

TP



Documento di Test Planning

Gruppo 7

Team Manager:	Roberto Di Russo	0510201086
Componenti:	Giovanni Maggiolini	0510201080
	Umberto Annunziata	0510201230
	Francesco Nicolao	0510200141
	Claudio Gargiulo	0510200990
	Vincenzo Iuliano	0510201428

Revision History

Data	Versione	Descrizione	Autore
20/05/2008	TP 0.1	Stesura della parte introduttiva	Tutti i componenti del gruppo, tranne quelli coinvolti nell'implementazione delle interfacce del SW
21/05/2008	TP 1.0	Definizione Test Cases Pianificazione Testing	Tutti i componenti del gruppo, tranne quelli coinvolti nell'implementazione delle interfacce del SW
03/06/2008	TP 1.1	Correzione e revisione finale	Tutti i componenti del gruppo

TP

Progetto di un sistema software per la gestione dei posti letto presso una clinica privata

1. Introduzione	pag. 4
2. Documenti correlati	pag. 5
2.1 Relazioni con il documento di analisi dei requisiti (RAD)	pag. 5
2.2 Relazioni con il documento di System Design (SDD)	pag. 5
3. Overview del Sistema	pag. 6
4. Funzionalità da testare e non	pag. 7
5. Criteri Pass/Failed	pag. 8
6. Approccio	pag. 9
6.1 Testing di Unità	pag. 9
6.2 Testing d'integrazione	pag. 10
6.3 Testing di Sistema	pag. 10
7. Sospensione e Ripresa	pag. 10
7.1 Criteri di sospensione	pag. 10
7.2 Criteri di ripristino	pag. 10
8. Materiale per il testing (requisiti Hardware/Software)	pag. 10
9. Casi di Test (Test cases).....	pag. 11
9.1 Login	pag. 11
9.2 InserimentoPaziente.....	pag. 13
9.3 ModificaPaziente/RicercaPaziente.....	pag. 17
9.4 VisualizzazionePaziente.....	pag. 23
9.5 InserimentoCartellaClinica.....	pag. 25
9.6 VisualizzazioneCartellaClinica	pag. 28
9.7 ModificaCartellaClinica	pag. 29
9.8 DimissionePaziente.....	pag. 31
9.9 AssegnazionePostoLetto.....	pag. 32
9.10 CambioReparto.....	pag. 33
10. Pianificazione del Testing.....	pag. 34
10.1 Determinazione dei ruoli.....	pag. 35
10.2 Determinazione dei rischi.....	pag. 35
10.3 Decomposizione gerarchica del sistema.....	pag. 36
10.4 Organizzazione delle attività del testing.....	pag. 37
10.5 Schedulazione delle attività di testing	pag. 38
11. Glossario.....	pag.39

1. Introduzione

Il testing è una tecnica di “fault detection” il cui scopo è appunto quello di rilevare errori in maniera pianificata all’interno del codice prodotto dal mapping.

L’obiettivo è, quindi, evitare che questi si presentino in fase di utilizzo da parte dell’utente finale.

Per questo motivo un test che ha successo è un test che ha identificato un errore, non un test che “garantisce” la correttezza del codice.

Lo scopo del presente documento è quello di descrivere e pianificare l’attività di testing del sistema software EasyClinique.

In esso vengono pianificate le attività di test relative a:

- GestioneAccessi
- GestionePazienti
- GestioneDegenze
- GestioneCartelleCliniche

Il testing di ogni funzionalità avverrà dal punto di vista di ciascuno degli utenti che può fare uso di essa.

Si noti, tuttavia, che verranno testate esclusivamente le funzionalità implementate e specificate nell’ODD 2.0.

Nel documento, oltre alla gestione dei test delle funzionalità, vengono anche pianificate le responsabilità dello staff e lo scheduling del test.

2. Documenti correlati

La relazione del test plan con gli altri documenti è notevole.

Innanzitutto, poiché si è deciso di passare alla fase di testing solo dopo che la fase di implementazione fosse ben avviata (in particolare nel momento in cui fossero già disponibili le interfacce utente), il test plan ha un forte legame con la fase di mapping oggetti – codice e con l'Object Design Document (ODD).

In secondo luogo, il testing ha il compito di rilevare le differenze tra il comportamento atteso e documentato nei system models con quello osservato nel momento dell'esecuzione del software.

Viene di seguito riportata una panoramica delle relazioni del testing con i documenti finora prodotti.

2.1 Relazioni con il documento di analisi dei requisiti (RAD)

I test delle funzionalità terranno conto delle specifiche espresse nel documento di analisi dei requisiti, cioè faranno riferimento sia ai requisiti funzionali che a quelli non funzionali.

2.2 Relazioni con il System Design Document (SDD)

La pianificazione dei test dei vari componenti dovrà mantenersi quanto più è possibile fedele alla decomposizione in sottosistemi specificata nello SDD, mentre il test d'integrazione farà riferimento ai package definiti nell'ODD.

3. Overview del Sistema

Come specificato nello SDD, il sistema verrà suddiviso in quattro sottosistemi: SottosistemaAmministratore, SottosistemaDirigente, SottosistemaResponsabile e SottosistemaMedicoParamedico.

In ognuno di essi le componenti sono suddivise in base ai diversi tipi di gestione sviluppati.

SottosistemaAmministratore

- GestioneAccount

SottosistemaDirigente

- GestioneReparti
- GestionePersonale
- GestioneStanze
- GestioneGrafici

SottosistemaResponsabile

- Gestione CartelleCliniche (nello SDD è chiamato CreazioneCartelleCliniche, ma poi nell'ODD è stato collassato nella GestioneCartelleCliniche)
- Gestione Pazienti
- Gestione Degenze

SottosistemaMedicoParamedico

- GestioneCartelleCliniche
- GestioneNotifiche

La GestionePazienti comprende anche la funzionalità RicercaPaziente a cui può accedere qualsiasi utente registrato.

Inoltre, sono presenti una funzionalità di Gestione Accessi e una di Gestione Problemi, non comprese in nessun sottosistema poiché condivise da tutti.

Come si evince dall' ODD, non tutte le funzionalità sono state implementate. Di seguito viene riportato un elenco delle funzionalità implementate.

- GestioneAccessi
- GestionePazienti (ma non viene implementata la stampa)
- GestioneDegenze
- GestioneCartelleCliniche

4. Funzionalità testate e non

Le funzionalità che verranno testate tra quelle implementate sono:

- **GestioneAccessi**
 - ✓ Login
- **GestionePazienti**
 - ✓ InserimentoPaziente
 - ✓ ModificaPaziente
 - ✓ Ricerca&VisualizzazionePaziente
- **GestioneDegenze**
 - ✓ DimissionePaziente
 - ✓ AssegnazionePostoLetto
 - ✓ CambioReparto
- **GestioneCartelleCliniche**
 - ✓ InserimentoCartellaClinica
 - ✓ ModificaCartellaClinica
 - ✓ Ricerca&VisualizzazioneCartellaClinica

Quelle che non verranno testate, sempre tra quelle implementate, sono invece:

- **Gestione degli accessi**
 - ✓ chiusura della sessione di lavoro (logout)

5. Criteri Pass/Failed

Di seguito vengono definiti i criteri per stabilire quando un input avrà superato un test e quando, invece, risulterà fallito.

Si noti che i dati di input del test verranno suddivisi in classi di equivalenza, ovvero verranno raggruppati in insiemi dalle caratteristiche comuni, per i quali sarà sufficiente testare un solo elemento rappresentativo.

Un input avrà superato un test se l'output risultante sarà quello atteso, cioè quello che è stato specificato nell'oracolo del test case.

Si noti che per oracolo si intende la persona (sviluppatore o membro del team di testing) che conosce quale dovrebbe essere l'output corretto.

In questo modo l'intera classe di appartenenza verificherà la correttezza del test.

Ovviamente, la stessa classe non supererà il test se l'output risultante sarà diverso da quello atteso.

6. Approccio

Le tecniche di testing adottate riguarderanno inizialmente il testing di unità dei singoli componenti, in modo da testare nello specifico la correttezza di ciascuna unità.

Seguirà il testing d'integrazione, che focalizzerà l'attenzione principalmente sul test delle interfacce delle suddette unità.

Infine verrà eseguito il testing di sistema, che vedrà come oggetto di testing l'intero sistema assemblato nei suoi componenti.

Quest'ultimo servirà soprattutto a verificare che il sistema soddisfi le richieste del committente.

6.1 Testing di Unità

Durante questa fase, verranno ricercate le condizioni di fallimento isolando i componenti ed usando test driver e test stub, cioè implementazioni parziali di componenti che dipendono o da cui dipendono le componenti da testare.

Al fine di minimizzare il numero dei test cases, come già accennato, i possibili input verranno partizionati in classi di equivalenza e per ogni classe verrà selezionato un test case (tecnica WECT).

La strategia utilizzata per il testing, dato il ristretto tempo a disposizione, si baserà esclusivamente sulla tecnica BlackBox, che si focalizza sul comportamento Input/Output, ignorando la struttura interna della componente.

Gli stati erronei scovati in questa, come in qualsiasi altra fase di testing, che comporteranno un fallimento del sistema dovranno essere tempestivamente comunicati agli implementatori al fine di correggerli e ripristinare il testing al più presto.

Si noti che le unità verranno anche testate utilizzando il modulo JUnit 3 messo a disposizione dal linguaggio Java. Tuttavia, per ragioni di tempo, questa tipologia di testing non verrà documentata.

Per quanto riguarda il WhiteBox Testing, la sua realizzazione è stata effettuata durante la fase implementativa in modo totalmente implicito da parte degli sviluppatori. Pertanto, la sua documentazione non viene riportata nel TestPlanning.

6.2 Testing d'integrazione

Si procederà all'integrazione delle componenti di una funzionalità che verranno testate nel complesso attraverso una strategia di tipo Sandwich, derivante dalla combinazione delle tecniche Bottom-Up e Top-Down.

Si passerà, poi, alla funzionalità successiva, fino ad esaurire le funzionalità implementate.

Quest'approccio mira principalmente a ridurre le dipendenze tra funzionalità differenti e a facilitare la ricerca di errori nelle interfacce di comunicazione tra sottosistemi.

6.3 Testing di Sistema

Lo scopo di questa fase del testing è quello di dimostrare che il sistema soddisfi effettivamente i requisiti richiesti e che sia, quindi, pronto all'uso.

Come per il testing di unità, si cercherà di testare le funzionalità più importanti per l'utente e per quelle che hanno una maggiore probabilità di fallimento.

Si noti che, come per lo Unit Testing, si procederà attraverso la tecnica BlackBox.

7. Sospensione e ricerca

7.1 Criteri di sospensione

Il testing sarà sospeso quando saranno state testate tutte le classi scelte per ogni possibile input e il risultato sarà quello atteso e sperato.

Nel momento in cui il testing rivelerà un errore si dovrà passare alla fase di correzione che dovrà essere seguita da una ripetizione dell'intero testing per rilevare la presenza di errori introdotti dalle correzioni stesse.

7.2 Criteri di ripristino

La ripresa in seguito alla fase di correzione avverrà, come già accennato, ripetendo l'intero testing per rilevare la presenza di errori introdotti dalle correzioni.

8. Materiale per il testing (Requisiti Hardware/software)

L'hardware necessario per l'attività di test è un PC, o possibilmente uno per ogni soggetto incaricato del testing, su cui sia installato il DBMS MySQL 5.0 e una Java Virtual Machine in grado di supportare Java 5 o superiori.

9. Casi di test (Test cases)

9.1 Login

La funzionalità da testare riguarda il componente GestioneAccessi, in particolare la funzionalità di Login, cioè di accesso la sistema.

Classi di equivalenza

Gli input del test sono l'userID e la password dell'utente.

Input	userID, password	
Classi valide	C_001	Stringa alfanumerica compresa tra 4 e 11 caratteri
Classi non valide	C_002	Stringa alfanumerica di lunghezza strettamente maggiore di 11 caratteri
	C_003	Stringa alfanumerica di lunghezza strettamente minore di 4 caratteri

Identificazioni Test Case

Test case: TCLLogin_01		
Input	Classe	Membro
userID	C_002	Francesco_nicolao85
Password	C_001	28_01_85

Test case: TCLLogin_02		
Input	Classe	Membro
userID	C_001	Umberto86
Password	C_002	Adriana_ti_amooooo

Test case: TCLLogin_03		
Input	Classe	Membro
userID	C_001	Roberto87
Password	C_001	16maggio

Test case: TCLLogin_04		
Input	Classe	Membro
userID	C_002	Lucariello88
Password	C_002	Gerardina_mia

Test case: TCLLogin_05		
Input	Classe	Membro
userID	C_003	Fra
Password	C_003	123

Test case: TCLLogin_06		
Input	Classe	Membro
userID	C_003	La stringa passata è null
Password	C_003	La stringa passata è null

9.2 InserimentoPaziente

Il test riguarda la componente GestionePaziente, in particolare la funzionalità di InserimentoPaziente.

Classi di equivalenza

Gli input sono: codice Fiscale, nome, cognome, dataNascita, luogoNascita, sesso, asl, regione

Input	codice Fiscale	
Classi valide	C_004	stringa alfanumerica di 16 caratteri
Classi non valide	C_005	stringa alfanumerica di lunghezza minore di 16 caratteri
	C_006	stringa alfanumerica di lunghezza maggiore di 16 caratteri
	C_007	stringa nulla

Input	data Nascita	
Classi valide	C_008	attributo di tipo date
Classi non valide	C_009	attributo di tipo diverso da date

Input	Regione	
Classi valide	C_010	stringa che identifica una regione italiana o “Eestero”
Classi non valide	C_011	stringa che non identifica né una regione italiana, né “Eestero”

Input	nome, cognome	
Classi valide	C_012	Stringa
Classi non valide	C_013	Viene passata una stringa nulla

Input	luogoNascita, sesso, indirizzo, asl	
Classi valide	C_014	Stringa
Classi non valide	C_015	Viene passata una stringa nulla

Identificazione Test Case

Test case: TCInserimentoPaziente_01		
Input	Classe	Membro
codice Fiscale	C_004	NCLFNC85A28F912B
Nome	C_012	Francesco
Cognome	C_012	Nicolao
data Nascita	C_008	28/01/85
Luogo Nascita	C_014	Nocera Inferiore
Sesso	C_014	M
Indirizzo	C_014	Via marco rossi
Asl	C_014	SA1
Regione	C_010	Campania

Test case: TCInserimentoPaziente_02		
Input	Classe	Membro
codice Fiscale	C_005	MC34GDKS56
Nome	C_012	Marco
Cognome	C_012	Rossi
data Nascita	C_009	12 Aprile 1985
Luogo Nascita	C_014	Palermo
Sesso	C_014	M
Indirizzo	C_014	Via ambrosini
Asl	C_014	PA6
Regione	C_011	Siciliana

Test case: TCInserimentoPaziente_03		
Input	Classe	Membro
codice Fiscale	C_006	LU88EDWF43DJFUFNN
Nome	C_012	Luana
Cognome	C_012	Sirio
data Nascita	C_008	12/03/83
Luogo Nascita	C_014	Roma
Sesso	C_014	F
Indirizzo	C_014	Via toledo
Asl	C_014	RM1
Regione	C_010	Lazio

Test case: TCInserimentoPaziente_04		
Input	Classe	Membro
codice Fiscale	C_004	RSSMRA89A01F839F
Nome	C_012	Mario
Cognome	C_012	Rossi
data Nascita	C_008	01/01/89
Luogo Nascita	C_014	Budapest
Sesso	C_014	M
Indirizzo	C_014	Via virno
Asl	C_014	BX5
Regione	C_010	Estero

Test case: TCInserimentoPaziente_05		
Input	Classe	Membro
codice Fiscale	C_004	GUTMRC90A17M912B
Nome	C_012	Marco
Cognome	C_012	Giusti
data Nascita	C_008	17/07/90
luogo Nascita	C_014	Parigi
Sesso	C_014	M
Indirizzo	C_014	Via dolce
Asl	C_014	PKK5
Regione	C_011	Francia

Test case: TCInserimentoPaziente_06		
Input	Classe	Membro
codice Fiscale	C_007	La stringa passata è nulla
Nome	C_013	La stringa passata è nulla
Cognome	C_013	La stringa passata è nulla
data Nascita	C_008	05/04/56
luogo Nascita	C_014	Siena
Sesso	C_014	F
Indirizzo	C_014	Via siono
Asl	C_014	SI1
Regione	C_010	Toscana

9.3 ModificaPaziente/RicercaPaziente

Il test riguarda il componente GestionePaziente, in particolare la funzionalità di ModificaPaziente.

La modifica del paziente si divide in due parti: RicercaPaziente, che verifica l'effettiva presenza del paziente nel database, e ModificaPaziente, per l'effettiva modifica.

Classi di equivalenza

Input	codice Fiscale	
Classi valide	C_016	stringa alfanumerica di 16 caratteri
Classi non valide	C_017	stringa alfanumerica di lunghezza minore di 16 caratteri
	C_018	stringa alfanumerica di lunghezza maggiore di 16 caratteri
	C_019	stringa nulla

Identificazione Test Case

Test case: TCRicercaPaziente_01		
Input	Classe	Membro
codice Fiscale	C_016	NCLFNC85A28F912B

Testcase: TCRicercaPaziente_02		
Input	Classe	Membro
codice Fiscale	C_017	NCLFNC85A28F

Testcase: TCRicercaPaziente_03		
Input	Classe	Membro
codice Fiscale	C_018	NCLFNC85A28F912BFRA

Testcase: TCRicercaPaziente_04		
Input	Classe	Membro
codice Fiscale	C_019	La stringa passata è nulla

Dopo la fase di ricerca è possibile, se il paziente è stato trovato , modificare i seguenti attributi: nome, cognome, dataNascita, luogoNascita, sesso, asl, regione

Classi di equivalenza

Input	nome, cognome	
Classi valide	C_020	Stringa
Classi non valide	C_021	Viene passata una stringa nulla

Input	dataNascita	
Classi valide	C_022	Attributo di tipo date
	C_023	Attributo di tipo diverso da date

Input	Regione	
Classi valide	C_024	Stringa che identifica una regione italiana o “Eestero”
	C_025	Stringa che non identifica una regione italiana o “Eestero”

Input	luogoNascita, sesso, indirizzo, asl	
Classi valide	C_026	Stringa
Classi non valide	C_027	Stringa null

Identificazioni Test Case

Test case: TCModificaPaziente_01		
Input	Classe	Membro
Nome	C_020	Francesco
Cognome	C_020	Nicolao
data Nascita	C_022	28/01/85
luogo Nascita	C_026	Nocera Inferiore
Sesso	C_026	M
Indirizzo	C_026	Via marco rossi
Asl	C_026	SA1
Regione	C_024	Campania

Test case: TCModificaPaziente_02		
Input	Classe	Membro
Nome	C_020	Marco
Cognome	C_020	Rossi
data Nascita	C_023	12 Aprile 1985
luogo Nascita	C_026	Palermo
Sesso	C_026	M
Indirizzo	C_026	Via ambrosini
Asl	C_026	PA6
Regione	C_024	Sicilia

Test case: TCModificaPaziente_03		
Input	Classe	Membro
Nome	C_020	Luana
Cognome	C_021	Stringa null
data Nascita	C_022	12/03/83
luogo Nascita	C_026	Roma
Sesso	C_026	F
Indirizzo	C_026	Via toledo
Asl	C_026	RM1
Regione	C_024	Lazio

Test case: **TCModificaPaziente_04**

Input	Classe	Membro
Nome	C_021	Stringa nulla
Cognome	C_020	Rossi
data Nascita	C_022	01/01/89
luogo Nascita	C_026	Budapest
Sesso	C_026	M
Indirizzo	C_026	Via Irno
Asl	C_026	BX3
Regione	C_024	Estero

Test case: **TCModificaPaziente_05**

Input	Classe	Membro
Nome	C_020	Marco
Cognome	C_020	Giusti
data Nascita	C_022	17/07/90
luogo Nascita	C_026	Parigi
Sesso	C_026	M
Indirizzo	C_026	Via dolce
Asl	C_026	PG5
Regione	C_025	Gran Bretagna

Test case: **TCModificaPaziente_06**

Input	Classe	Membro
Nome	C_020	Marco
Cognome	C_020	Giusti
data Nascita	C_023	05/04/5643
luogo Nascita	C_027	La stringa passata è nulla
Sesso	C_026	M
Indirizzo	C_026	Via siono
Asl	C_026	SI1
Regione	C_024	Toscana

Test case: TCModificaPaziente_07		
Input	Classe	Membro
Nome	C_021	La stringa passata è nulla
Cognome	C_021	La stringa passata è nulla
data Nascita	C_023	05/04/5643
luogo Nascita	C_026	Siena
Sesso	C_026	F
Indirizzo	C_026	Via siono
Asl	C_026	SI1
Regione	C_024	Toscana

9.4 VisualizzazionePaziente

Il test riguarda la componente GestionePaziente, in particolare la funzionalità di VisualizzazionePaziente.

Classi di equivalenza

Input	codice Fiscale	
Classi valide	C_028	stringa alfanumerica di 16 caratteri
Classi non valide	C_029	stringa alfanumerica con lunghezza minore di 16 caratteri
	C_030	stringa alfanumerica con lunghezza maggiore di 16 caratteri
	C_031	stringa nulla

Identificazione Test Case

Testcase: TCVisualizzazionePaziente_01		
Input	Classe	Membro
codice Fiscale	C_028	NCLFNC85A28F912B

Testcase: TCVisualizzazionePaziente_02		
Input	Classe	Membro
codice Fiscale	C_029	NCLFNC85A28F

Testcase: TCVisualizzazionePaziente_03		
Input	Classe	Membro
codice Fiscale	C_030	NCLFNC85A28F912BFRA

Testcase: TCVisualizzazionePaziente_04		
Input	Classe	Membro
codice Fiscale	C_031	La stringa passata è nulla

9.5 InserimentoCartellaClinica

La funzionalità da testare riguarda la componente GestioneCartelleCliniche, in particolare l’InserimentoCartellaClinica.

Classi di equivalenza

Gli input del test sono: numeroCartella, dataRicovero, dataDimissione.

Input	numeroCartella	
Classi valide	C_032	Stringa alfanumerica nella forma NNNNN/AAAA
Classi non valide	C_033	Stringa alfanumerica che non rispetta la forma NNNNN/AAAA
	C_034	*Stringa vuota*

Input	dataRicovero	
Classi valide	C_035	Attributo di tipo date
Classi non valide	C_036	Attributo di tipo diverso da date

Identificazione Test Case

Test case: TCInserimentoCartellaClinica_01		
Input	Classe	Membro
numeroCartella	C_032	00035/2001
dataRicovero	C_035	05/03/2001

Test case: TCInserimentoCartellaClinica_02		
Input	Classe	Membro
numeroCartella	C_033	35duemilauno
dataRicovero	C_035	05/03/2001

Test case: TCInserimentoCartellaClinica_03		
Input	Classe	Membro
numeroCartella	C_033	35duemilauno
dataRicovero	C_036	data12

Test case: TCInserimentoCartellaClinica_04		
Input	Classe	Membro
numero Cartella	C_032	00037/2004
data Ricovero	C_036	data12

Test case: TCInserimentoCartellaClinica_05		
Input	Classe	Membro
numero Cartella	C_033	2004
data Ricovero	C_036	data

Test case: TCInserimentoCartellaClinica_06		
Input	Classe	Membro
numero Cartella	C_034	*Stringa vuota*
data Ricovero	C_036	data12

Test case: TCInserimentoCartellaClinica_07		
Input	Classe	Membro
numero Cartella	C_034	*Stringa vuota*
data Ricovero	C_035	05/03/2001

9.6 VisualizzazioneCartellaClinica

La funzionalità da testare riguarda la componente GestioneCartelleCliniche, in particolare la Visualizzazione dei dati di una cartella clinica esistente.

Classi di equivalenza

Input		
Classi valide	C_037	Dall'elenco dei pazienti ne viene selezionato uno che ha già una cartella clinica associata.
Classi non valide	C_038	Non viene selezionato nessun paziente dalla lista
	C_039	Viene Selezionato un paziente che non ha nessuna cartella clinica associata

Identificazione Test Case

Test case: TCVisualizzazioneCartellaClinica_02	
Classe	Membro
C_037	Dall'elenco ne viene selezionato uno che ha già una cartella clinica associata

Test case: TCVisualizzazioneCartellaClinica_02	
Classe	Membro
C_038	Non viene selezionato nessun paziente dalla lista

Test case: TCVisualizzazioneCartellaClinica_03	
Classe	Membro
C_039	Viene Selezionato un paziente che non ha nessuna cartella clinica associata

9.7 ModificaCartellaClinica

La funzionalità da testare riguarda la componente GestioneCartelleCliniche, in particolare la ModificaCartellaClinica.

Per modificare una cartella clinica occorre selezionare da un elenco il paziente associato ad essa.

Si ha un errore se non viene selezionato nessun paziente.

Classi di equivalenza

Gli input sono diagnosiEntrata, cureSomministrate, diagnosiUscita, diarioMedico, diarioInterventi, note.

Input	diagnosiEntrata, cureSomministrate, diagnosiUscita, diarioMedico, diarioInterventi, note	
Classi valide	C_040	Stringa di lunghezza minore o uguale a 1000 caratteri (anche nulla)
Classi non valide	C_041	Stringa di lunghezza strettamente maggiore di 1000 caratteri

Identificazione Test Case

Test case: TCModificaCartellaClinica_01		
Input	Classe	Membro
diagnosiEntrata	C_040	Colica renale destra
cureSomministrate	C_040	Flebo con antispastici
diagnosiUscita	C_040	Calcolosi renale
diarioMedico	C_040	Manovra di Giordano positiva a destra, stranguria con ematuria
diarioInterventi	C_040	Ecografia renale
Note	C_040	Dolenzia lombare da diversi giorni

Test case: TCModificaCartellaClinica_02		
Input	Classe	Membro
diagnosiEntrata	C_041	*stringa superiore ai 1000 caratteri*
cureSomministrate	C_040	
diagnosiUscita	C_040	
diarioMedico	C_040	
diarioInterventi	C_040	100 cc di adrenalina intracardiaca
Note	C_040	

Test case: TCModificaCartellaClinica_03		
Input	Classe	Membro
diagnosiEntrata	C_040	Emicrania con episodi di labirintite
cureSomministrate	C_041	*stringa superiore ai 1000 caratteri*
diagnosiUscita	C_040	
diarioMedico	C_040	
diarioInterventi	C_040	
Note	C_040	

Test case: TCModificaCartellaClinica_04		
Input	Classe	Membro
diagnosiEntrata	C_040	
cureSomministrate	C_040	
diagnosiUscita	C_041	*stringa superiore ai 1000 caratteri*
diarioMedico	C_040	
diarioInterventi	C_040	
Note	C_040	Psoriasi diffusa sugli arti

Test case: TCModificaCartellaClinica_05		
Input	Classe	Membro
diagnosiEntrata	C_040	
cureSomministrate	C_040	Fisioterapia
diagnosiUscita	C_040	
diarioMedico	C_041	*stringa superiore ai 1000 caratteri*
diarioInterventi	C_040	
Note	C_040	

Test case: TCModificaCartellaClinica_06		
Input	Classe	Membro
diagnosiEntrata	C_040	
cureSomministrate	C_040	
diagnosiUscita	C_040	Leggere cicatrici sul torace
diarioMedico	C_040	
diarioInterventi	C_041	*stringa superiore ai 1000 caratteri*
Note	C_040	

Test case: TCModificaCartellaClinica_07		
Input	Classe	Membro
diagnosiEntrata	C_040	
cureSomministrate	C_040	
diagnosiUscita	C_040	
diarioMedico	C_040	Ricaduta in data 24/4/2007
diarioInterventi	C_040	
Note	C_041	*stringa superiore ai 1000 caratteri*

9.8 DimissionePaziente

La funzionalità da testare riguarda il componente GestioneDegenze, in particolare la dimissione di un paziente.

Classi di equivalenza

Gli input del test sono: codiceFiscale

Input	codiceFiscale	
Classi valide	C_042	Stringa alfanumerica di 16 caratteri
Classi non valide	C_043	Stringa alfanumerica con lunghezza > 16 caratteri
	C_044	Stringa alfanumerica con lunghezza < 16 caratteri
	C_045	Stringa vuota

Identificazione Test Case

Test case: TCDimissionePaziente_01		
Input	Classe	Membro
codice Fiscale	C_042	NNNMRT87L28I483T

Test case: TCDimissionePaziente_02		
Input	Classe	Membro
codice Fiscale	C_043	NFNGOSNFJJDKEIDKLS

Test case: TCDimissionePaziente_03		
Input	Classe	Membro
codice Fiscale	C_044	JKFNSODF

Test case: TCDimissionePaziente_04		
Input	Classe	Membro
codice Fiscale	C_045	*Stringa vuota*

9.9 AssegnazionePostoLetto

La funzionalità da testare riguarda il componente GestioneDegenze, in particolare l'assegnazione di un posto letto ad un paziente.

Classi di equivalenza

Gli input del test sono: numeroCartella.

Input	numero Cartella	
Classi valide	C_046	Stringa alfanumerica nella forma NNNNN/AAAA
Classi non valide	C_047	Stringa alfanumerica che non rispetta la forma NNNNN/AAAA
	C_048	Stringa vuota

Identificazione Test case

Test case: TCAssegnazionePostoLetto_01		
Input	Classe	Membro
numeroCartella	C_046	00030/2004

Test case: TCAssegnazionePostoLetto_02		
Input	Classe	Membro
numeroCartella	C_047	sette

Test case: TCAssegnazionePostoLetto_03		
Input	Classe	Membro
numeroCartella	C_048	*Stringa vuota*

9.10 Cambio di un reparto

La funzionalità da testare riguarda il componente GestioneDegenze, in particolare lo spostamento di un paziente da un reparto ad un altro.

Tale funzionalità non viene testata in quanto il suo test corrisponde esattamente a quello effettuato per la DimissionePaziente

10. Pianificazione del Testing

Il team che effettuerà il testing deve essere composto da persone che abbiano una conoscenza almeno discreta del sistema e delle tecniche di testing .

Tali tecniche dovranno essere applicate nel rispetto dei vincoli di tempo, budget e qualità stabiliti.

Per quanto riguarda il tempo, il team avrà a disposizione non più di tre giorni, mentre, essendo un progetto a scopi prettamente didattici, non ci saranno limitazioni a livello di budget.

Rispetto alla qualità, invece, si cercherà di scovare il più alto numero possibile di errori, sempre nei limiti di tempo stabiliti.

Normalmente i componenti del team di testing sono persone che non siano state coinvolte nella fase di sviluppo, in quanto è statisticamente provato che uno sviluppatore è molto restio a modificare una sua “creatura”.

Nel caso specifico, tuttavia, tale approccio non potrà essere rispettato poiché il numero dei componenti del team, sei per la precisione, è insufficiente per coprire interamente i task di sviluppo ed anche perché, dati i fini didattici del progetto, si preferisce rendere partecipe ad ogni fase dello sviluppo il più alto numero possibile di componenti del team.

Faranno parte del team di testing, quindi, gli sviluppatori che per primi completeranno l'implementazione, in particolare quelli che non hanno partecipato allo sviluppo delle interfacce.

Per quanto riguarda la documentazione, gli eventuali fault di una componente verranno notificati agli sviluppatori della componente stessa, per consentire la correzione del sistema.

Dopo la correzione e la revisione, il sistema verrà nuovamente testato per verificare se le modifiche apportate per risolvere gli errori individuati non ne abbiano introdotti nuovi.

Data l'importanza del testing è fondamentale una buona schedulazione di esso.

10.1 Determinazione dei ruoli

L'attività di testing di sistema sarà svolta, come detto, dai componenti del team che per primi completeranno l'implementazione.

Quasi sicuramente, non faranno parte del sub-team gli sviluppatori impegnati nella definizione delle interfacce grafiche, compito comunque oneroso e dinamico.

Il team manager avrà funzioni di coordinatore e revisore.

10.2 Determinazione dei rischi

Si cercherà di minimizzare i rischi di fallimento, pianificando al meglio il testing.

In caso di ritardi dovuti all'individuazione di un numero troppo elevato di errori (failure), si terrà in considerazione la possibilità di rilasciare in un primo momento un numero minore di funzionalità completamente testate.

Si cercherà, comunque, di evitare situazioni di questo genere, in quanto le funzionalità implementate e documentate nell'ODD sono già di gran lunga ridotte rispetto al progetto originario.

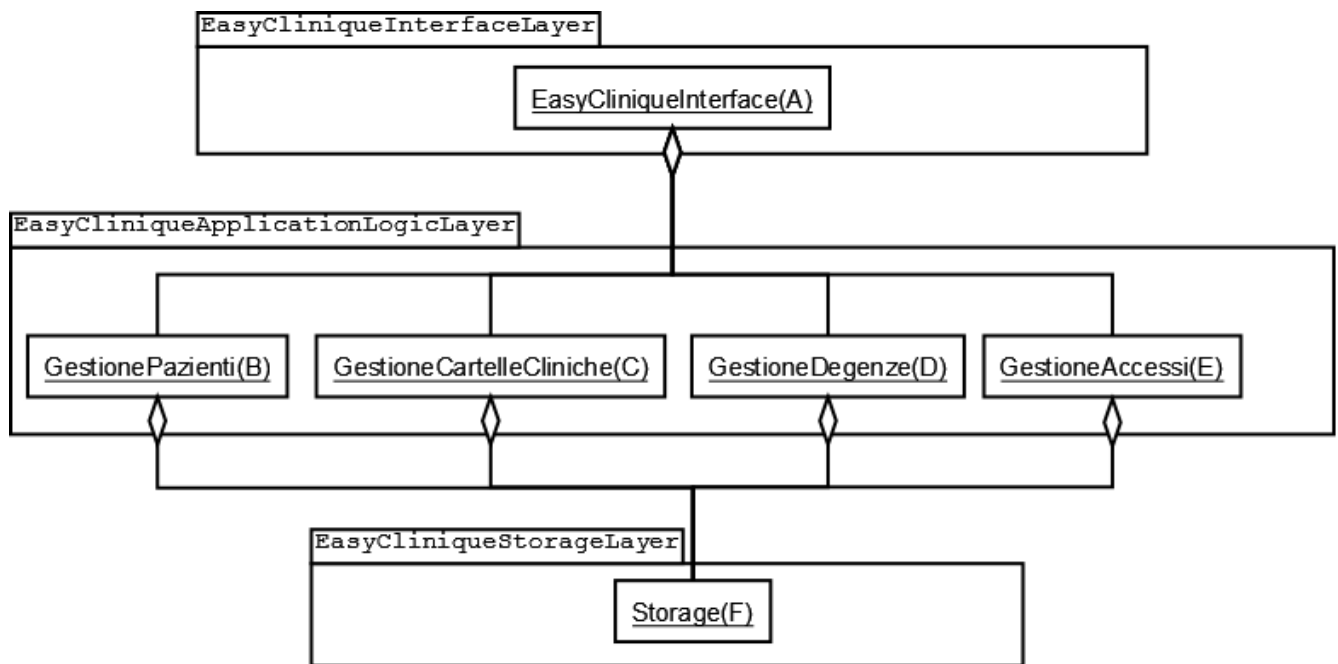
Sul piano tecnico, si cercherà di fare un uso intelligente dei driver e degli stub, in quanto una loro applicazione eccessiva o scorretta potrebbe portare ad appesantimenti o funzionamenti fittizi del sistema.

10.3 Decomposizione gerarchica del sistema

La decomposizione in sottosistemi effettuata nell'attività di design è stata rimappata sulla

base di tre livelli gerarchici:

- InterfaceLayer (1)
- ApplicationLogicLayer (2)
- StorageLayer (3)



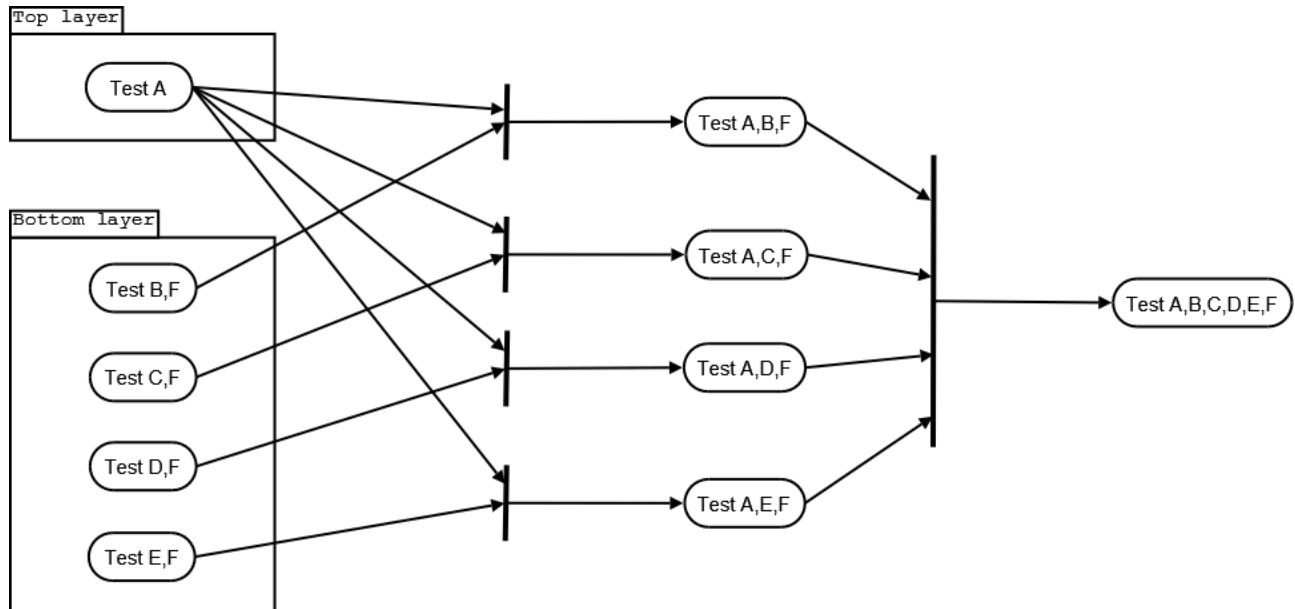
Come si evince dall'ODD, EasyCliniqueApplicationLogicLayer e EasyCliniqueStorageLayer saranno trattati come un unico livello, in quanto la comunicazione tra i due strati è gestita da JDBC, ed è quindi totalmente trasparente agli sviluppatori.

Anche per il Testing, quindi, ciascuna componente di Gestione verrà testata insieme alla relativa componente di memorizzazione. Facendo riferimento all'ODD, ad esempio, la componente CartellaClinica e DBCartellaClinica verranno testate con un unico test.

10.4 Organizzazione delle attività di testing

Le attività verranno organizzate secondo uno schema di tipo Sandwich, cioè combinando le strategie Top-Down e Bottom-Up, in modo da far avanzare i test in modo parallelo.

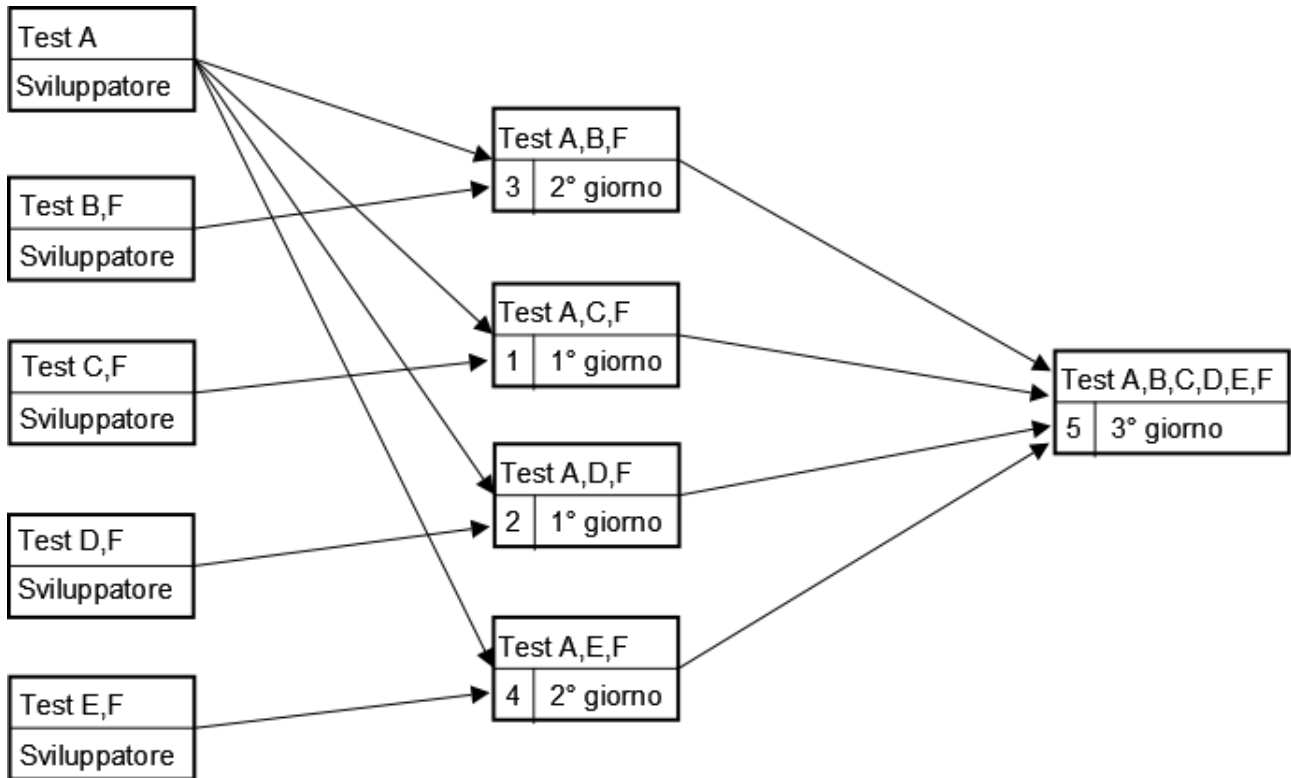
Ne viene di seguito fornita una rappresentazione grafica.



10.5 Schedulazione delle attività di testing

Le attività di testing verranno schedulate mediante l'uso del diagramma di PERT e basandosi sul diagramma delle attività del paragrafo 10.4.

Come già accennato, l'attività di testing verrà effettuata in un massimo di tre giorni.



11. Glossario

Terms

Definitions

Black Box Testing

Tipo di testing incentrato sul comportamento input/output che trascura la struttura interna delle componenti.

Bottom-Up Testing

Tipo di testing di integrazione, in cui i componenti vengono progressivamente integrati a partire da quelli di livello più basso.

Classi di equivalenza/ Equivalence classes

Insiemi di oggetti con proprietà comuni rispetto ad una funzionalità. Il risultato del testing su un oggetto è valido per l'intera classe di appartenenza.

Layer

Ogni layer (o strato) rappresenta una parte dei sottosistemi risultanti dalla decomposizione effettuata nel System Design.

Un layer può dipendere solo dai layer al di sotto di lui.

ODD

Acronimo di Object Design Document, documento di progettazione pre-implementativa del sistema.

RAD

Acronimo di Requirements Analysis Document, documento di analisi e raccolta dei requisiti.

Sandwich Testing

Tipo di testing di integrazione che combina i testing bottom-up e top-down.

SDD	Acronimo di System Design Document, documento di progettazione del sistema
Testing di Integrazione	Fase in cui le singole componenti vengono integrate e testate insieme.
Testing di Sistema	Fase finale del testing, in cui tutti i componenti sono stati integrati e il sistema viene testato nella sua interezza.
Testing di Unità	Fase in cui ogni componente viene testata singolarmente.
Top-Down Testing	Tipo di testing di integrazione in cui i componenti vengono progressivamente integrati a partire da quelli di livello più alto.
TP	Acronimo di Test Plan, il presente documento.
White Box Testing	Tipo di testing incentrato sulla struttura interna della componente. Viene testato ogni stato presente nel modello dinamico dell'oggetto.
Test Stub	Implementazione parziale di una componente da cui la componente testata dipende
Test Driver	Implementazione parziale di una componente che dipende da quella testata