**Progetto Ingegneria del Software - Documentazione**



Medical

Environment

Database

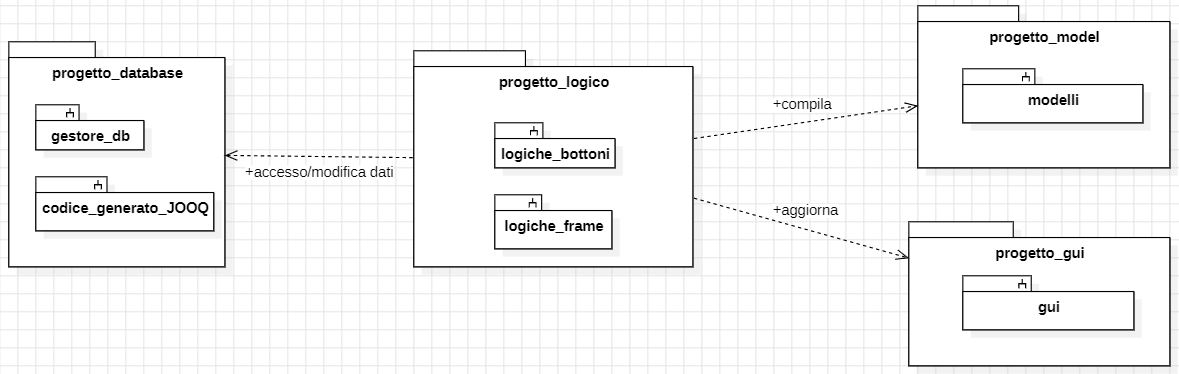
1. **Ciclo di vita del software**

Il processo di sviluppo seguito nel corso della progettazione è stato di tipo agile, più precisamente SCRUM.

Lo sviluppo del software è stato suddiviso inizialmente in 3 pacchetti di lavoro principali (database, logica, GUI), ai quali è stato aggiunto successivamente un quarto (modello).

Tutte le modifiche alla baseline definita inizialmente sono state discusse durante le riunioni tenute giornalmente, come previsto dal modello SCRUM.

La definizione dell’architettura dei pacchetti di lavoro è stata dunque riassunta nel seguente diagramma UML:



In seguito si è rivelato necessario aggiungere ulteriori sotto-pacchetti all’interno di alcuni dei progetti, ma la struttura e il funzionamento complessivi sono rimasti invariati.

1. **Gestione della configurazione**

I sistemi utilizzati per gestire la configurazione del progetto sono stati GitHub e l’applicazione GitHub desktop.

Inizialmente le operazioni di commit sono state eseguite direttamente sul main branch, ma successivamente abbiamo iniziato a gestire gli aggiornamenti del codice attraverso l’apertura di branch e l’utilizzo di pull request.

Nel corso dell’implementazione alcuni errori e problematiche inattese sono state notificate e susseguentemente gestite attraverso l’utilizzo delle issues;

queste sono state aperte dal componente del team che riscontrava il problema e assegnate da parte dello SCRUM master a uno o più membri in grado di completare la richiesta.

Per monitorare lo stato di avanzamento dei pacchetti di lavoro è stata utilizzata una kanban board, aggiornata progressivamente ogni qual volta veniva raggiunta una milestone.

1. **Organizzazione del team e gestione del personale**

La configurazione del team di sviluppo adotta il paradigma della "Squadra Agile", conformemente al processo di sviluppo software selezionato.

In particolare, i ruoli assegnati ai vari membri del gruppo sono i seguenti:

* **Gotti Daniele**, matricola 1079011 -> principale programmatore dell’interfaccia grafica;
* **Bolis Filippo Antonio**, matricola 1079493 -> product owner, principale programmatore delle funzionalità logiche;
* **Mazzoleni Gabriele**, matricola 1079514 -> SCRUM master, principale programmatore della gestione del database, assistente programmatore delle funzionalità logiche;
* **Masinari Gabriele**, matricola 1079692 -> principale redattore della documentazione UML, assistente programmatore dell’interfaccia grafica.

Per ottimizzare al meglio le tempistiche, abbiamo deciso di suddividere i compiti secondo i principali pacchetti di lavoro previsti inizialmente.

L’organizzazione dello sprint backlog è stata comunque generalmente rispettata, essendo stata ideata per permettere il lavoro parallelo e collaborativo sui differenti componenti del software.

1. **Qualità del software**

La qualità del prodotto software è stata testata progressivamente e valutata al termine di ogni ciclo di lavorazione (sprint).

In particolare, per quanto riguarda l’utilizzo da parte degli utenti finali, il prodotto finito deve garantire:

* correttezza, permettendo l’accesso ai dati dei pazienti secondo i vari criteri definiti dai requisiti funzionali, senza la ripetizione o l’omissione di dati utili;
* affidabilità, la garanzia che il software risponda entro tempistiche ammissibili (nell’ordine di pochi secondi nei peggiori dei casi) senza riscontrare perdite di dati forniti in input dagli utenti;
* efficienza, riferita all’uso delle risorse della macchina che ospita il software, oltre che a un efficace e rapido accesso alla base di dati;
* integrità, cioè la limitazione dell’accesso ai dati al solo personale registrato e la limitazione delle interazioni possibili con il sistema al solo personale competente in tale specifica attività;
* usabilità, che permetta a un qualsiasi utente autorizzato di operare sui dati in maniera intuitiva, senza la necessità di spiegazioni estese.

[to be continued]

1. **Ingegneria dei requisiti**

I requisiti sono stati elicitati tramite intervista diretta con un possibile cliente, un operatore in campo ospedaliero, e attraverso l’analisi delle funzionalità del software ospedaliero attualmente utilizzato in alcune strutture del territorio.

**Obiettivo:**

Il software gestisce il percorso clinico del paziente dal suo arrivo alla sua dimissione.

**Specifica dei requisiti funzionali:**

* Per accedere al software è richiesto il log-in del personale (Segreteria, Infermiere, Medico).

Il log-in è presentato come una finestra in cui viene chiesto Id personale e password fornite dall’ospedale all’assunzione.

* All’arrivo del paziente viene richiesta la tessera sanitaria collegata al database statale dalla quale il personale di accoglienza (segreteria) inserirà a mano nel software i dati del paziente (Nome, Cognome e Data di nascita). La segreteria inserisce anche il grado di urgenza (Verde, Giallo, Rosso).
* I medici assegnati al pronto soccorso possono visualizzare i pazienti in attesa della prima visita all’interno della sezione “in pronto soccorso”. I medici possono anche osservare l’urgenza assegnata e prendere a carico i pazienti per ordine di urgenza.
* Durante la prima visita il medico curante compila la diaria medica, non eliminabile, nella quale inserisce il motivo del ricovero, lo storico del paziente, eventuali allergie, il reparto di destinazione, la lista dei farmaci e dei trattamenti che ritiene necessari. Il sistema terrà traccia automaticamente della data di inserimento e della matricola del medico che sta eseguendo l’operazione.
* La diaria medica può poi essere modificata quando necessario dal medico di turno curante.
* Al completamento della prima diaria medica, il paziente viene automaticamente spostato nella sezione “da prendere in carico” visibile anche dal personale infermieristico.
* Un operatore o un membro della segreteria si occupa di assegnare al paziente un modulo e un posto letto in un reparto della struttura, in modo che corrisponda a quello consigliato all’interno della diaria medica; il paziente è quindi spostato automaticamente nella sezione “in reparto”.
* Ogni reparto è composto da N moduli ognuno con 15 posti letto. Il paziente può essere spostato di posto letto e di modulo ma non di reparto. Per spostare il paziente da un reparto all’altro è infatti necessario compilare la dimissione del paziente, inserendo come motivo “altro reparto”.
* Il software aiuta gli infermieri consentendo di filtrare i pazienti per reparto e per modulo. Nel proprio turno di lavoro gli infermieri selezionano un paziente dalla tabella “reparto” e, consultando le informazioni sulla cura prescritta, compilano la diaria inserendo il farmaco somministrato ed eventuali note riguardanti lo stato del paziente, con la possibilità di assegnare un flag di “importanza” alla nota inserita.
* L’infermiere, se richiesto dal medico, può aggiungere i dati delle rilevazioni effettuate sui pazienti (glicemia, temperatura, pressione, frequenza cardiaca, dolore). Inoltre, anche nella diaria infermieristica e nelle rilevazioni verrà automaticamente salvata la matricola dell’infermiere che le compila e l’orario di somministrazione del farmaco o delle rilevazioni.
* I medici osservano periodicamente lo storico delle diarie e delle rilevazioni per tenere traccia dell’andamento della cura, con la possibilità di modificare la diaria medica come sopra citato.
* Al termine della degenza il medico compila il modulo di “dimissione” in cui indica data, ora e motivazione (ordinaria, deceduto, altra destinazione, altro reparto).
* Alla dimissione il paziente è rimosso dal database; nel caso in cui sia stato dimesso per essere posto in un altro reparto il paziente deve risostenere la registrazione e la visita medica in pronto soccorso.

1. **Modellazione UML**

La presentazione della modellazione è avvenuta attraverso l’impiego dei seguenti diagrammi UML:

* Casi d’uso
* Attività
* Classe
* Sequenza
* Macchina a stati (che evidenzia gli stati del paziente all’interno del database)
* Componenti
* Diagramma ER
* Package

Nell’esempio qui sotto, viene illustrata la modellazione del diagramma dei componenti, suddiviso in quattro parti per rappresentare l’architettura del software:

Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamente

1. **Architettura del software**

a

1. **Design del software**

La descrizione dei design del software è stata eseguita con UML, qui sotto è mostrato il diagramma delle attività, che rappresenta le attività svolte da chi utilizza il software:

Immagine che contiene testo, diagramma, Piano, Parallelo

Descrizione generata automaticamente

1. **Testing del software**

a

1. **Manutenzione del software**

La manutenzione del software si è svolta effettuando delle operazioni di refactoring, le principali eseguite sono le seguenti:

* estrazione di una superclasse astratta;
* ridenominazione di campi, variabili, progetti e classi;
* aumento della qualità del codice con l’aggiunta di commenti JavaDoc.

Inoltre, essa è stata effettuata in maggior parte in fase di sviluppo andando ad attuare al meglio le esigenze del cliente e andando a ridurre il più possibile gli errori che si sono presentati.

In una possibile futura manutenzione del software si ipotizza un miglioramento della lunghezza complessiva del codice e un adattamento ai cambiamenti hardware e software più moderni.