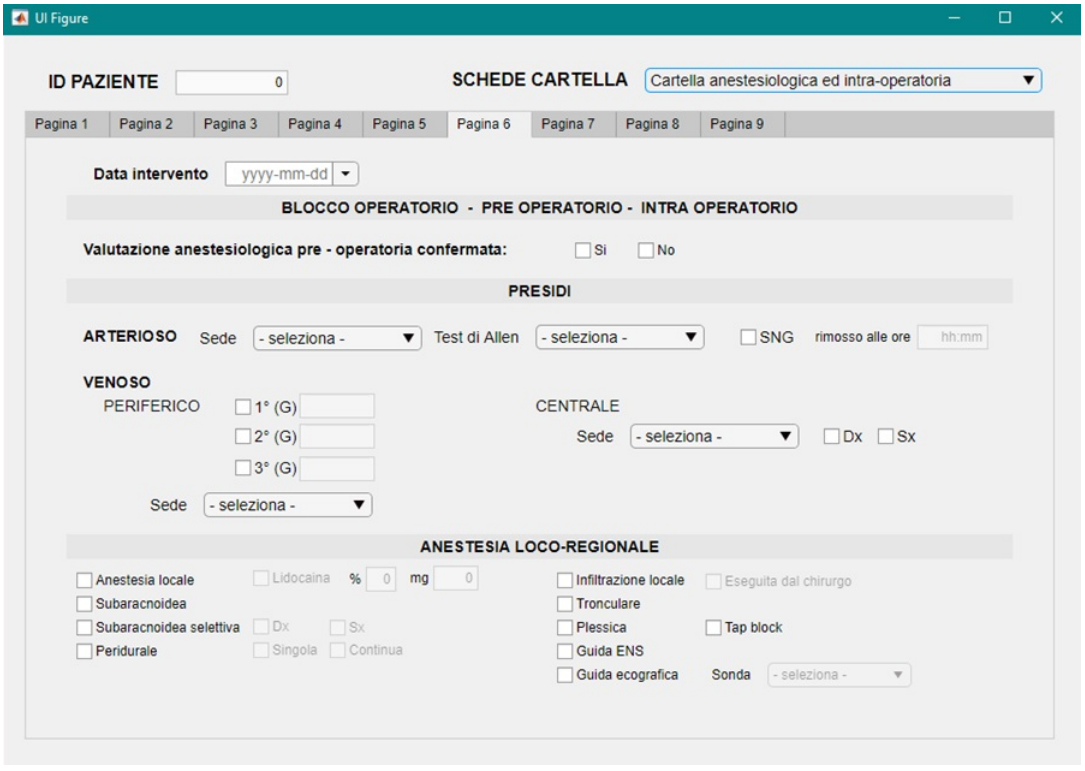


Figura 13: *Cartella anestesiologica ed intra-operatoria: Pagina 6*

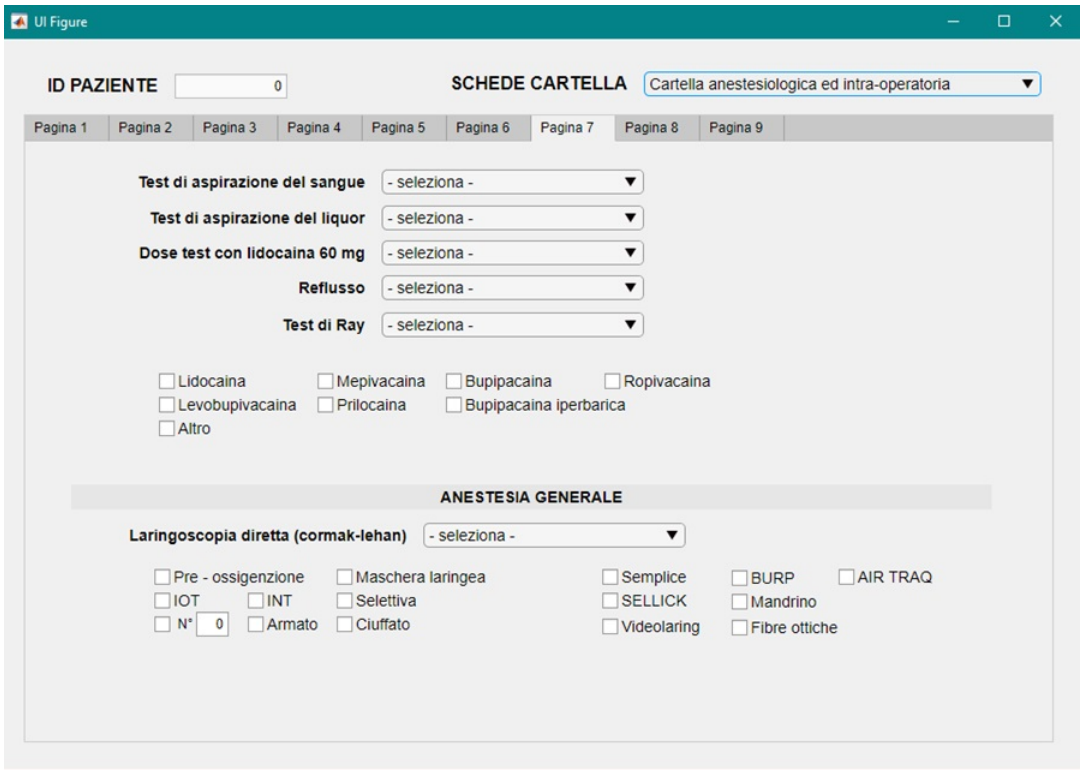


Figura 14: *Cartella anestesiologica ed intra-operatoria: Pagina 7*

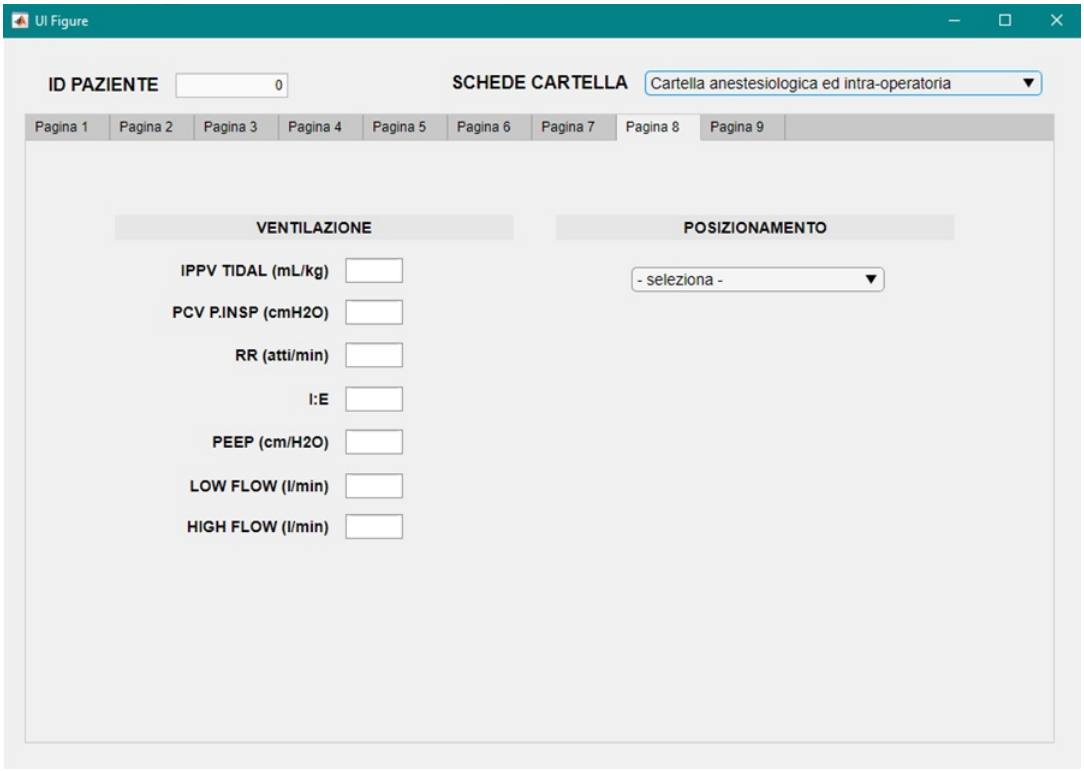


Figura 15: *Cartella anestesiologica ed intra-operatoria: Pagina 8*

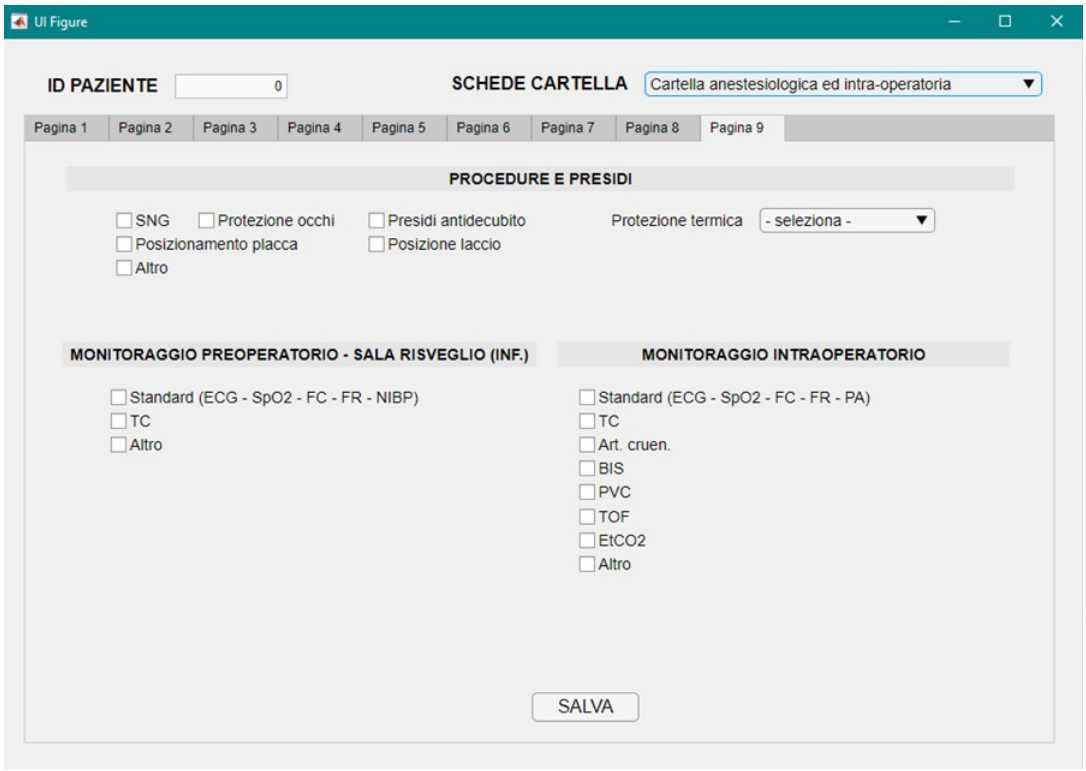


Figura 16: *Cartella anestesiologica ed intra-operatoria: Pagina 9*

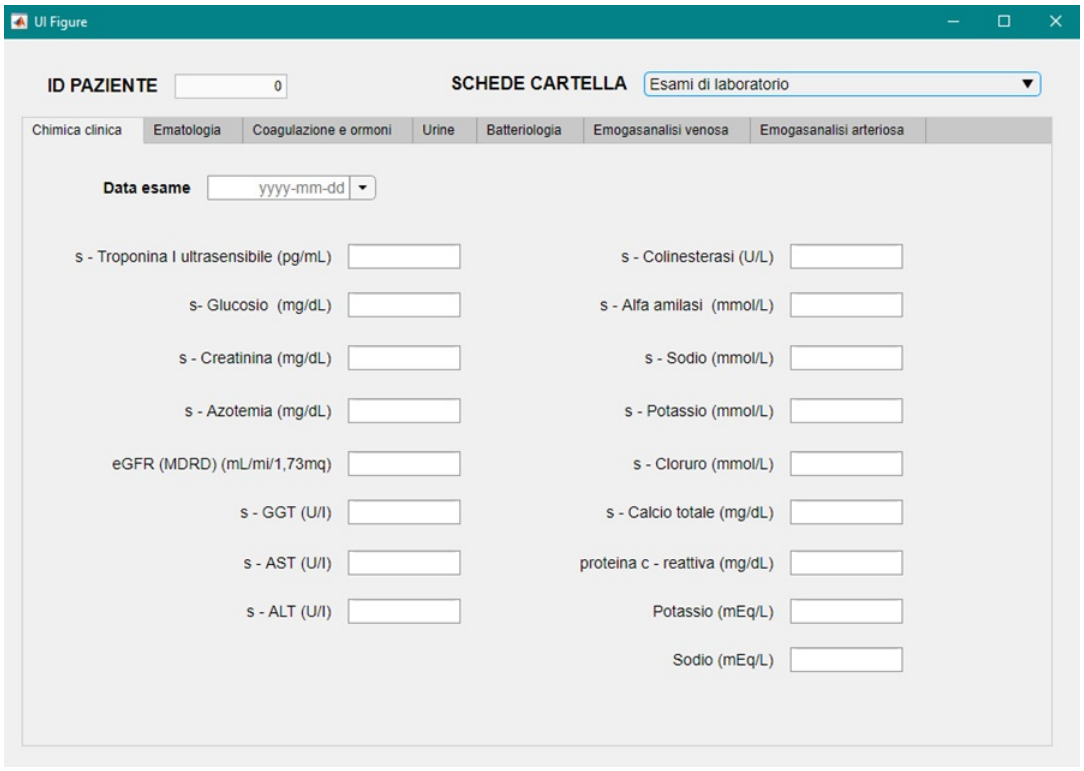


Figura 17: *Esami di laboratorio: Chimica clinica*

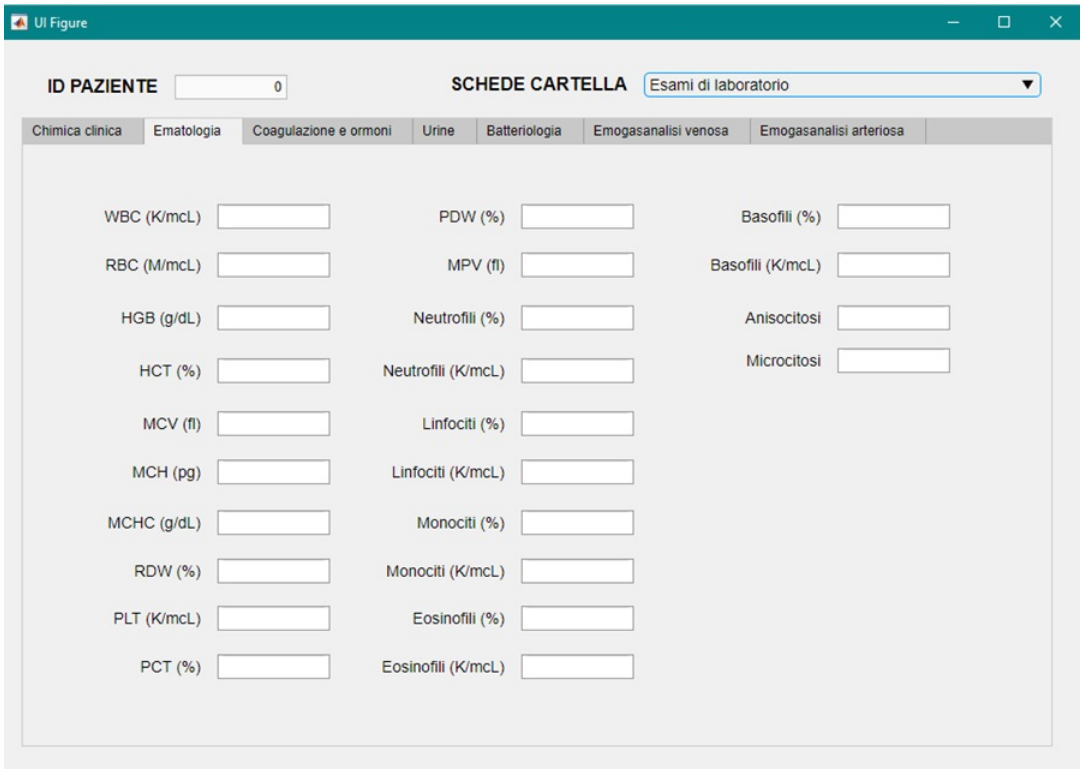


Figura 18: *Esami di laboratorio: Ematologia*

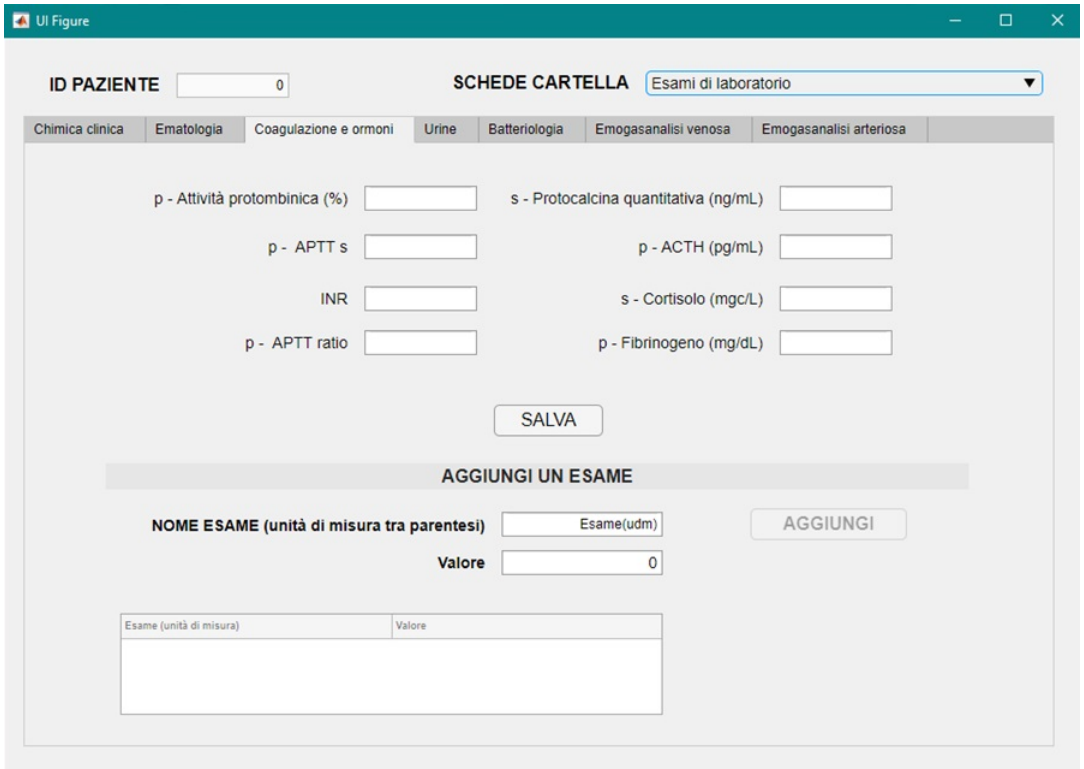


Figura 19: *Esami di laboratorio: Coagulazione ed ormoni*

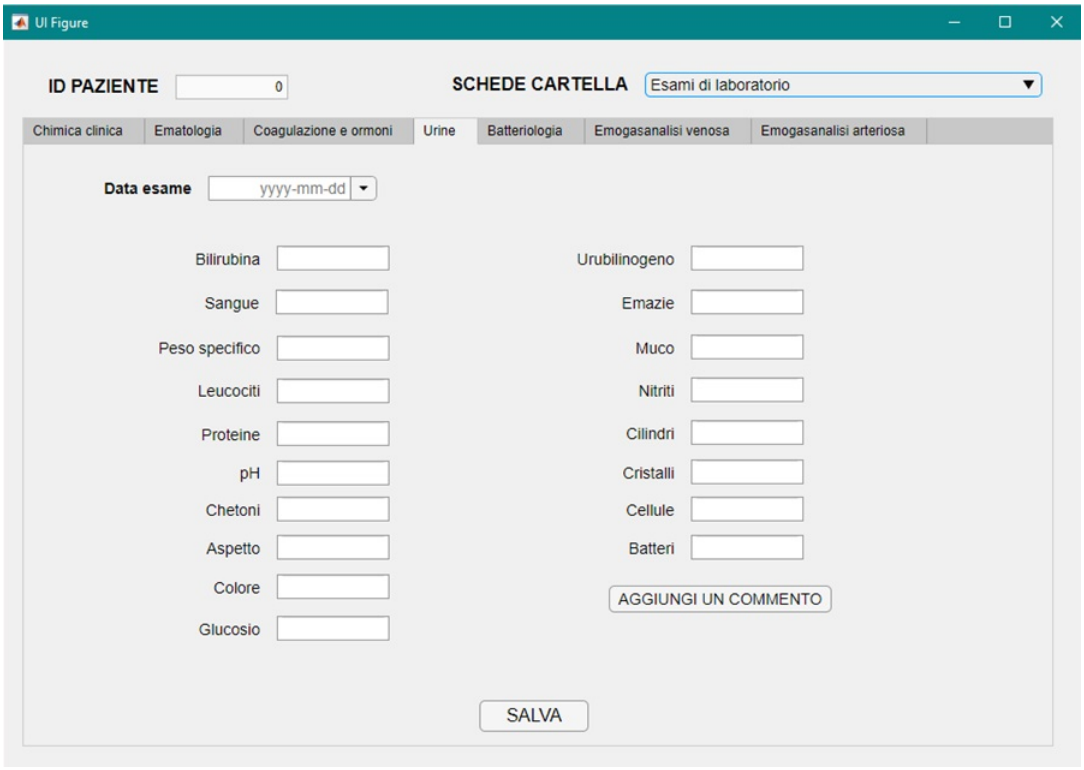


Figura 20: *Esami di laboratorio: Urine*

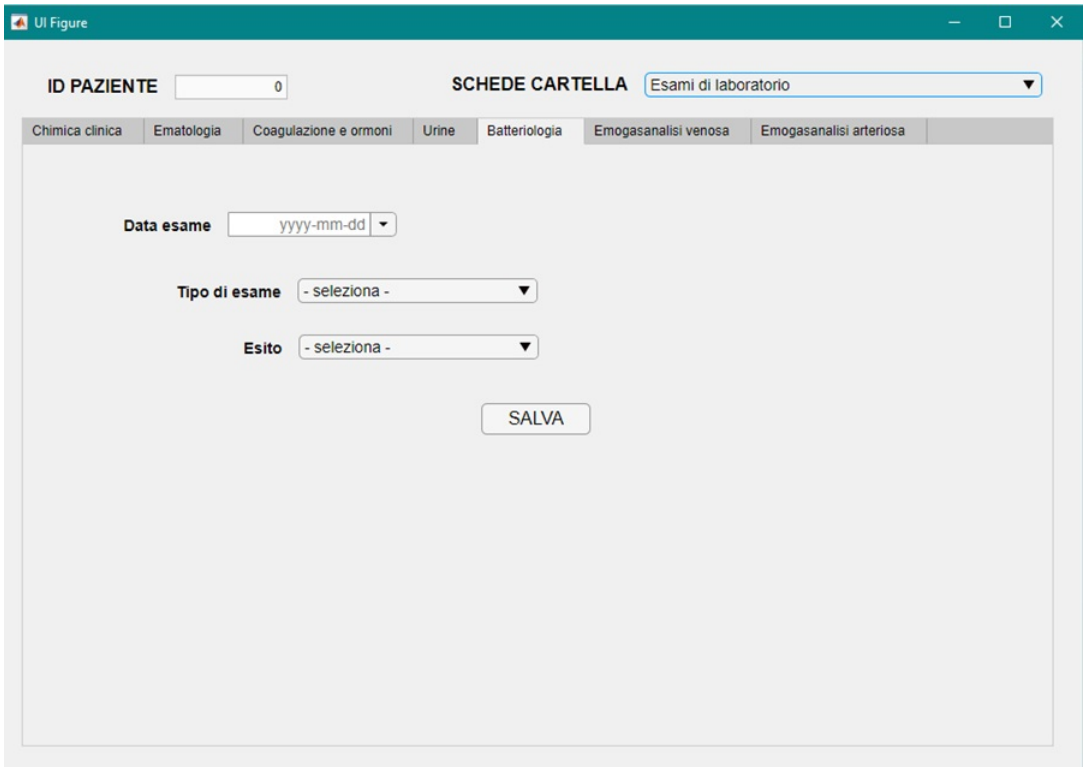


Figura 21: *Esami di laboratorio: Batteriologia*

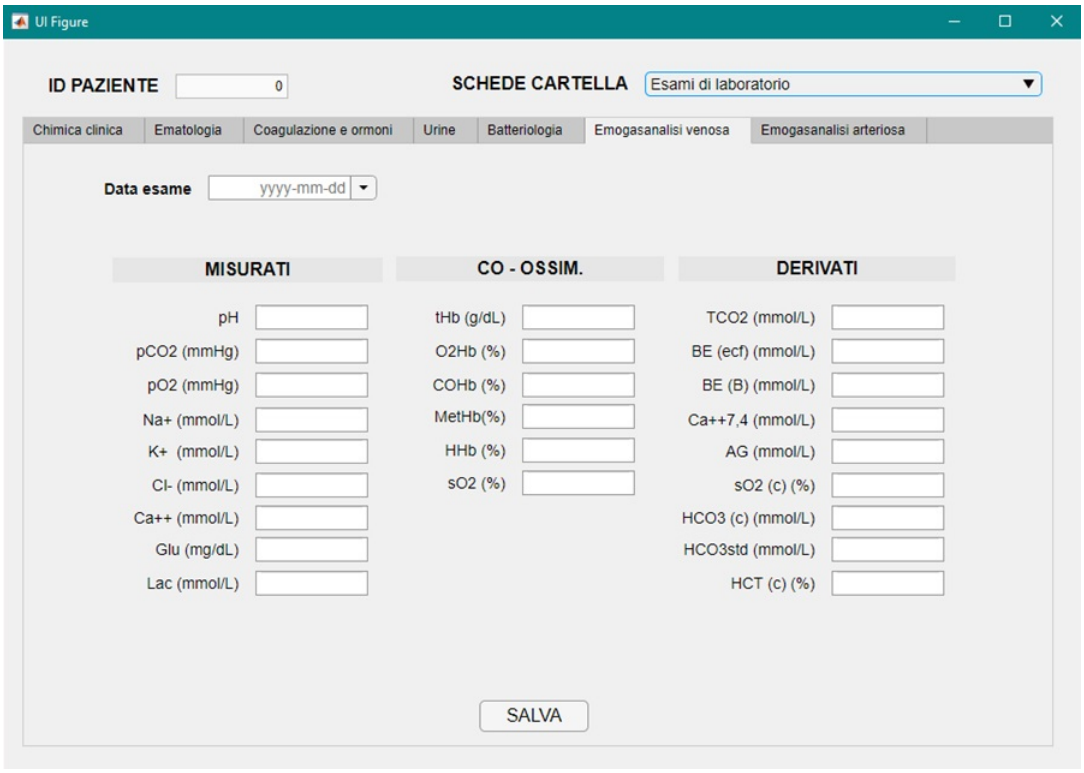


Figura 22: *Esami di laboratorio: Emogasanalisi venosa*

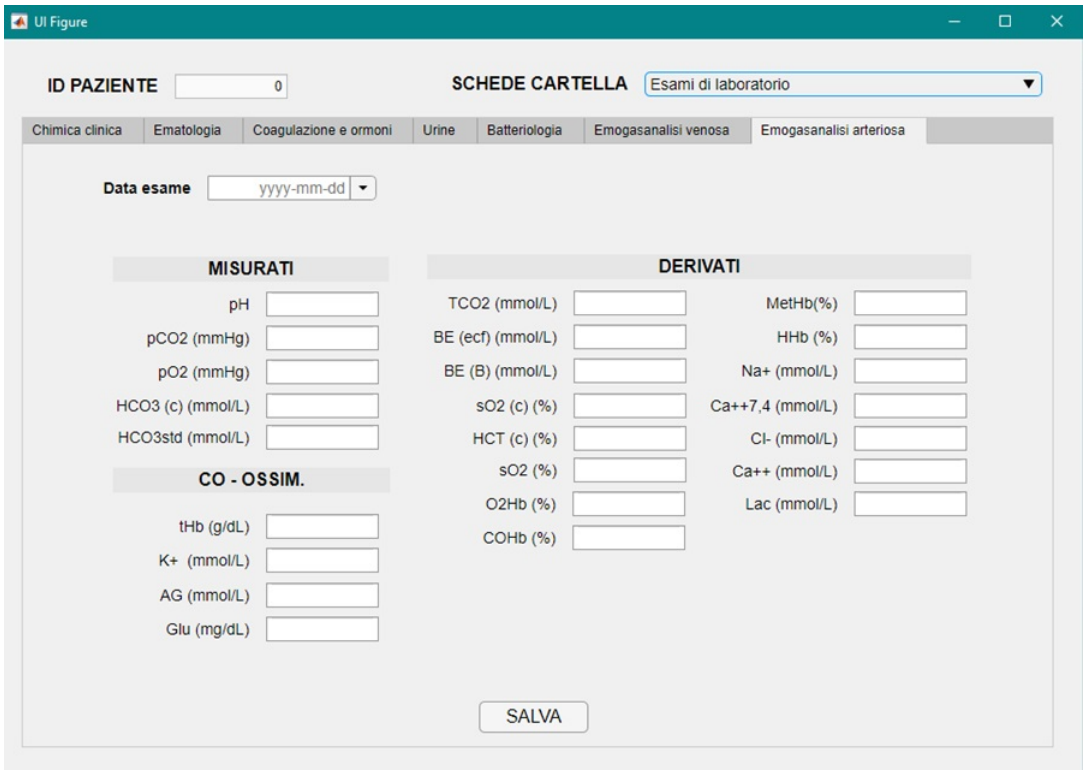


Figura 23: *Esami di laboratorio: Emogasanalisi arteriosa*

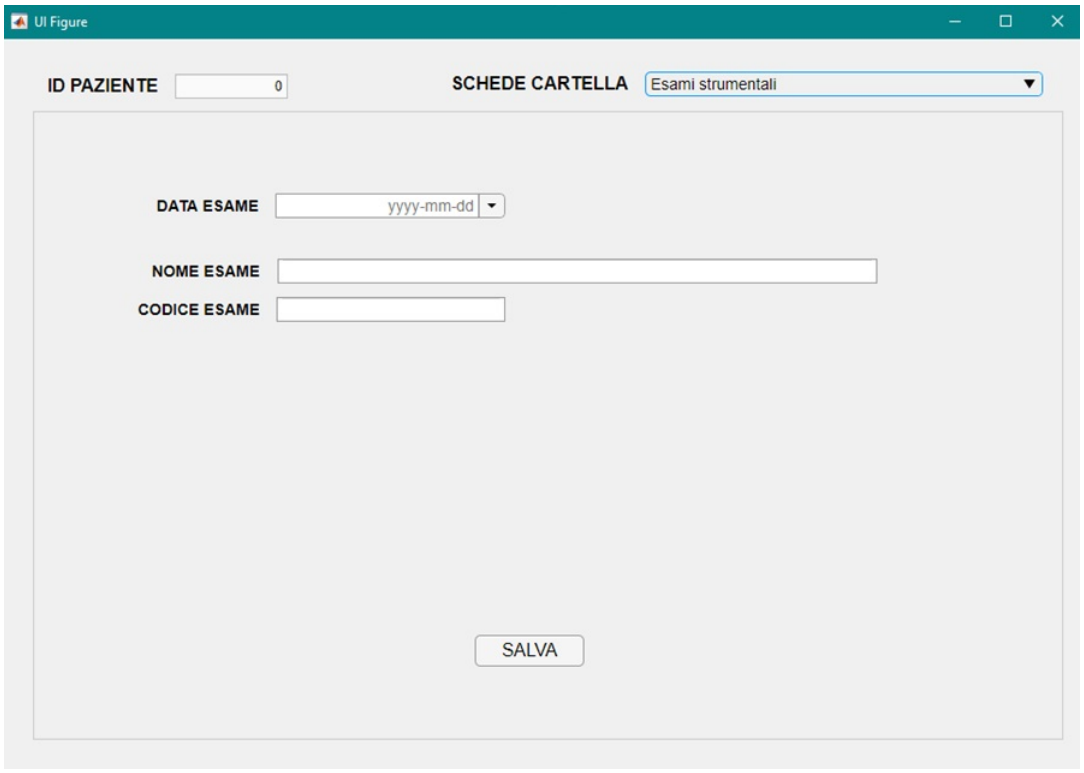


Figura 24: *Esami strumentali*

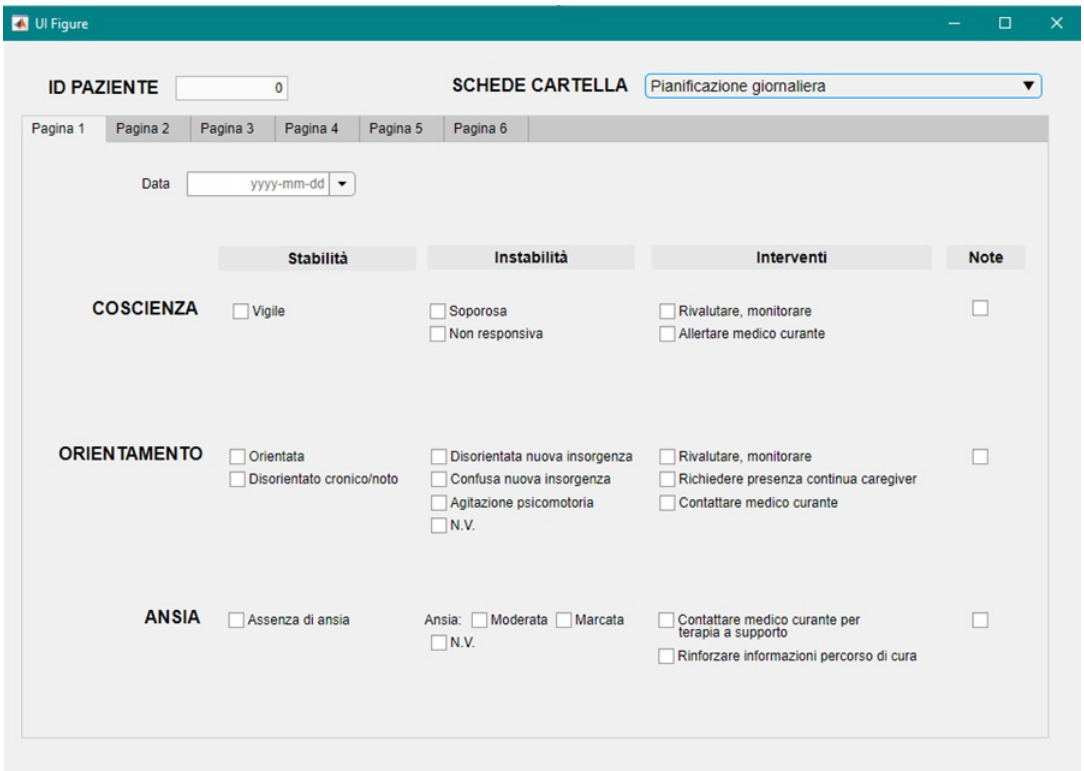


Figura 25: *Pianificazione giornaliera: Pagina 1*

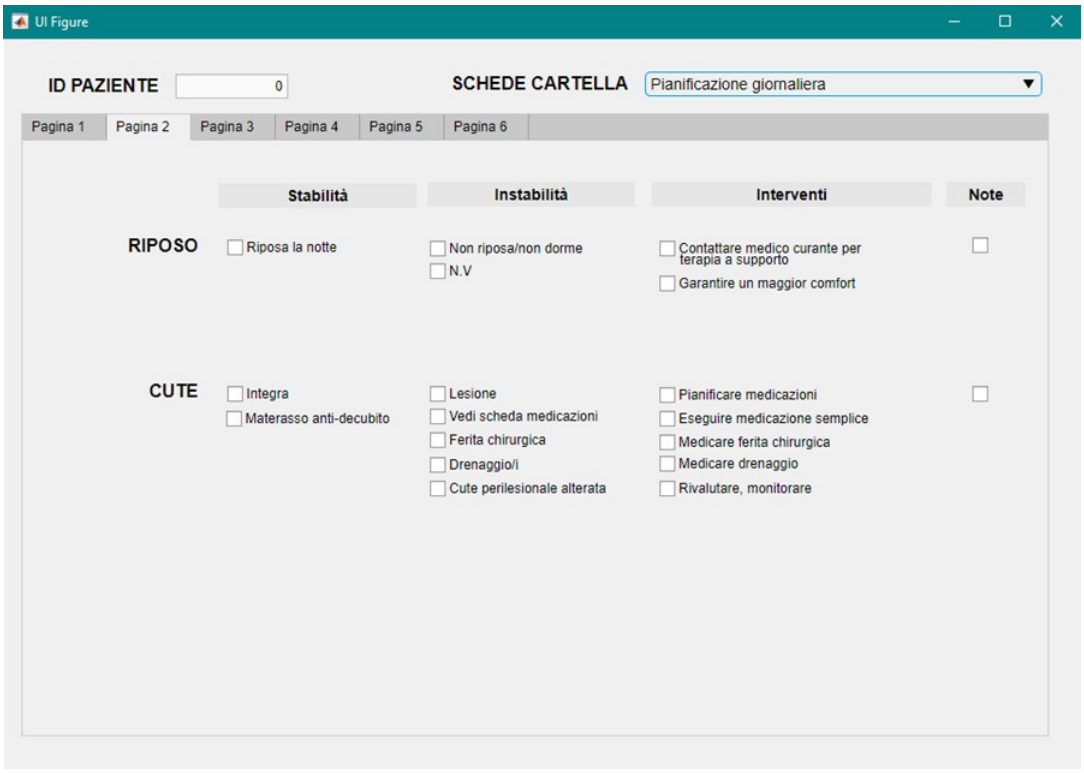


Figura 26: *Pianificazione giornaliera: Pagina 2*

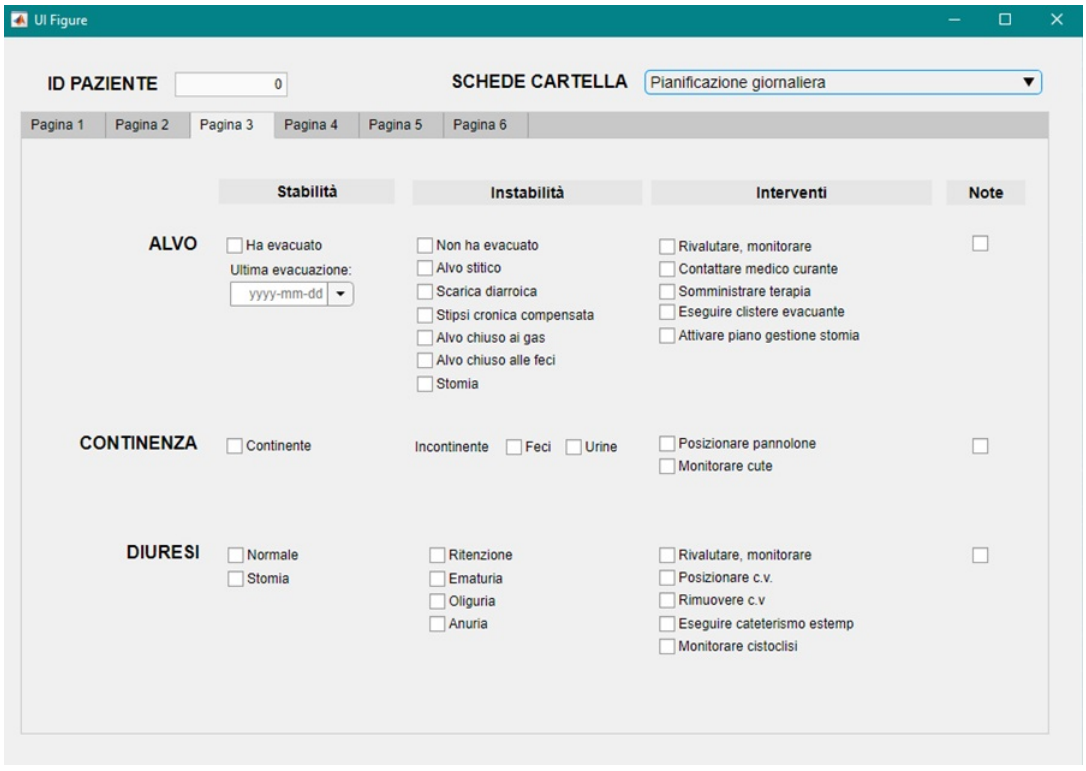


Figura 27: *Pianificazione giornaliera: Pagina 3*

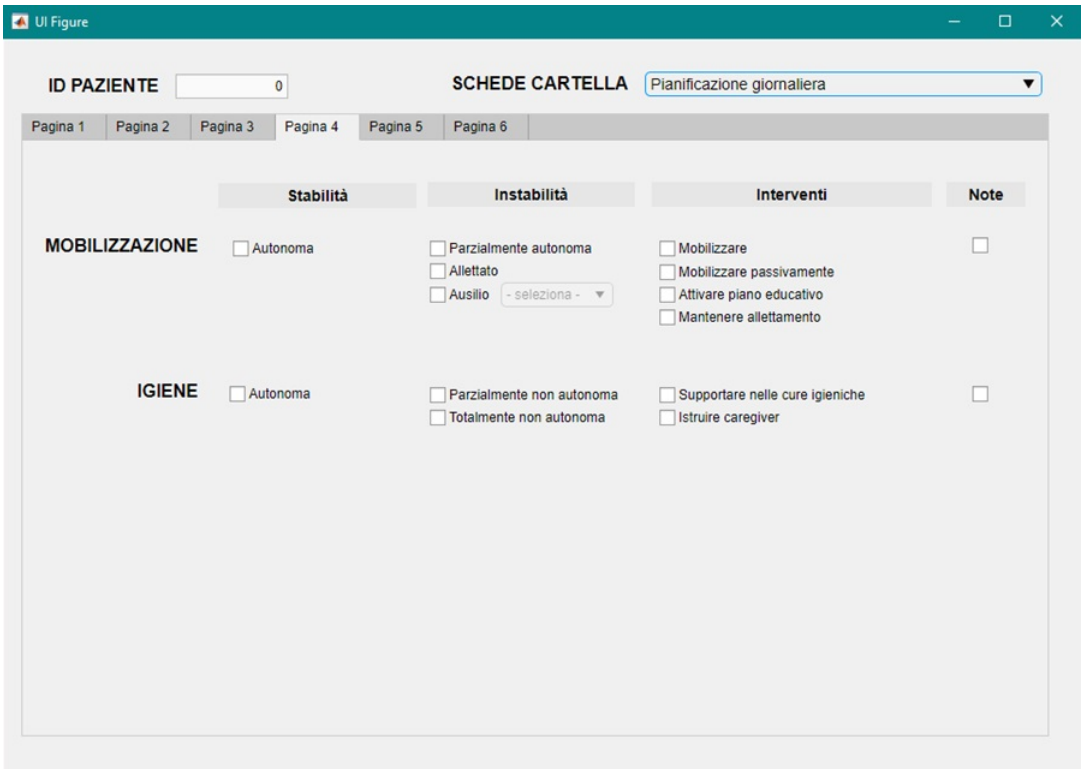


Figura 28: *Pianificazione giornaliera: Pagina 4*

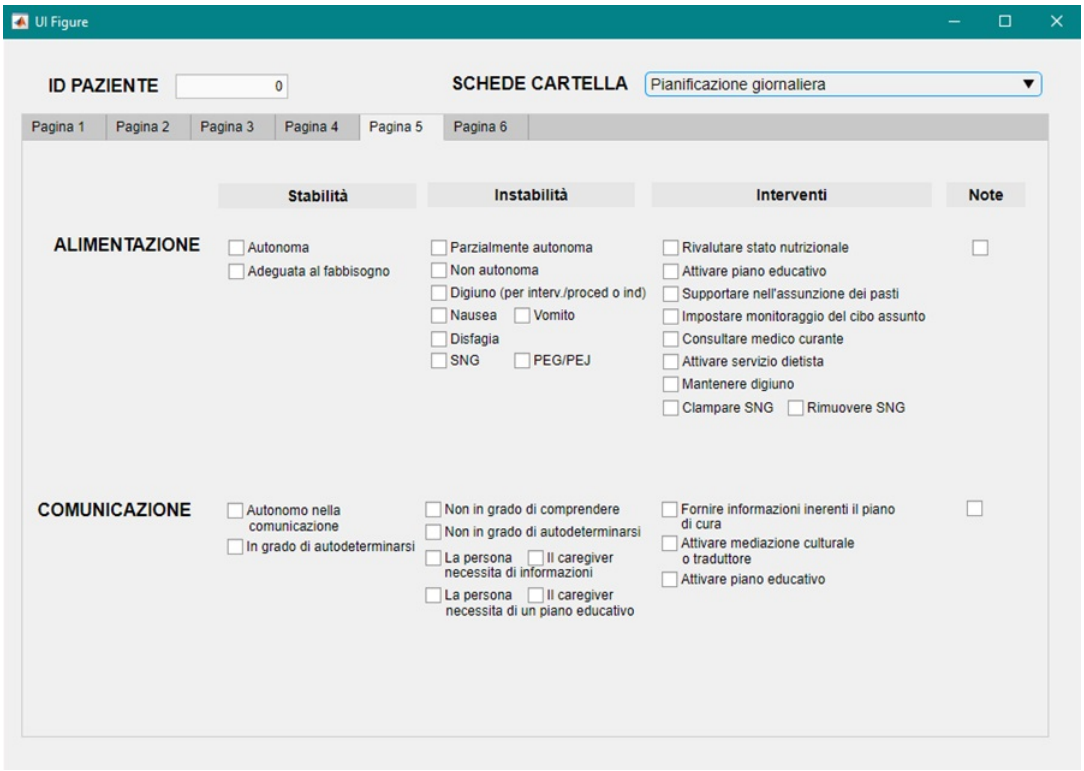


Figura 29: *Pianificazione giornaliera: Pagina 5*

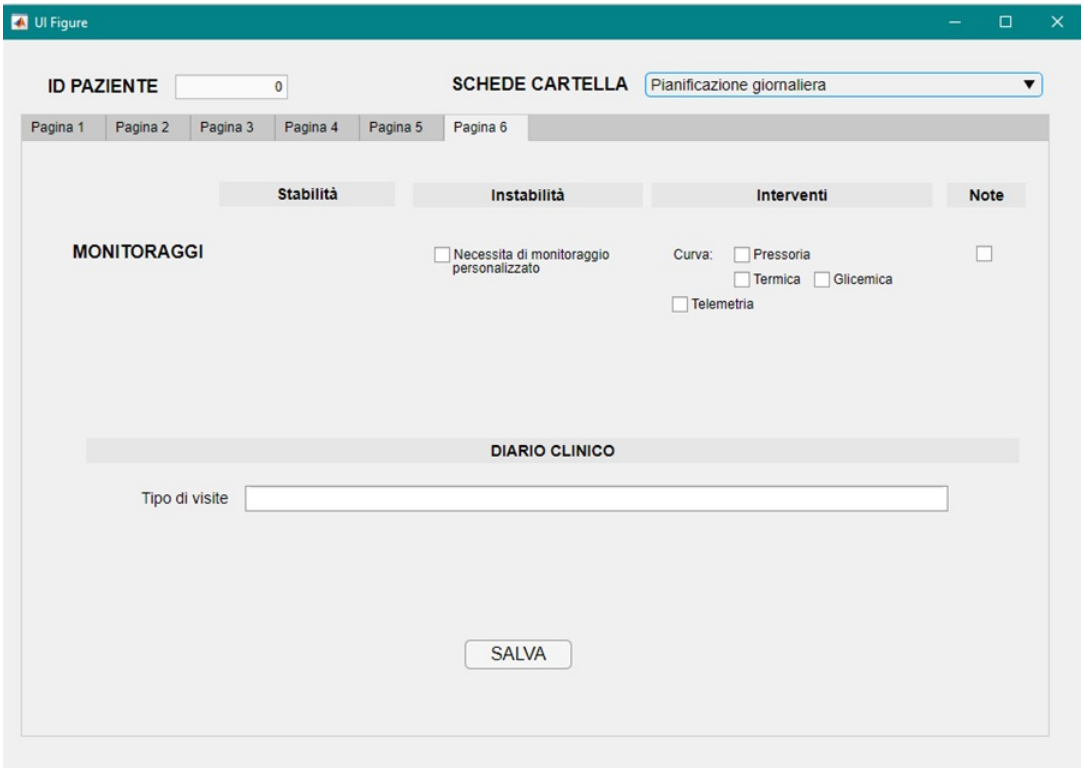


Figura 30: *Pianificazione giornaliera: Pagina 6*

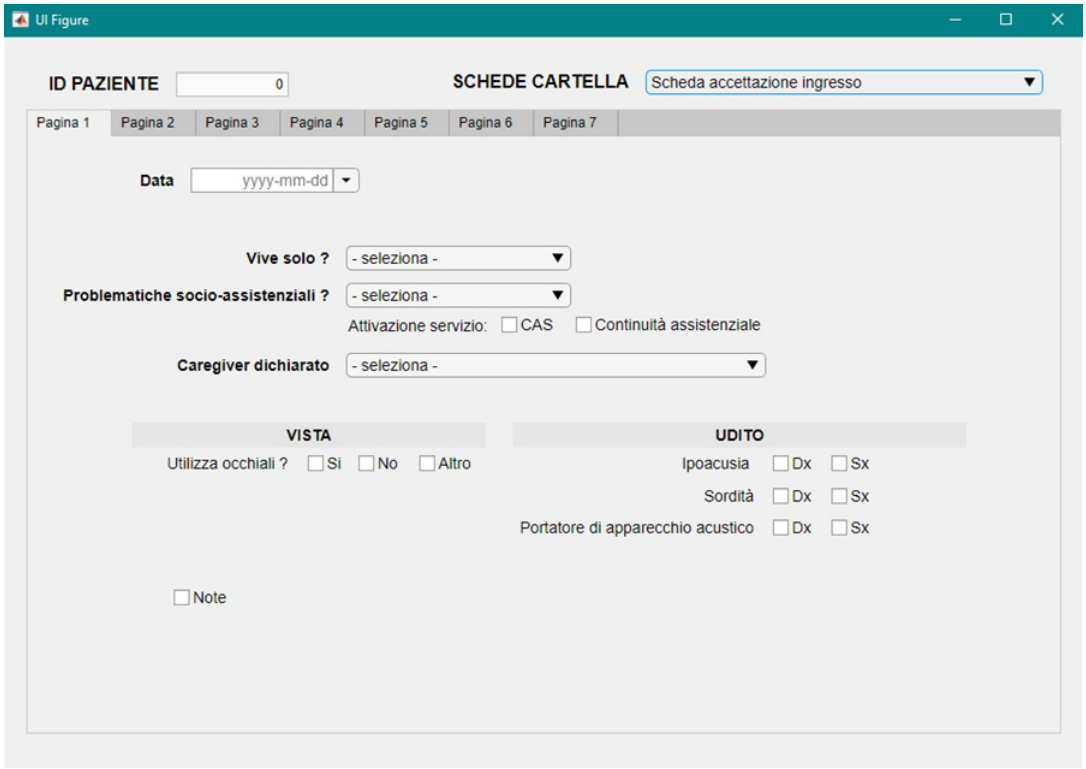


Figura 31: *Scheda accettazione ingresso: Pagina 1*

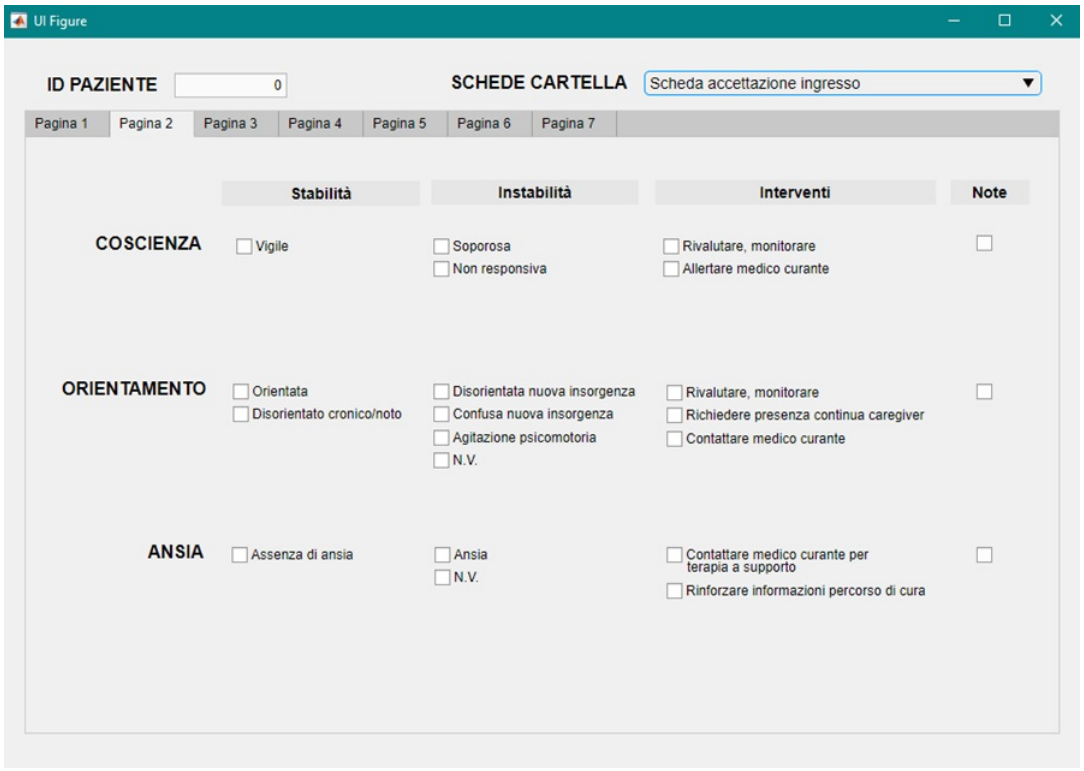


Figura 32: *Scheda accettazione ingresso: Pagina 2*

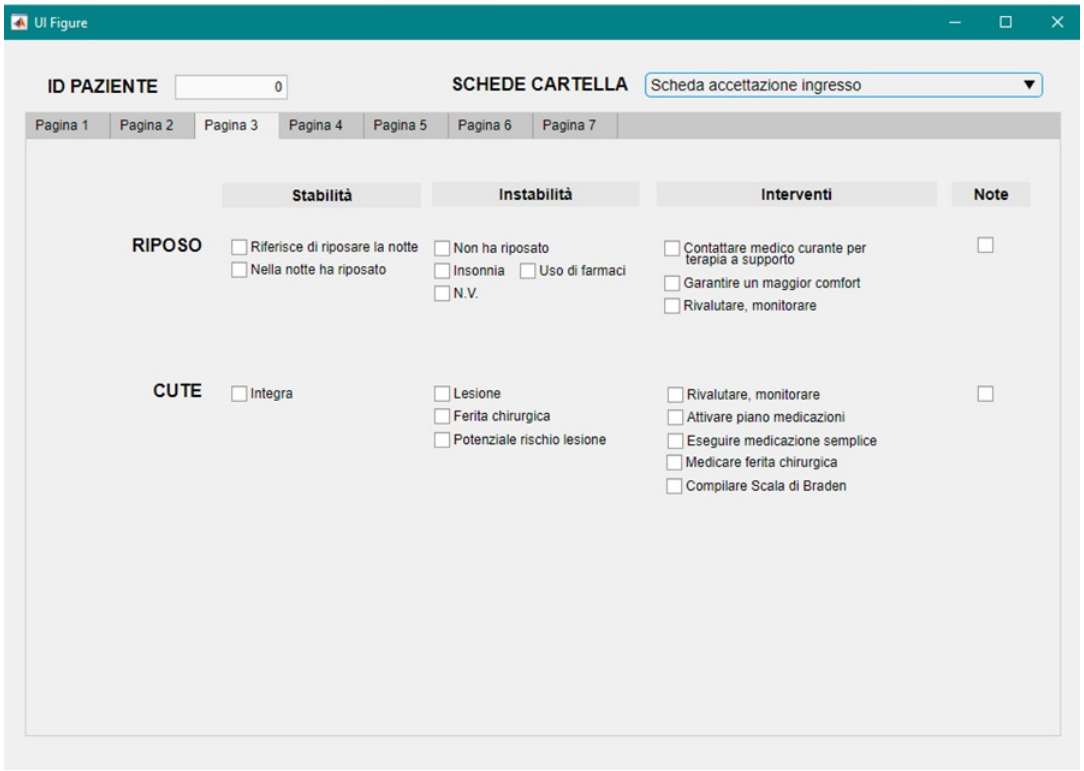


Figura 33: *Scheda accettazione ingresso: Pagina 3*

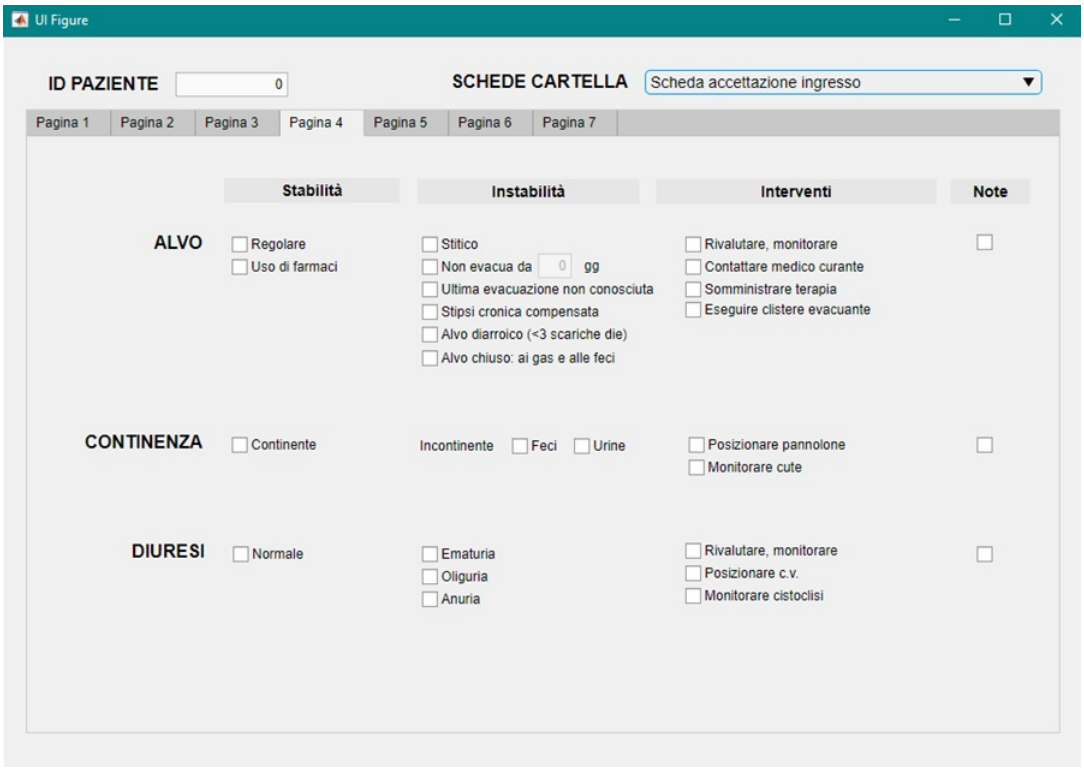


Figura 34: *Scheda accettazione ingresso: Pagina 4*

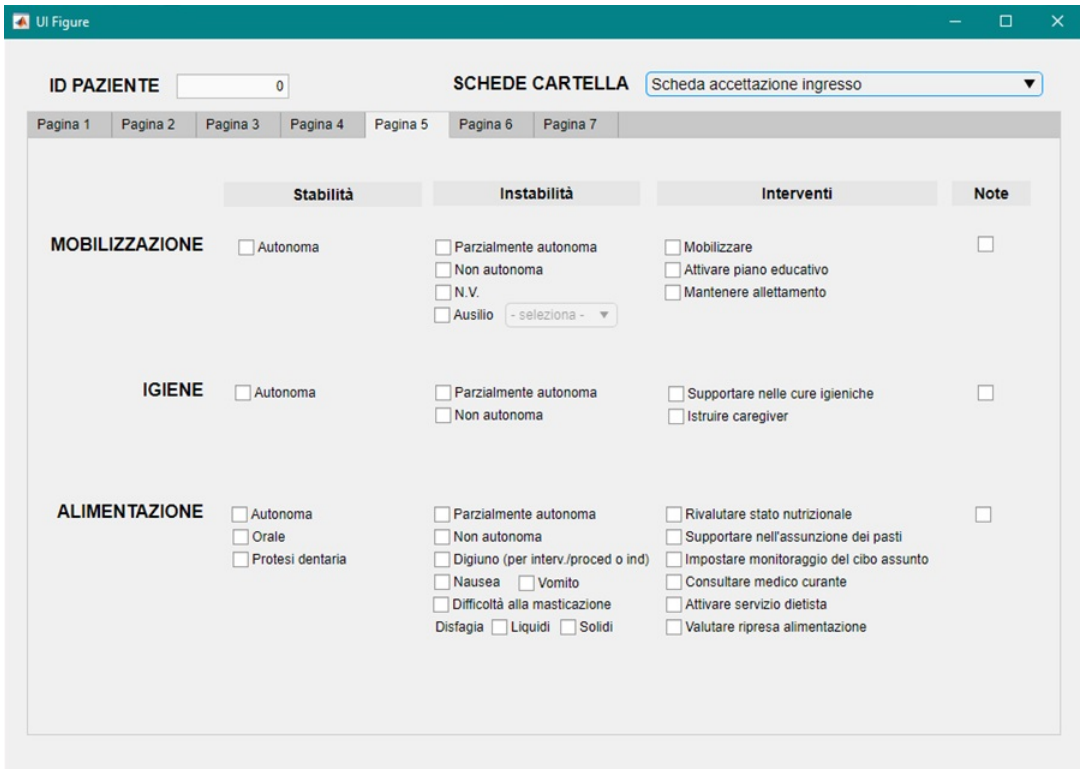


Figura 35: *Scheda accettazione ingresso: Pagina 5*

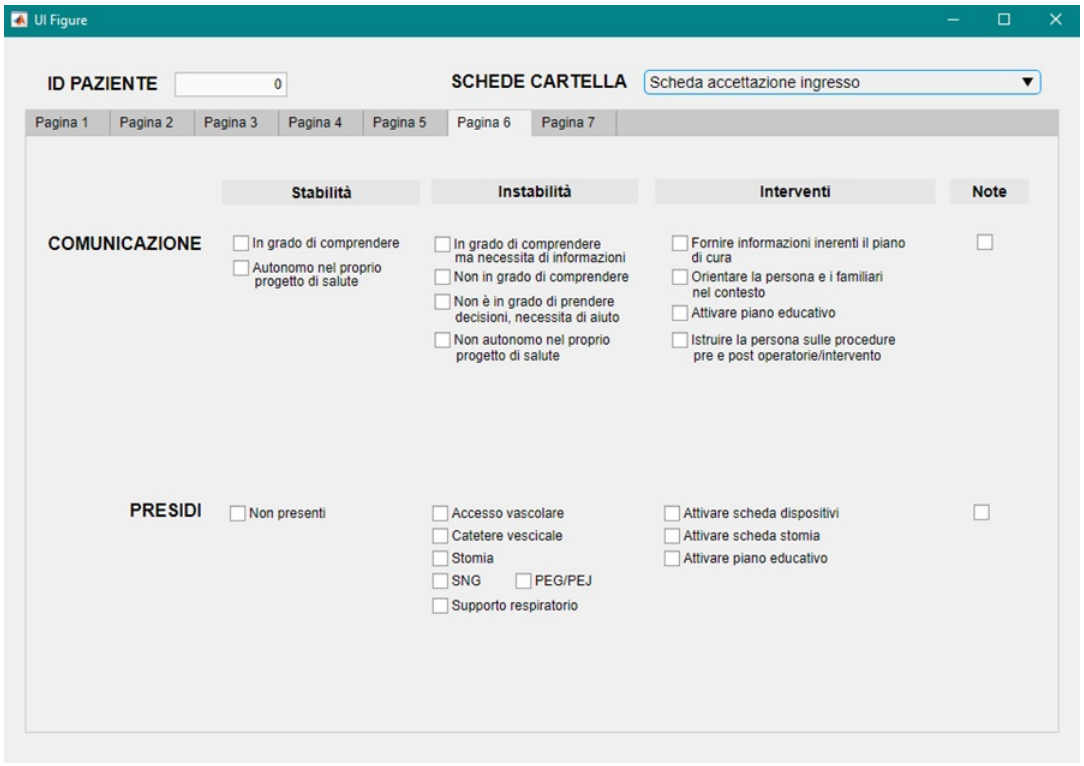


Figura 36: *Scheda accettazione ingresso: Pagina 6*

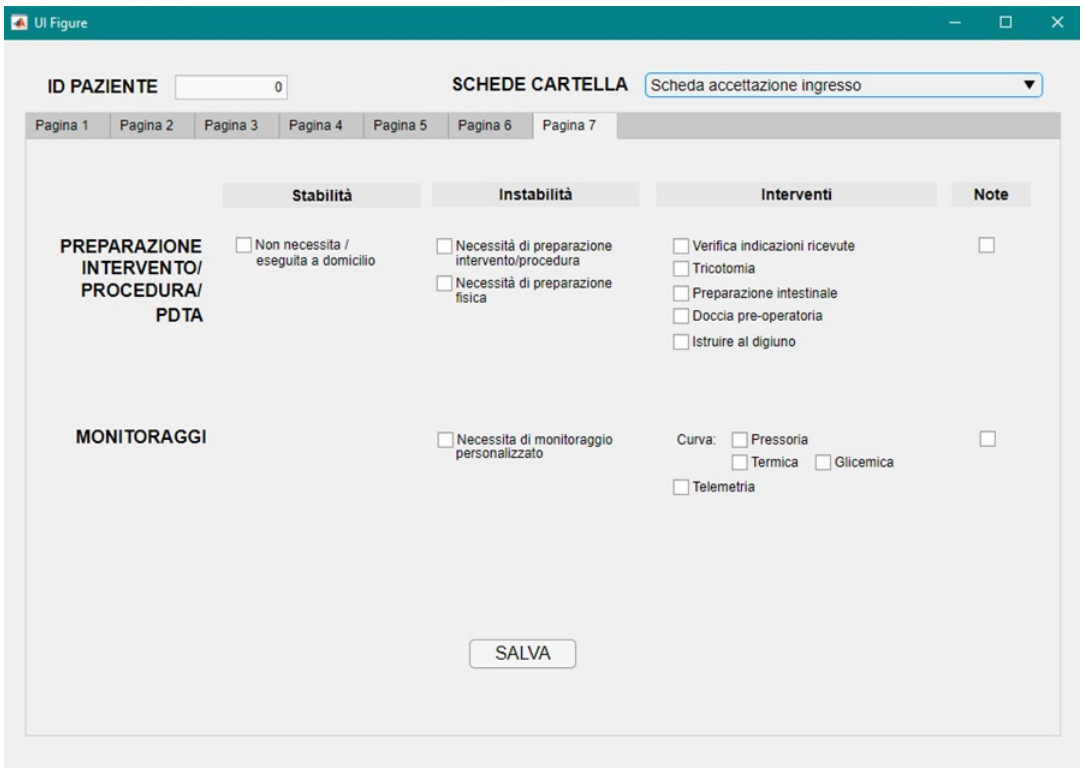


Figura 37: *Scheda accettazione ingresso: Pagina 7*

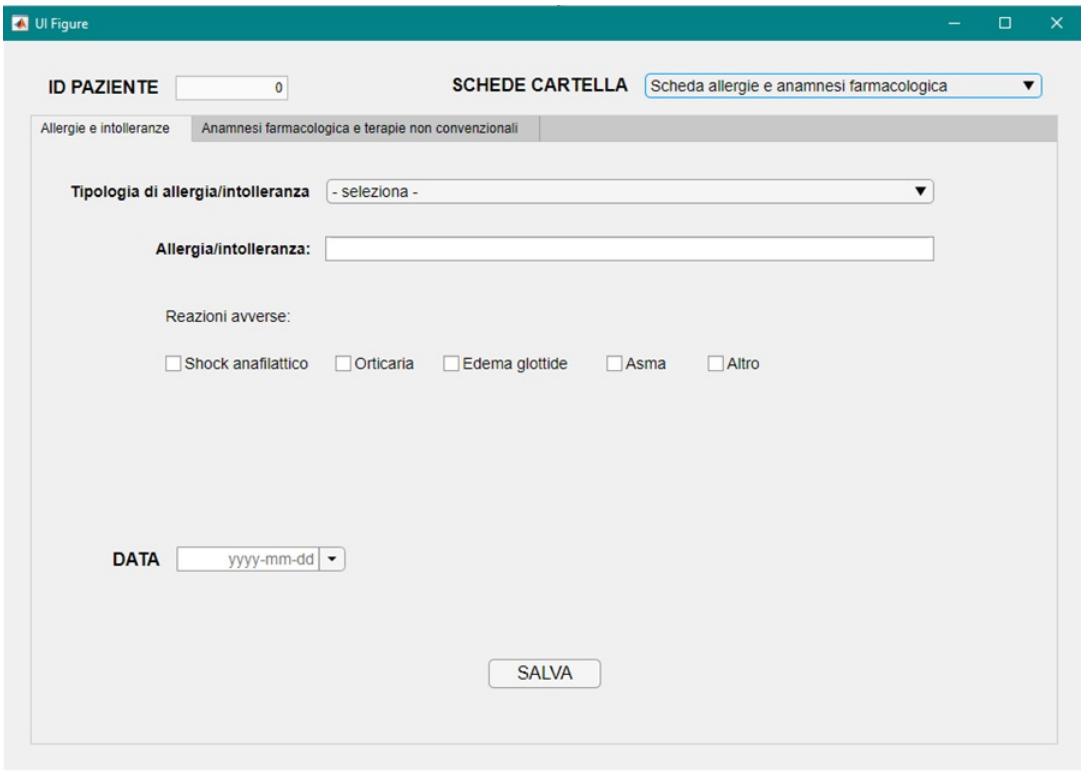


Figura 38: *Scheda allargie e anamnesi farmacologica: Allergie e intolleranze*

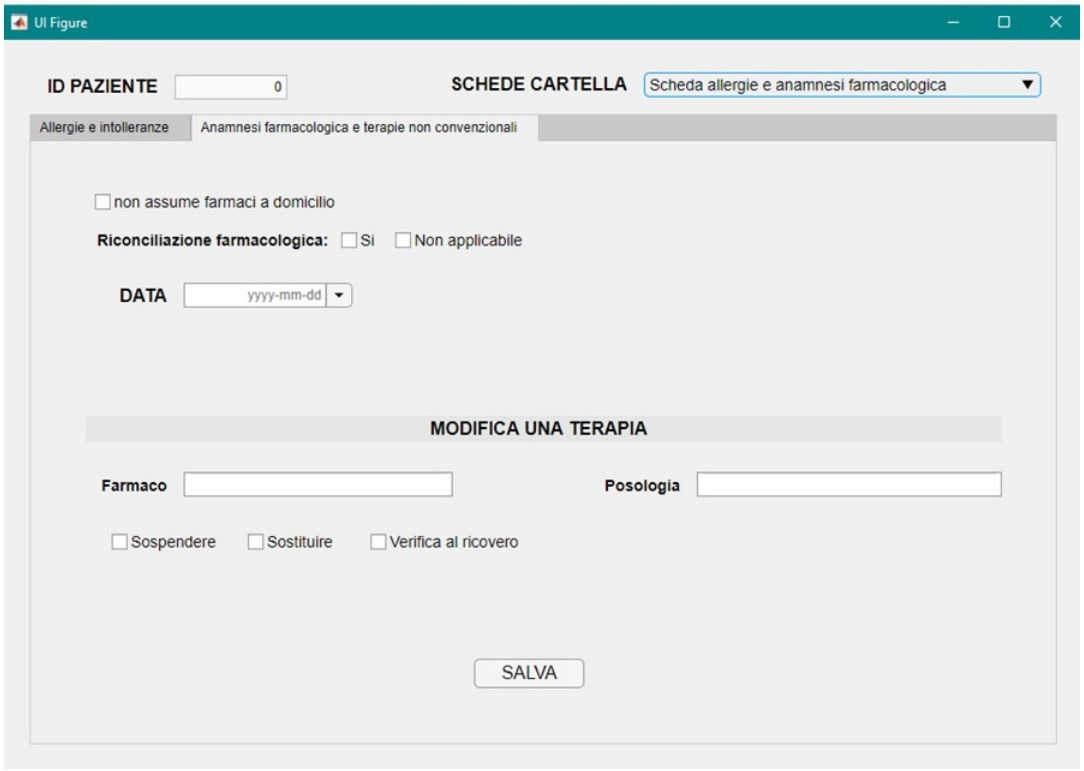


Figura 39: *Scheda allargie e anamnesi farmacologica: Anamnesi farmacologica e terapie non convenzionali*

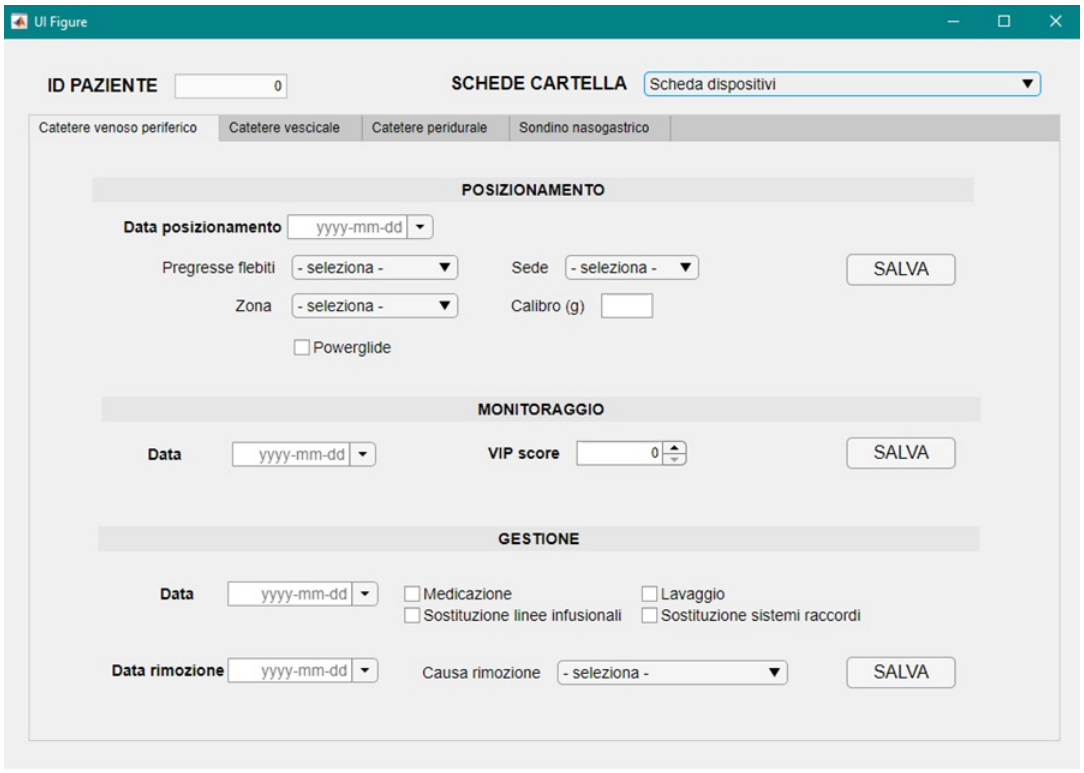


Figura 40: *Scheda dispositivi: Catetere venoso periferico*

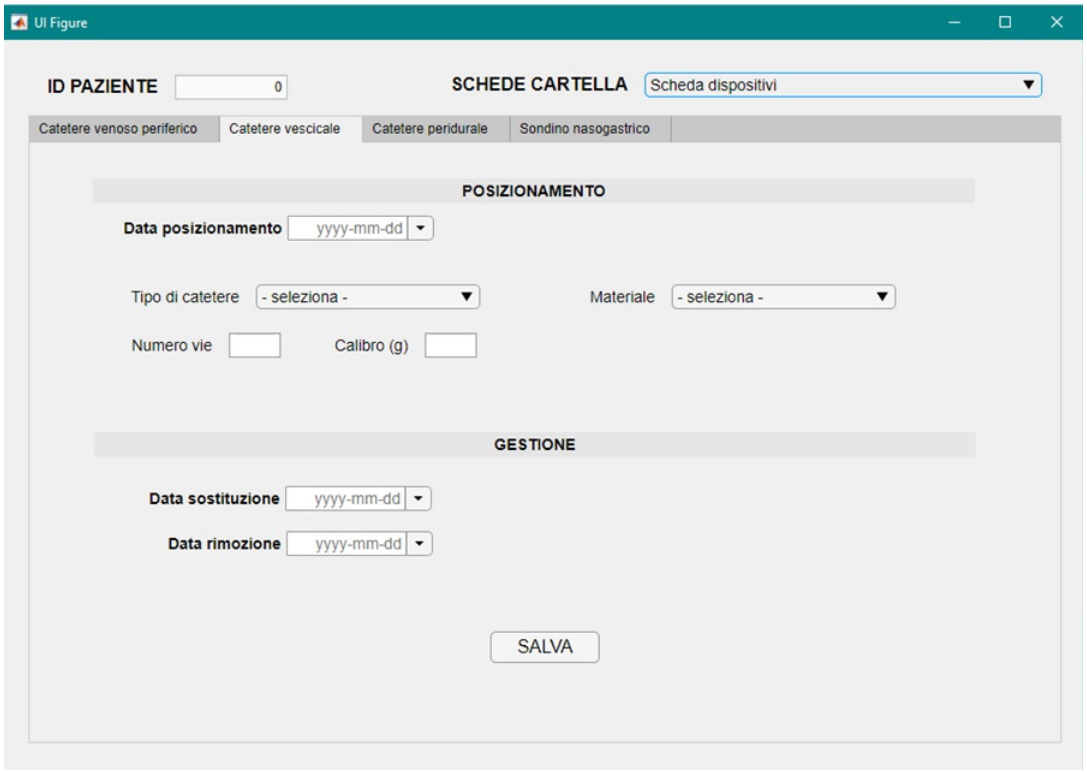


Figura 41: *Scheda dispositivi: Catetere vescicale*

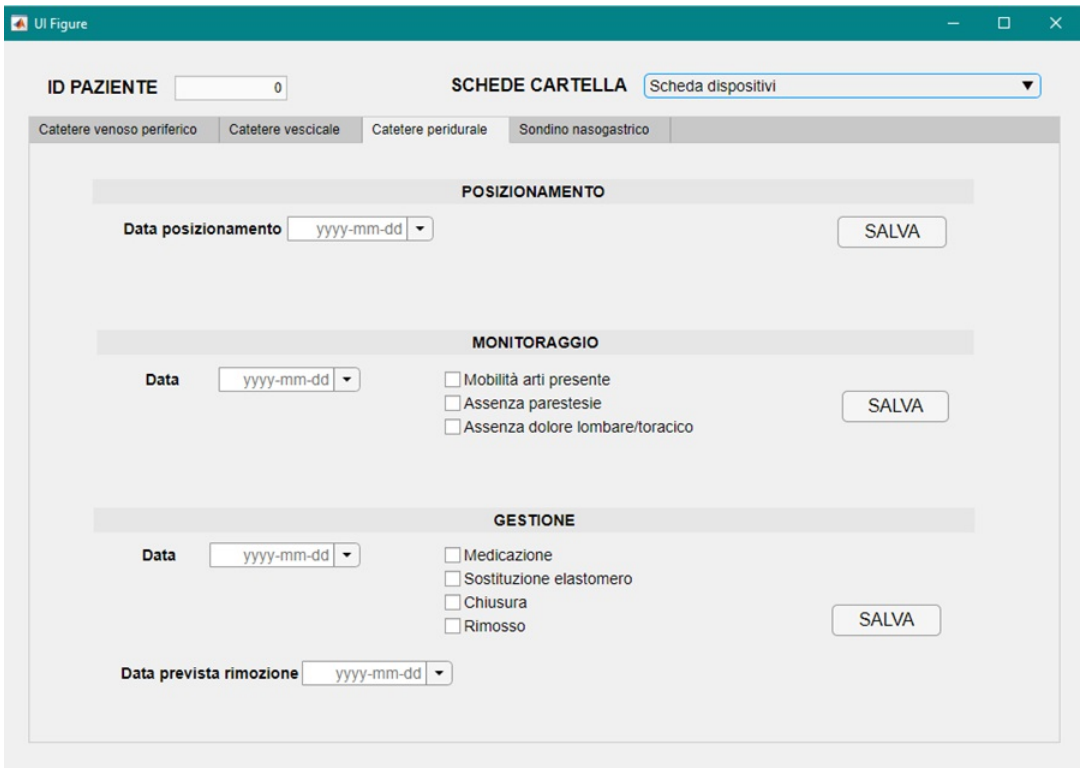


Figura 42: *Scheda dispositivi: Catetere peridurale*

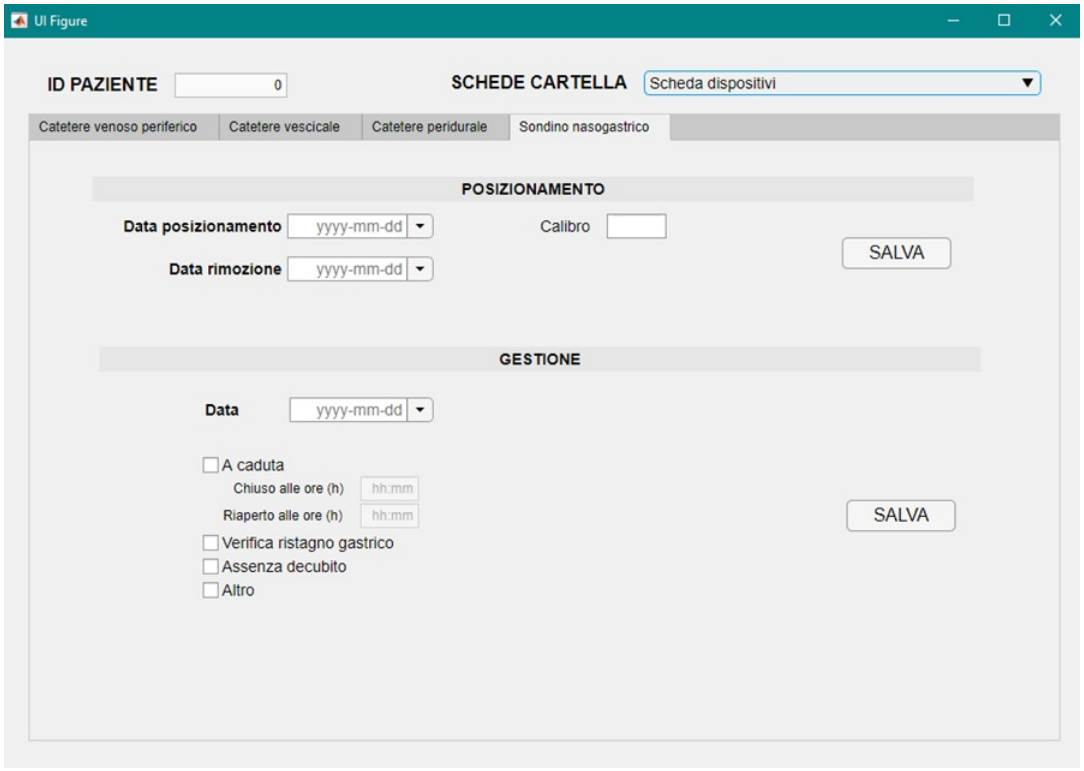


Figura 43: *Scheda dispositivi: Sondino nasogastrico*

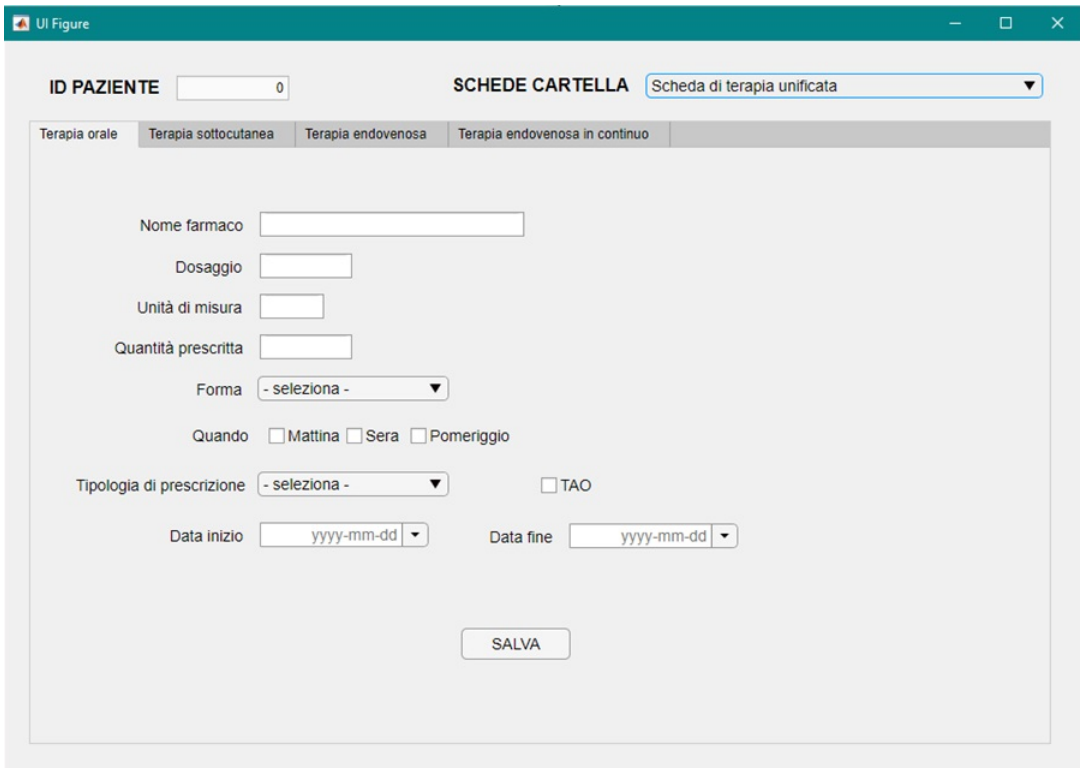


Figura 44: *Scheda di terapia unificata: Terapia orale*

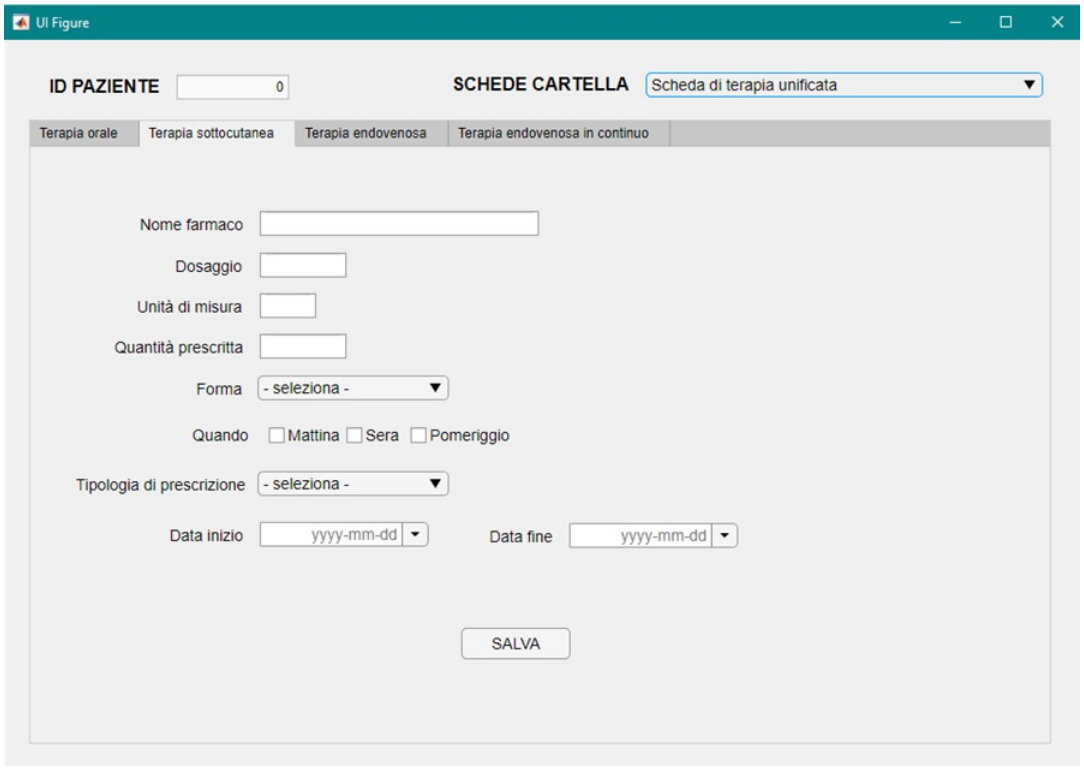


Figura 45: *Scheda di terapia unificata: Terapia sottocutanea*

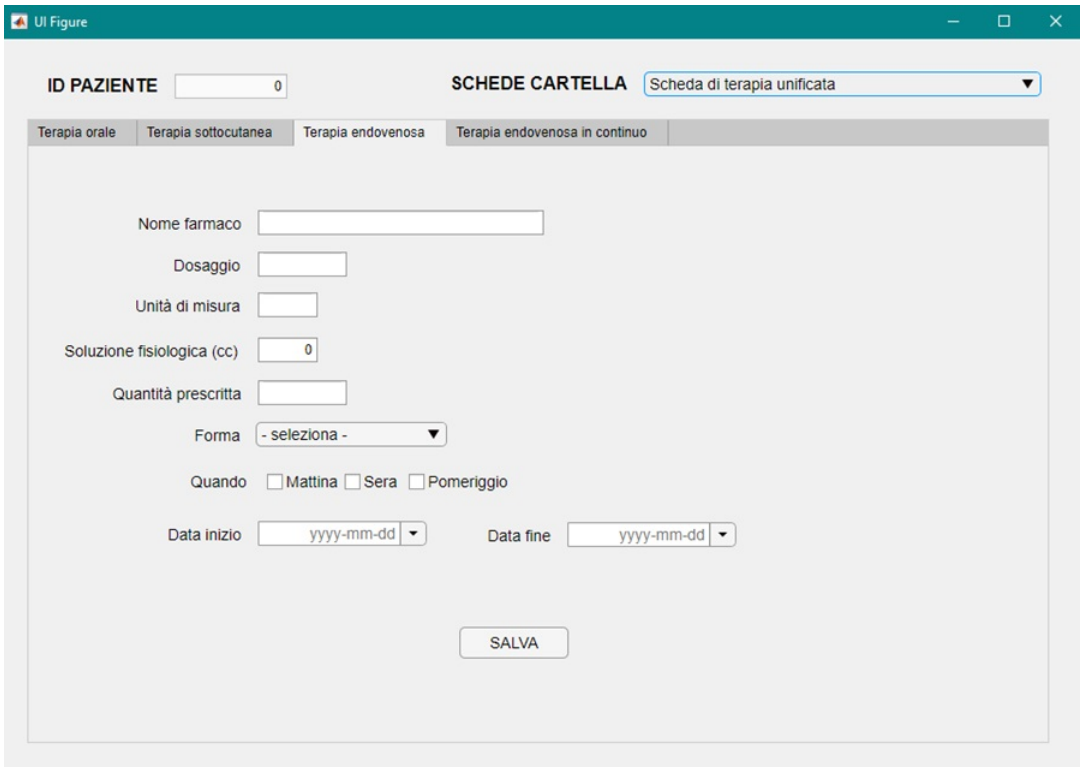


Figura 46: *Scheda di terapia unificata: Terapia endovenosa*

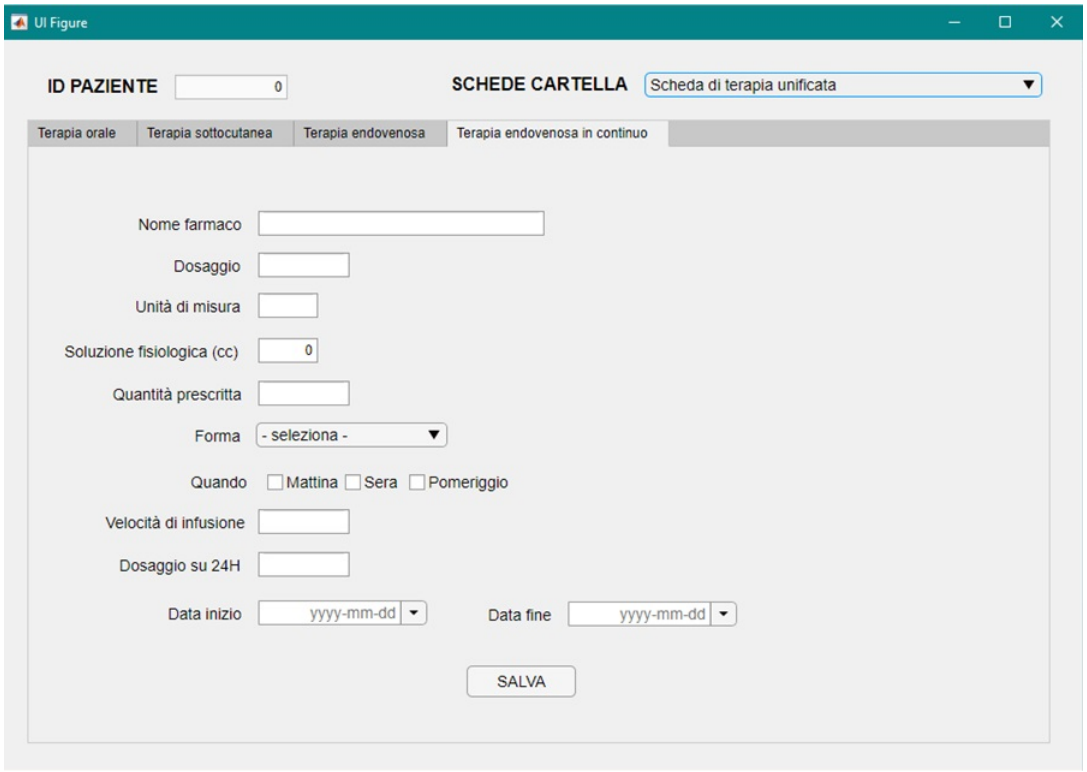


Figura 47: *Scheda di terapia unificata: Terapia endovenosa in continuo*

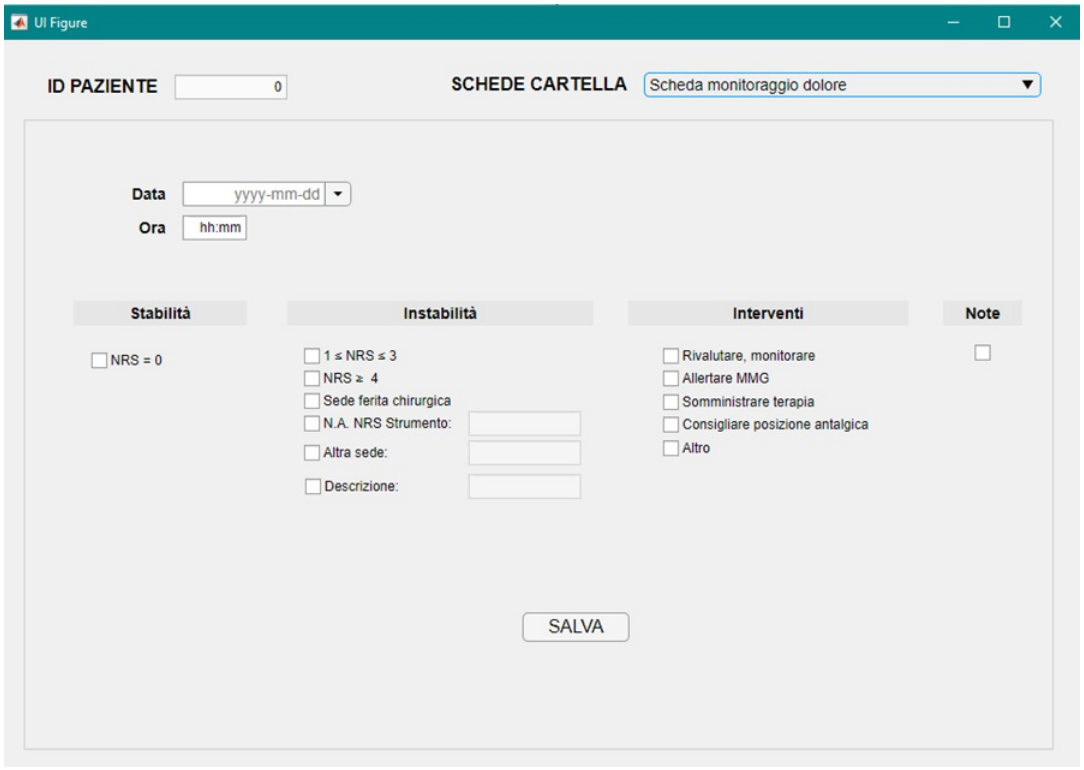


Figura 48: *Scheda monitoraggio dolore*

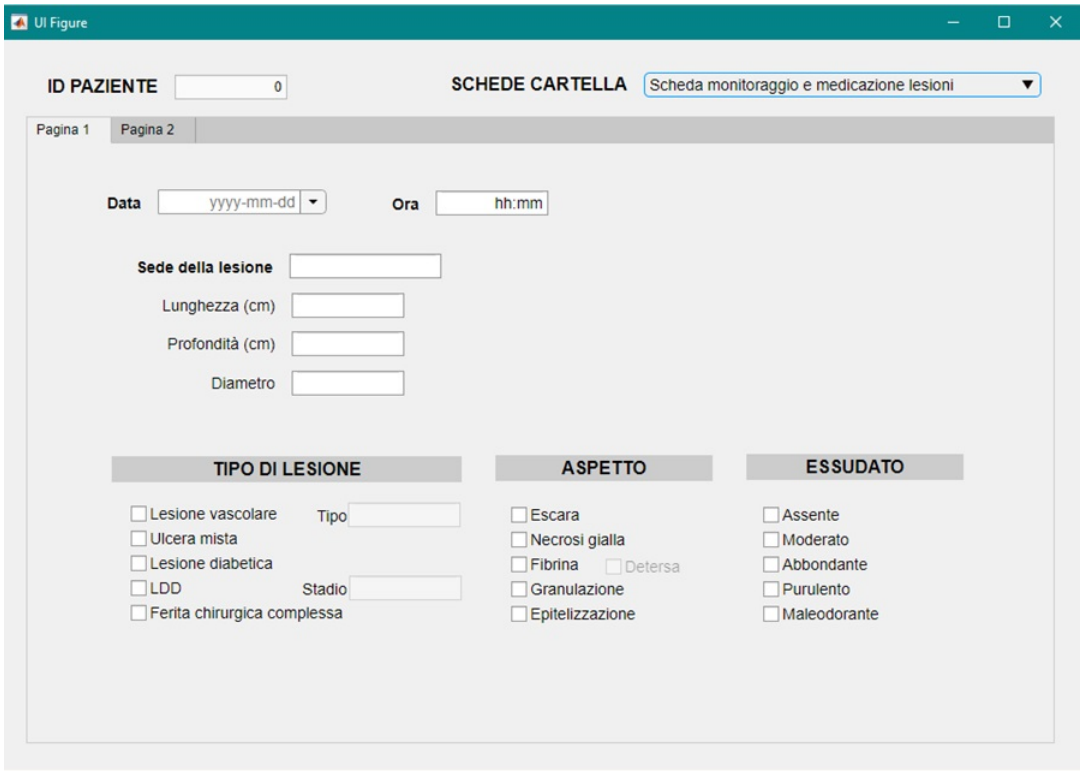


Figura 49: *Scheda monitoraggio e medicazione lesioni: Pagina 1*

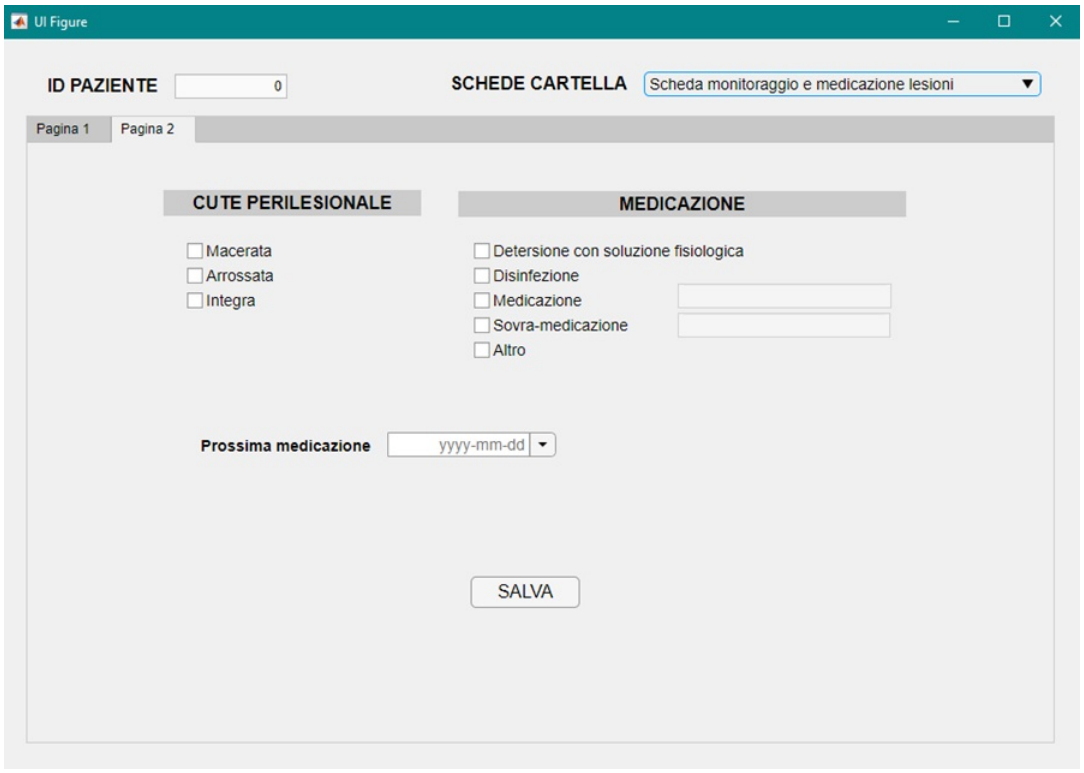


Figura 50: *Scheda monitoraggio e medicazione lesioni: Pagina 2*

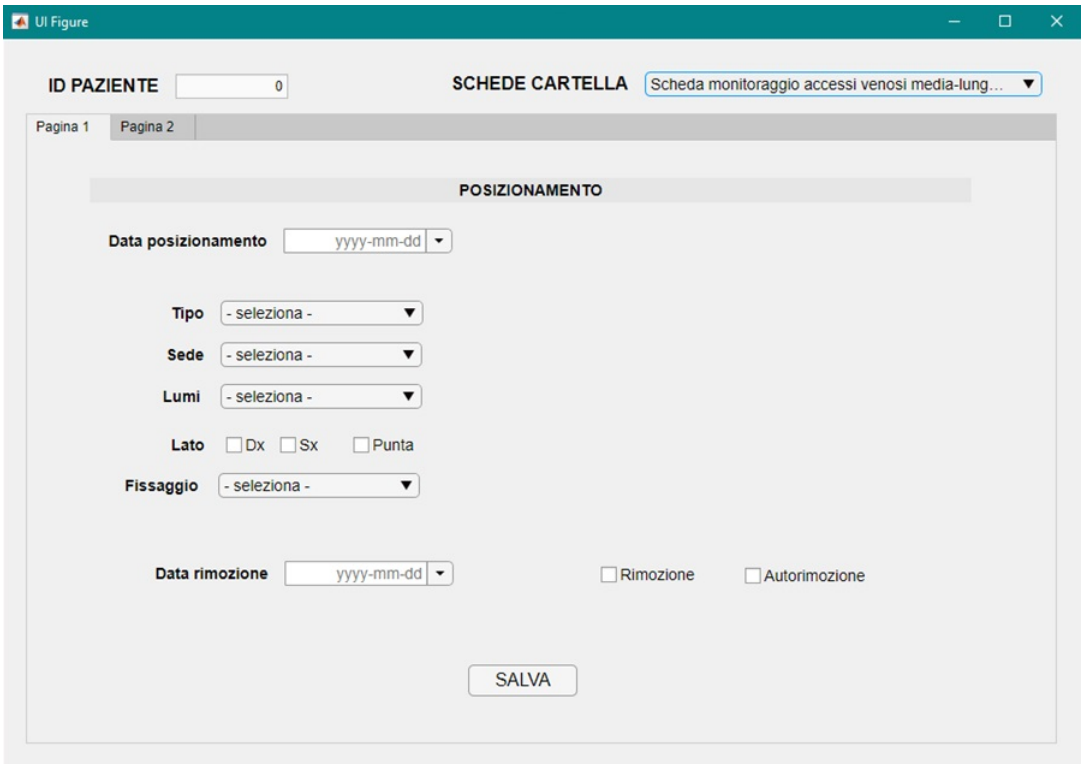


Figura 51: *Scheda monitoraggio accessi media-lunga permanenza: Pagina 1*

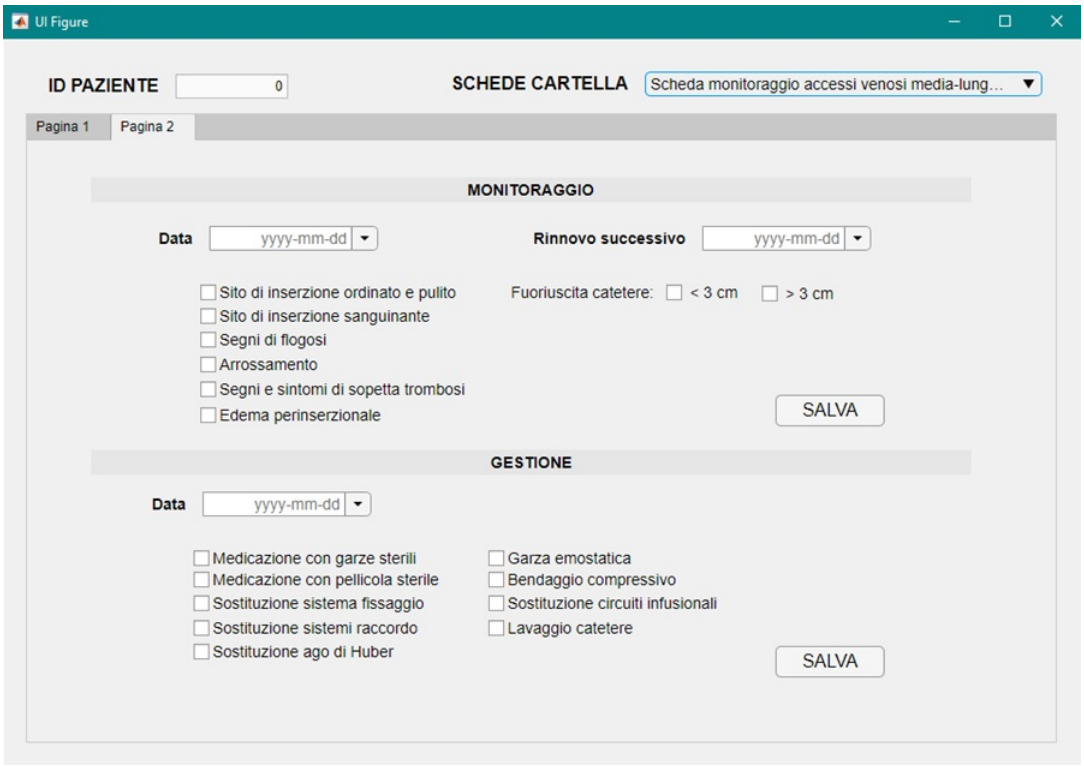


Figura 52: *Scheda monitoraggio accessi media-lunga permanenza: Pagina 2*

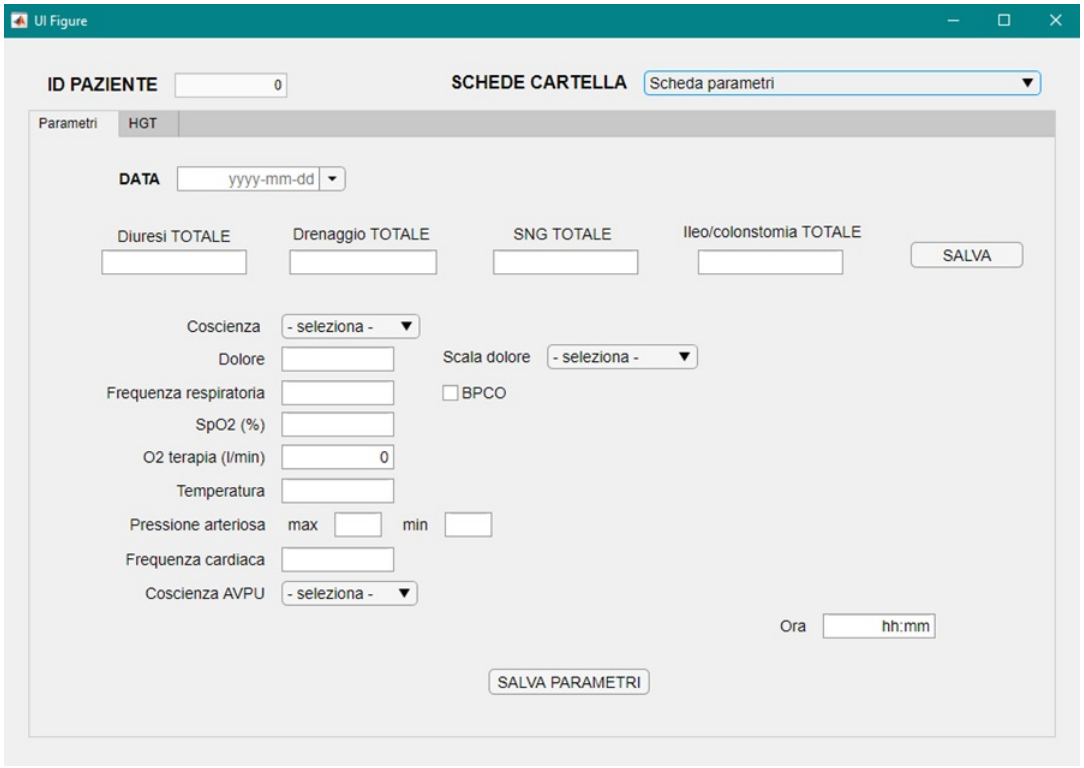


Figura 53: *Scheda parametri: Parametri*

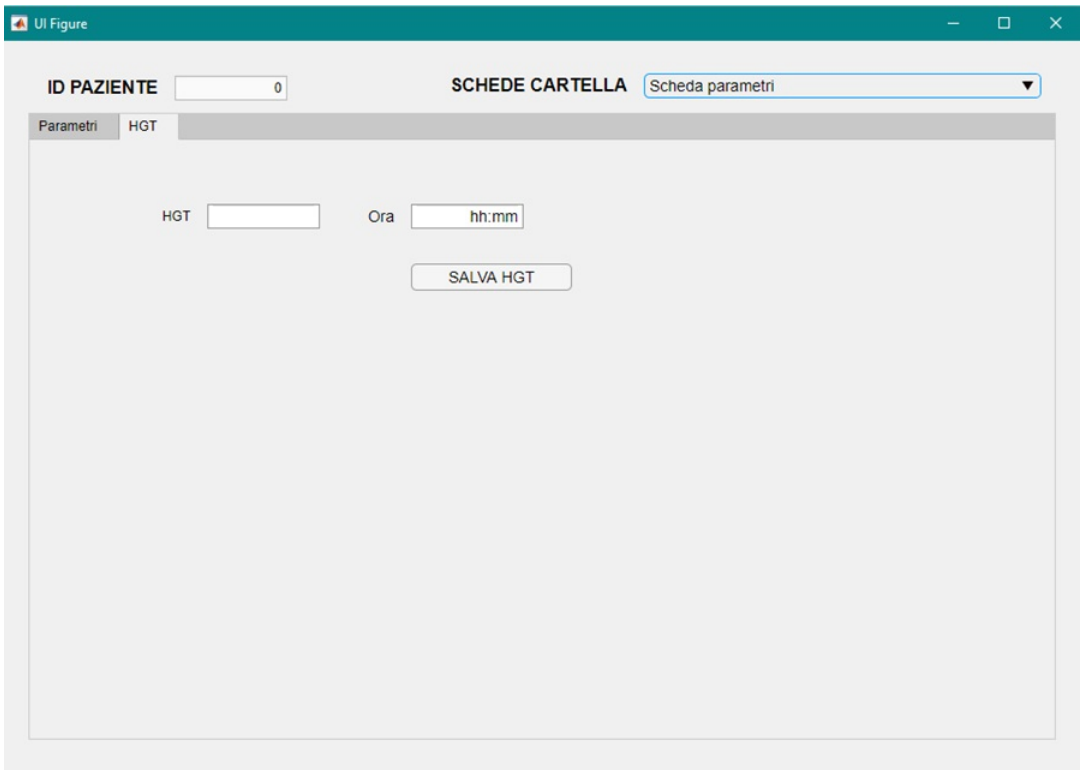


Figura 54: *Scheda parametri: HGT*

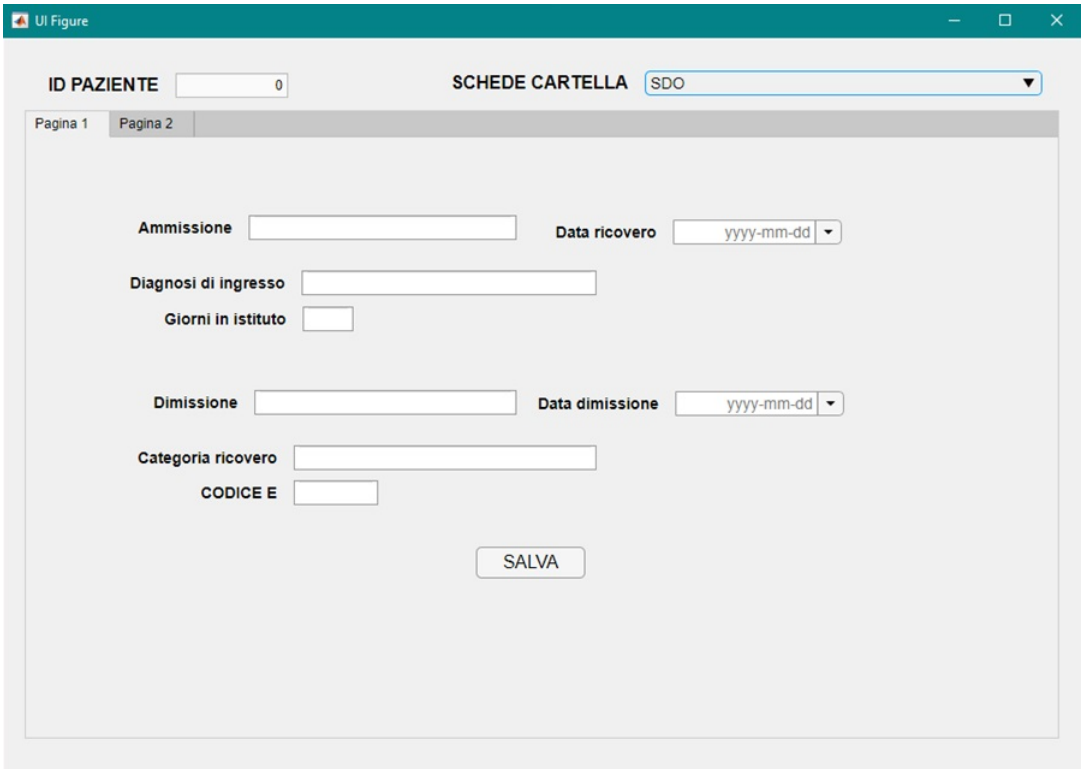


Figura 55: *Scheda di dimissione ospedaliera: Pagina 1*

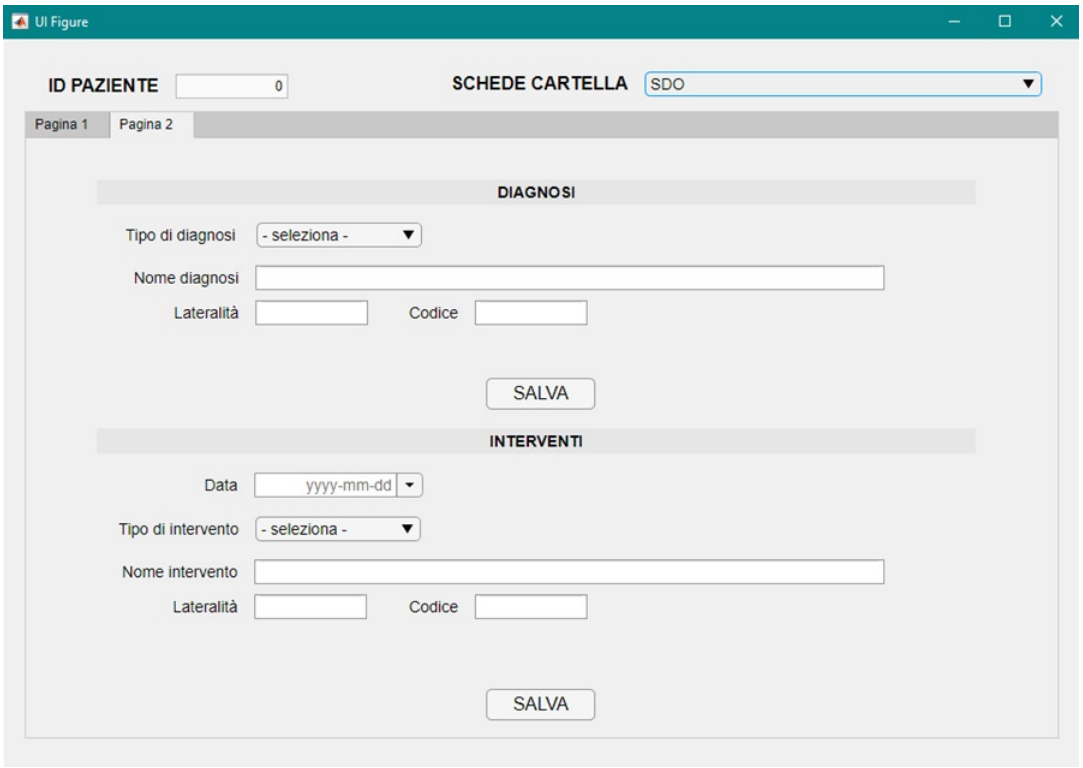


Figura 56: *Scheda di dimissione ospedaliera: Pagina 2*

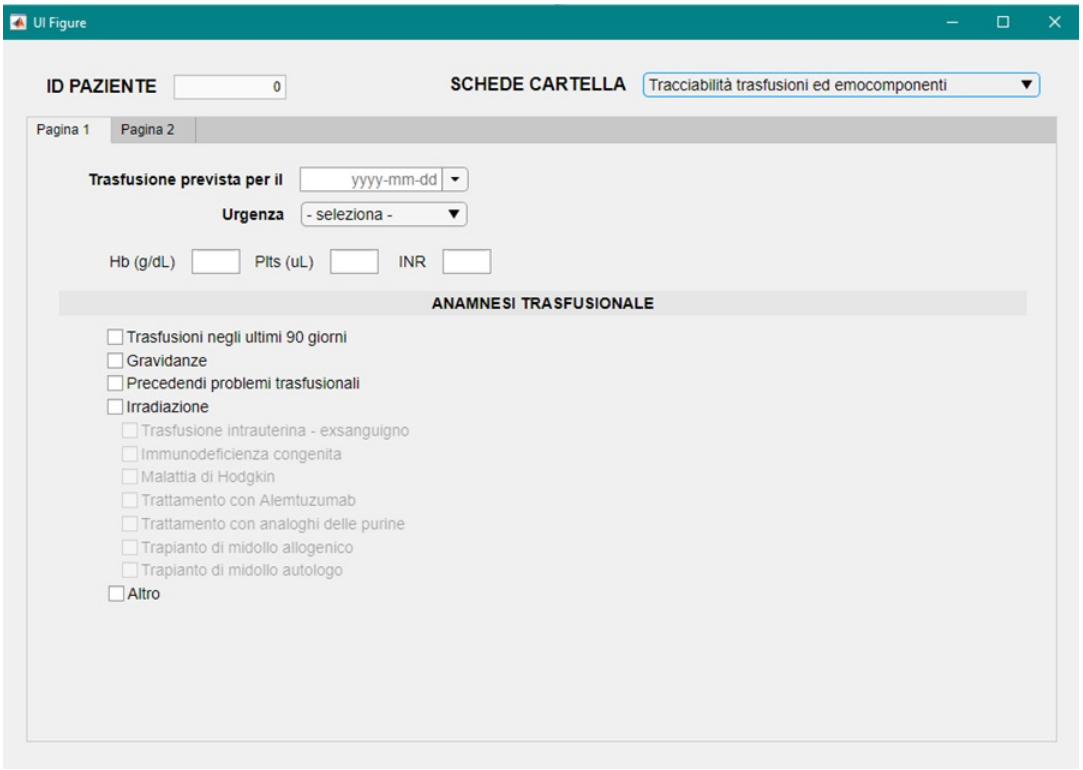


Figura 57: *Tracciabilità trasfusioni ed emocomponenti: Pagina 1*

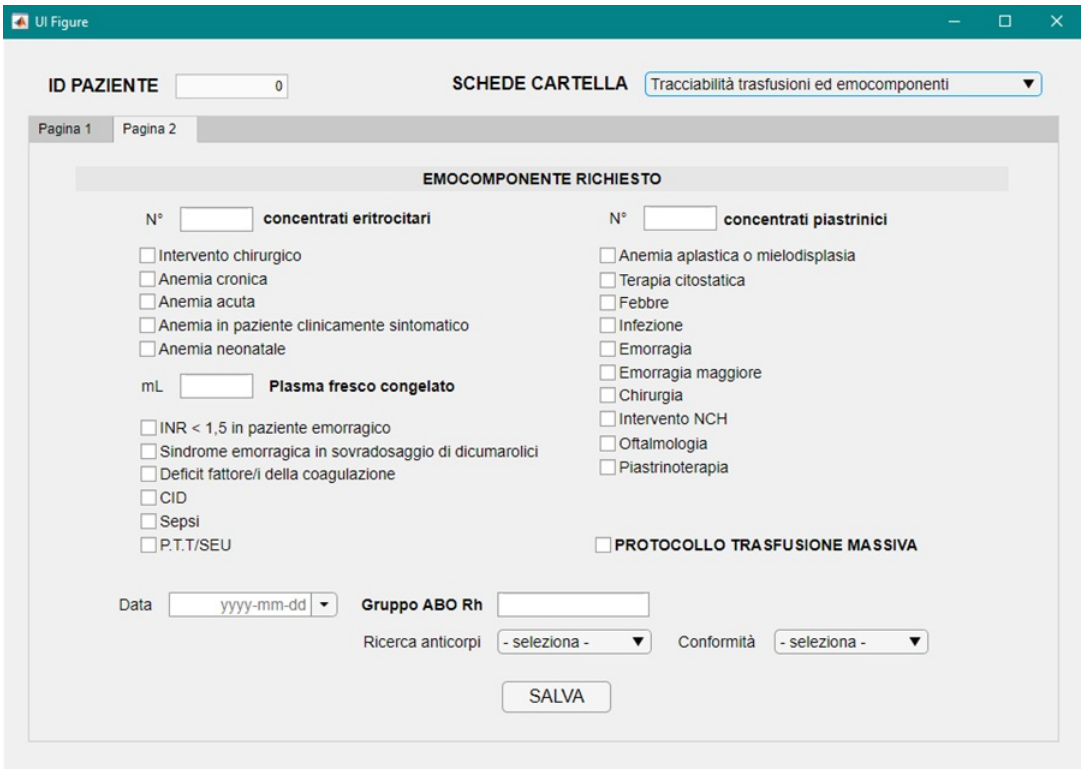


Figura 58: *Tracciabilità trasfusioni ed emocomponenti: Pagina 2*

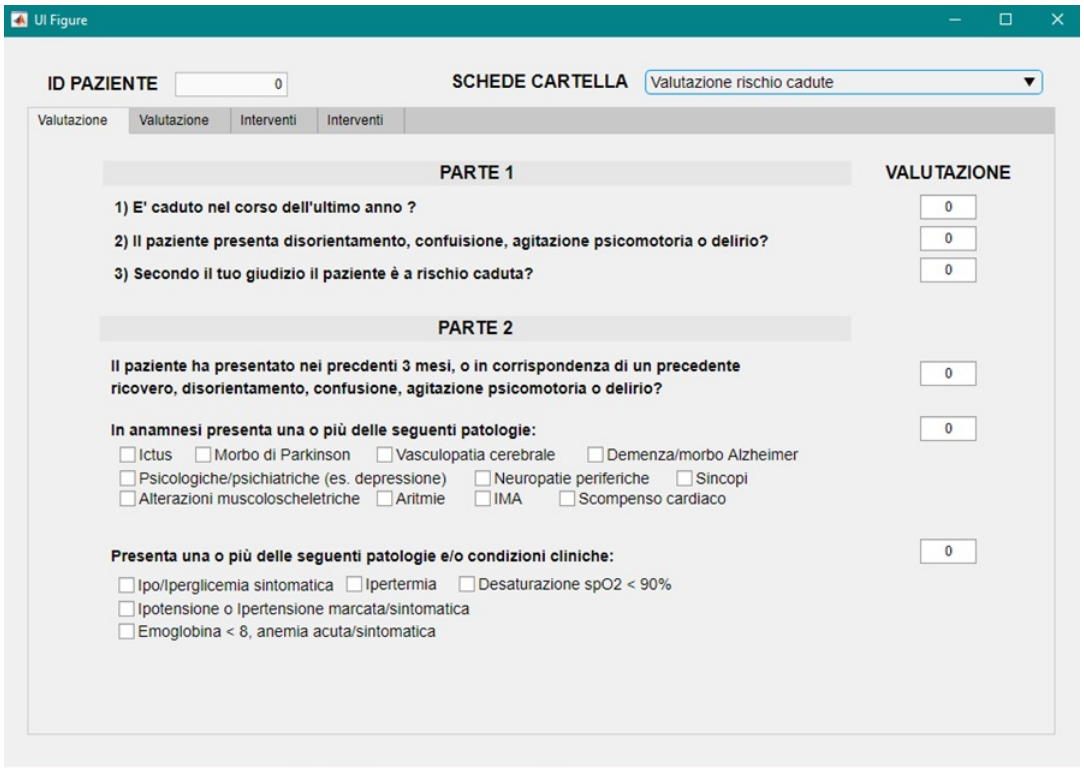


Figura 59: *Valutazione rischio cadute: Valutazione - pagina 1*

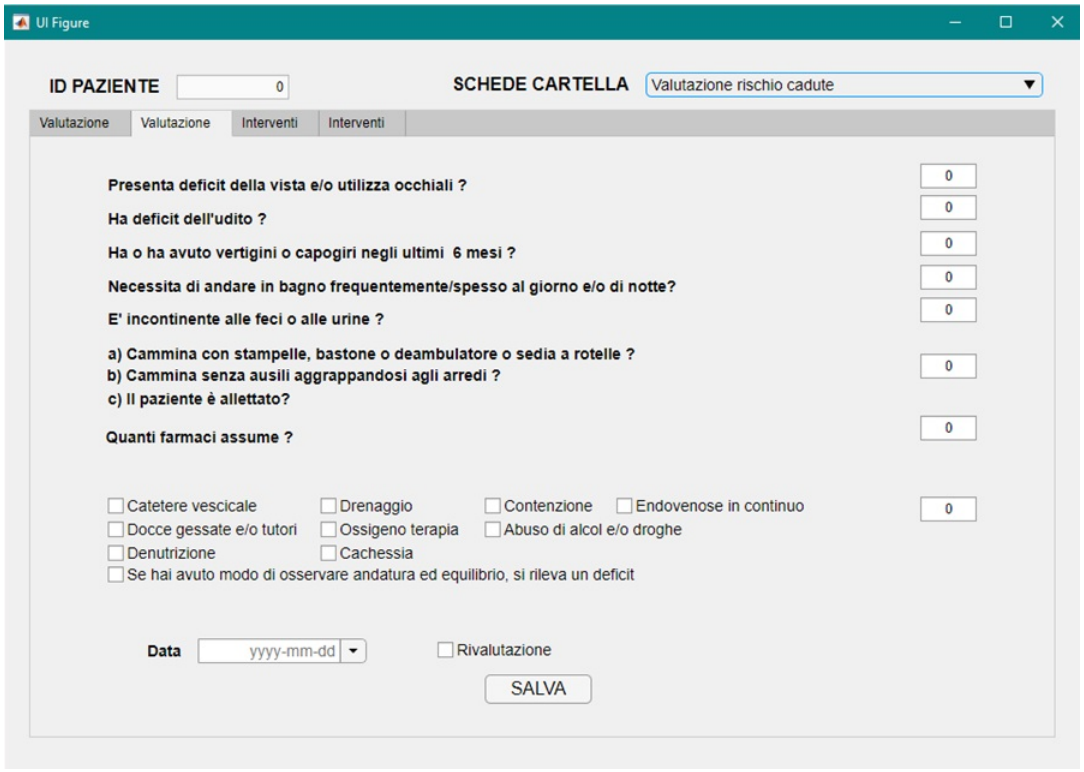


Figura 60: *Valutazione rischio cadute: Valutazione - pagina 2*

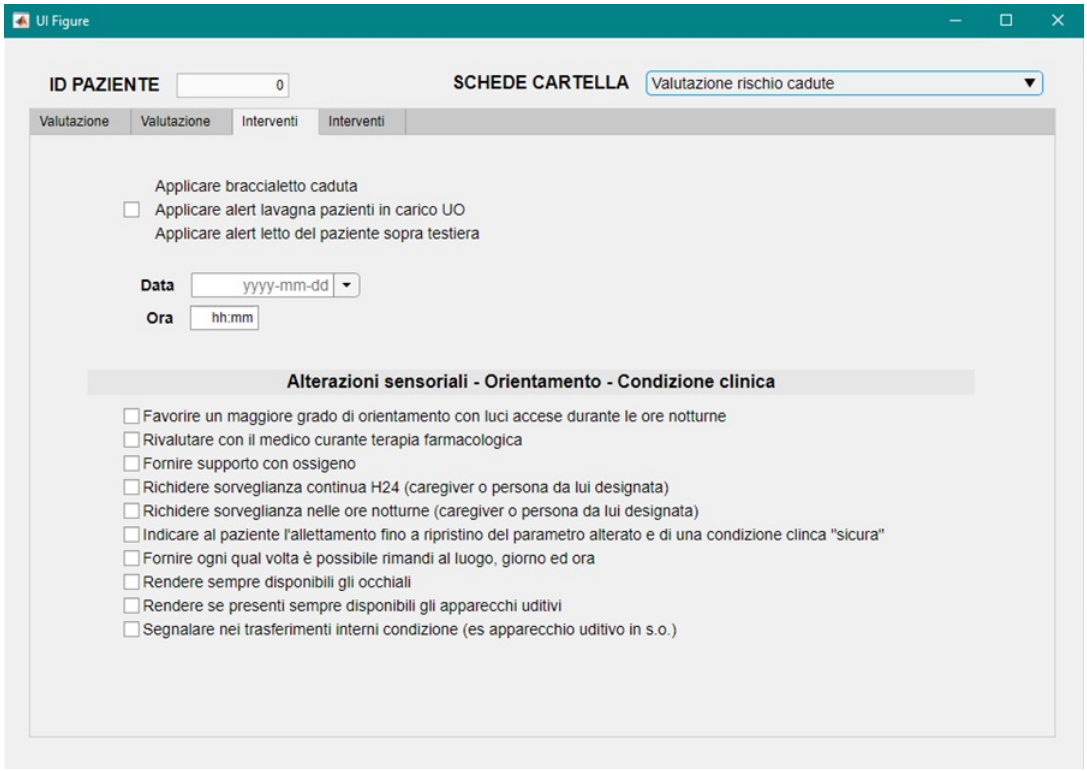


Figura 61: *Valutazione rischio cadute: Interventi - pagina 1*

Figura 62: *Valutazione rischio cadute: Interventi - pagina 2*

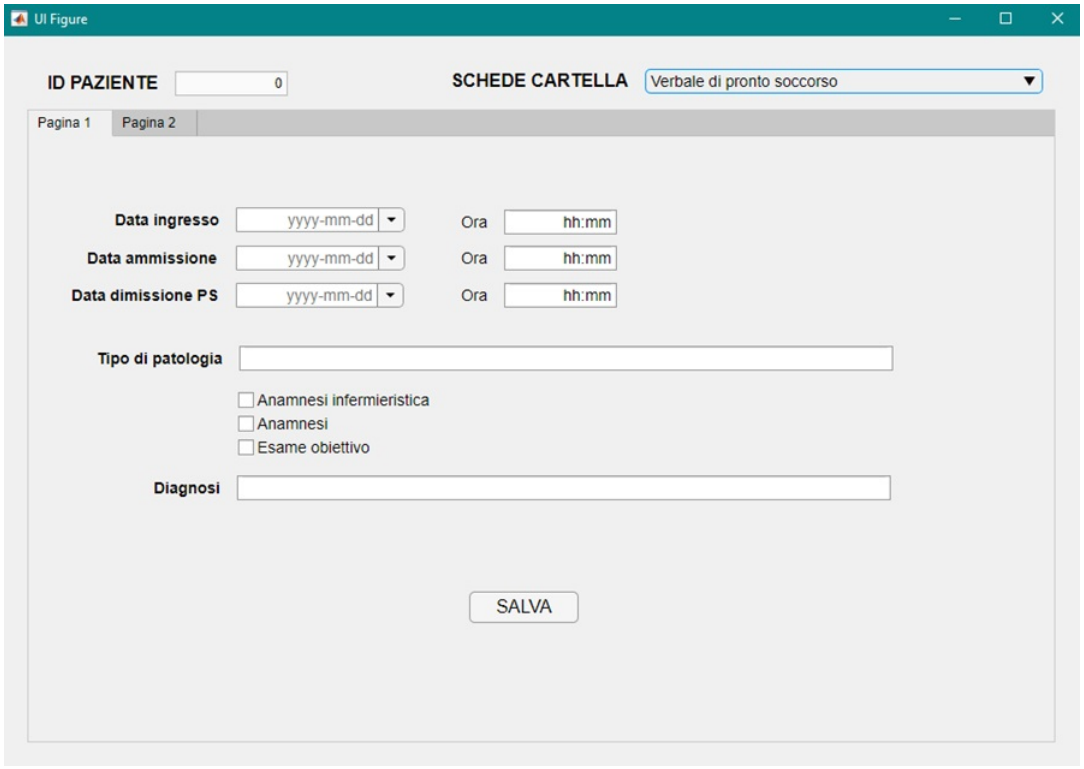


Figura 63: *Verbale di pronto soccorso: Pagina 1*

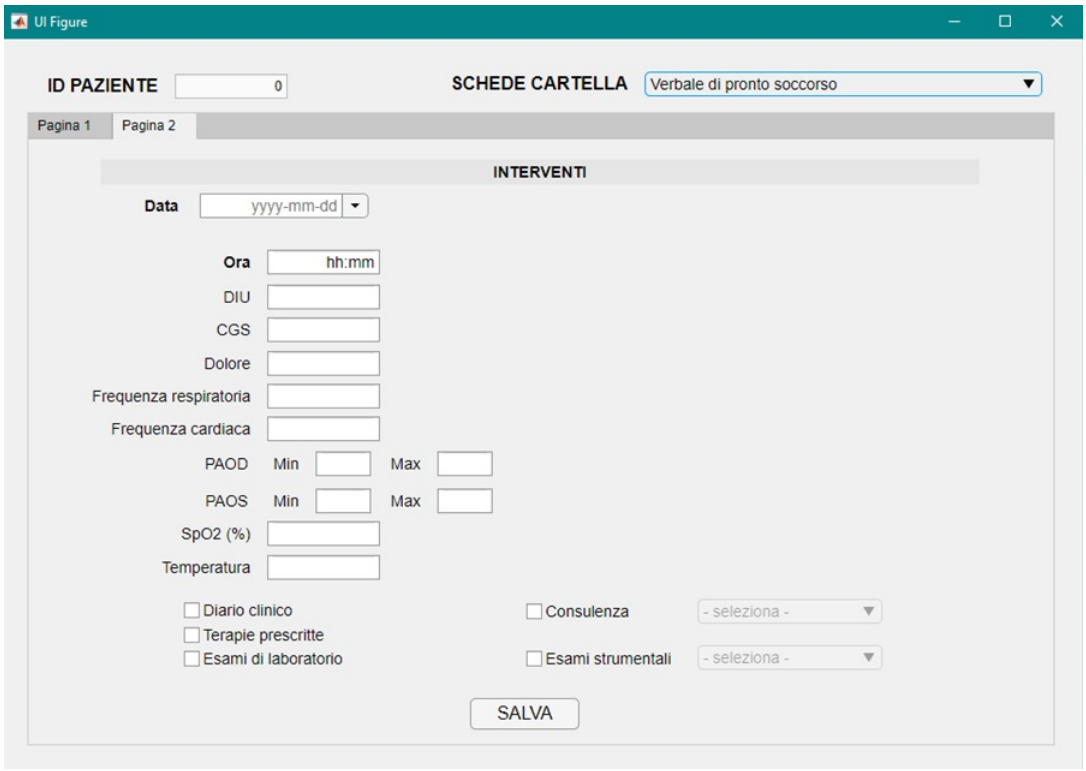


Figura 64: *Verbale di pronto soccorso: Pagina 2*

Appendice B

Confusion matrix

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	55	3
	Classe 1	18	5

Tabella B.1: Testo 2 - GATE

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	73	0
	Classe 1	0	8

Tabella B.2: Testo 2 - spaCy

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	73	2
	Classe 1	0	6

Tabella B.3: Testo 2 - TINT

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	40	4
	Classe 1	12	4

Tabella B.4: Testo 3 - GATE

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	52	2
	Classe 1	0	6

Tabella B.5: Testo 3 - spaCy

		Classe reale	
		Classe 0	**Classe 1**
Classe predetta	**Classe 0**	51	5
	Classe 1	1	3

Tabella B.6: Testo 3 - TINT

Bibliografia

[1] Tragni, E., Sala, F. e Casula, M. «IL PAZIENTE ANZIANO COMPLESSO: DATI EPIDEMIOLOGICI E DI CONSUMO DEI FARMACI Elders with multiple chronic conditions: epidemiology and drug use». In: *Giornale Italiano di Farmacoeconomia e Farmacoutilizzazione* 6.3 (2014), pp. 5–16.

[2] Istat. *Aspetti di vita degli over 75. Condizioni di salute, vicinanza ai figli, disponibilità di spazi esterni all'abitazione, cani in casa*. 2020.

[3] Lindmeier, C. e Brunier, A. «WHO: number of people over 60 years set to double by 2050; major societal changes required». In: *World Health Organization* (2015).

[4] Memini, F. *La complessità del paziente anziano in ospedale*. Università del Piemonte Orientale. 2020. URL: https://www.agingproject.uniupo.it/la-complessita-del-paziente-anziano-in-ospedale/ (visitato il 11 gen. 2021).

[5] Allegri, M., Bevere, F. et al. «Criteri di appropriatezza clinica, tecnologica e strutturale nell'assistenza del paziente complesso». In: *Quanderni del Ministero della salute*. Vol. 23. 2013.

[6] Spavanello, A. *Chi è il paziente fragile*. Fondazione Salvatore Maugeri. 2019. URL: https://www.fsm.it/il-paziente-fragile/ (visitato il 11 gen. 2021).

[7] Acquaviva, S. et al. *Relazione sullo Stato Sanitario del Paese 2009-2010*. Relazione. Roma: Ministero della Salute. Direzione Generale del Sistema Informativo e Statistico Sanitario, 2011, pp. 135–463.

[8] Berti, E., La Porta, P., Serra, V. et al. *Guida per i valutatori alla verifica dei Percorsi Diagnostico Terapeutici Assistenziali (PDTA) nell'ambito delle visite di accreditamento*. Regione Emilia-Romagna. Servizio Sanitario Regionale Emilia-Romagna. 2013.

[9] *Percorsi diagnostici terapeutici assistenziali (PDTA)*. European Pathway Association, E-P-A. 2015. URL: http://e-p-a.org/sito-internet-e-p-a/percorsi-diagnostici-terapeutici-e-assistenziali-pdta/ (visitato il 12 gen. 2021).

[10] *Process mining*. Wikipedia. URL: https://it.wikipedia.org/wiki/Process_mining (visitato il 2 mar. 2021).

[11] *Process mining*. IONOS. 2019. URL: https://www.ionos.it/digitalguide/online-marketing/vendere-online/process-mining/ (visitato il 2 mar. 2021).

[12] Martelli, S., Laura, B. et al. *Raccomandazioni per la costruzione di Percorsi Diagnostico Terapeutici Assistenziali (PDTA) e Profili Integrati di Cura (PIC) nelle Aziende Sanitarie della Regione Piemonte*. Regione Piemonte, Agenzia Regionale per i Servizi Sanitari (Aress). 2007.

[13] Silva Rebuge, A. J. da. «Business Process Analysis in Healthcare Environments». Dissertation for the degree of Master of Science. Technical University of Lisbon, 2012.

[14] Eisner, M. *How to Perform a Business Process Analysis*. Process Maker. 2020. URL: https://www.processmaker.com/blog/how-to-perform-a-business-process-analysis (visitato il 16 gen. 2021).

[15] Aalst et al. «Process Mining Manifesto». In: vol. 99. Ago. 2011, pp. 169–194. ISBN: 978-3-642-28107-5. DOI: 10.1007/978-3-642-28108-2_19.

[16] *Data mining*. Intelligenza Artificiale, Il portale dedicato all'intelligenza artificiale. URL: https://www.intelligenzaartificiale.it/data-mining/ (visitato il 16 gen. 2021).

[17] *Process mining e Business Process Management: differenze e interazioni*. Integris. 2019. URL: https://blog.integris.it/process-mining-e-business-process-management-differenze-e-interazioni (visitato il 16 gen. 2021).

[18] Zeppilli, V. «La cartella clinica». In: *Studio Cataldi: il diritto quotidiano* (2018).

[19] Rudnitckaia, J. «Process Mining. Data science in action». In: *University of Technology, Faculty of Information Technology* (2015), pp. 1–11.

[20] Celesti, R. *LA CARTELLA CLINICA*. Università degli Studi di Genova, DISSAL - Dipartimento di Scienze della Salute, Sezione di Medicina Legale.

[21] *La cartella clinica - La conservazione, gli archivi e la circolazione*. Ordine provinciale dei medici chirurghi e degli odontoiatri di Messina.

[22] Ghirardini, A. et al. *Sviluppo di un modello di Cartella Paziente Integrata*. Ministero della Salute. 2018.

[23] Paternostro, G. «Analisi di cartelle cliniche di pazienti geriatrici sottoposti ad intervento chirurgico». Tesi di Laurea Magistrale. Politecnico di Torino, 2020.

[24] Balestra, G. *Construction process*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

[25] *Che cos'è un database*. Oracle. URL: https://www.oracle.com/it/database/what-is-database/ (visitato il 15 gen. 2021).

[26] *Progettare i Database - SQL e PHP*. Altervista. URL: http://www.silvioionta.altervista.org/SilvioDBEXE/index.html (visitato il 15 gen. 2021).

[27] Lorenzi, A. e Cavalli, E. «L'architettura a 3 livelli dei sistemi per database». In: *Progettazione dei database, linguaggio SQL, dati in rete: Access, MySQL, Pagine ASP, Pagine Php*. A cura di Atlas. 2011. Cap. 1. Organizzazione degli archivi e basi di dati.

[28] Balestra, G. *Requirements analysis*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

[29] Balestra, G. *Requirements elicitation*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

[30] Balestra, G. *Testing*. Materiale didattico. Politecnico di Torino, BioLab, DET - Dipartimento di Elettronica e Telecomunicazioni, 2019.

[31] *Natural Language Processing*. Wikipedia. URL: https://en.wikipedia.org/wiki/Natural_language_processing (visitato il 25 gen. 2021).

[32] Cambria, E. e White, B. «Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]». In: *IEEE Computational Intelligence Magazine* 9.2 (2014), pp. 48–57. DOI: 10.1109/MCI.2014.2307227.

[33] Rosati, S. *Natural Language Processing*. Politecnico di Torino.

[34] Tono, R. «Natural Language Processing e tecniche semantiche per il supporto alla diagnosi: un esperimento». Tesi di Laurea Magistrale. Università degli studi di Padova, 2010.

[35] Wolff, R. *11 Natural Language Processing (NLP) Applications in Business*. MonkeyLearn. 2020. URL: https://monkeylearn.com/blog/natural-language-processing-applications/ (visitato il 26 gen. 2021).

[36] *Natural Language Processing Applications*. Expert.ai. 2020. URL: https://www.expert.ai/blog/natural-language-processing-applications/ (visitato il 26 gen. 2021).

[37] Guts, Y. *Natural Language Processing*. NLP Morning@Lokiha, 2016.

[38] Schütze, Hinrich, Manning, Christopher D e Raghavan, Prabhakar. *Introduction to information retrieval*. Vol. 39. Cambridge University Press Cambridge, 2008. URL: https://nlp.stanford.edu/IR-book/.

[39] Elia, F. *Constituency Parsing vs Dependency Parsing*. Baeldung. 2020. URL: https://www.baeldung.com/cs/constituency-vs-dependency-parsing (visitato il 26 gen. 2021).

[40] Palshikar, G. K. «Techniques for Named Entity Recognition: A Survey». In: vol. 1. Gen. 2012, pp. 191–217. ISBN: 9781466636057. DOI: 10.4018/978-1-4666-3604-0.ch022.

[41] Singh, S. «Natural language processing for information extraction». In: *arXiv preprint arXiv:1807.02383* (2018).

[42] *Entity Linking*. Wikipedia. URL: https://en.wikipedia.org/wiki/Entity_linking (visitato il 27 gen. 2021).

[43] Laporte, E. «Symbolic Natural Language Processing». In: *Applied Combinatorics on Words*. A cura di Lothaire. Cambridge University Press, 2005, pp. 164–209.

[44] *Feature Extraction in Natural Language Processing with Python*. Medium. 2019. URL: https://medium.com/@eiki1212/feature-extraction-in-natural-language-processing-with-python-59c7cdcaf064 (visitato il 28 gen. 2021).

[45] Wang, Y., Liu, S. et al. «A comparison of word embeddings for the biomedical natural language processing». In: *Journal of biomedical informatics* 87 (2018), pp. 12–20.

[46] *Word2vec*. Wikipedia. URL: https://it.wikipedia.org/wiki/Word2vec (visitato il 28 gen. 2021).

[47] *Ambiguità sintattica*. Wikipedia. URL: https://it.wikipedia.org/wiki/Ambiguit%5C%C3%5C%A0 (visitato il 28 gen. 2021).

[48] Viani N. & Velupillai, S. *Natural language processing methods for clinical text*. NHS - National Institute for Health Research.

[49] Derczynski, L. «Complementarity, F-score, and NLP Evaluation». In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 2016, pp. 261–266.

[50] Powers, D. «Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation». In: *Journal of Machine Learning Technologies* 2(1) (2008), pp. 37–63.

[51] Cunningham, H. et al. «Developing Language Processing Components with GATE Version 9 (a User Guide)». In: (2002). URL: https://gate.ac.uk/sale/tao/split.html (visitato il 2 feb. 2021).

[52] *LinguA*. Italian Natural Language Processing Lab. URL: http://www.italianlp.it/demo/linguistic-annotation-tool (visitato il 2 feb. 2021).

[53] *ISST-TANL Tagsets*. URL: http://www.italianlp.it/docs/ISST-TANL-POStagset.pdf (visitato il 2 feb. 2021).

[54] *ISST-TANL dependency tagset*. URL: http://www.italianlp.it/docs/ISST-TANL-DEPtagset.pdf (visitato il 2 feb. 2021).

[55] Dell'Orletta, F. «Ensemble system for Part-of-Speech tagging». In: *Proceedings of EVALITA 2009 - Evaluation of NLP and Speech Tools for Italian 2009* 9 (dic. 2009), pp. 1–8.

[56] Attardi, G. e Dell'Orletta, F. «Reverse Revision and Linear Tree Combination for Dependency Parsing». In: *NAACL-HLT 2009 – North American Chapter of the Association for Computational Linguistics – Human Language Technologies*. Boulder, Colorado: Association for Computational Linguistics, giu. 2009, pp. 261–264.

[57] Attardi, G. et al. «Accurate Dependency Parsing with a Stacked Multilayer Perceptron». In: *Proceedings of EVALITA – Evaluation of NLP and Speech Tools for Italian 2009* 9 (dic. 2009), pp. 1–8.

[58] *TINT - The Italian NLP Tool*. DH, Digital Humanities. URL: https://dh.fbk.eu/research/tint/ (visitato il 2 feb. 2021).

[59] Palmero, A. e Moretti, G. «Italy goes to Stanford: a collection of CoreNLP modules for Italian». In: *ArXiv e-prints* (2016). Provided by the SAO/NASA Astrophysics Data System. eprint: 1609.06204. URL: http://adsabs.harvard.edu/abs/2016arXiv160906204P.

[60] *spaCy: Industrial-strength NLP*. URL: https://v2.spacy.io/ (visitato il 2 feb. 2021).

[61] *Stanza – A Python NLP Package for Many Human Languages*. Stanford NLP Group. URL: https://stanfordnlp.github.io/stanza/ (visitato il 2 feb. 2021).

[62] Qi, P. et al. «Stanza: A Python Natural Language Processing Toolkit for Many Human Languages». In: Association for Computational Linguistics (ACL) System Demonstrations, 2020.

[63] Bosco, C. et al. *UD Italian ISDT*. Universal Dependencies.

[64] Basili, R., Pazienza, M.T. e Zanzotto, F.M. «Evaluating a robust parser for Italian». In: *Carroll, Basili, et al* (1998).

Ringraziamenti

Per concludere questo elaborato, vorrei dedicare questo spazio a chi durante questi mesi di lavoro con pazienza e delicatezza mi ha appoggiata e supportata.

Innanzitutto vorrei ringraziare la mia relatrice, la Professoressa Balestra, che mi ha dato la possibilità di apportare il mio contributo allo sviluppo di questo importante progetto che il Politecnico di Torino sta portando avanti in collaborazione con l'Ospedale Humanitas Gradenigo di Torino. La ringrazio inoltre per la disponibilità mostrata nelle occasioni di confronto, che sono state fondamentali per proseguire il lavoro.

Ringrazio la Professoressa Rosati in qualità di correlatrice per avermi fornito la documentazione e per avermi guidato soprattutto nella seconda parte del lavoro, per definire il percorso e le attività da affrontare nell'ambito dello studio condotto sulle tecniche di Natural Language Processing.

Vorrei ringraziare anche i miei amici e colleghi di università con i quali ho affrontato tutti i progetti didattici e grazie ai quali questi anni sono stati davvero pieni di felicità, di spensieratezza e di arriccchimento spirituale e culturale. Un grazie speciale va a Stefania, con la quale è nato un rapporto di amicizia importante, soprattutto in questi mesi di distanza.

Un ringraziamento particolare, ma comunque insufficiente, va alla mia famiglia e ai miei genitori, che nonostante tutte le difficoltà mi hanno consentito di poter intraprendere questo percorso di studi, lasciandomi libera di poter fare le mie scelte e di poter prendere le mie decisioni. Grazie per aver avuto pazienza e fiducia in me, spero di potervi presto ripagare di tutti gli sforzi fatti in questi anni.

Infine, ci tengo tanto a ringraziare Riccardo, la persona che in questi cinque anni è stata spettatrice della mia crescita personale ed accademica e che mi ha aiutato, guidato ed accompagnato sia nella realizzazione di questo elaborato sia nel superamento di tutti gli ostacoli e di tutti i momenti importanti che fino ad ora ho dovuto superare. Grazie per essermi stato sempre accanto, anche a distanza, e per avermi sorretto e spronato nei periodi difficili.