Il dottore artificiale

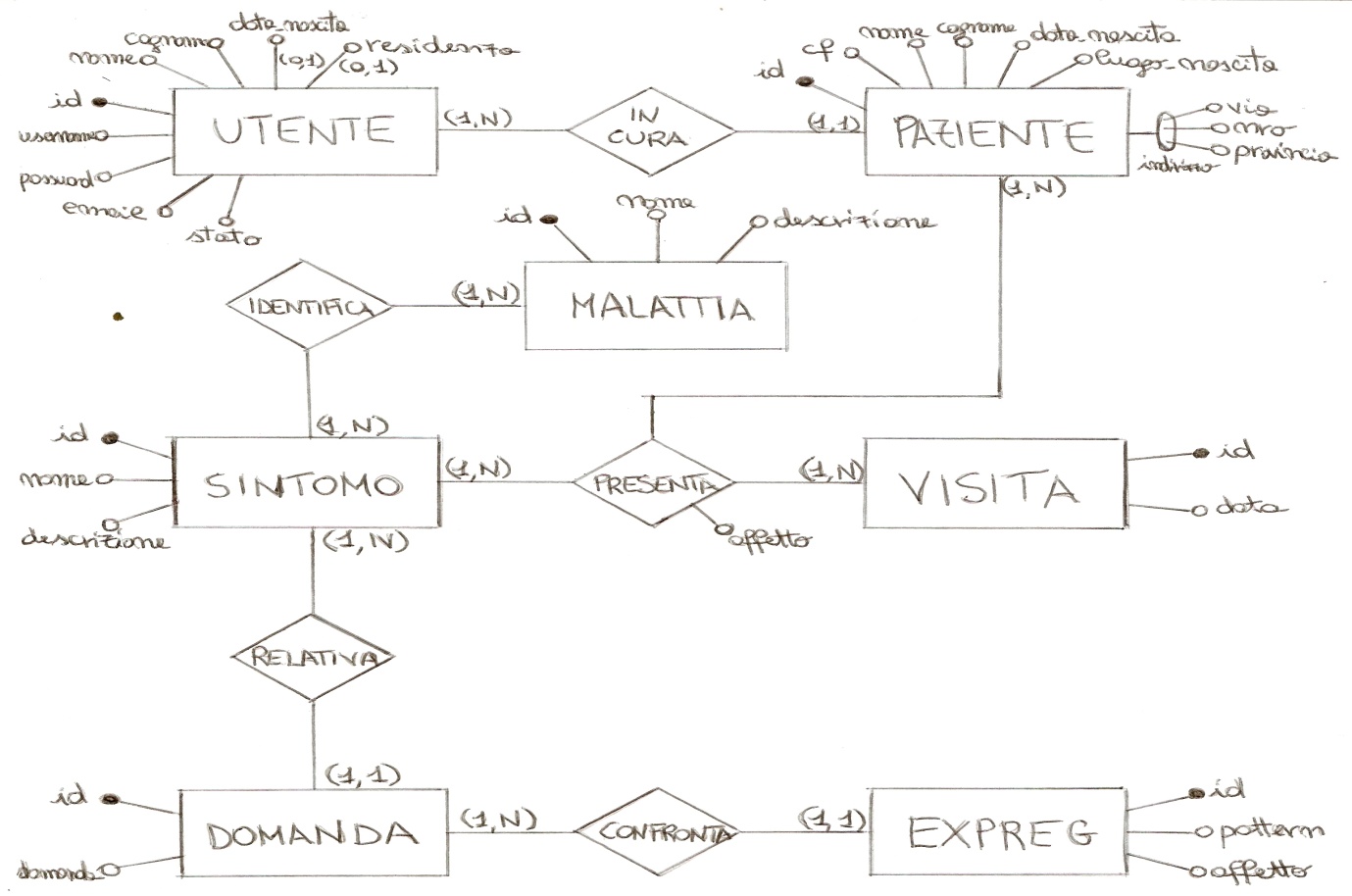
1. Requisiti applicazione

L'applicazione realizza in maniera semplificata un sistema esperto in ambito medico basato su regole, con lo scopo di identificare le malattie di cui un paziente può essere affetto partendo dai sintomi che presenta.

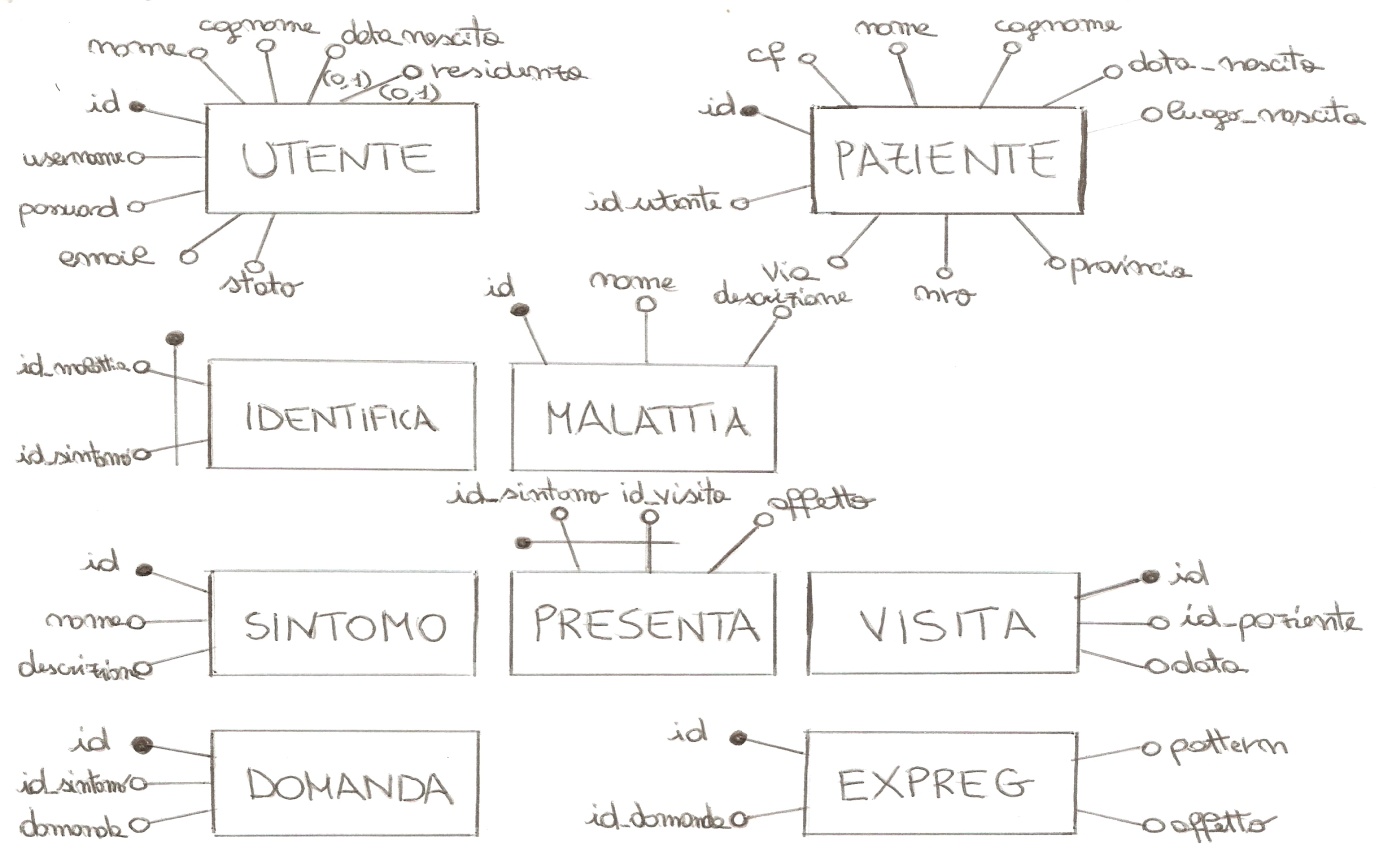
L'identificazione dei sintomi avviene attraverso un 'dialogo' tra applicazione e paziente in linguaggio naturale. Il 'dialogo' sarà caratterizzato da una serie di domande molto specifiche a cui il paziente dovrà rispondere con una breve frase. Le domande devono essere poste con lo scopo di identificare la malattia di cui il paziente è affetto con il minor numero di domande necessario. Le risposte ottenute dal paziente vengono analizzate attraverso un ampio uso delle espressioni regolari, allo scopo di capire se il paziente è affetto o meno da quel sintomo.

Ogni utente iscritto al sito come un dottore a tutti gli effetti ha in cura più pazienti e per ognuno di essi ha a disposizione la possibilità di visualizzarne le visite precedentemente svolte, attraverso una rappresentazione sintetica dei sintomi su cui erano state poste domande e le malattie pertanto possibili.

1. Progetto concettuale



1. Progetto logico



1. Descrizione di ogni altro aspetto significativo del sistema

4.1 Sistema esperto

Un **sistema esperto** è un programma che cerca di riprodurre le prestazioni di una o più persone esperte in un determinato campo di attività, ed è un'applicazione o una branca dell'intelligenza artificiale.

I programmi utilizzati dai sistemi esperti sono in grado di porre in atto procedure di **confronto tra le** **condizioni particolari del caso** **e quelle conosciute** (per previo inserimento del programmatore o per casi precedentemente risolti) traendo così conclusioni adeguate alla risoluzione di problemi particolarmente complessi. Ciò implica che tale sistema possa avvalersi in modo risoluto e autorevole della conoscenza di cui è fornito che soggiace al corretto funzionamento del programma, cosicché sia capace di superare le incertezze e le difficoltà su cui volge la propria attività.

Il sistema esperto si compone in sostanza di tre sezioni:

* una **base di conoscenza**, in cui sono accumulate le regole deduttive e i dettami procedurali di cui il sistema si serve nel suo operato.
* un **motore inferenziale**, in cui il programma si occupa di applicare in concreto le nozioni contenute nella base dati (facendo il confronto suddetto).
* un'**interfaccia utente**, che permette l’interazione fra il soggetto umano e il programma che deve dare risposta ai sui problemi.

Queste informazioni sono piuttosto generiche, ed estremamente flessibili ciò è necessario visto l’ampio campo applicativo dei sistemi esperti.

Evidentemente non possono esistere sistemi capaci di soddisfare nella sua interezza il tipo di conoscenza e il processo decisionale che dovrebbe caratterizzare un esperto. Difatti, nella maggior parte dei programmi non è possibile attenere il rigore connaturato ad un [algoritmo](http://it.wikipedia.org/wiki/Algoritmo), in quanto nelle situazioni altamente complicate sarebbe troppo dispendioso analizzare ogni possibilità; si ricorre così allo stratagemma dell'[euristica](http://it.wikipedia.org/wiki/Euristica), che, tramite ragionamenti approssimativi, sacrifica la sicurezza dell'algoritmo per giungere a risultati altamente probabili, ma comunque fallibili.

È da notare che un sistema esperto non è "intelligente" nel senso comune della parola, ossia in modo creativo. Le deduzioni di un sistema esperto non possono uscire dall'insieme di nozioni immesse inizialmente e dalle loro conseguenze. Ciò che li rende utili è che, come i calcolatori elettronici, possono maneggiare una gran quantità di dati molto velocemente e tenere quindi conto di una miriade di regole e dettagli che un esperto umano può ignorare, tralasciare o dimenticare.

4.1.1 Sistemi esperti basati su regole

I sistemi esperti basati su regole sono dei programmi composti da regole nella forma:

IF <condizione> THEN <azione>

La parte “if” presenta una **condizione** che deve essere vera affinché la regola sia applicabile. La parte “then” presenta un’**azione**, che viene eseguita quando la regola è applicabile. Una regola può contenere più condizioni e/o più azioni attraverso l’uso di AND (congiunzione) e OR (disgiunzione).

Per esempio, supponiamo di avere un problema di salute, forniamo al sistema esperto i seguenti fatti:

* ho mal di testa
* sono raffreddato
* la temperatura corporea è di 38 °C

il sistema esperto assume i fatti e sceglie una regola così formata:

IF ((mal di testa) AND (raffreddore) AND (temperatura = 38)) THEN

(Il suo problema è INFLUENZA)

L’ordine in cui le regole sono selezionate dipende dal motore inferenziale. In particolare, con la concatenazione in avanti (**forward chaining**) l’interprete cerca nella base di conoscenza i fatti coincidenti con le premesse delle regole, deducendo poi nuovi fatti e conclusioni. Con la concatenazione all’indietro (**backward chaining**), l’interprete cerca di dimostrare un obiettivo tentando di confermare le condizioni che conducono a quell’obiettivo.

Questo sistema è molto diffuso perché permette con molta semplicità di riprodurre il ragionamento umano e ogni regola nella base di conoscenza prescinde dalle altre. Tuttavia non tutte le forme di conoscenza sono esprimibili sotto forma di regole.

4.1.2 Sistemi esperti basati su alberi

Un sistema esperto basato su alberi, partendo da un nodo iniziale, che rappresenta la condizione di partenza, genera un ampio **albero** che raffigura **tutte le possibili scelte** che il sistema ha a disposizione. Ogni dato ricevuto in ingresso porterà poi a sfoltire l’albero e giungere così alla conclusione desiderata. Questa potrebbe magari venir raggiunta in variegati modi starà poi al motore inferenziale stabilire quale sia la più efficiente e se anche quando non si è ancora giunti alla soluzione valutare tutte le possibilità o effettuare una ricerca **euristica**.

4.2 Il dottore artificiale

A titolo esemplificativo delle potenzialità di un sistema esperto in campo medico, ho sviluppato un sito web che ne implementa in maniera semplificata molte caratteristiche.

4.2.1 Interfaccia utente

L’utente comunica attraverso un dialogo con il Dr. Artificiale, che gli pone una serie di domande, che si interromperà solo a visita finita, ossia quando si riesce ad escludere tutte le malattie ad eccezione di una. A tali domande si risponde attraverso l’immissione di una frase in una textarea, un particolare input predisposto dal linguaggio HTML che permette l’inserimento di ampi messaggi, sebbene, al fine del corretto funzionamento e interpretazione della risposta, convenga inserire periodi semplici privi di subordinate.

4.2.2 Base di conoscenza

È il ‘luogo’ dove vengono memorizzate le regole deduttive di cui il sistema si serve nel suo operato. Nel mio particolare caso, lo implemento fisicamente con un database relazionale nel quale inserisco tutti i dati necessari al corretto funzionamento del mio sistema esperto. Conterrà quindi i vari sintomi, di cui un paziente può essere affetto, strettamente legati mediante una tabella relazionale alle malattie che vanno ad individuare. Per ogni sintomo saranno necessarie più domande attraverso le quali è possibile chiedere esplicitamente se si è affetti da un determinato sintomo andando a direzionare il discorso e limitare poi i pattern su cui dovrò cercare il riscontro della risposta ricevuta dal client. Saranno quindi necessarie più espressioni regolari usate come pattern che serviranno a far ‘capire’ se il paziente è affetto, o meno, di quel determinato sintomo.

Essendo questa applicazione a scopo unicamente esemplificativo, una sorta di Demo, tale database sarà abbastanza spoglio e avente solo i sintomi e le malattie più comunemente diffuse.

4.2.3 Motore inferenziale

Rappresenta il centro nevralgico di tutto il sistema e nel mio specifico caso è scritto in codice php, che andrà poi a interrogare e ad eseguire varie operazioni anche sul database per mezzo del linguaggio mySQL.

Generalmente tali applicazioni vengono sviluppate in linguaggi molto particolari, che permettono associazioni logiche tra stringhe e una gestione delle regole ad un livello astratto superiore certo a quelle del php, tuttavia a livello esemplificativo le funzionalità del php sono sufficienti.

La pagina che si occupa di analizzare i messaggi immessi dall’utente è “analizza\_msg.php”. La quale assume tre stati principali:

* L’inizio della visita, ossia quando l’utente esterno inserisce il codice fiscale del paziente, che si vuole visitare.
* Il riconoscimento dei sintomi, ossia tutta la fase che segue in cui le domande vertono nell’individuare se il paziente è affetto, o meno, dal sintomo a cui è legata la domanda.
* La selezione della domanda da fare all’utente

4.2.3.1 Inizio della visita

Si prende il messaggio ricevuto dall’utente e lo si confronta con delle espressioni regolari, generate dinamicamente in base ai pazienti che quell’utente ha in cura, formate inserendo tra i caratteri di inizio e fine di un espressione regolare “/” il codice fiscale, prima del 1° paziente, poi del 2° e così via. Verificando così se nel messaggio immesso dall’utente vi è un codice fiscale inerente e capendo così a chi effettuare la visita.

Se non vi fosse alcuna corrispondenza si mostra tale problema all’utente permettendo la reimmissione del messaggio, perché presumibilmente l’inserimento precedente è errato.

Se non vi fossero pazienti in cura si chiede all’utente, gentilmente, di inserirne almeno uno prima di tornare a effettuare la richiesta di una visita.

Se si riconosce il paziente si crea una nuova visita a lui legata, si crea una sessione che memorizzi il numero di visita che si sta effettuando, e si crea una tabella copia di tutte le malattie, che durerà per tutta la visita sarà necessaria per l’esclusione delle malattie all’aumentare dei sintomi conosciuti.

4.2.3.2 Riconoscimento dei sintomi

Si prende il messaggio immesso dall’utente e lo si confronta con le sole espressioni regolari che sono legate al sintomo da cui è stata scelta la domanda precedentemente, attraverso la funzione preg\_match (messa a disposizione dal linguaggio php). A rigor di logica solo un espressione regolare potrà soddisfare la condizione e pertanto appena si trova la prima occorrenza il ciclo si interrompe, andando poi a inserire nella tabella relazionale “presenta” un elemento avente id\_visita e id\_sintomo uguale a quelli salvati nella sessione e affetto uguale alla booleana presente nell’espressione regolare che ha soddisfatto la condizione. Così facendo sarà possibile memorizzare i sintomi riconosciuti in una visita e visualizzare nella pagina delle visite l’anamnesi del paziente.

Entra qui in gioco la tabella “copia”, che ho nel punto 3.3.1 citato, infatti, per rendere più veloci le operazioni di selezione delle malattie possibili per i sintomi di cui conosciamo lo stato basterà eseguire una delete sulla tabella a ogni nuovo sintomo conosciuto; nel caso in cui ne sia affetto, cancellando tutte le malattie che non prevedono quel sintomo, nel caso in cui non ne sia affetto, cancellando tutte le malattie dove quel sintomo è presente.

Se a seguito di questo rimane solo una malattia, si termina la visita comunicandone i risultati all’utente, si elimina la tabella “copia” e la sessione relativa al sintomo e la visita e si riporta la pagina che gestisce le visite alla situazione iniziale.

4.2.3.3 Selezione della domanda

Tale fase a differenza delle precedenti non si occupa della gestione del messaggio ricevuto o delle espressioni regolari, il suo compito è quello di scegliere il sintomo logicamente migliore, cioè che sia nel caso in cui il paziente ne è affetto, sia nel caso in cui non lo è, mi permetta di eliminare il maggior numero di malattie. Nel caso ottimo, quindi, mi permette di escludere, se la risposta è riconosciuta da qualche pattern, il 50% delle malattie. Ciò permette un’ottimizzazione del numero di domande a cui l’utente deve rispondere, che nel caso limite in cui tutte le risposte mi permettano di dimezzare il numero di malattie raggiunge log2(N° malattie).

Tale processo viene svolto creando una tabella temporanea che, al termine dell’esecuzione della pagina php, viene rimossa, e contiene tutti i sintomi che indicano le malattie presenti nella tabella “copia” (descritta nel punto 3.3.1). A ogni sintomo saranno poi affiancati due numeri:

1. Affetto (che è il nome dell’attributo della tabella temporanea) e indica il numero di malattie che, tra quelle presenti nella tabella “copia”, sono identificate da quel sintomo se il paziente ne fosse affetto.
2. Non affetto che consiste nella differenza tra il numero di tutte le malattie presenti nella tabella “copia” e il valore della colonna Affetto. Così facendo si ottiene il numero di malattia che vado a escludere se il paziente non fosse affetto da quel sintomo.

Di tale tabella seleziono come sintomi su cui effettuare la domanda solo quelli che rendono minima la differenza tra il valore nella colonna Affetto e Non affetto, così facendo selezionerò i sintomi logicamente migliori su cui applico un casuale per selezionarne solo uno, che andrà a settare la sessione del sintomo. La strada risulta essere ora tutta in discesa, infatti basterà fare un ulteriore casuale tra le domande legate al sintomo prescelto e comunicarne all’utente la “fortunata”.

Tutta la serie di istruzioni sopra commentate verranno eseguite solo nel caso in cui vi siano più malattie (come già accennato nel paragrafo 3.3.2)