responsabile@hourglabs.org

Piano di Qualifica

Versione 4.0

Informazioni documento

Nome	*Piano_ di_ Qualifica_ 4.0.pdf*
Versione	4.0
Data creazione	2012-12-16
Data ultima modifica	2013-09-21
Stato del Documento	Formale ad uso esterno
Redazione	Giovanni Morlin Sasa Ilievsky Riccardo Cesarotto
Verifica	Sasa Ilievski
Approvazione	Thomas Rossetto
Distribuzione	**hourglass** Prof. Tullio Vardanega Prof. Riccardo Cardin

Registro delle modifiche

Data	Versione	Ruolo	Descrizione	Autore
2013-09-21	4.0	Responsabile	Approvazione Documento	Thomas Rossetto
2013-09-15	3.1	Verificatore	Aggiornamento a seguito delle attività di verifica e resoconto dei test di sistema	Giovanni Morlin
2013-07-14	3.0	Responsabile	Approvazione Documento	Gioele Lorenzo Cresce
2013-07-13	2.8	Verificatore	Verifica Documento	Thomas Rossetto
2013-07-12	2.7	Progettista	Ampliamento delle tabelle sui test di unità e integrazione nella sezione pianificazione test	Riccardo Cesarotto
2013-07-12	2.6	Progettista	Ampliamento della tabella test di integrazione nella sezione pianificazione test	Riccardo Cesarotto
2013-07-3	2.5	Verificatore	Ampliamento della tabella test di unità nella sezione pianificazione test	Sasa Ilievsky
2013-07-2	2.4	Progettista	Redazione della sezione test di unità nella sezione pianificazione test	Giovanni Morlin
2013-02-11	2.1	Verificatore	Correzioni emerse dalla valutazione della **RP**	Sasa Ilievski
2013-01-29	2.0	Responsabile	Approvazione Documento	Giovanni Morlin
2013-01-25	1.4	Verificatore	Verifica Documento	Gioele Lorenzo Cresce
2013-01-23	1.3	Verificatore	Modifica Obiettivi di qualità - Affidabilità	Riccardo Cesarotto

Data	Versione	Ruolo	Descrizione	Autore
2013-01-20	1.2	Verificatore	Inserimento sezioni: Revisioni di progetto e Pianificazione ed esecuzione del collaudo	Sasa Ilievski
2013-01-19	1.1	Responsabile	Prima correzione delle incongruenze emerse da valutazione **RR**	Thomas Rossetto
2012-12-20	1.0	Responsabile	Approvazione documento	Paolo Bustreo
2012-12-17	0.3	Verificatore	Inserimento sezioni: Introduzione e Visione generale della strategia di Verifica	Riccardo Cesarotto
2012-12-16	0.2	Verificatore	Aggiunta sezione: Gestione amministrativa della revisione	Riccardo Cesarotto
2012-12-16	0.1	Verificatore	Prima stesura del Documento	Sasa Ilievski

Indice

1 Introduzione

1.1 Scopo del documento

Il presente documento definisce gli obiettivi di qualità e processo che il gruppo **hourglass** si impone di raggiungere mediante opportune strategie, qui definite, per garantire la qualità del prodotto **MyTalk**.

1.2 Scopo del prodotto

Il prodotto **MyTalk** costituisce una Web Application orientata all'interazione in tempo reale di tipo multimediale tra utenti. Caratteristica fondamentale del prodotto stesso sarà la possibilità di avviare e partecipare a conversazioni disponendo esclusivamente di browser e connessione Internet, senza la necessità, quindi, di alcuna installazione di software aggiuntivi o plug-in.

1.3 Glossario

Per migliorare la comprensione dei documenti ed evitare ogni ambiguità riguardante il linguaggio e i termini utilizzati, si allega il glossario nel file *Glossario_ v4.0.pdf*, al cui interno sarà possibile trovare una descrizione dei suddetti. Ogni ricorrenza di un termine da glossario sarà segnalata con l'utilizzo di *sottolineatura*.

1.4 Riferimenti

1.4.1 Normativi

- Capitolato d'appalto: **MyTalk**, rilasciato dal proponente **Zucchetti S.p.A**, reperibile all'indirizzo: http://www.math.unipd.it/ tullio/IS-1/2012/Progetto/C1.pdf
- Le Norme di Progetto, disponibili nel documento *Norme di Progetto_ v4.0.pdf*.
- Il Piano di Progetto, disponibili nel documento *Piano di Progetto_ v4.0.pdf*.

1.4.2 Informativi

- ISO/IEC 9126 http://en.wikipedia.org/wiki/ISO/IEC_9126
- ISO/IEC 15504 http://en.wikipedia.org/wiki/ISO/IEC_15504

2 Obiettivi di qualità

2.1 Qualità di prodotto

In questa sezione verranno descritti gli obiettivi di qualità di prodotto, e per ogni obiettivo verranno descritte le misure e metriche stabilite per poterli verificare in modo quantificabile. Prendendo come riferimento lo standard *ISO/IEC 9126*, il gruppo **hourglass** si impegna a garantire i seguenti indici di qualità per il prodotto **MyTalk**:

- Funzionalità
- Affidabilità
- Usabilità
- Manutenibilità

2.1.1 Funzionalità

L'obiettivo di Funzionalità rappresenta la capacità del prodotto software di soddisfare tutti i requisiti individuati nel documento Analisi dei Requisiti in modo completo ed economico.

2.1.2 Affidabilità

L'obiettivo di affidabilità rappresenta la capacità del prodotto software di mantenere un livello di prestazioni accettabile quando usato sotto determinate condizioni.

2.1.3 Usabilità

L'obiettivo di usabilità rappresenta la capacità del prodotto software di essere capito, appreso, usato e gradito dall'utente, quando usato sotto condizioni specificate

2.1.4 Manutenibilità

L'obiettivo di manutenibilità rappresenta la capacità del prodotto software di essere modificato. Per modifiche si intende correzioni, miglioramenti o adattamenti del software dovuti a cambiamenti nell'ambiente operativo, nei requisiti o nelle specifiche funzionali.

2.2 Qualità di processo

Per la realizzazione del prodotto, il gruppo **hourglass** intende utilizzare metriche di verifica di qualità.
Per il controllo della qualità dei processi, il gruppo utilizzerà la tecnica PDCA.
Questo metodo è in grado di migliorare, in modo continuo, i processi di sviluppo, facendo emergere eventuali problematiche nell'esecuzione del processo analizzato ed eventuali peggioramenti nella qualità di processo rilevata.
In particolare come metriche di valutazione del processo si dovrà prestare attenzione a:

- Tempo impiegato per il completamento del lavoro (Misura dell'Efficienza)
- Risorse utilizzate
- Soddisfacimento degli obiettivi decisi nella fase di planning

Una volta ottenuti i dati sarà compito dell'Amministratore valutarli e intervenire di conseguenza per migliorare il processo esaminato.

3 Visione generale della strategia di verifica

3.1 Organizzazione, pianificazione strategica e temporale, responsabilità

3.1.1 Organizzazione

Le attività di verifica verranno effettuate durante tutto il ciclo di vita del software. La verifica sarà suddivisa nelle seguenti quattro parti:

- **Controllo dei documenti:** durante l'intero ciclo di vita del software si avrà la produzione di documentazione. Le attività di controllo verranno effettuate al verificarsi di una delle seguenti condizioni:
  - Stesura della prima versione del documento.
  - Aggiunta di una modulo ad un documento.
  - Modifica sostanziale che porti il documento ad una versione parziale successiva.

  I documenti verranno controllati secondo la seguente strategia:
  1. Verrà controllata la correttezza ortografica grazie ad Aspell[1].
  2. Verrà controllata la correttezza lessicale per mezzo di un'accurata rilettura.
  3. Verrà controllata la correttezza dei contenuti rispetto alle aspettative del documento per mezzo di un'accurata rilettura.
  4. Verrà verificato il contenuto di ogni figura o tabella e controllato se ad ogni rappresentazione grafica corrisponda una didascalia.
  5. Verrà verificato che ogni documento rispetti le Norme di Progetto utilizzando gli strumenti più appropriati per ogni verifica. Al verificatore sarà assegnata la verifica di un documento tramite ticket, nelle modalità indicate nel documento *Norme Di Progetto*, e sarà suo compito notificare al redattore del documento eventuali consigli e correzioni tramite lo strumento sopra citato. Per far evolvere il documento alla versione definitiva, quest'ultimo deve essere prima approvato dal responsabile, che lo controllerà nella sua interezza, prima di procedere con l'approvazione.

- **Analisi dei Requisiti:** alla conclusione dell'attività di analisi, che comporta la produzione del documento *Analisi dei Requisiti*, i Verificatori avranno il compito di verificare la copertura e la tracciabilità dei requisiti.

- **Progettazione:** alla conclusione dell'attività di progettazione, i Verificatori avranno il compito di verificare la necessità e completezza dei componenti sviluppati durante la fase di progettazione. I requisiti non mappati su alcuna unità software sviluppata e i componenti software che non sono necessari a soddisfare alcun requisito verranno segnalati dal Verificatore tramite l'uso di ticket ai Progettisti. Il Verificatore dovrà stabilire i test da effettuare durante la fase di realizzazione per ogni unità e insieme di unità.

[1] http://aspell.net/

- **Realizzazione:** durante la fase di realizzazione, ciascuna unità software dovrà essere testata tramite i test definiti dai Verificatori nella fase precedente. Sarà compito del Verificatore verificare che i test siano adeguati e, nel caso in cui tali proprietà non siano verificate, segnalare tramite apposito ticket i problemi individuati. Nelle fasi di realizzazione o alla conclusione delle stesse, secondo quanto previsto dal Piano di progetto, si prevedono comunque i seguenti test: di unità, di integrazione, di sistema, di regressione, di accettazione.

3.1.2 Pianificazione strategica e temporale

La pianificazione delle attività di verifica è compito del Responsabile, il quale è incaricato di realizzare i primi ticket di verifica secondo quanto indicato nel *Piano di Progetto*.
Il metodo di creazione di tali ticket e la loro gestione sono descritti nelle *Norme di Progetto* e nella sezione *14 - Gestione milestones e ticketing* di tale documento.

3.1.3 Responsabilità

Le responsabilità delle attività di verifica e validazione sono identificate nelle figure di:

- **Amministratore**: Egli ha il compito di assicurare il corretto funzionamento dell'ambiente di lavoro per la creazione di tutte le parti del sistema e di gestire la documentazione e il controllo del corretto *versionamento* della stessa.
- **Responsabile**: Il Responsabile ha il compito di gestire la pianificazione delle attività di verifica e il controllo dell'effettiva esecuzione delle stesse.

3.2 Risorse necessarie

3.2.1 Risorse Umane

La verifica di qualità è svolta dai seguenti ruoli:

- **Responsabile di Progetto:** avrà il compito di accertare che le date di scadenza siano rispettate e che i documenti rispettino le *Norme di Progetto*. Respinge o approva, inoltre, le modifiche effettuate dai Verificatori sui documenti redatti e deve garantire al Committente un buono e corretto sviluppo del prodotto.
- **Verificatore:** segnala ai Programmatori eventuali errori di codice e controlla i documenti realizzati applicando i processi di verifica e di validazione per un corretto svolgimento dei processi. I problemi sollevati saranno presentati al Responsabile di Progetto per essere successivamente risolti.
- **Amministratore di Progetto:** stabilisce le norme delle attività di verifica, gestendo e risolvendo eventuali anomalie riscontrate. Il suo compito sarà anche quello di far rispettare le funzionalità lavorative.

- **Programmatore:** risolve le problematiche del codice segnalate dai Verificatori oppure riscontrate facendo il debugging del codice da lui prodotto.

3.2.2 Risorse software

Si useranno :

- software di verifica parzialmente o completamente automatici per la verifica dei documenti.
- software di controllo metriche e misurazione del codice realizzato
- Framework per i test di unità, integrazione e sistema.

3.2.3 Risorse hardware

Le macchine utilizzate devono costituire un ambiente di sviluppo adatto per la realizzazione del progetto, secondo quanto indicato nelle *Norme di Progetto*.

3.3 Risorse disponibili

3.3.1 Risorse software

I software utilizzati per la realizzazione del prodotto sono descritti nel documento *Norme di Progetto*.

3.3.2 Risorse hardware

Verranno utilizzati per lo svolgimento del progetto i computer personali (fissi o portatili) di ciascun membro del gruppo.

4 Tecniche

Nella seguente sezione verranno descritte le tecniche utilizzate durante le attività di verifica.

4.1 Analisi Statica

L'Analisi Statica verrà effettuata durante tutto l'intero ciclo di vita del software. Avrà lo scopo di analizzare il codice senza aver necessità di eseguirlo. Verrà applicata in questi due seguenti modi:

- **Inspection:** questa tecnica ha come finalità una lettura mirata del codice e dei documenti per rilevare eventuali tipi di errori. La verifica è fatta da Verificatori distinti e separati dai Programmatori. Si focalizza la ricerca su presupposti. L'Inspection è gestita con una lista di controllo che elenca i tipi di errore che possono essere fatti durante un processo. La correzione degli errori va documentata con un rapporto delle attività eseguite.
- **Walkthrough:** questa tecnica, basata sull'esperienza, ha come finalità la rilevazione e la presenza di difetti eseguendo una lettura critica del codice, a largo spettro. Si percorre il codice simulandone le possibili esecuzioni. Viene eseguita da gruppi misti di ispettori e sviluppatori, ma con ruoli ben distinti. Prima di applicare la correzione dei difetti rilevati e di effettuare le dovute modifiche ogni errore viene discusso tra Sviluppatori e Verificatori per evitare incomprensioni. Infine ogni attività svolta verrà memorizzata.

4.2 Analisi Dinamica

L'Analisi Dinamica ha lo scopo di verificare e validare il software tramite dei test per osservare il suo comportamento durante l'esecuzione. Ogni test deve essere ripetibile, ovvero dato lo stesso input e lo stesso ambiente la sua esecuzione deve fornire gli stessi risultati.

- **Test di Unità:** si controlla la correttezza di piccole unità di codice scritte dai Programmatori verificandone la coerenza con i requisiti fissati. Per fare ciò ci si aiuta con l'uso di driver e di stub.
- **Test di Integrazione:** verifica che l'integrazione delle unità di codice che hanno superato i test di unità non produca errori. Se si verificano problemi la soluzione andrà così cercata nell'interfaccia che li integra.
- **Test di Sistema:** verifica che i requisiti software descritti nell'*Analisi dei Requisiti* siano rispettati, procedendo così alla validazione del prodotto.
- **Test di Regressione:** a ogni modifica effettuata a una parte di codice precedentemente testato e funzionante, sarà opportuno eseguire nuovamente i test di unità e di integrazione. Se necessario, è possibile aggiungere nuovi test ma cercando di non perdere tempo su test simili ai precedenti ma meno efficaci. Per far ciò occorre mantenere un alto grado di incapsulamento e un basso grado di accoppiamento.

- **Test di Accettazione:** collaudo svolto dal Committente. Se superato, il prodotto si può rilasciare.

4.3 Metodi

Nella seguente sezione verranno indicati i metodi di analisi statica e dinamica che il gruppo decide di impiegare per verificare i documenti e il software prodotto.

4.3.1 Analisi dei documenti

- **Controllo sintattico e del periodo:** È noto che TexMaker e Aspell consentano di segnalare ed eliminare in modo efficiente gli errori di grammatica più evidenti nei documenti, ma non si può dire lo stesso degli errori di sintassi o di sostituzione, che provocano la creazione di parole grammaticalmente corrette ma non inerenti al contesto (es: 'file' e 'filo' sono grammaticalmente corretti ma non intercambiabili nel contesto). Stessa cosa per l'individuazione di periodi con costruzione artificiosa o sbagliata . Per questa ragione ciascun documento dovrà essere sottoposto ad un walkthrough da parte dei verificatori per individuare tali errori e correggerli nel modo più opportuno.
- **Rispetto delle norme di progetto:** molte delle regole contenute nelle *Norme di progetto* sono di carattere generale e impongono una struttura dei documenti che non può essere verificata in maniera automatica.

Per tutte quelle norme per cui non è stato definito uno strumento automatico di verifica si impone che i verificatori eseguano inspection sul rispetto di quelle norme in ciascun documento. Vi sono poi problematiche specifiche che verranno automatizzate con con degli script, per evitare attività di verifica inutile e probabilmente fallimentare dovuta alla natura umana.

4.3.2 Analisi statica del software

Oltre alle metriche che verranno chiarite nella sezione 5 il gruppo tenterà di fornire anche delle dimostrazioni formali di correttezza del codice per le funzionalità più critiche o più difficilmente verificabili tramite analisi dinamica. Nel caso in cui tale intento fallisse verrà cambiata la pianificazione dei test e fornita un insieme di test sufficientemente completi da garantire qualità minima del software.

5 Metriche

Le metriche permettono di ottenere delle misure che diano informazioni quantitative relative ai prodotti di ogni processo produttivo. A seguito verranno indicati i vicoli sulle metriche.

5.1 Metriche software

Le metriche software sono le seguenti:

- **Complessità ciclomatica:** rappresenta la complessità di un metodo. Misura il numero di cammini linearmente indipendenti che attraversano il grafo di flusso di controllo. Un valore di elevato della complessità ciclomatica riduce la manutenibiltà e rende difficile il controllo del flusso.
  Numero massimo desiderabile: 8.

- **Numero di parametri:** indica il numero di parametri formali per metodo. Un elevato numero di parametri di un metodo potrebbe essere indice del fatto che il metodo o la classe relativi vadano progettati nuovamente, così da presentare metodi più specializzati e quindi rendere il codice più manutenibile.
  Numero massimo desiderabile: 8.

- **Numero di campi dati per classe:** un numero elevato di attributi interni ad una classe potrebbe palesare la necessità di incapsulamento da parte di essi in nuove classi coese, che forniscano inoltre metodi utili al loro utilizzo.
  Numero massimo desiderabile: 12.

- **Numero di livelli di annidamento:** rappresenta il numero di livelli di annidamento dei metodi. Metodi troppo annidati complicano di molto l'attività di verifica, nonché le dipendenze da altri metodi che potrebbe dare margini a propagazione di errori.
  Numero massimo desiderabile: 6.

- **Indice di utilità:** numero di classi esterne al package che dipendono da classi interne ad esso. Se troppo basso indicherà che il package non fornisce molte funzionalità al suo esterno e quindi potrebbe essere scarsamente utile; se troppo alto, indicherà che altre classi sono strettamente dipendenti dal package in questione, e quindi potrebbe accadere che eventuali modifiche ad esso comportino costi elevati di adattamento delle classi che vi dipendono, qualora non fosse stato progettato adeguatamente il sistema di interfacce.

- **Indice di dipendenza:** numero di classi interne al package che dipendono da classi esterne ad esso. In generale va sempre minimizzato, aumentando quindi le funzionalità proprie di un package, senza la necessità di affidarsi al servizio offerto da altre classi esterne.

6 Gestione amministrativa della revisione

6.1 Comunicazione e risoluzione di anomalie

L'individuazione di un'anomalia determinata dalla violazione di una norma o di un'incoerenza nello sviluppo di una parte del sistema dovrà, in ogni occasione, essere segnalata perché vi si ponga rimedio.
Come già accennato nelle norme, il Verificatore dovrà aprire un ticket nel sistema di ticketing offerto GitHub[2] il quale, una volta accertato e validato dal Responsabile, sarà assegnato al

[2] https://github.com/

componente del gruppo impiegato nella risoluzione del problema.
Il Verificatore dovrà descrivere, nella sezione Descrizione, in maniera più semplice e concisa possibile le motivazioni della creazione del ticket.
Ogni ticket di correzione dovrà, nella sezione Commenti, contenere le seguenti note:

- Documento (o file) in cui è presente l'anomalia
- Descrizione dell'anomalia
- Comportamento atteso
- Comportamento rilevato

Il Verificatore dovrà prestare particolare attenzione se verrà rilevata una discrepanza tra il prodotto realizzato e quello atteso.

6.2 Trattamento discrepanze

Se l'anomalia individuata riguarderà il mancato soddisfacimento di un requisito di una parte precedentemente progettata, il Verificatore dovrà segnalare tale anomalia sempre tramite un ticket, recapitato al Responsabile, segnalando in modo dettagliato:

- L'origine della discrepanza
- Il requisito non soddisfatto
- Il motivo del mancato soddisfacimento dello stesso

Sarà dunque compito del Responsabile valutare la gravità della discrepanza segnalata, dando una stima dei costi e del tempo necessari alla risoluzione della stessa. Verrà quindi elaborato un piano correttivo che descriverà il metodo di risoluzione della discrepanza.
A seguito della modifiche effettuate, il Verificatore dovrà verificare nuovamente i documenti (o il codice) modificati.

6.3 Procedure di controllo della qualità

Il controllo della qualità di processi e prodotti verrà eseguito come di seguito esplicato. Il controllo di qualità del processo verrà garantito grazie al già descritto ciclo PDCA. Il controllo di qualità del prodotto verrà garantito da:

- Quality Assurance : l'insieme delle attività attuate per garantire il raggiungimento degli obbiettivi di qualità. Sarà descritto nelle prossime versioni del documento corrente ma prevederà sicuramente tecniche di analisi statica e analisi dinamica. Verifica e Validazione : sono i due processi che determinano se il sistema viene costruito nel modo corretto (Verifica) e se il sistema è corretto secondo le richieste espresse dal proponente (Validazione). La verifica andrà eseguita costantemente durante l'intera durata del progetto e prevede le modalità già descritte in precedenza nel presente documento.

7 Resoconto delle attività di verifica

I contenuti di questa sezione sono i resoconti delle attività di verifica effettuate durante le fasi di sviluppo del progetto.

7.1 Revisione dei Requisiti

Durante la fase di *Analisi dei Requisiti* sono stati prodotti documenti testuali e sono stati controllati applicando tecniche di analisi statica e rispettando metodi descritti nelle *Norme di Progetto*. Il tracciamento dei requisiti è stato svolto come descritto nella sezione *8.4.1 - Tracciamento dei Requisiti* del documento *Norme di Progetto_ 2.0.pdf*. Il Verificatore nei documenti svolti ha trovato principalmente errori grammaticali e/o di battitura, oltre al mancato rispetto di alcune regole di formattazione descritte nella sezione *8.2 - Ambiente dei Documenti* del documento *Norme di Progetto_ 2.0.pdf*. I documenti sono stati quindi corretti. Alcune sezioni presentate in modo poco comprensibile sono state segnalate così da permettere una più chiara formulazione.

7.2 Revisione di Progettazione

Nella fase di Progettazione sono state applicate, come nella fase precedente, le attività di analisi statica per la verifica dei documenti ma in modo più efficiente ed efficace. Ciò è stato possibile grazie a un miglioramento del processo di verifica aumentandone i controlli. Abbiamo in programma di affinare ulteriormente queste tecniche automatizzando i controlli in fase di commit, sia dei documenti che dei codici sorgenti. Auspicabilmente questo avverrà durante l'attività di PDC. Abbiamo utilizzato lo strumento hourglass Requrements Tool [3] per il vari tracciamenti quali : Test-Requisito, Requisito - Modalità di verifica, Requisito - Test.

7.3 Revisione di Qualifica

In questa fase oltre alle normali attività di verifica della documentazione è stata svolta l'analisi statica sul codice sorgente tramite la misurazione delle metriche definite nella sezione 5.1. Inoltre sono stati effettuati i test di integrazione e di unità pianificati precedentemente: non appena un test alla fine della propria esecuzione dava esito positivo ne veniva riportato il successo attribuendogli lo stato di Superato. Maggiori informazioni riguardo i test effettuati per l'analisi statica, test di unità e test di integrazione sono disponibile nel documento allegato denominato denominato *Esito_ dei_ test_ v1.0.pdf*.

[3] http://www.hourglabs.org/

8 Pianificazione dei test

8.1 Test di sistema

Nella presente sezione verranno presentati i test di sistema che dovranno essere effettuati prima del rilascio del prodotto. Tramite i seguenti test si vuole dimostrare di aver soddisfatto i requisiti forniti in *Anlisi dei requisiti* e quindi le aspettative del cliente. Il prodotto potrà essere proposto al cliente quando tutti i test avranno sollevato tutte le anomalie presenti.

Codice test	Descrizione	Stato	Requisito
TS-1	Viene verificato se il sistema effettivamente risponde in maniera positiva alla pressione del pulsante di richiesta di collegamento e all' inserimento dell'ID-univoco da parte dell'utente.	Superato	RUFUO 1
TS-2	Viene verificato se il sistema effettivamente risponde in maniera positiva alla pressione del pulsante chiamata e all'inserimento dell'ID-univoco da parte dell'utente.	Superato	RUFUO 1.1
TS-3	Viene verificato se un utente riesce a inviare un messaggio testuale ad un altro utente, e se questo lo visualizza correttamente.	Superato	RUFUD 1.2
TS-4	Viene verificato se un utente riesce a invitare un nuovo utente in una chiamata già in atto. Viene verificato poi se ogni partecipante riesce a condivide il proprio audio e video con gli altri.	Superato	RUFUD 1.3
TS-5	Viene verificato se un utente visualizza una notifica che lo avvisi di una chiamata in arrivo. L'utente dovrà poter accettare o rifiutare la chiamata, o ignorare la chiamata (tramite comando apposito o attendendo un intervallo di 30 secondi).	Superato	RUFUO 1.4
TS-6	Viene verificato se un Utente Autenticato durante una video-chiamata con un altro utente visualizza delle statistiche sulla connessione come durata,latenza e numero di byte inviati e ricevuti.	Superato	RUFUO 1.5
TS-7	Viene verificato che tutte le funzioni siano raggiungibili in meno tempo possibile per aumentare l'usabilità del prodotto software.	Superato	RGQD 17
TS-8	Viene verificato che tutti i dati sensibili presenti nel database non siano in chiaro, ma criptati con un algoritmo di criptazione sufficiente a garantire un'adeguata sicurezza agli utilizzatori del prodotto software.	Superato	RGQD 18

Codice test	Descrizione	Stato	Requisito
TS-9	Viene verificato che sia possibile registrare un nuovo utente.	Superato	RUFUO 2
TS-10	Viene verificato se un utente riesce ad in inserire le informazioni necessarie, quali username(carattere di tipo alfanumerico da 8 a 32 caratteri), password(carattere di tipo alfanumerico da 8 a 32 caratteri), email,nome(carattere di tipo alfanumerico da 2 a 32 caratteri), cognome(carattere di tipo alfanumerico da 2 a 32 caratteri), data di nascita e sesso riuscendo poi a completare la registrazione.	Superato	RUFUO 2.1
TS-11	Viene verificato se dopo aver seguito le istruzioni di conferma inviate via mail, l'utente visualizza il proprio profilo come valido una volta eseguita l'autenticazione.	Superato	RUFUO 2.2
TS-12	Viene verificato se dopo aver inserito username e password l'amministratore ha accesso alla sezione amministratore.	Superato	RAFUD 26
TS-13	Viene verificato che l'amministratore, dopo la pressione del tasto logout, perda i sui diritti di accesso e venga reindirizzato alla pagina principale del sistema .	Superato	RAFUD 26.3
TS-14	Viene verificato che l'amministratore possa visualizzare tutti gli utenti connessi al sistema in quel momento.	Superato	RAFUD 27
TS-15	Viene verificato se l'utente, dopo aver inserito username e password negli appositi campi, riesce ad accedere alla sua pagina personale e ad accedere alle funzioni a lui permesse.	Superato	RUFUO 3
TS-16	Viene verificato che l'utente, dopo la pressione del tasto logout, perda i sui diritti di accesso e venga reindirizzato alla pagina principale del sistema .	Superato	RUFUO 3.4
TS-17	Viene verificato se un utente autenticato riesce a modificare i parametri del suo account.	Superato	RUFUO 4
TS-18	Viene verificato se un utente autenticato riesce a modificare la propria password. Se la modifica va a buon fine l'utente perde i diritti di accesso e viene indirizzato alla pagina principale del sistema per poi dargli la possibilità di autenticarsi con le nuove credenziali.	Superato	RUFUO 4.1

Codice test	Descrizione	Stato	Requisito
TS-19	Viene verificato se un utente autenticato riesce a modificare la mail a cui è associato il suo account e se ciò avviene correttamente viene verificato che arrivi la mail che chiede di confermare l'operazione tramite le istruzioni inviate.	Superato	RUFUO 4.2
TS-20	Viene verificato se un utente autenticato riesce a cancellare il proprio account. Dopo tale operazione l'utente deve essere automaticamente deautenticato e indirizzato alla pagina principale del sistema.	Superato	RUFUD 4.3
TS-21	Viene verificato se l'utente, dopo aver fatto l'accesso alla pagina di recupero password, fornendo l'email associata al suo profilo, riceve una mail con le istruzioni da eseguire per avere di nuovo accesso alla sua area privata.	Superato	RUFUD 4.4
TS-22	Viene verificato se un utente autenticato dopo aver modificato la propria immagine personale possa vederla modificata nella sua pagina profilo.	Superato	RUFUD 4.5
TS-23	Viene verificato se un utente autenticato dopo aver modificato il proprio stato(online, occupato oppure offline) , possa vedere la modifica, e che tutti i suoi contatti nella rubrica (se ne ha) possano vedere il cambiamento.	Superato	RUFUD 4.7
TS-24	Si deve verificare che l'utente abbia correttamente accesso alle informazioni del suo profilo.	Superato	RUFUO 4.9
TS-25	Viene verificato se l'utente autenticato può accedere alla sua rubrica, per poi gestirla.	Superato	RUFUD 5
TS-26	Viene verificato se l'utente autenticato, una volta aggiunto un altro utente autenticato alla sua rubrica, sia aggiunto in automatico alla rubrica dell'utente inserito.	Superato	RUFUD 5.1
TS-27	Viene verificato che se un utente crea un gruppo di contatti personalizzato nella sua rubrica quest'ultimo possa essere visibile dopo averla creata.	Superato	RUFUD 5.4
TS-28	Viene verificato che la web application sia compatibile con il browser Chrome versione 24 o superiore mediante l'apertura di quest'ultimo, verificandone poi l'effettiva compatibilità.	Superato	RGVO 8

Codice test	Descrizione	Stato	Requisito
TS-29	Viene verificato che l'amministratore possa bloccare un utente per un periodo di tempo definito, e che effettivamente l'utente bloccato non possa avere accesso al sistema.	Superato	RAFUD 27.1.4

Di seguito viene presentata la tabella che associa ad ogni requisito il tipo di verifica che servirà per validarlo.

Codice requisito	Test associato
RUFUO 1	TS-1
RUFUO 1.1	TS-2
RUFUD 1.1.2	Test non tracciabile
RUFUD 1.1.3	Test non tracciabile
RUFUD 1.2	TS-3
RUFUD 1.2.1	Verifica Statica
RUFUD 1.2.2	Verifica Statica
RUFUD 1.3	TS-4
RUFUD 1.3.1	Verifica Statica
RUFUO 1.4	TS-5
RUFUO 1.4.1	Verifica Statica
RUFUO 1.4.2	Verifica Statica
RUFUO 1.4.3	Verifica Statica
RUFUO 1.5	TS-6
RUFUO 1.5.1	Verifica Statica
RUFUD 1.5.2	Verifica Statica
RUFUO 1.5.3	Verifica Statica
RGVO 10	Test non tracciabile
RGVO 10.1	Test non tracciabile
RGVF 11	Verifica Statica
RGVF 12	Verifica Statica
RGVF 13	Verifica Statica
RGVF 14	Verifica Statica
RGVF 15	Verifica Statica
RGVO 16	Test non tracciabile
RGQD 17	TS-7
RGQD 18	TS-8
RGQO 19	Test non tracciabile
RUFUO 2	TS-9
RUFUO 2.1	TS-10
RUFUO 2.1.1	Verifica Statica
RUFUO 2.1.1.1	Verifica Statica

Codice requisito	Test associato
RUFUO 2.1.1.2	Verifica Statica
RUFUO 2.1.1.3	Verifica Statica
RUFUO 2.1.2	Verifica Statica
RUFUO 2.1.2.1	Verifica Statica
RUFUO 2.1.2.2	Verifica Statica
RUFUO 2.1.2.3	Verifica Statica
RUFUO 2.1.3	Verifica Statica
RUFUO 2.1.4	Verifica Statica
RUFUO 2.1.4.1	Verifica Statica
RUFUO 2.1.4.2	Verifica Statica
RUFUO 2.1.4.3	Verifica Statica
RUFUO 2.1.5	Verifica Statica
RUFUO 2.1.5.1	Verifica Statica
RUFUO 2.1.5.2	Verifica Statica
RUFUO 2.1.5.3	Verifica Statica
RUFUO 2.1.6	Verifica Statica
RUFUO 2.1.7	Verifica Statica
RUFUO 2.2	TS-11
RUFUO 2.2.1	Verifica Statica
RGQD 20	Test non tracciabile
RGQO 21	Test non tracciabile
RGQO 22	Test non tracciabile
RGQO 23	Test non tracciabile
RGQO 24	Test non tracciabile
RGQO 25	Test non tracciabile
RAFUD 26	TS-12
RAFUD 26.1	Verifica Statica
RAFUD 26.2	Verifica Statica
RAFUD 26.3	TS-13
RAFUD 27	TS-14
RAFUD 27.1	Verifica Statica
RAFUD 27.1.1	Verifica Statica
RAFUD 27.1.2	Verifica Statica
RAFUD 27.1.3	Verifica Statica
RAFUD 27.1.3.1	Verifica Statica
RAFUD 27.1.4	TS-29
RAFUD 27.1.5	Verifica Statica
RUFUO 3	TS-15
RUFUO 3.1	Verifica Statica
RUFUO 3.2	Verifica Statica
RUFUF 3.3	Verifica Statica
RUFUO 3.4	TS-16

Codice requisito	Test associato
RUFUO 4	TS-17
RUFUO 4.1	TS-18
RUFUO 4.2	TS-19
RUFUO 4.2.1	Verifica Statica
RUFUD 4.3	TS-20
RUFUD 4.4	TS-21
RUFUD 4.5	TS-22
RUFUF 4.6	Verifica Statica
RUFUF 4.6.1	Verifica Statica
RUFUF 4.6.2	Verifica Statica
RUFUF 4.6.3	Verifica Statica
RUFUF 4.6.4	Verifica Statica
RUFUF 4.6.5	Verifica Statica
RUFUF 4.6.6	Verifica Statica
RUFUD 4.7	TS-23
RUFUD 4.8	Verifica Statica
RUFUO 4.9	TS-24
RUFUD 5	TS-25
RUFUD 5.1	TS-26
RUFUD 5.1.1	Verifica Statica
RUFUD 5.1.2	Verifica Statica
RUFUF 5.2	Verifica Statica
RUFUF 5.3	Verifica Statica
RUFUD 5.3.1	Verifica Statica
RUFUD 5.3.2	Verifica Statica
RUFUF 5.3.3	Verifica Statica
RUFUF 5.3.4	Verifica Statica
RUFUD 5.3.5	Verifica Statica
RUFUD 5.3.6	Verifica Statica
RUFUD 5.4	TS-27
RUFUF 5.4.1	Verifica Statica
RUFUD 5.4.2	Verifica Statica
RUFUF 5.5	Verifica Statica
RUFUF 5.5.1	Verifica Statica
RUFUF 5.5.2	Verifica Statica
RUFUF 5.5.3	Verifica Statica
RGVO 6	Test non tracciabile
RGVO 7	Test non tracciabile
RGVO 8	TS-28
RGVO 9	Test non tracciabile
RGVO 9.1	Test non tracciabile

8.2 Test di unità

Le classi riguardanti la grafica del software non sono verranno testate poichè dotate di una scarsa complessità, di conseguenza le classi del package `it.hourglass.myTalk.client.view` e `it.hourglass.myTalk.client.event` non saranno soggette a test di unità: la loro correttezza verrà visualizzata interagendo direttamente con l'applicativo.
Lo stesso vale per le classi dei package `it.hourglass.myTalk.client.wrappers` e `it.hourglass.myTalk.client.communicate` che presentano caratteristiche proprie del *framework* *GWT* e della tecnologia *webRTC* per le quali il team non ha trovato *tool* e strategie utili per effettuare un'azione di testing consistente ed efficiente. La seguente sezione presenta i test d'unità pianificati per poter verificare il corretto funzionamento delle classi implementate.
Legenda delle abbreviazioni utilizzate per i nomi dei pacchetti software:

ELB: it.hourglass.model.server.elaborator
ITF: it.hourglass.model.server.interf
DAO:it.hourglass.myTalk.model.DAO
SHA:it.hourglass.myTalk.client.shared;
PRE:it.hourglass.myTalk.client.presenter
PRO:it.hourglass.myTalk.client.presenter.ProfileMangment

Nome test	Descrizione	Stato
DialerTest	**Registrazione nickname lato server:** Viene verificato se il server registra in modo corretto un nickname ricevuto.	Superato
SwitchTest	**Elaborazione JSON:** Viene verificato se il server gestisce ed elabora in maniera corretta un JSON creato ad-hoc.	Superato
ParserInTest	**Analisi tipi di JSON:** Test sulla corretta analisi dei tipi dei JSON da parte del server	Superato
ParserOutTest	**Comunicazione Websocket:** Controlla che vengano inviati correttamente messaggi via Websocket	Superato
DaoImplTest	**Operazioni con il DAO:** Test sulla corretta interazione del software con il proprio database	Superato
ProfileTest	**Gestione messaggi e lista amici dei profili utente:** Viene verificata al corretta gestione della lista amici di un utente e della propria lista messaggi	Superato
UserlistPreTest	**Gestione stato UserList:** Verifica del corretto cambiamento di uno stato relativo ad un oggetto della classe PRE.UserlistPresenter	Superato

Nome test	Descrizione	Stato
SignedMenuPreTest	**Gestione sessioni con il server e menù:** Controlla che venga creata una sessione attiva con il server	Superato
PopupMessagePreTest	**Gestione invio messaggi:** Verifica che la classe PRE.PopupMessagePresenter effettui le istruzioni prestabilite all'invio di un messaggio	Superato
SignedMenuPreTest	**Gestione sessioni con il server e menù:** Controlla che venga creata una sessione attiva con il server	Superato
BaseViewPreTest	**Gestione della view base:** Viene controllata la gestione delle view dopo la chiamata di alcuni metodi	Superato
UserListPreTest	**Gestione status UserLis:** Viene controllata lapossibilità di cambiare lo status ad un utente	Superato
SignedMenuPreTest	**Gestione sessioni con il server e menù:** Viene controllata la possibilità di instaurare una sessione con il server	Superato
PopupMessagePreTest	**Gestione invio messaggi:** Verifica il corretto comportamento del presenter in questione, all'invio di un messaggio.	Superato
RegistrationPresenterTest	**Accertamento delle funzionalità di controllo dei dati in fase di registrazione:** Verifica la possibilità di registrare un nuovo utente nella piattaforma	Superato
PresenterLoginTest	**Accertamento nell'uso dei parametri di accesso alla piattaforma:** Verifica che i parametri recuperati dalla view non subiscano mutamenti durante l'operazione di autenticazione	Superato
ContactListPresenterTest	**Verifico le operazioni disponibili sulla lista contatti:** Controllo che operazioni disponibili non facciano emergere stati si inconsistenza nella piattaforma	Superato
TestValuesCheck	**Accertamento della verifica dei parametri nella fase di registrazione:** Controllo che la classe effettui correttamente i docuti controlli su parametri in input	Superato
ExtendedDataProfileTest	**Accertamento della verifica dei parametri inseriti dall'utente:** Verifica che i campi opzionali dell'account personale, inseriti dall'utente, abbiano una lunghezza limitata	Superato

Nome test	Descrizione	Stato
AvatarProfileManagment PresenterTets	**Accertamento della creazione di oggetto della classe PRE.AvatarProfileManagmentPresenter:** Verifica che sia possibile istanziare un'oggeto `PRE.AvatarProfileManagmentPresenter`	Superato
PersonalDataProfile ManagmentPresenterTest	**Accertamento della verifica dei parametri della classe :** Vengono verificati i parametri recuperati dalla view associata	Superato
NewEmailConfirmPopup PresenterTest	**Accertamento della creazione di un oggetto PRE.NewEmailConfirmPopupPresenter:** Verifica che sia possibile istanziare un'oggeto `PRE.NewEmailConfirmPopupPresenterTest`	Superato

8.3 Test di integrazione

Si testeranno i componenti a seconda delle funzionalità da loro offerte, mappando i moduli alla specifica funzionalità: l'integrazione avverrà in simultanea per certi moduli in modo da minimizzare la ricerca della causa dell'eventuale errore. Si fa presente che i seguenti test rappresentano una piccola porzione di tutti i test effettuabili e non hanno l'ambizione di essere quelli definitivi. Essi verranno espansi e dettagliati in modo definitivo nel corso della prima parte della 1PDC.
A pagina seguente sono elencati i test di integrazione pianificati.

Codice componente	Test di integrazione	Esito
MPTest	Vengono testate le funzionalità offerte dal model al presenter mediante l'uso di metodi remoti. In questo senso è stato possibile verificare che l'aggregazione tra queste due componenti non ha evidenziato la presenza di errori. Sono stati usati i seguenti metodi per testare le diverse funzionalità • `testLoginPresenter()`: Verifica che tutte le operazione dedite all'autenticazione dell'utente siano eseguite senza provocare malfunzionamenti ocomportamenti inattesi dell'applicativo. • `testBaseViewPresenter()`: Verifica che il controllo sulla sessione, tra client e server una volta che l'utente è autenticato, non dia luogo a inconsistenze o falle di sicurezza. • `testContactProfilePrsenter()`: Verifica che tutte le azioni volte al recupero delle informazioni su un contatto di un utente registrato alla piattaforma. • `testFriendContactPresenter()`: Verifica tutte le azioni che interessano il recupero della lista amici di un utente registrato alla piattaforma. • `testRegistrationPresenter()`: Verifica nell'insieme le operazione che servono a sbloccare un'account appena creato. • `testTextMessagePresenter()`: Verifica le operazioni di cancellazione di un messaggio inviato tra due utenti registrati nella piattaforma. • `testSignedMenuPresenter()`: Verifica le operazioni che portano alla cancellazione della sessione tra un client e un server. • `testContactManagmentPresenter()`: Verifica l'operazione di richiesta di un'amicizia tra due utenti registrati alla piattaforma. • `testPopupMessagePresenter()`: Verifica le operazioni di invio e ricezione di messaggi tra due utenti registrati nella piattaforma. • `testPasswordRecoverPresenter()`: Verifica che le operzioni per il cambio di una password dimenticata.	Superato